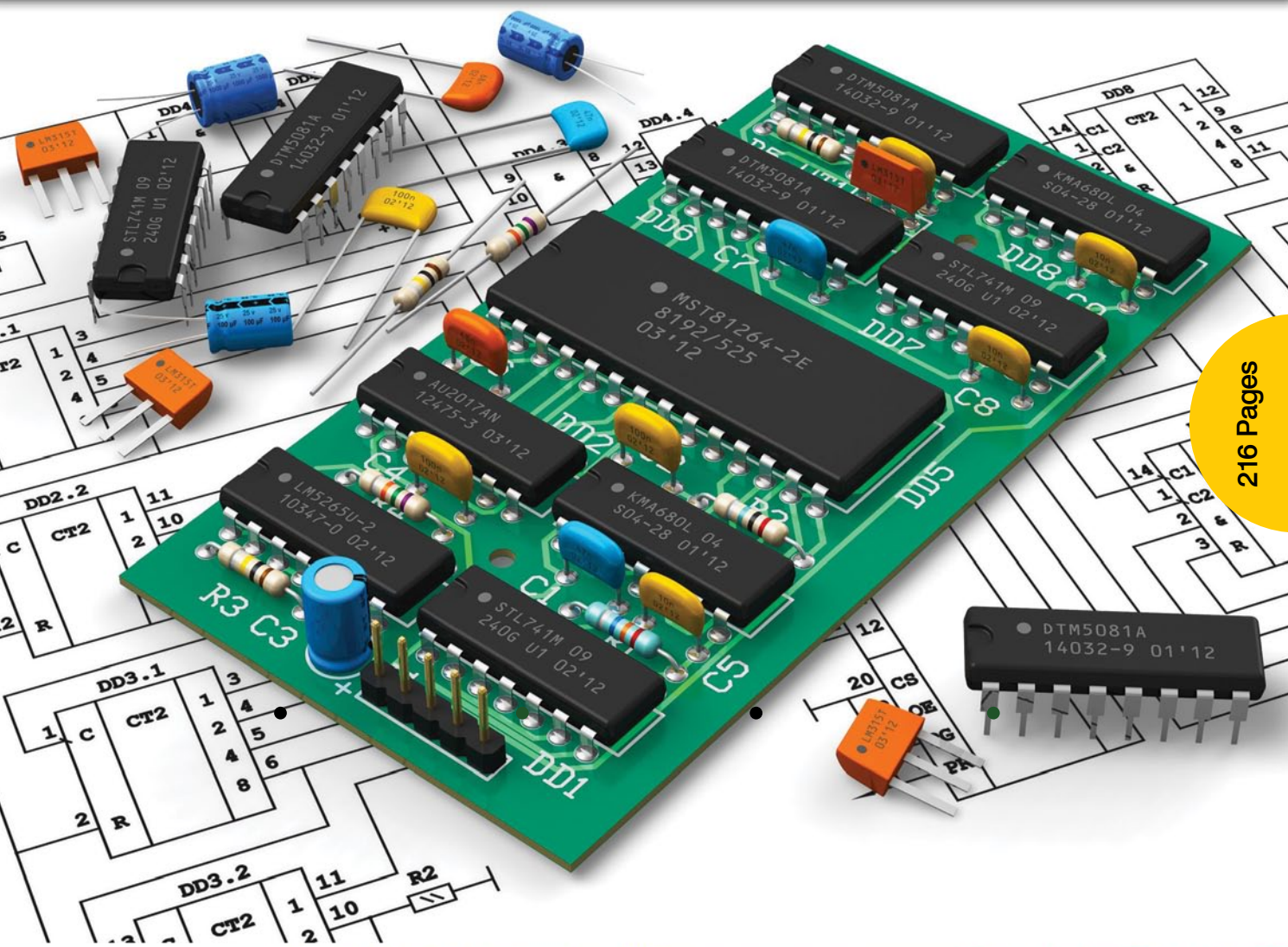
An EFY Group Publication

ISBN 978-81-88152-26-1

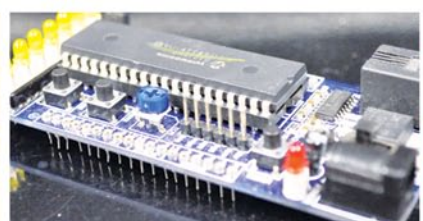
Price $ 10

Electronics Projects VOLUME 26

A Compilation of 21 tested Electronic Construction Projects and 71 Circuit Ideas for Electronics Professionals and Enthusiasts



216 Pages



efy ♥ electronics

efy group started in 1969 with the launch of electronics for you magazine. the love and support we received from indian electronics fraternity motivated us to launch many more products and services. some of them have grown into leaders in their respective categories...

electronics for you
india's #1 electronics magazine. leader in whole of south asia. read by half-a-million techies with a keen interest in electronics.

electronics bazaar
india's #2 electronics magazine and #1 b2b magazine. read by decision makers who either manufacture, buy or sell electronics goods or components.

efy yellow pages
india's only b2b yellow pages for electronics. it lists 2500 product categories, and the 7000-plus organisations in india, which supply them. excellent resource for buyers.

efy expo
launched in 2011, it is india's fastest growing electronics expo, and the only one that gets the engineering, manufacturing and trading communities under a single roof.

electronicsforu.com
india's #1 website for electronics engineers. 60% of its traffic is from india, the rest from all across the globe.

efy awards
india's leading awards for electronics brands and organisations–26 are decided by the readers of efy magazine, and 4 by a knowledgeable jury.

elcina-efy awards
a joint effort with india's leading electronics association (ELCINA), these awards are decided through self-nomination followed by a jury review.

facebook.com/designelectronics
world's #1 facebook community for electronics engineers! 100,000+ design engineers are friends of this page. 66% are from India, the rest from all across the globe.

FOR ENQUIRIES, CALL ATUL AT: +91-11-40596600 (EXT 228) OR 8800094211
EMAIL: efyenq@efyindia.com

ELECTRONICS PROJECTS
VOL. 26

First Published in this Edition, November 2013

ISBN 978-81-88152-26-1

Published by Ramesh Chopra for EFY Enterprises Pvt Ltd,
D-87/1, Okhla Industrial Area, Phase 1, New Delhi 110020
Typeset at EFY Enterprises Pvt Ltd

ELECTRONICS PROJECTS
VOL. 26

EFY GROUP
Technology Drives Us
EFY Enterprises Pvt Ltd
D-87/1 Okhla Industrial Area, Phase 1
New Delhi 110020

About EFY Labs

EFY Group has modern lab setup for R&D and testing various electronics projects for publications. All the projects published in EFY were tested at EFY Labs. Apart from this online edition, all the print versions including Microcontroller-Based Projects (First edition), Simple Projects You Can Make At Home, Electronics Pojects Volume 1 through 25, Chip-Talk and Learn to Use Microprocessors books were compiled by EFY Labs.

About EFY Group

Electronics For You, South Asia's most popular electronics magazine is one of the products of EFY Group. The Group currently offers a bouquet of specialised publications which include Open Source For You, Electronics Bazaar and Facts For You. The publications enjoy a huge readership and have managed to attract non-technical readers with their simple language and easy-on-the-eye design.

The Group also publishes directories and books, and organises several leading technology events. Its web-portals, which include electronicsforu.com, efytimes.com, eleb2b.com, linuxforu.com, electronicsb2b.com, investinelectronics.in and electronicsofthings.com have become leaders in their respective categories. The EFY Expo India, EFY Awards, Open Source India, Electronics Rocks and ELCINA-EFY Awards are some of the annual events organised by the Group.

Foreword

This volume of Electronics Projects is the twenty sixth in the series published by EFY Enterprises Pvt Ltd. It is a compilation of 21 construction projects and 71 circuit ideas published in Electronics For You magazine during 2005.

In keeping with the past trend, all modifications, corrections and additions sent by the readers and authors have been incorporated in the articles. It is a sincere endeavour on our part to make each project as error-free and comprehensive as possible. However, EFY is not responsible if readers are unable to make a circuit successfully, for whatever reason.

This collection of tested circuit ideas and construction projects in a handy volume would provide all classes of electronics enthusiasts—be they students, teachers, hobbyists or professionals—with a valuable resource of electronic circuits, which can be fabricated using readily-available and reasonably-priced components. These circuits could either be used independently or in combination with other circuits described in this and other volumes. We are confident that this volume, like its predecessors, will generate tremendous interest amongst the readers.

Table of Contents

Section A: Construction Projects

Section B: Circuit Ideas

SECTION A:
CONSTRUCTION PROJECTS

MICROCONTROLLER-BASED REAL-TIME CLOCK

■ K.S. SANKAR

In most applications, a microcontroller can satisfy all the system requirements with no additional integrated circuits. Due to their low cost and a high degree of flexibility, microcontrollers are finding way into many applications that were previously accomplished by mechanical means or combinational logic. One such application is a real-time clock.

Here's a real-time clock using Atmel AT89S8252. The software for the microcontroller is written in Bascom51 (a powerful BASIC compiler), which is capable of creating a hex file. The hex file code can be burnt into the microcontroller using any commonly available programmer or kit.

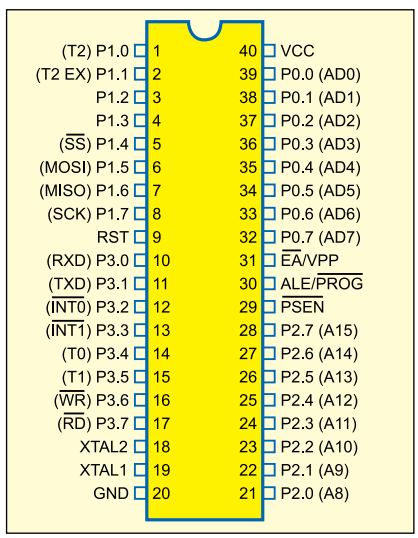


Fig. 1: Pin assignments of AT89S8252

IC AT89S8252 is a low-power, high-performance CMOS 8-bit microcontroller. It is manufactured using Atmel's high-density non-volatile memory technology and is compatible with the industry-standard 80C51 instruction set and pin-out. The powerful AT89S8252 microcontroller provides a highly flexible and cost-effective solution to many embedded control applications. Its main features are:

1. Compatibility with MCS-51 products
2. 8kB in-system reprogrammable downloadable Flash memory with SPI serial interface for program downloading and
3. 2kB EEPROM with endurance of 100,000 write/erase cycles
4. 4V–6V operating range
5. Fully static operation: 0 Hz to 24 Mhz
6. Three-level program memory lock
7. 256×8-bit internal RAM
8. 32 programmable I/O lines
9. Three 16-bit timer/counters
10. Nine interrupt sources
11. Programmable UART serial channel
12. SPI serial interface
13. Low-power idle and power-down modes
14. Interrupt recovery from power-down
15. Programmable watchdog timer
16. Dual data pointer
17. Power-off flag

Fig. 1 shows the pin assignments of AT89S8252.

Fig. 2 shows the block diagram of the real-time clock using AT89S8252 microcontroller and a few external components to display the time in HH.MM.SS format on six 7-segment displays. Switches S2, S3, S4 and S5 are used for hour increment, hour decrement, minute increment and minute decrement, respectively, while switch S6 is used for resetting the clock display to all zeroes.

PARTS LIST

Semiconductors:
IC1 - 7805, 5V regulator
IC2 - AT89S8252 microcontroller
IC3 - 74LS244 octal line driver
IC4 - ULN2803 octal transistor array
DIS1-DIS6 - LTS543 commoncathode 7-segment display
LED1 - Red LED

Resistors (all ¼-watt, ±5% carbon):
R1 - 1-kilo-ohm
R2 - 10-kilo-ohm
R3-R11 - 100-ohm

Capacitors:
C1 - 100µF, 25V electrolytic
C2 - 0.1µF ceramic
C3, C4 - 22pF ceramic
C5 - 10µF, 10V electrolytic

Miscellaneous:
X_{TAL} - 6MHz crystal
S1-S6 - Push-to-on switch

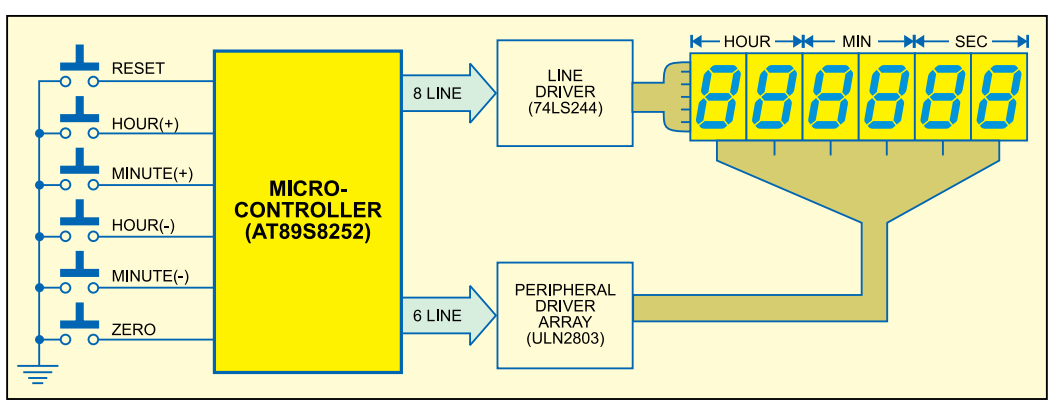


Fig. 2: Block diagram of real-time clock using AT89S8252 microcontroller

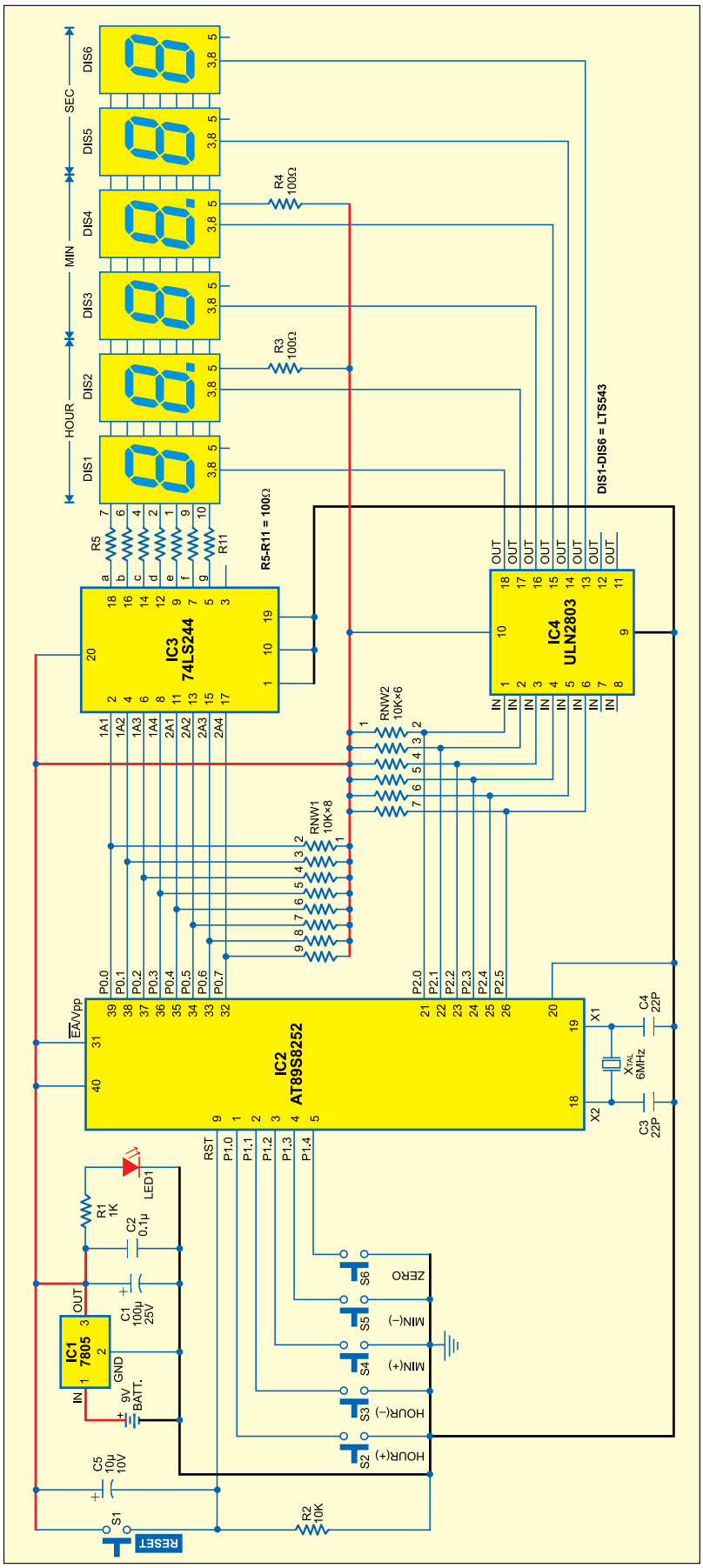


Fig. 3: Circuit of the real-time clock built around AT89S8252 microcontroller

Out of the three ports of the microcontroller, one port is used for setting the time and the other two ports are used for displaying the time. Line driver and Darlington driver array are used to drive the segment data and enable the 7-segment display, respectively.

Ciruit discription

Fig. 3 shows the circuit of the real-time clock built around AT89S8252 microcontroller (IC2). The power supply from the 9V battery is down converted and regulated by IC 7805 (IC1) to provide regulated 5V to the circuit. Glowing of LED1 indicates that power to the circuit is switched on. Resistor R1 acts as the current limiter.

Switch S1 is used to manually reset the microcontroller, while the power-on reset signal for the microcontroller is derived from the combination of capacitor C5 and resistor R2. EA/Vpp pin (pin 31) of the microcontroller is connected to Vcc to enable internal program execution. Pins 19 and 18 are input and output pins of the built-in inverting amplifier, respectively, which can be configured for use as an on-chip oscillator. A 6MHz crystal is used to generate the clock frequency for the microcontroller.

AT89S8252 has four bidirectional 8-bit ports, of which only three ports (0 through 2) have been used in this circuit. Port 0 is an 8-bit open-drain bidirectional I/O port. As an output port, each pin can sink eight TTL inputs. Port 0 can also be configured as the multiplexed low-order address/data bus during accesses to the external program and data memory. External pullups are required during data outputs.

Port 0 is used to drive the segments of all the 7-segment common-cathode displays. Pin 1 of the RNW1 resistor network is connected to Vcc and pins 2 through 9 are connected to port-0 pins 39 down through 32 of IC2 as external pull-ups. Pins 39 down through 32 of port 0 are also connected to the input pins of octal

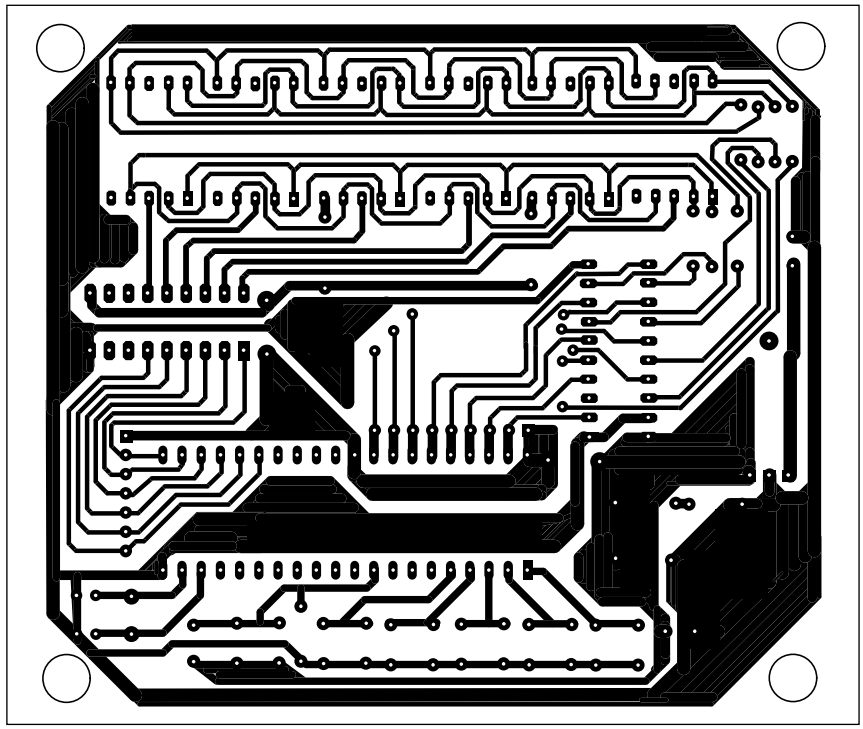


Fig. 4: Actual-size, single-side PCB for the real-time clock using AT89S8252 microcontroller

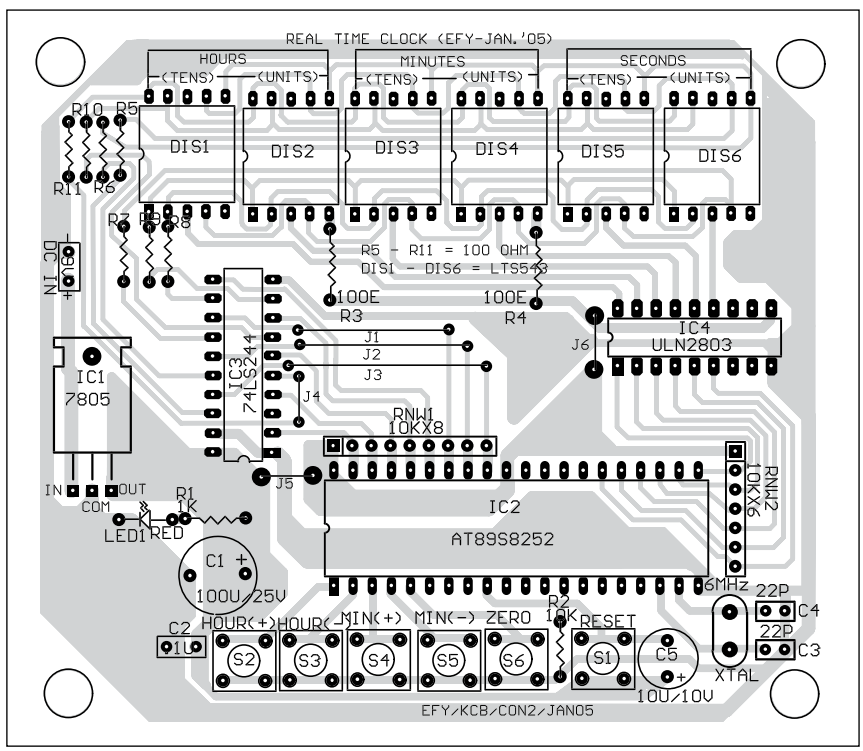


Fig. 5: Component layout for the PCB

line driver IC 74LS244 (IC3).

Segments 'a' through 'g' of 7-segment displays DIS1 through DIS6 are joined and connected to the output pins of IC3 via resistors R5 through R11, respectively. IC3 acts as an octal buffer between the microcontroller and the displays to increase the current level. Resistors R5 through R11 limit the current through the 7-segment displays. Each display comprises seven light emitting diodes (LEDs) with their common cathodes connected together, hence termed as the common-cathode, 7-segment display.

Port 2 acts as the multiplexer to select a particular 7-segment display using octal Darlington transistor array ULN2803 (IC4). Pins 21 through 26 of port 2 are pulled up by the RNW2 resistor network and also connected to pins 1 through 6 of IC4. IC4 outputs a low signal to light up the segments of the 7-segment display selected by the port-2 data.

Ports 0 and 2 provide the segment data and enable signal simultaneously for displaying a particular number on the 7-segment display. Decimal-point pin 5 of displays DIS2 and DIS4 is enabled by Vcc through resistors R3 and R4, respectively, to differentiate the hour, minute and second.

Port 1 detects pressing of the switches to increment/decrement hours and minutes and reset the display to '00:00:00' by pulling the port pins to ground. The software detects pressing of the switches and sets the time accordingly. Pull-up resistors on port 1 have been avoided since the port already has internal pull-ups.

An actual-size, single-side PCB for the real-time clock is shown in Fig. 4 and its component layout in Fig. 5.

Software

The software for the real-time clock is written in Bascom51 version. Those who have knowledge of Basic, Basic-A, GW-Basic or QBasic language (used to run on the good old 286 and 386 PCs with DOS 2.x to 6.2) can understand the program easily. The demo version of Bascom-8051 is available on Website 'www.mcselec.com/ download_8051.htm.'

Fig. 6 shows the flow-chart of the program. Step-wise explanation of how the program works is given below:

1. Define the port pins and where

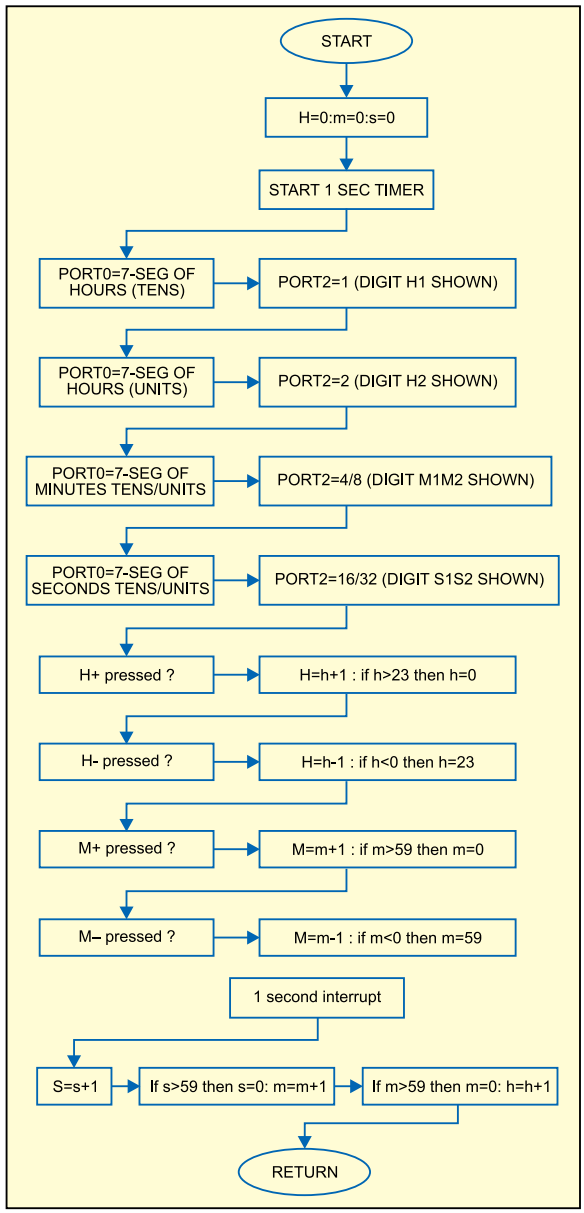


Fig. 6: Flow-chart of the program

these are connected.

2. Include the header file for the microcontroller

3. Define the crystal speed.

4. Declare the variables as bits, bytes and words.

5. Initialise all ports to 0, except port 1, which is turned high to act as an input port.

6. Run a diagnostic subroutine to test the segments of all the digits.

7. Configure the internal timer as an interrupt generator to get a one-second-activity source.

8. Initialise hour, minute and second variables to zero.

9. Get into a perpetual Do loop to display the time in 'HH:MM:SS' format. (Since there are no BCDto-7-segment converter ICs and no latch ICs, it is up to the software to show the clock display without being interrupted.)

10. Set the input switches to activate the respective subroutines using the built-in command of Bascom's key debounce statement.

11. Check when second, minute and hour variables exceed their limits and increment them accordingly.

12. Activate the digits one by one through port 2 and show the corresponding number on the display using port 0.

13. Declare subroutines for detection of the switches pressed to adjust hours and minutes.

14. Declare the main display subroutine. Since we have not used a 7segment converter IC, a quick table check using read and data concept in Basic is performed to get the correct byte value for the digit to be displayed.

15. Declare the internal timer interrupt subroutine. This subroutine is called 2000 times in a second using a 6MHz crystal, and to generate an accurate one-second variable, we set the flag only once every 2000 times. This variable is used to detect the seconds change and increment the time in the main Do loop routine. The accuracy of the clock depends on the timer subroutine.

Other possible uses

The circuit and the software can be improved to convert this real-time clock into an alarm clock. With port 3 activated, it can be used as a multichannel industrial timer.

Download source code: http://www.efymag.com/admin/issuepdf/Real%20Time%20Clock.zip

EFYCLK11.BAS

```
'-----------------------------------------------------------
' EFYclk.bas  18-10-04
' REAL TIME CLOCK DISPLAY ON six 7-SEG DISPLAYS
' BY k.s.sankar www.mostek.biz  for EFY
' written using BASCOM-51 from MSC electronics Netherlands
'-----------------------------------------------------------


'Connect common cathode LED displays as following :
' port-0  (red)
'a = P0.0
'b = P0.1
'c = P0.2
'd = P0.3
'e = P0.4
'f = P0.5
'g = P0.6
'dp= p0.7
'


'88 88 88
'hh mm ss  port-2 (green) p2.0 /1  : 2/3 : 4/5
'12 34 56  digit number

' yellow port-1 set switches
'P1.0=H+ P1.1=H-
'P1.2=M+ P1.3=M-
'P1.4= 00 00 00 ( reset to 00 00 00)

'-----------------------------------------------------------
$regfile = "89s8252.dat"
$crystal = 6000000
'6 mhz crstal

Dim Once_a_sec As Bit
Dim Clock_word As Word
Dim Hours As Byte , Minutes As Byte , Seconds As Byte
Dim Red As Byte , Green As Byte
Dim Count As Byte , X As Byte , Segment As Byte
Dim Number As Byte , Digit_select As Byte
Dim Del As Byte , Diagdelay As Byte
Dim Large As Word

Del = 1
' delay variable in milliseconds
' all ports 0
P0 = 0
'red
P1 = 255
'yellow   all high for sw inputs
P2 = 0
'green
P3 = 0
'blue not used

Config Debounce = 30
```

```
' key debounce time in milli seconds
 Config Timer0 = Timer , Gate = Internal , Mode
= 2
 'Timer0           use timer 0
 'Gate = Internal    no external interrupt
 'Mode = 2          8 bit auto reload


 Gosub Diag
 ' diagnostic routine

 ' set t0 internal interrupt
 On Timer0 Timer_0_int
 Load Timer0 , 250
 Priority Set Timer0
 Enable Interrupts
 Enable Timer0
 Start Timer0


 Hours = 0
 Minutes = 0
 Seconds = 0
 Clock_word = 0



 Do
 ' yellow port-1 key inputs for setting
 Debounce P1.0 , 0 , Hup , Sub
 Debounce P1.1 , 0 , Hdown , Sub
 Debounce P1.2 , 0 , Mup , Sub
 Debounce P1.3 , 0 , Mdown , Sub
 Debounce P1.4 , 0 , Zero , Sub


 If Once_a_sec = 1 Then
 ' once_a_sec=calculation every second
 Once_a_sec = 0
 'update hh mm ss

 inc seconds
 If Seconds = 60 Then
 Seconds = 0
 inc minutes
    If Minutes = 60 Then
    Minutes = 0
    inc hours
       If Hours = 24 Then
       Hours = 0
       End If
    End If
 End If
 End If

 ' display time constantly
 ' hours

 Number = Hours / 10
 P2 = 1
 Gosub Disp
 Waitms Del
 P0 = 0
 '-------
 Number = Hours Mod 10
 P2 = 2
 Gosub Disp
 Waitms Del
 P0 = 0
 '-------

 'minutes
 Number = Minutes / 10
 P2 = 4
 Gosub Disp
 Waitms Del
 P0 = 0

 '-------
 Number = Minutes Mod 10
 P2 = 8
 Gosub Disp
 Waitms Del
 P0 = 0
 '-------

 'SECONDS

 Number = Seconds / 10

 P2 = 16
 Gosub Disp
 Waitms Del
 P0 = 0
 '-------

 Number = Seconds Mod 10
 P2 = 32
 Gosub Disp
 Waitms Del
 P0 = 0
 '-------
 Loop
 ' - - - - - - - - - - - - - - - - - -

 ' set keys below

 Hup:
 Incr Hours
 If Hours >= 24 Then
  Hours = 0

 End If
  Return

 Hdown:
 Decr Hours
 If Hours = 255 Then
  Hours = 23
 End If
  Return

 Mup:
 Incr Minutes
 If Minutes >= 60 Then
 Minutes = 0

 End If
 Return

 Mdown:
 Decr Minutes
 If Minutes = 255 Then
 Minutes = 59

 End If
 Return

 Zero:

 Hours = 0 : Minutes = 0 : Seconds = 0
 Return

 ' - - - - - - - - - - - - - - - - - - - - - -
 Diag:
 'diagnostics

 'if zero button pressed then goto zero label and
return
 Diagdelay = 121

 For Seconds = 1 To 5

 Diagdelay = Diagdelay - 20

 P2 = 1
 For Green = 0 To 5
 P0 = 1
 For Red = 0 To 7
 Debounce P1.4 , 0 , Zero
  Waitms Diagdelay
 Rotate P0 , Left

 Next Red

 Rotate P2 , Left
 Next Green

 Next Seconds

 ' next diag show 000000 to 999999 on all digits
 ' - - - - - - - - - - - - - - - - - - - - - - - -

 For Number = 0 To 9
 P2 = 1

 For Large = 1 To 50

 ' approx 1 second time loop with 200 in large
 For Green = 0 To 5
 Debounce P1.4 , 0 , Zero
  Gosub Disp
 Waitms Del
 Rotate P2 , Left
 Next Green
 Next Large

 Next Number

 Return


 'Displaying routine

 Disp:

 Restore Tabela


 ' scan 7-seg table to get byte for the digit to
display
 For X = 0 To 9
 Read Segment
    If X = Number Then
   'if X = value to display
      P0 = Segment
     'then set this value to Port0-red
      Exit For
   'and exit FOR loop
    End If
 Next

 Return



 ' int subroutine -----------------
 Timer_0_int:
 Incr Clock_word

 If Clock_word > 2000 Then
 Clock_word = 0
 Once_a_sec = 1
 End If
 Return


 '---- data for 7-seg LED display ------
 Tabela:
 Data 63 , 6 , 91 , 79 , 102 , 109 , 125 , 7 , 127 , 111
 ' end of program
 ' -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
```

STANDALONE SCROLLING DISPLAY USING AT90S8515 AVR

■ **SHUBHIKA TANEJA, DEEPA CHAWLA**

Microcontrollers are being extensively used in many industrial and household applications. Here, we've used an AVR microcontroller (AT90S8515) from Atmel Corp. for controlling four 5x7 dot-matrix displays. The microcontroller is based on true reduced instruction set computer (RISC) architecture. Any message entered by the user through the keyboard of a PC into regulated 5V DC.

4. The string of characters entered through the keyboard is stored in the EEPROM. The stored message can be displayed on the dot-matrix display just by clicking the scud button on the terminal program while it is connected to the PC.

5. Any message entered from the PC's keyboard gets stored in the EEPROM of the AVR and can be scrolled at any time without the use of a PC, i.e. you just need to switch on the embedded system.

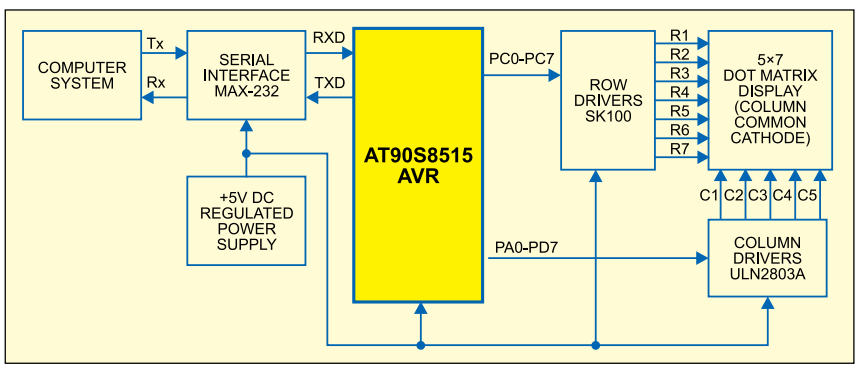


Fig. 1: Block diagram of standalone scrolling display using AT90S8515 AVR

scrolls elegantly through the displays even after disconnection of the circuit from the PC.

This display can be used in public places such as railway stations and restaurants to convey messages to the public. The microcontroller is interfaced to the PC keyboard through its serial port. The embedded system software is written in 'C.'

The circuit has the following features:

1. It accepts any message entered through the keyboard of the PC for display.

2. User interface is provided through the PC's RS-232 serial port (COM port).

3. The circuit derives power from 230V AC mains, which is converted

6. RXD and TXD pins of the microcontroller are used to communicate with the PC through MAX-232 IC and TX and RX pins of COM port. All the four ports (ports A, B, C and D) of the AVR are programmed as output ports.

Fig. 1 shows the block diagram of the AT90S8515-based standalone scrolling display system. It consists of an AVR microcontroller, row display drivers, column display drivers, four 5x7 dot-matrix displays and power supply section. The AVR compiler, in-system programmer (ISP) and terminal program are installed in the computer. The display control program, written in 'C' using AVR C compiler, is loaded into the microcontroller by using parallelport pins of the PC.

PARTS LIST

Semiconductors:

IC1	-	AT90S8515 AVR microcontroller
IC2-IC6	-	ULN2803A Darlington array LED driver
IC7	-	MAX232 RS-232 serial interface
T1-T7	-	SK100B pnp transistor

Resistors (all ¼-watt, ±5% carbon):

R1	-	220-ohm
R2-R8	-	1-kilo-ohm
R9-R15	-	220-ohm
R16	-	620-ohm

Capacitors:

C1	-	100µF, 16V electrolytic capacitor
C2, C3	-	22pF ceramic capacitor
C4	-	0.1µF ceramic capacitor
C5-C9	-	1µF, 16V electrolytic capacitor

Miscellaneous:

XTAL	-	8MHz crystal
DIS1-DIS4	-	5×7 dot-matrix (column common cathode) display
LED1	-	Red power indicator
S1	-	SPST on/off switch
S2, S3	-	Tactile switch

Circuit description

Fig. 2 shows the circuit of AVR AT90S8515-based scrolling display system.

AT90S8515 AVR microcontroller. AT90S8515 is a 40-pin, 8-bit microcontroller from Atmel. It has 512 bytes of SRAM, 512 bytes of EEPROM and 8kB Flash with 32 programmable input/output (I/O) lines. AVR microcontrollers are in-system programmable through RS-232C serial port (COM port) of the PC. The programmable Flash memory and EEPROM of the AVR can be programmed using a simple software and just four wires from parallel port of the PC to your target board containing AVR. Easy in-circuit programmability combined with Flash memory makes it easy to update the code during development. Since we require a minimum of 27 output pins (20 columns and 7 rows),

AT90S8515 suits this application as it has 32 programmable I/O lines. Pin details of this AVR are shown in Fig. 4. The AVR marked on the IC with 8PI or 8PC indicates the value of the crystal to be used, which in this case is 8 MHz. The baud rate in the communication software should be selected as per the following relationship:

$$\text{Baud rate} = \frac{f_{CLK}}{16(V_{BRR}+1)}$$

where f_{CLK} is crystal frequency and V_{BRR} is the value of contents of the UART baud rate register.

Serial interface. The serial interface comprises 9-pin D-type female connector, IC MAX-232, five 1µF electrolytic capacitors and 3-core cable as shown in Fig. 3.

Display drivers. Seven SK100B transistors along with 220-ohm (output current limitor) and 1k-ohm resistors (base current limiter) are used for controlling the rows of LED array, and five ULN2803 ICs (IC2 through IC6) are used for controlling the columns of dot-matrix displays.

Dot-matrix displays. Four 5x7 dot-matrix LEDs (with common cathodes as the columns) such as KLP2057 from Kwality Electronics (India) are used for the display. The displays need seven row drivers and 20 column drivers. These displays are identical, with cathodes shorted along the column and anodes shorted along the row (refer Fig. 5).

Since the human eye cannot perceive changes carried out at frequencies greater than 20 Hz, each column must be refreshed at a minimum rate of 20 Hz. Here, we have set the refresh rate (the rate at which the display from one column to the next) at about 400 Hz. In case only one LED glows in a particular col-

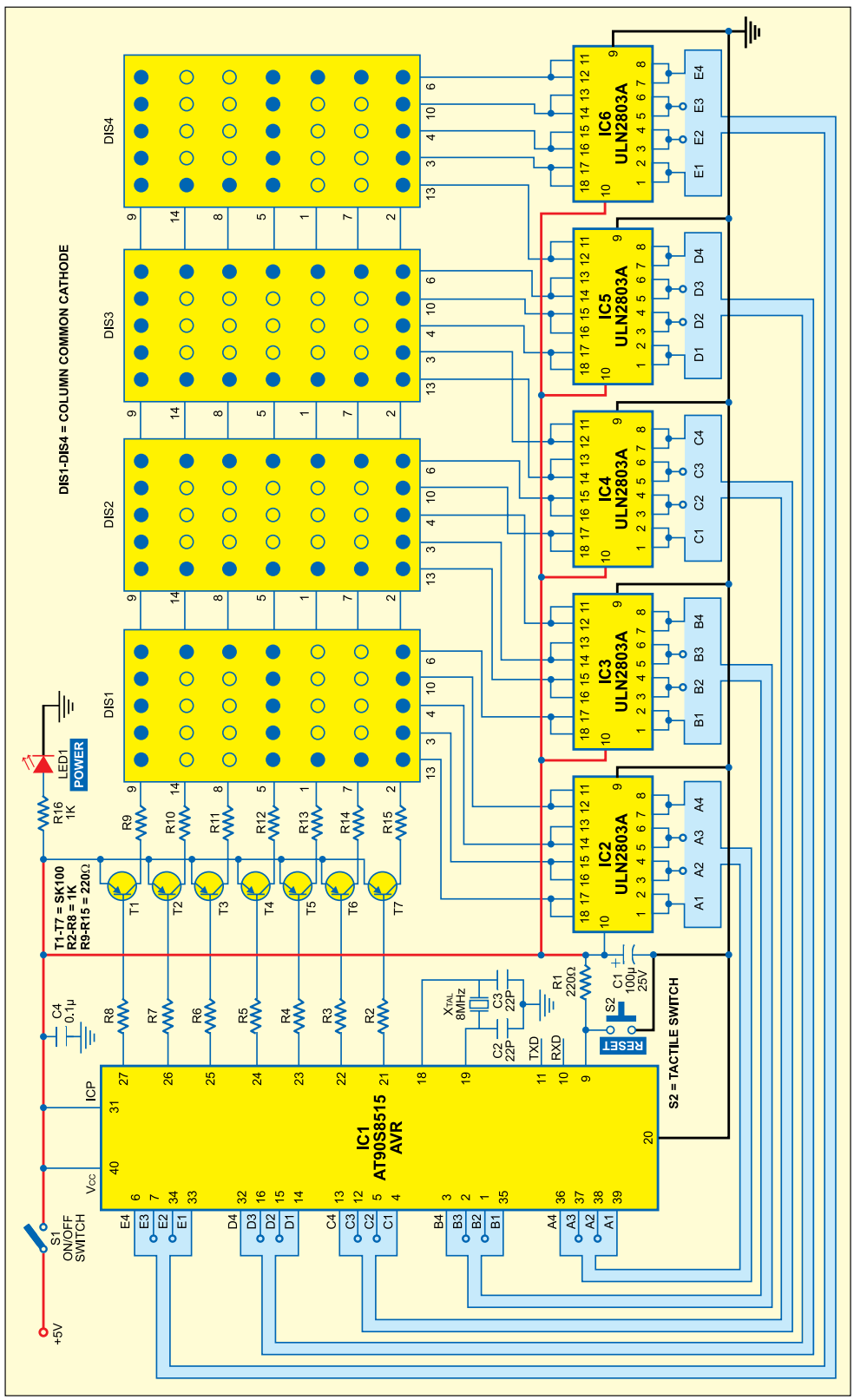


Fig. 2: Circuit for standalone scrolling display using AT90S8515

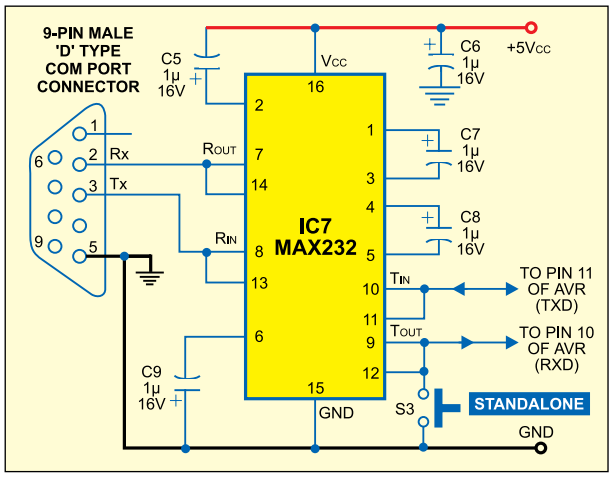


Fig. 3: RS-232 interface circuit

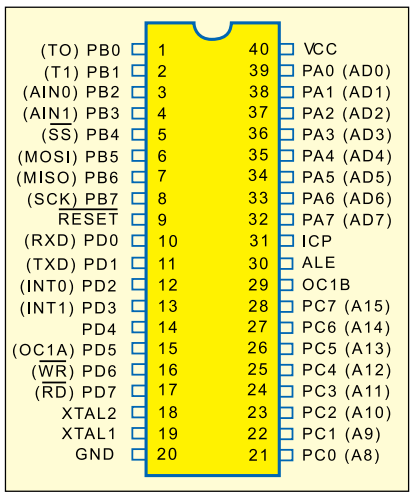


Fig. 4: Pin details of AT90

umn, that particular data line will have to handle 20mA current.

Since there are 20 LEDs in a row, 400mA current could flow through a particular column at a particular instant. The circuit has to be designed keeping the value of this peak current in mind. Since 400mA current cannot be sourced by the port pin of AVR (maximum current sourced or sinked by the AVR's I/O ports is 20 mA), the display cannot be directly connected to the AVR port. We thus use SK100B pnp transistors along with 220-ohm current-limiting resistors.

For obvious reason, we've used five ULN2803 ICs to increase the current sinking capacity. These ICs are connected to the columns of the displays. Each IC has eight Darlington pairs. Pairs of input and output pins of ULN 2803 are connected in parallel to increase the current sinking capability. The transistors are turned on by the TTL voltages applied by the input/output ports of the AVR to their bases through 1-kilo-ohm resistors.

Power source. A 5V DC regulated power supply is used in this circuit, which has to be supplied externally.

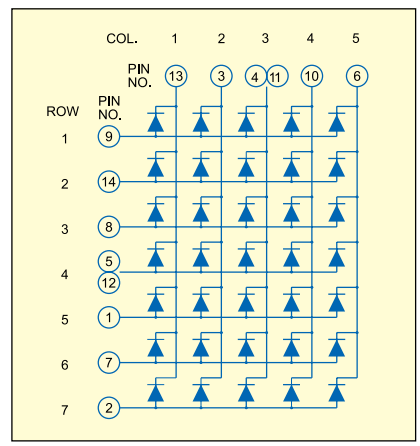


Fig. 5: Column common cathode

Connecting the AVR to the PC's serial port

The microcontroller needs to communicate with the PC's RS-232 port to scroll the string entered through the keyboard of the PC. AT90S8515 has a built-in serial port. The processor takes care of serialising and shifting out of the data on the output pin and assembling of the incoming data into a byte. Since the RS-232 signals are bipolar in nature, they cannot be fed directly to the controller. We have used a very popular RS-232 line driver and receiver MAX232 (IC7) for converting the PC's RS-232 compatible signals into TTL levels for AVR and vice versa. T_{IN} (TTLinput) and TOUT (TTL output) pins of MAX232 are connected to the transmitter (TXD) and receiver (RXD) pins of the AVR, respectively.

The transmitter (T_X) and receiver (R_X) pins of the PC's Com port are connected to the R_{IN} (RS-232 input) and R_{OUT} (RS-232 output) pins of MAX232, respectively. A 9-pin D-type male connector is attached to the PCB board, whose pins 2, 3 and 5 are soldered to R_{OUT}, R_{IN} and ground of IC7, respectively.

Two 9-pin D-type female connectors are required for connection between the PCB board and the PC's serial port. The communication between the PC and the circuit board for display is done through a terminal program software such as 'Terminal v1.9b,' which can be downloaded for free from the Website 'bray.velenje.cx/avr/terminal.' Using this software, up to 130 characters can be typed in at a time for transmission to the display circuit for the scrolling display.

Programming the AVR

Getting started with the AVR requires nothing more than the free assembler/compiler, a simple programmer such as the one by Jerry Meng (available on 'www.qsl.net/ba1fb/') and a target board. The target board can be as simple as a few parts since the AVR is highly integrated. Since it is easy to reprogram the flash memory, you can develop code and test without the need for an expensive in-circuit emulator. This is done by a built-in interface in the AVR chip, which enables you to write and read the contents of the programmed Flash and the built-in-EEPROM. This interface works serially and needs mainly three signal lines from the AVR to PC's printer port for programming:

1. SCK: A clock signal that shifts the bits to be written to the memory into an internal shift register, and that shifts out the bits to be read from another internal shift register.

2. MOSI: The data signal that sends the bits to be written to the AVR.

3. MISO: The data signal that receives the bits read from the AVR.

The connections for program-

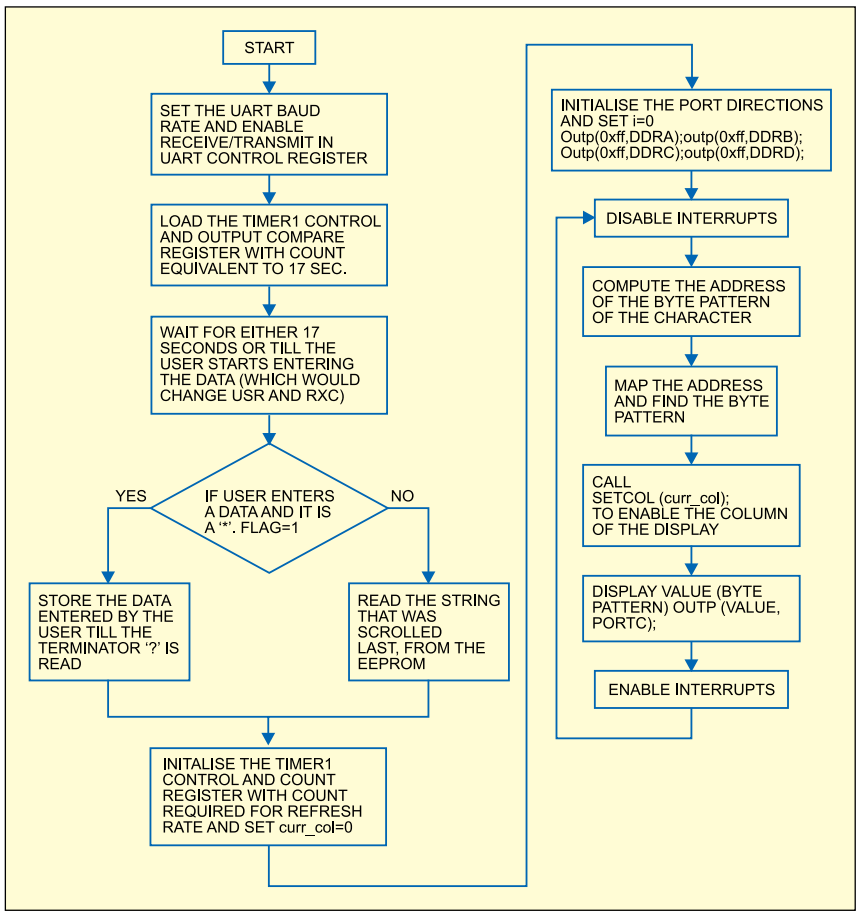


Fig. 6: Flow-chart of the program

ming are simple but there are various standards adopted by the industry. In this project, the ISP10 standard is used on the STK200 programmer board (from KANDA Systems) for programming. The STK200 board consists of the zif socket for the AVR and a 10-pin header box. The dongle is used to connect the port of the PC to the 10- in header connector on the STK200 board. Along with this STK200 board, you need a compiler/assembler such as AVREdit 3.5 and Atmel AVR ISP 2.65 software to be installed into your system for programming the AVR chip. The required software tools can be downloaded from the Website 'www.avrfreaks.net.' The STK200 dongle is available on the Website 'elm-chan.org/works/avrx/report_e.html.'

EFY note. A simple dongle circuit used in EFY Lab for programming the AVR will be published in the next issue.

Software Program

The software has the following features:

1. Initially waits for 17 seconds for the user to enter the string.

2. Receives data from UART sent through the serial port of the PC connected to MAX232 by a 9-pin connector.

3. Stores the string entered by the user. Else, retrieves the previously stored string from the EEPROM.

4. Stores the byte-patterns of characters 'A' through 'Z,' 'a' through 'z' and '0' through '9' in the 16-bit programmable flash memory.

5. Initialises the interrupts for refresh rate and scroll rate.

6. Maps the byte pattern of each character from the program memory as a function of the scroll parameter and then sends the values to the ports.

The flow-chart of the program is shown in Fig. 6.

The 8-bit timer/counter of the AVR is used to implement refreshing of the display. As the minimum refresh rate for flicker-free view is 20 Hz, we have chosen prescale as Clk/64, thus giving us the refresh rate in kilohertz, where 'Clk' is the oscillator clock frequency of the crystal used.

Wait interrupt has been implemented by the 16-bit timer/counter with clk/1024 as the pre-scaler and output-compare register (OCR). This gives us an initial wait period of 17 seconds.

Sub-modules of the code. During the 17-second waiting period, the program waits for the user to send data through the UART. Hence, the program waits in while loop 'While (! (USR&(1<<RXC))&& (q! =0));' and keeps checking the RXC bit (UART Receiver Complete) of the UART status register (USR) until either the user enters a data byte (RXC bit will be set) or the 16-bit timer/counter output compare interrupt is generated and the while loop terminates. The 16-bit timer/ counter is initialised as 'TCCR1B=5; OCR1AH=10;' which defines the prescaler of 'clk/64.'

To receive data from UART sent from the serial port of the PC, first the UART baud rate and UART control register (UCR) are set to enable the receiver and the transmitter as 'UBRR=25; UCR=(1<<RXEN) | (1<<TXEN);' where UBRR is the UART baud rate register.

If the user sends a new string, it will first be received from the UART data register (UDR) and stored in SRAM, then it will be written into the EEPROM, which, in turn, overwrites the previously stored string. The following lines enable storing of the string in SRAM:

```
While ((count1<100) && (str1 [k]! = 63))
{
if(USR & (1<<RXC))
flag=1;
```

If the string entered is in the correct format, the flag is set to '1.' Else, the flag remains '0' and the previously

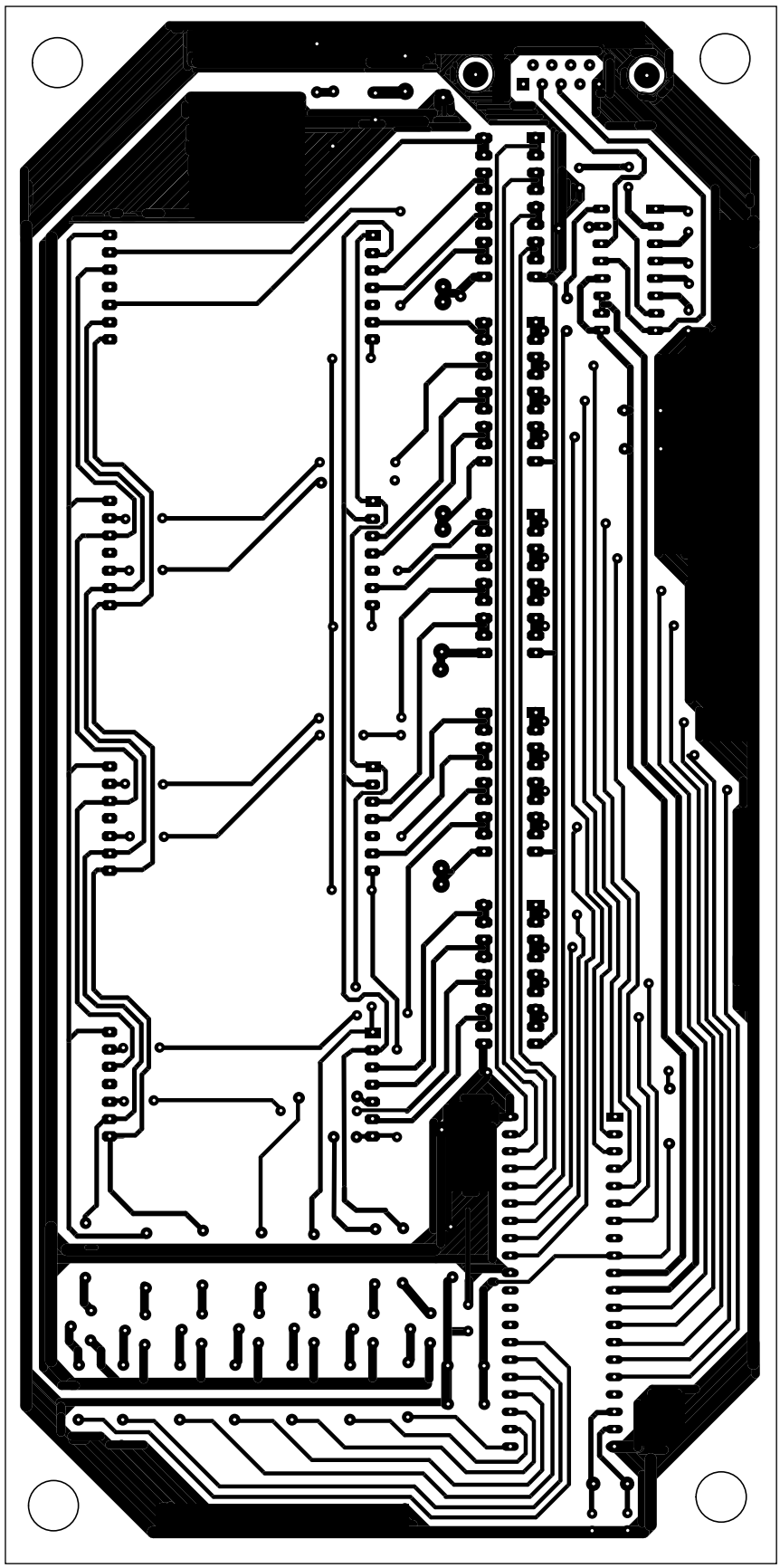


Fig. 7: Combined actual-size, single-side PCB layout for Figs 2 and 3

stored string will be displayed. To store the string in EEPROM, the string is written character-by-character in the EEPROM starting from location '0x0001.'

If the previously stored string is to be scrolled, the same routine is executed, except that data is only 'read from' instead of 'written to' the EEPROM. The following program lines perform these actions:

```
address = 0x0001;
EEREAD( address, str+x);
EEWRITE(address,str1[x]);
//Store the string
in EEPROM
```

To store the byte patterns of characters 'a' and 'b' in the 16-bit programmable flash memory, an extract from the program is reproduced below:

```
typedef unsigned char u08;
u08 __attribute__ ((progmem)) leds[]={
0xe0, 0xd7, 0xb7, 0xd7, 0xe0, //a
0x80, 0xb6, 0xb6, 0xb6, 0xc9, //b
```

The program lines "t = str[i]; addr = (t-'A')*5;" are used to retrieve the starting address of the byte-pattern of any character, where 'A' is the base address.

Initialisation of interrupts for refresh rate and scroll rate is as follows:

```
TCNT0 = 200;
TIMSK |= 1<<TOIE0 ;
TCCR0=3;//Timer/Counter Control Register
```

An 8-bit timer/counter (TCNT0) is used in the program, whose value can be changed to increase the intensity of the display. The scroll rate has been taken as a multiple of refresh rate. This multiple is taken as '2000.' When the string to be scrolled is known, first the input/output ports are set by the following instructions:

```
outp(0xff,DDRA);
outp(0xff,DDRB);
outp(0xff,DDRC);
outp(0xff,DDRD);
```

To map the byte pattern of each character of the string from the program memory as a function of the scroll parameter (named as offset here) and then send the values to the ports, the following section of the program is a critical section. As we don't want the interrupts to occur

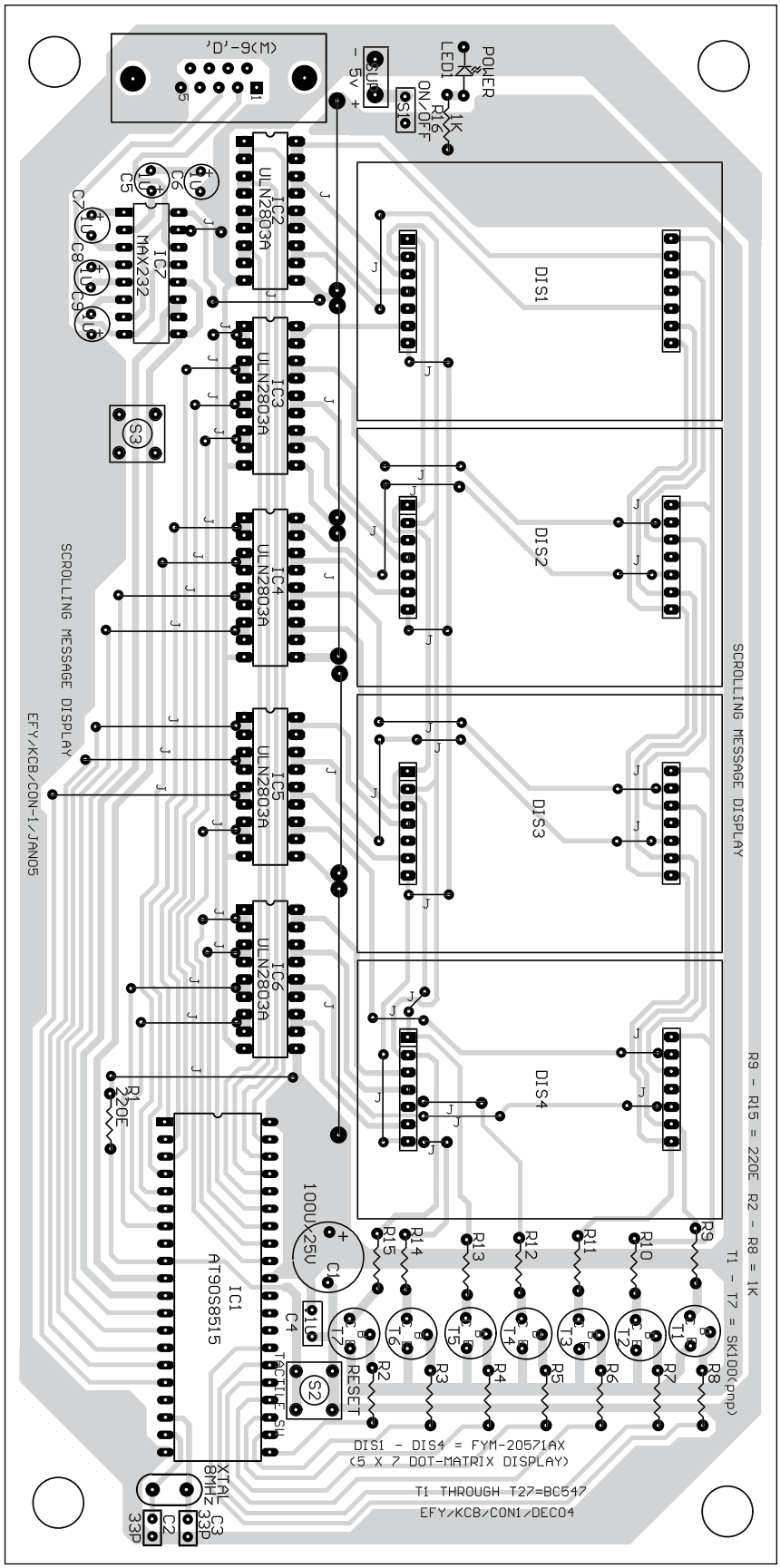


Fig. 8: Component layout for the PCB

during their execution, we use cli () and sei ():

```
"cli();//disable interrupt in
critical section
if( j == 2000
t = str[i];
if(t>=65&& t<=91);// Characters
between A
and Z
addr=(t-'A')*5;//i is being
incremented in interrupt
else if(t>=97&& t<='122);
// Characters between a and z
else if(t>=48&& t<=57);
// Characters between 0 and 9
curr_col_temp=(curr_col<5)?
curr_col:curr_col%5;
m = offset + curr_col_temp;
if(m>=5) m=m-5;
addr = addr + m;
value = PRG_RDB(&leds[addr]);
outp( value, PORTC);
setcol(curr_col);
sei();//enable interrupt"
```

The function 'setcol(int col)' is called to send appropriate values to the ports to drive the column LEDs.

Construction

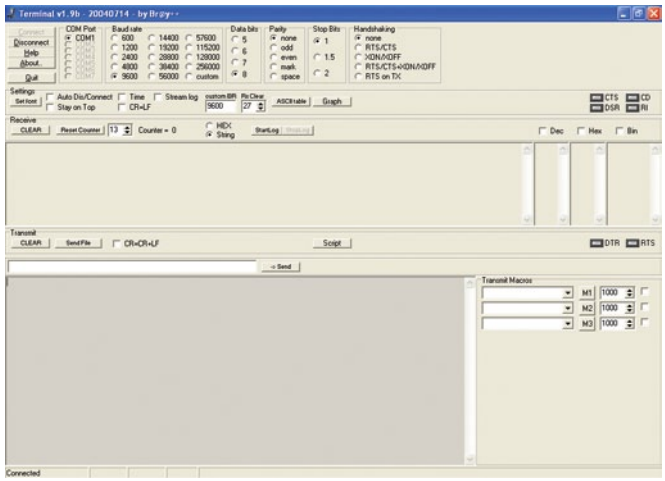
The circuit can be constructed on any general-purpose PCB. A 3-core serial cable is used for communication with the PC's keyboard. The 9-pin male connector is soldered on the PCB to interface with the cable. 5V DC regulated power supply is required for the circuit as well as programming the circuit, which can be constructed on a separate PCB.

An actual-size, solder-side combined PCB layout for the display and interface circuits (Figs 2 and 3) is shown in Fig. 7 and its component layout in Fig. 8.

Testing procedure

After having mounted all the components, except AVR on the PCB, you have to perform the initial test (optional) to check the connections of the 5x7 dot-matrix displays. The 'check.c' program given below can be programmed into the AVR for this checking. The various steps involved are:

1. Download the 'AvrEdit3.5' software and Atmel AVR ISP and



Screenshot of terminal program

install in your system. The 'AvrEdit' and 'Avrtools' folders automatically get created in the respective software.

2. Create another folder, say, 'Discheck,' under the 'AvrEdit' folder and copy the 'check.c' file into the 'Discheck' folder.

3. Run 'AvrEdit' from the desktop, open the 'check.c' program and click 'Run' in the menu bar for compilation. After compilation, the 'Check.Rom' file is automatically generated under the 'Discheck' folder.

4. Now, connect the STK200 (dongle) to the parallel port of the PC and insert the AVR into the zip socket of the STK200 board.

5. Run the Atmel AVR ISP from the desktop, select 'New Project' to load the 'Check.Rom' file from the 'AvrEdit' folder.

6. From 'Program' menu bar of the ISP, select 'Program Device' to program the AVR.

Remove the programmed AVR from the STK200 board. The AVR, when inserted into the populated PCB, will light up all the LEDs in the display devices if the circuit connections are correct.

Now, to program the main program 'ScrollD.c' into the AVR chip, create a folder, say, 'Scroll' under the 'AvrEdit' folder. Copy 'ScrollD.c' into the 'Scroll' folder, run 'AvrEdit' and follow steps 2 through 6 as mentioned above. After programming the AVR, remove it from the STK200 board and insert into the main circuit.

7. Connect the 9-pin D-type female connector from the main circuit to the COM port of your PC.

8. Download the 'Terminalv1.9b' communication software and install it in your PC. An application file icon named 'Terminal' will be created on the desktop.

9. Switch on the power to the circuit and run 'Terminal' from the desktop. Choose the baud rate of this application as 9600 and parity bit as none (refer to the screenshot).

10. Click 'Connect' button and type '*New Year 2005?' in the transmit box. Note that the message should always be enclosed between '*' and '?' before transmission.

11. Click 'Send' button to transmit the characters for display on the dot-matrix displays.

12. To enter new characters for display, click 'Disconnect' button, press reset switch S2 and type new message in the transmit/edit box. Click 'Connect' button followed by 'Send' button.

13. If a particular string is to be scrolled again and again, disconnect the circuit from the PC. Whenever the circuit is switched on, the display system will wait for 17 seconds and the previous string stored in the EEPROM will scroll on the displays without the need of serial cable, Terminal program and PC. This feature makes this embedded system a standalone system.

EFY note. 1. It was observed that a momentary low pulse is required to be provided at pin 10 (RXD) of the AVR through switch S3 to initiate the display without PC.

Download source code: http://www.efymag.com/admin/issuepdf/SCROLL%20DISPLAY.zip

SCROLLD.C

```
 // Code for AVR PROJECT of  Scrolling Display
 #include <eeprom.h>
 // Offset b/w 0 and 4
 #include <io.h>
 #include <progmem.h>
 #include<interrupt.h>
 #include<sig-avr.h>
 #include<ina90.h>
 //offset is the beginnig pointer
 // global varables
 int curr_col,i=0,j=0,offset=0,temp=0,q=1;
 unsigned char str[100], str1[100];
 int count=0, address,x,x1 ;
 void EEWRITE( int address,char value);
 void EEREAD( int address,char *val);
 void setcol(int col);
 SIGNAL(SIG_OUTPUT_COMPARE1A)
 {q=0;}
 SIGNAL(SIG_OVERFLOW0)
 {
 int k;
  setcol(-1);
 curr_col++;
 j++;
 if ( curr_col==20)
 {
 curr_col=0;
 if( offset ==0)
  {
   if( i>=3) i=i-3;
  else i=i+count-3;
  //offset++;
  }
 else
 if(offset==4 && j== 2000)
 {i=temp+1;
 temp=i;
 }
  else
  { i--;
  k = 20 - offset;
  while( k>=5){ k=k-5; i--; if(i<0) i=i+count; }
  }
 }

 else
 {
  int x = (curr_col<5)? curr_col: curr_col%5 ;
  if( (x!=0&&(x+offset)%5==0) ||(offset==0 && (
curr_col==5 || curr_col==10 ||curr_col==15 ||
curr_col==20)))
  i++;//char shift
  if(i==count ) i=0;
  }
  if(i==count)//added now
  i=0;
  TCNT0 = 230;
  }

  typedef unsigned char  u08;
  u08 __attribute__ ((progmem)) leds[]={
  0xe0, 0xd7, 0xb7, 0xd7, 0xe0,
  0x80, 0xb6, 0xb6, 0xb6, 0xc9,  //b
  0xc1, 0xbe ,0xbe, 0xbe, 0xdd,  //c
  0x80, 0xbe ,0xbe, 0xbe, 0xc1,  //d
  0x80, 0xb6, 0xb6, 0xb6, 0xbe, //e
  0x80, 0xb7, 0xb7, 0xb7, 0xbf,  //f
  0xc1, 0xbe, 0xba, 0xba, 0xd9,  //g
  0x80, 0xf7, 0xf7, 0xf7, 0x80,  //h
  0xbe, 0xbe, 0x80, 0xbe, 0xbe,  //i
```

```
0xb9, 0xbe, 0xbf, 0x81, 0xbf, //j
0x80, 0xf7, 0xeb, 0xdd, 0xbe, //k
0x00, 0xfe, 0xfe, 0xfe, 0xfe, //l
0x80, 0xdf, 0xe7, 0xdf, 0x80, //m
0x80, 0xef, 0xf7, 0xfb, 0x80, //n
0xc1, 0xbe, 0xbe, 0xbe, 0xc1, //o
0x80, 0xb7, 0xb7, 0xb7, 0xcf, //p
0xc1, 0xbe, 0xba, 0xbc, 0xc0, //q
0x80, 0xb7, 0xb3, 0xb5, 0xce, //r
0xce, 0xb6, 0xb6, 0xb6, 0xd9, //s
0xbf, 0xbf, 0x80, 0xbf, 0xbf, //t
0x81, 0xfe, 0xfe, 0xfe, 0x81, //u
0x83, 0xfd, 0xfe, 0xfd, 0x83, //v
0x00, 0xfd, 0xfb, 0xfd, 0x00, //w
0x1c, 0x6b, 0x77, 0x6b, 0x1c, //x
0xbf, 0xdf, 0xe0, 0xdf, 0xbf, //y
0xbc, 0xba, 0xb6, 0xae, 0x9e,
0xf0,0xee,0xee,0xf1,0xfe,//a
0x00,0xf6,0xf6,0xf6,0xf6,//b
0xf1,0xee,0xee,0xee,0xff,//c
0xf0,0xf6,0xf6,0xf6,0x00,//d
0xe1,0xd6,0xd6,0xd6,0xe6,//e
0xf7,0x00,0x37,0x37,0xdf,//
0xcf,0x37,0x31,0x36,0xc0,//g
0xff,0x00,0xf7,0xf7,0xf8,//h
0xff,0xff,0xd0,0xff,0xff,//i
0xfd,0xfa,0x20,0xff,0xff,//j
0xff,0x00,0xfb,0xf5,0xee,//k
0xfb,0x00,0x2e,0xdd,0xff,//l
0xf0,0xef,0xf0,0xef,0xf0,//m
0x6f ,0x70 ,0x6f ,0x6f ,0x70,//n
0xf9,0xf6,0xf6,0xf6,0xf9,//o
0x00, 0x6d, 0x6b, 0x77, 0x7f,//p
0x4f, 0x37, 0x37 ,0x00 ,0x7b,//q
0xf7 ,0xf4 ,0xfa ,0xf4 ,0xf7,//r
0xf5 ,0xea ,0xea ,0xf4 ,0xff,//s
0xf7 ,0xf7, 0x00, 0xf6, 0xf5,//t
0xf1, 0xfe ,0xfe, 0xfe, 0xf1,//u
0xef ,0xf1 ,0xfe ,0xf1 ,0xef,//v
0xe1, 0xfe, 0xf9 ,0xfe, 0xe1,//w
0xee, 0xed ,0xf3 ,0xed, 0xee,//x
0xcf ,0xf7, 0xf5, 0xf2, 0xc0,//y
0xee ,0xec ,0xea ,0xe6 ,0xee,//z
0x00,0x3e,0x3e,0x3e,0x00,//0
0xff,0xff,0x00,0xff,0xff,//1
0xb0,0xb6,0xb6,0xb6,0x86,//2
0xb6,0xb6,0xb6,0xb6,0x00,//3
0x87,0xf7,0xf7,0xf7,0x80,//4
0x06, 0x36 ,0x36 ,0x36 ,0x30,//5
0xf1, 0xee, 0xd6 ,0xb8, 0x7f,//6
 0xbd, 0xbb ,0xb7 ,0xaf ,0x9f,//7
0xc9 ,0xb6 ,0xb6 ,0xb6 ,0xc9,//8
 0xcd ,0xb6, 0xb6, 0xb6, 0xc1//9

 };

 /* interrupts 1. refresh rate 2. scroll rate */
 /* End of interrupts */
 int main(void)
 {
 unsigned char first_byte,count1,k=0,flag=0;
 count1=0;
 UBRR=25;
 UCR= (1<<RXEN)|(1<<TXEN);
 TIFR=TIFR;
 TIMSK=1<<OCIE1A;
 TCCR1B=5;
 OCR1AH=10;
 // OCR1AL=0;
 _SEI();
 while( !(USR&(1<<RXC))&& (q!=0 ));//timer1
will count till 2^16-1
 first_byte=UDR;
 if(first_byte == 42) //is *
   {
       while((count1<100) && (str1[k] != 63))  //
enter not pressed
       {
       if(USR & (1<<RXC))
          {
            str1[count1]=UDR;
            k=count1;
            count1++;

          }
       }
 flag=1;//if string entered in correct format ok
else flag remains 0 & prevoiusly stored string will
be displayed
 }
 if(str1[k] == 63)
 str1[k]='\0';
 address = 0x0001;
 x=0;
 if(flag==1)
 {do
 {
 EEWRITE(address,str1[x]);
 EEREAD( address, str+x);
 address++;
 x1=x;
 x++;
 }
 while( str1[x1] !='\0');
 count = x;
 }//end of if flag==1

 if(flag==0)
 {do
 {EEREAD( address, str+x);
 address++;
 x1=x;
 x++;
 }
 while(str[x1]!='\0');
 count = x;
 }//end of flag==0

 TIFR = TIFR;
 TCNT0 = 230;
 TIMSK |= 1<<TOIE0 ;
 TCCR0 = 3;

    int addr, curr_col_temp,m;
    u08  value;
    outp(0xff,DDRA);
    outp(0xff,DDRB);
    outp(0xff,DDRC);
    outp(0xff,DDRD);
    char t;
    curr_col=0;
 setcol(-1);
 while(1)
 {
    cli();
 if( j == 2000)
 {
 //if( offset == 4 ) temp= offset;
 offset++;
  j=0;
 }//multiple of refresh(19),make para 1900 or 2000

 if(offset >=5)
 {offset=0;
 //   temp++;
 if(temp>=count)
 temp=0;
 }

    t = str[i];
     if( t>=65 && t<=91)
      addr  = (t-'A')*5;//i is being incremented in
interrupt
     else
  if( t>=97 && t<=122) // c b/w a and z
    addr = (t-71)*5;
  else
     if( t>=48 && t<=57)  // c b/w 0 and 9
     addr = (t-48+52)*5;

  else
  addr = -325;
  curr_col_temp=(curr_col<5)?curr_col:curr_col%5;
     m = offset + curr_col_temp;
     if(m>=5) m=m-5;
     addr = addr + m;
     value = PRG_RDB(&leds[addr]);
     outp( value, PORTC);
   // curr_col = curr_col+1;
     setcol(curr_col);
     sei();
 }
 }

 void setcol( int col)
 {

 //initially switch off all coloumns
 switch (col)
 {
 case -1: PORTA=0x00;PORTB=0x00;PORTC=0xF
F;PORTD=0x00;break;
 case 0:  PORTA = 0x01; break;
 case 1:  PORTA = 0x02; break;
 case 2:  PORTA = 0x04; break;
 case 3:  PORTA = 0x08; break;
 case 4:  PORTA = 0x10; break;
 case 5:  PORTB = 0x01; break;
 case 6:  PORTB = 0x02; break;
 case 7:  PORTB = 0x04; break;
 case 8:  PORTB = 0x08; break;
 case 9:  PORTB = 0x10; break;
 case 10: PORTD = 0x04; break;
 case 11: PORTD = 0x08; break;
 case 12: PORTD = 0x10; break;
 case 13: PORTD = 0x20; break;
 case 14: PORTD = 0x40; break;
 case 15: PORTA = 0x80; break;
 case 16: PORTA = 0x40; break;
 case 17: PORTA = 0x20; break;
 case 18: PORTB = 0x40; break;
 case 19: PORTB = 0x20; break;
 default : break;
 }
 }

 void EEWRITE(int address, char value)
 {
 while(EECR&(1<<EEWE));
 eeprom_wb(address, value);
  EECR |=(1<<EEMWE);
  EECR|=(1<<EEWE);
 }

 void  EEREAD (int address,char *val)
 {
  while(EECR&(1<<EEWE));
  EEAR=address;
  EECR=(1<<EERE);
 *val= EEDR;
 }
```

CHECK.C

```
// Program for checking Dot matrix Display //
#include<io.h>
#include<sig-avr.h>
#include<ina90.h>
int main(void)
{
DDRA=0xFF;
DDRB=0xFF;
DDRC=0XFF;
DDRD=0xFF;
PORTA=0XFF;
PORTB=0XFF;
PORTD=0XFF;
PORTC=0X00;
for(;;)
{
}
}
```

REMOTE-CONTROLLED DIGITAL AUDIO PROCESSOR

■ **KULAJIT SARMA**

These days most audio systems come with remote controllers. However, no such facility is provided for normal audio amplifiers. Such audio controllers are not available even in kit form. This article presents an infrared (IR) remote-controlled digital audio processor. It is based on a microcontroller and can be used with any NEC-compatible full-function IR remote control.

This audio processor has enhanced features and can be easily customised to meet individual requirements as it is programmable. Its main features are:

1. Full remote control using any NEC-compatible IR remote control handset
2. Provision for four stereo input channels and one stereo output
3. Individual gain control for each input channel to handle different sources
4. Bass, midrange, treble, mute and attenuation control
5. 80-step control for volume and 15-step control for bass, midrange and treble
6. Settings displayed on two 7-segment light-emitting diode (LED) displays and eight individual LEDs
7. Stereo VU level indication on 10-LED bar display
8. Full-function keys on-board for audio amplifier control
9. All settings stored on the EEPROM
10. Standby mode for amplifier power control

Circuit description

Fig. 1 shows the block diagram of the remote-controlled digital audio processor. The system comprises Atmel's AT89C51 microcontroller (IC1), TDA7439 audio processor from SGS-Thomson (IC4) and I²C bus compatible MC24C02 EEPROM (IC5). The microcontroller chip is programmed to control all the digital processes of the system. The audio processor controls all the audio amplifier functions and is compatible with I²C bus. All the commands from the remote control are received through the IR sensor. The audio amplifier can also be controlled using the on-board keys.

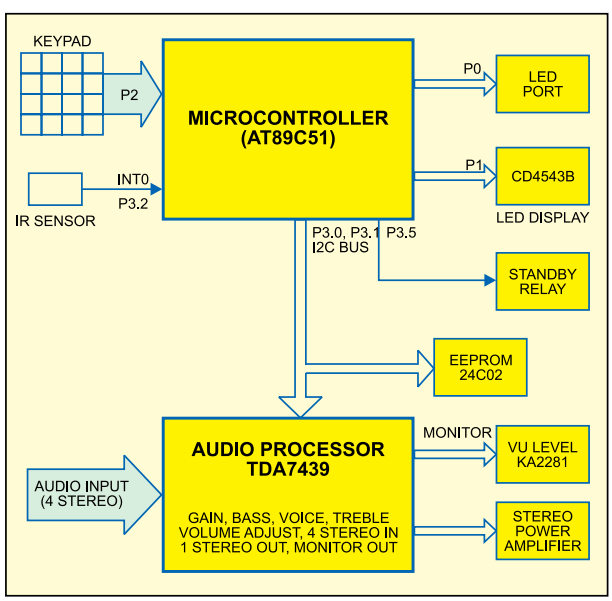


Fig. 1: Block diagram of the remote-controlled digital audio processor

Microcontroller. The function of the microcontroller is to receive commands (through port P3.2) from the remote handset, program audio controls as per the commands and update the EEPROM. A delay in updating the EEPROM is deliberately provided because normally the listener will change the value of a parameter continuously until he is satisfied.

The 40-pin AT89C51 microcontroller has four 8-bit input/output (I/O) ports.

Port 0 is used for indicating through LEDs the various functions selected via the remote/on-board keys.

PARTS LIST

Semiconductors:

IC1	-	AT89C51 microcontroller
IC2, IC3	-	CD4543 7-segment decoder/driver
IC4	-	TDA7439 audio processor
IC5	-	MC24C02 I²C EEPROM
IC6	-	KA2281 2-channel level meter driver
IC7	-	TSOP1238 IR receiver module
IC8	-	7809 9V regulator
IC9	-	7805 5V regulator
IC10	-	LM317 variable regulator
T1	-	BC558 pnp transistor
T2, T3, T5	-	BC547 npn transistor
T4	-	BD139 pnp transistor
BR1	-	W04M bridge rectifier
D1-D6	-	1N4004 rectifier diode
DIS1, DIS2	-	LTS543 7-segment display
DIS3	-	10-LED bargraph display
LED1-LED8	-	Red LED
LED9	-	Green LED

Resistors (all ¼-watt, ±5% carbon):

R1	-	8.2-kilo-ohm
R2-R24, R40-R49	-	1-kilo-ohm
R25, R28, R50, R53	-	10-kilo-ohm
R26, R29, R30, R34	-	2.7-kilo-ohm
R27	-	100-ohm
R31, R35	-	5.6-kilo-ohm
R32, R33	-	4.7-kilo-ohm
R36-R39	-	22-kilo-ohm
R51	-	220-kilo-ohm
R52	-	2.2-kilo-ohm

Capacitors:

C1, C2	-	33pF ceramic disk
C3, C10	-	10µF, 16V electrolytic
C4-C6, C39-C41	-	100nF ceramic disk
C7	-	4.7µF, 16V electrolytic
C8, C9	-	2.2µF, 16V electrolytic
C11, C20	-	5.6nF polyester
C12, C19	-	18nF polyester
C13, C18	-	22nF polyester
C14, C17	-	100nF polyester
C21-C28	-	0.47µF polyester
C29-C32	-	4.7µF, 25V electrolytic
C33, C34	-	10µF, 25V electrolytic
C35	-	1000µF, 25V electrolytic
C36	-	4700µF, 25V electrolytic
C37, C38	-	0.33µF ceramic disk
C42	-	470µF, 25V electrolytic

Miscellaneous:

X1	-	230V AC primary to 12V, 1A secondary transformer
RL1	-	9V, 160Ω, 2 C/O relay
X_{TAL}	-	12MHz crystal
S1-S7	-	Push-to-on switch
S8	-	On/Off switch
Remote	-	Creative's remote (NEC-compatible format)

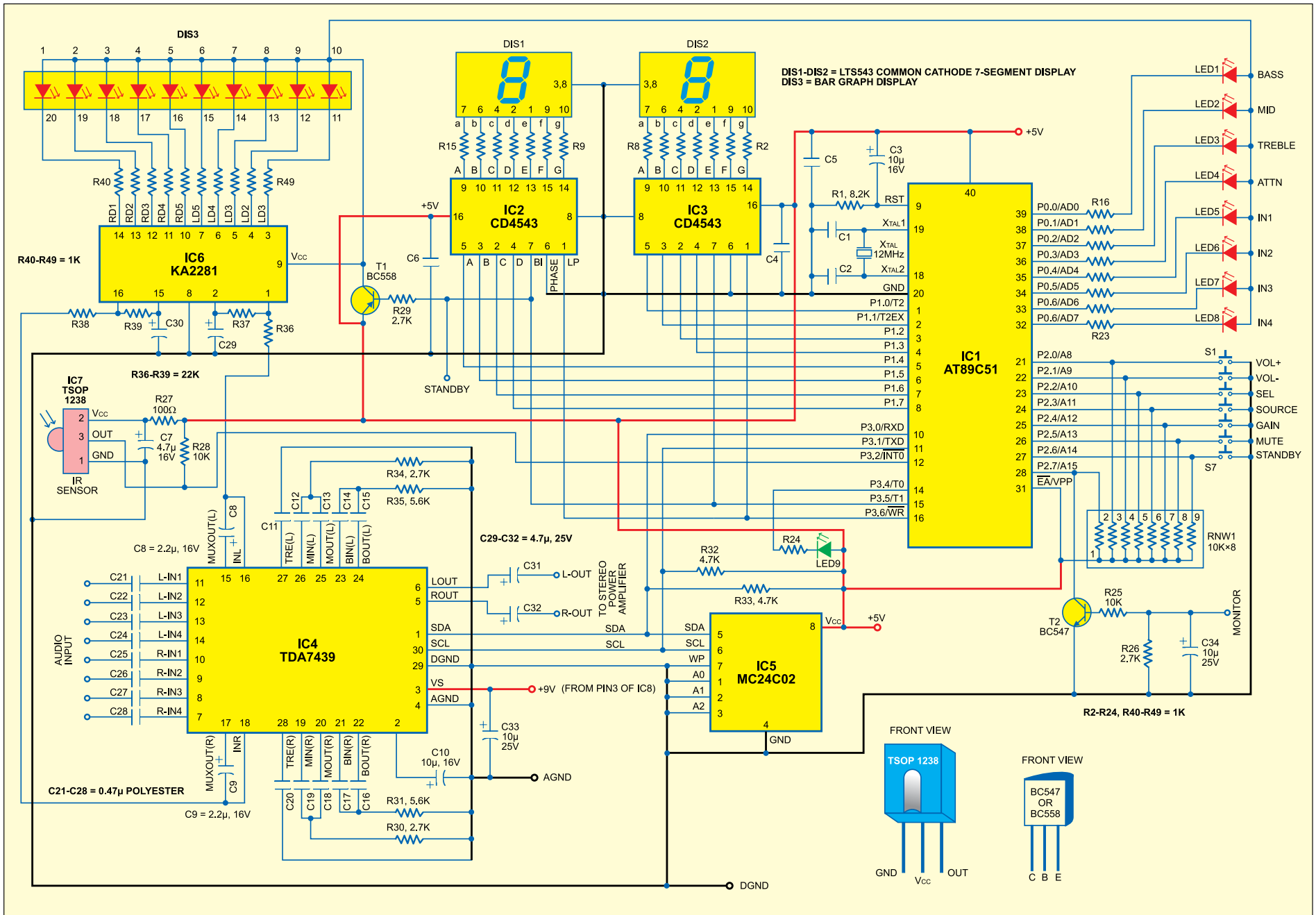


Fig. 2: Circuit diagram of the remote-controlled digital audio processor

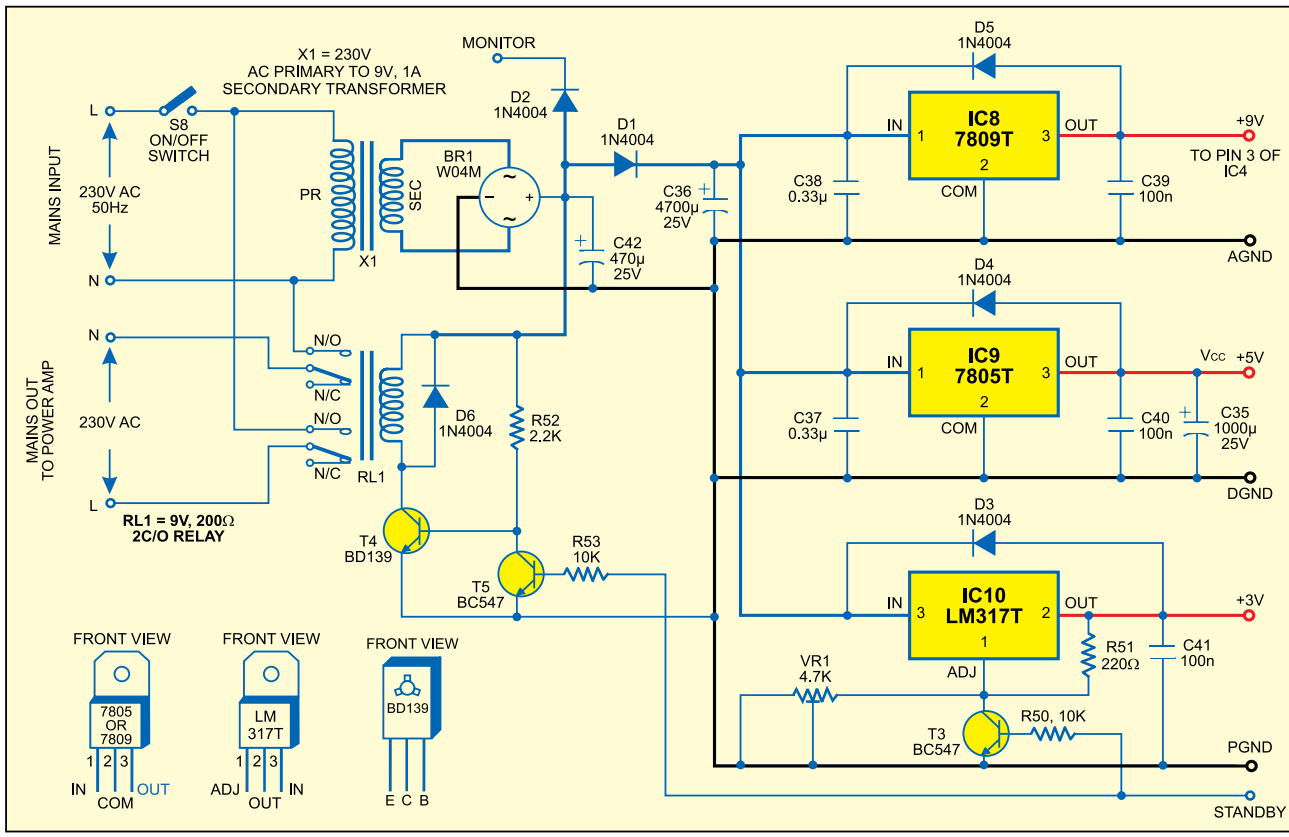


Fig. 3: Power supply

Port 1 drives the 7-segment display using 7-segment latch/decoder/driver IC CD4543.

Port 2 is pulled up via resistor network RNW1 and used for manual key control.

Pins P3.0 and P3.1 of the microcontroller are used as serial data (SDA) and serial clock (SCL) lines for the I²C bus for communicating with the audio processor (TDA7439) and EEPROM (MC24C02). These two lines are connected to pull-up resistors, which are required for I²C bus devices. P3.2 receives the remote commands through the IR receiver module. Pin P3.4 is used for flashing LED9 whenever a remote command is received or any key is pressed.

The microcontroller also checks the functioning of the memory (MC24C02) and the audio processor (TDA7439). If it is not communicating with these two ICs on the I²C bus, it flashes the volume level on the 7-segment displays.

Memory. IC MC24C02 is an I²C-bus compatible 2k-bit EEPROM organised as 256×8-bit that can retain data for more than ten years. Various parameters can be stored in it.

To obviate the loss of latest settings in the case of power failure, the microcontroller stores all the audio settings of the user in the EEPROM. The memory ensures that the microcontroller will read the last saved settings from the EEPROM when power resumes. Using SCL and SDA lines, the microcontroller can read and write data for all the parameters.

For more details on I²C bus and memory interface, please refer to the MC24C02 datasheet. Audio parameters can be set using the remote control handset or the on-board keys as per the details given under the 'remote control' section.

Audio processor. IC TDA7439 is a single-chip I²C-bus compatible audio controller that is used to control all the functions of the audio amplifier. The output from any (up to four) stereo preamplifier is fed to the audio processor (TDA7439). The microcontroller can control volume, treble, bass, attenuation, gain and other functions of each channel separately. All these parameters are programmed by the microcontroller using SCL and SDA lines, which it shares with the memory IC and the audio processor.

Data transmission from the microcontroller to the audio processor (IC TDA7439) and the memory (MC24C02) and vice versa takes place through the two-wire I²C-bus interface consisting of SDA and SCL, which are connected to P3.0 (RXD) and P3.1 (TXD) of the microcontroller, respectively. Here, the microcontroller unit acts as the master and the audio processor and the memory act as slave devices. Any of these three devices can act as the transmitter or the receiver under the control of the master.

Some of the conditions to communicate through the I²C bus are:

1. Data validity: The data on the SDA line must be stable during the high period of the clock. The high and low states of the data line can change

only when the clock signal on the SCL line is low.

2. Start and Stop: A start condition is a high-to-low transition of the SDA line while SCL is high. The stop condition is a low-to-high transition of the SDA line while SCL is high.

3. Byte format: Every byte transferred on the SDA line must contain eight bits. The most significant bit (MSB) is transferred first.

4. Acknowledge: Each byte must be followed by an acknowledgement bit. The acknowledge clock pulse is generated by the master. The transmitter releases the SDA line (high) during the acknowledge clock pulse. The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains low during the high period of this clock pulse.

To program any of the parameters, the following interface protocol is used for sending the data from the microcontroller to TDA7439. The interface protocol comprises:

1. A start condition (S)

2. A chip address byte containing the TDA7439 address (88H) followed by an acknowledgement bit (ACK)

3. A sub-address byte followed by an ACK. The first four bits (LSB) of this byte indicate the function selected (e.g., input select, bass, treble and volume). The fifth bit indicates incremental/non-incremental bus (1/0) and the sixth, seventh and eighth bits are 'don't care' bits.

4. A sequence of data followed by an ACK. The data pertains to the value for the selected function.

5. A stop condition (P)

In the case of non-incremental bus, the data bytes correspond only to the function selected. If the fifth bit is high, the sub-address is automatically incremented with each data byte. This mode is useful for initialising the device. For actual values of data bytes for each function, refer to the datasheet of TDA7439.

Similar protocol is followed for sending data to/from the microcontroller to MC24C02 EEPROM by using its chip address as 'A0H'.

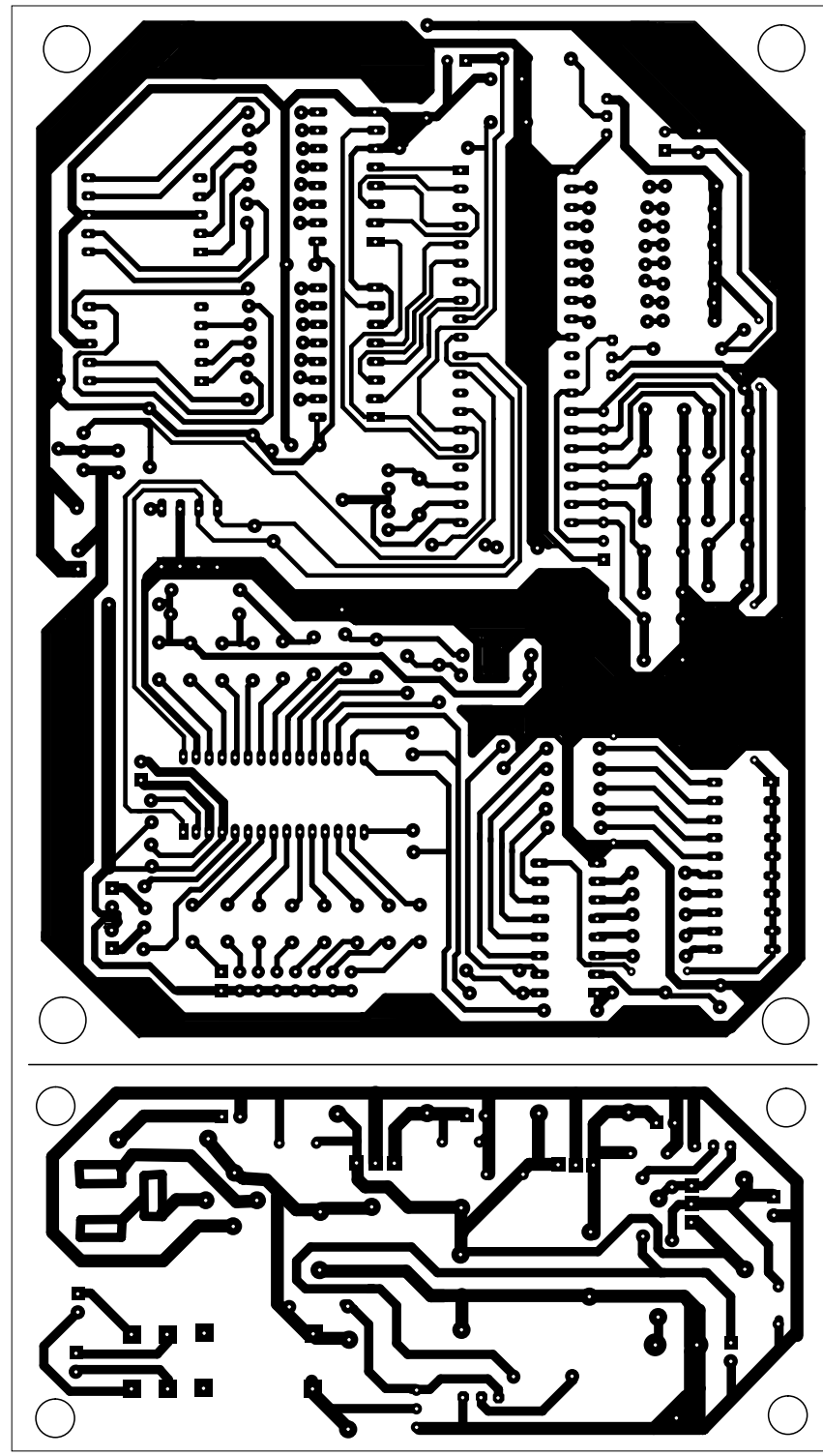


Fig. 4: Combined actual-size, single-side PCB for the remote-controlled digital audio processor (Fig. 2) and power supply (Fig. 3)

Power supply. Fig. 3 shows the power supply circuit for the remote-controlled digital audio processor. The AC mains is stepped down by transformer X1 to deliver a secondary output of 9V AC at 1A. The transformer output is rectified by full-wave bridge rectifier BR1 and filtered by capacitor C42. Regulators IC8 and IC9 provide regulated 5V and 9V power supplies, respectively. IC10 acts as the variable power supply regulator. It is set to pro-

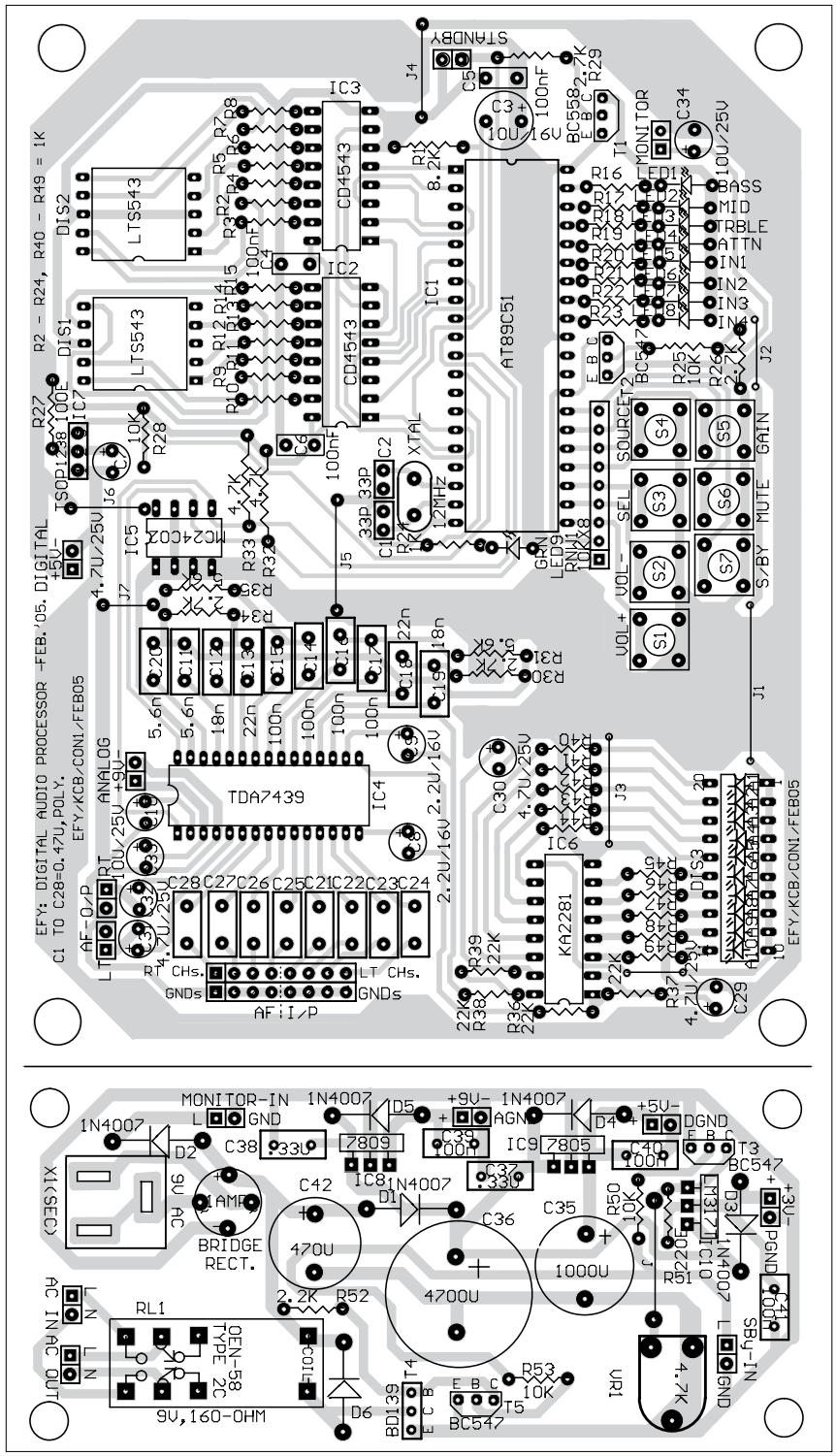


Fig. 5: Component layout for the PCB of Fig. 4

vide 3V regulated supply by adjusting preset VR1. Capacitors C39, C40 and C41 bypass any ripple in the regulated outputs. This supply is not used in the circuit. However, the readers can use the same for powering devices like a Walkman.

As capacitors above 10 µF are connected to the outputs of regulator ICs, diodes D3 through D5 provide protection to the regulator ICs, respectively, in case their inputs short to ground. Relay RL1 is normally energised to provide mains to the power amplifier. In standby mode, it is de-energised. Switch S2 is the 'on'/'off' switch.

Software

The software was assembled using Metalink's ASM51 assembler, which is freely available for download. The source code has been extensively commented for easier understanding. It can be divided into the following segments in the order of listing:

1. Variable and constant definitions
2. Delay routines
3. IR decoding routines
4. Keyboard routines
5. TDA7439 communication
6. MC24C02 communication
7. I²C bus routines
8. Display routines
9. IR and key command processing
10. Timer 1 interrupt handler
11. Main program

On reset, the microcontroller executes the main program as follows:

1. Initialise the microcontroller's registers and random-access memory (RAM) locations.

2. Read Standby and Mute status from the EEPROM and initialise TDA7439 accordingly.

3. Read various audio parameters from the EEPROM and initialise the audio processor.

4. Initialise the display and LED port.

5. Loop infinitely as follows, waiting for events:

- Enable the interrupts.
- Check the monitor input for AC power-off. If the power goes off, jump to the power-off sequence routine.
- Else, if a new key is pressed, call the DO_KEY routine to process the key. For this, check whether the NEW_KEY bit is set. This bit is cleared after the command is processed.
- Else, if a new IR command is received, call the DO_COM routine to process the remote command. For this, check whether the NEW_COM (new IR command available) bit is set. This bit is cleared after the command is processed.

• Jump to the beginning of the loop.

6. Power-off sequence. Save all the settings to the EEPROM, and turn off the display and standby relay.

Since the output of the IR sensor is connected to pin 12 (INT0) of the microcontroller, an external interrupt occurs whenever a code is received. The algorithm for decoding the IR stream is completely implemented in the 'external interrupt 0' handler routine. This routine sets NEW_COM (02H in bit memory) if a new command is available. The decoded command byte is stored in 'Command' (location 021H in the internal RAM). The main routine checks for NEW_COM bit continuously in a loop. Timer 0 is exclusively used by this routine to determine the pulse timings.

Decoding the IR stream involves the following steps:

1. Since every code is transmitted twice, reject the first by introducing a delay of 85 milliseconds (ms) and start timer 0. The second transmission is detected by checking for no-overflow timer 0. In all other cases, timer 0 will overflow.

2. For second transmission, check the timer 0 count to determine the length of the leader pulse (9 ms). If the pulse length is between 8.1 ms and 9.7 ms, it will be recognised as valid. Skip the following 4.5ms silence.

3. To detect the incoming bits, timer 0 is configured to use the strobe signal such that the counter runs between the interval periods of bits. The value of the counter is then used to determine whether the incoming bit is '0', '1' or 'Stop.' This is implemented in the RECEIVE_BIT routine.

4. If the first bit received is 'Stop,' repeat the last command by setting the NEW_COM bit.

5. Else, receive the rest seven bits. Compare the received byte with the custom code (C_Code). If these don't match, return error.

6. Receive the next byte and compare with the custom code. If these don't match, return error.

7. Receive the next byte and store in 'Command.'

8. Receive the next byte and check whether it is complement value of 'Command.' Else, return error.

9. Receive 'Stop' bit.

10. Set NEW_COM and return from interrupt.

Other parts of the source code are relatively straightforward and self-explanatory.

Remote control. The micro-controller can accept commands from any IR remote that uses NEC transmission format. These remote controllers are readily available in the market and use µPD6121, PT2221 or a compatible IC. Here, we've used Creative's remote handset.

All the functions of the system can be controlled fully using the remote or the on-board keys. By default, the display shows the volume setting and LEDs indicate the channel selected. LED9 glows momentarily whenever a command from the remote is received or any key is pressed.

Function adjustments are detailed below:

1. Volume: Use Vol+/Vol- key to increase/decrease the volume. The volume settings are shown on the two-digit, 7-segment display. Steps can be varied between '1' and '80.'

2. Mute and Standby: Using 'Mute' and 'Standby' buttons, you can toggle the mute and standby status, respectively. If 'Mute' is pressed, the display will show '00.' In 'Standby' mode, the relay de-energises to switch off the main amplifier. All the LEDs and displays, except LED9, turn off to indicate the standby status.

3. Input Select: To select the audio input source, press 'Channel' key until the desired channel is selected. The LED corresponding to the selected channel turns on and the input gain setting for that channel is displayed for five seconds. Thereafter, the volume level is displayed on the 7-segment display.

4. Input Gain set: Press 'Gain' key. The LED corresponding to the channel will start blinking and the gain value is displayed. Use Vol+/Vol- key to increase/decrease the gain for that channel. Note that the gain can be varied from '1' to '15.' If you press 'Gain' key once more, and no key is pressed for five seconds, it will exit the gain setting mode and the volume level is displayed.

5. Audio: Press 'Audio Set' (Menu) key to adjust bass, middle, treble and attenuation one by one. Each time 'Audio Set' key is pressed, the LED corresponding to the selected function turns on and the function value is displayed. Once the required function is selected, use Vol+ and Vol- to adjust the setting. Bass, middle and treble can be varied from '07' to '7.' Values '0' through '7' indicate 'Boost' and '00' through '07' indicate 'Cut.' Attenuation can be varied from '0' to '40.'

Construction

The circuit can be easily assembled on any PCB with IC base. Before you install the microcontroller, memory and audio processor in their sockets and solder the IR receiver module, make sure that the supply voltage is correct. All parts, except the audio processor (TDA7439), require 5V DC supply. The audio processor is powered by 9V DC.

Download source code: http://www.efymag.com/admin/issuepdf/Audio%20Processor.zip ●

DEVICE CONTROL THROUGH PC'S PARALLEL PORT USING VISUAL BASIC

■ **ADEEB RAZA**

Here is a Windows-based program developed in Microsoft Visual Basic programming language for controlling eight devices through the PC's parallel port or Line Printer Port (LPT). The program accepts the input in decimal number and outputs in binary form across the data pins of the PC's parallel port for controlling the connected devices/ appliances.

PC's parallel port

The standard parallel port comprises four control lines, five status lines and eight data lines (refer to the table). It is found on the back of the PC as a D-type 25-pin female connector. Here, we are concerned only with data lines D0 through D7 terminated at pins 2 through 9. These data lines are the primary means of sending information out of the port. Pins 18 through 25 of the connector are grounded.

Control lines of the parallel port are used to provide control signals such as 'form feed' and 'initialise' to the printer.

The five status lines are the only input lines of the standard parallel port. These allow the printer to send signals such as 'error,' 'paper out' and 'busy' to the PC.

Circuit description

Fig. 1 shows the block diagram for device control through the PC's parallel port using Visual Basic. The data output port of the PC's parallel port is used for controlling the devices or appliances. The interface circuit requires regulated 6V DC to drive the loads. Eight MCT2E opto-osolator ICs are used to prevent damage to the parallel port from short-circuit that may occur across the interface circuit. Darlington array IC ULN2803 is used to drive the relays for controlling the devices.

Fig. 2 shows the circuit for device control using the PC's parallel port programmed in Visual Basic. To get the power supply for the circuit, 230V AC mains is stepped down by transformer X1, rectified by bridge rectifier R3151 and filtered by capacitor C1 (1000µF, 25V). The filtered output is fed to input pin 1 of regulator IC 7806. The regulated 6V DC is used to power the interface circuit comprising ICs MCT2E (IC2 through IC9) and ULN2803 (IC1). Optocoupler MCT2E can be replaced with 4N35.

LED1 through LED8 connected across data output pins 2 through 9, respectively, are used to indicate the

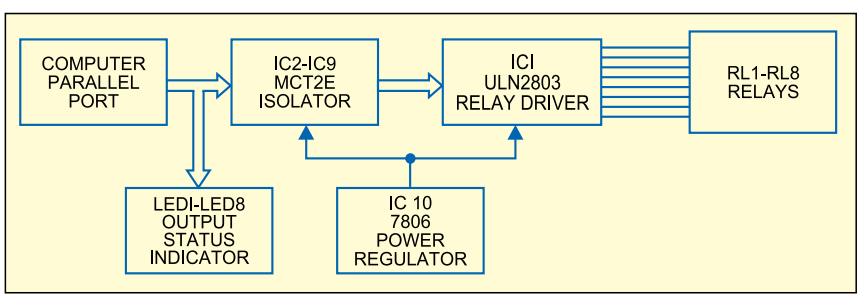


Fig. 1: Block diagram of device control through PC's parallel port using Visual Basic

Parallel-Port Pin Details

Pin number	Traditional use	Port name	Read/Write	Port address	Port bit
2-4	Data out	Data port	W	Base	D0-D2
5-9	Data out	—	W	Base	D3-D7
1	Strobe	Control port	R/W	Base+2	C0
14	Auto feed	—	R/W	Base+2	C1
16	Initialise	—	R/W	Base+2	C2
17	Select input	—	R/W	Base+2	C3
15	Error	Status port	R	Base+1	S3
13	Select	—	R	Base+1	S4
12	Paper end	—	R	Base+1	S5
10	ACK	—	R	Base+1	S6
11	Busy	—	R	Base+1	S7

PARTS LIST

Semiconductors:
IC1 - ULN2803 relay driver
IC2-IC9 - MCT2E optocoupler
IC10 - 7806 voltage regulator
BR1 - 1A bridge rectifier

Resistors (all ¼-watt, ±5% carbon):
R1-R16 - 220-ohm resistor

Capacitors:
C1 - 1000µF, 25V electrolytic capacitor
C2 - 0.1µF ceramic type capacitor

Miscellaneous:
X1 - 230V AC primary to 0-9V, 250mA secondary transformer
S1 - On/Off switch
RL1-RL8 - 6V, 100-ohm, 1C/O relay
- 25-pin, D-type parallel-port male connector

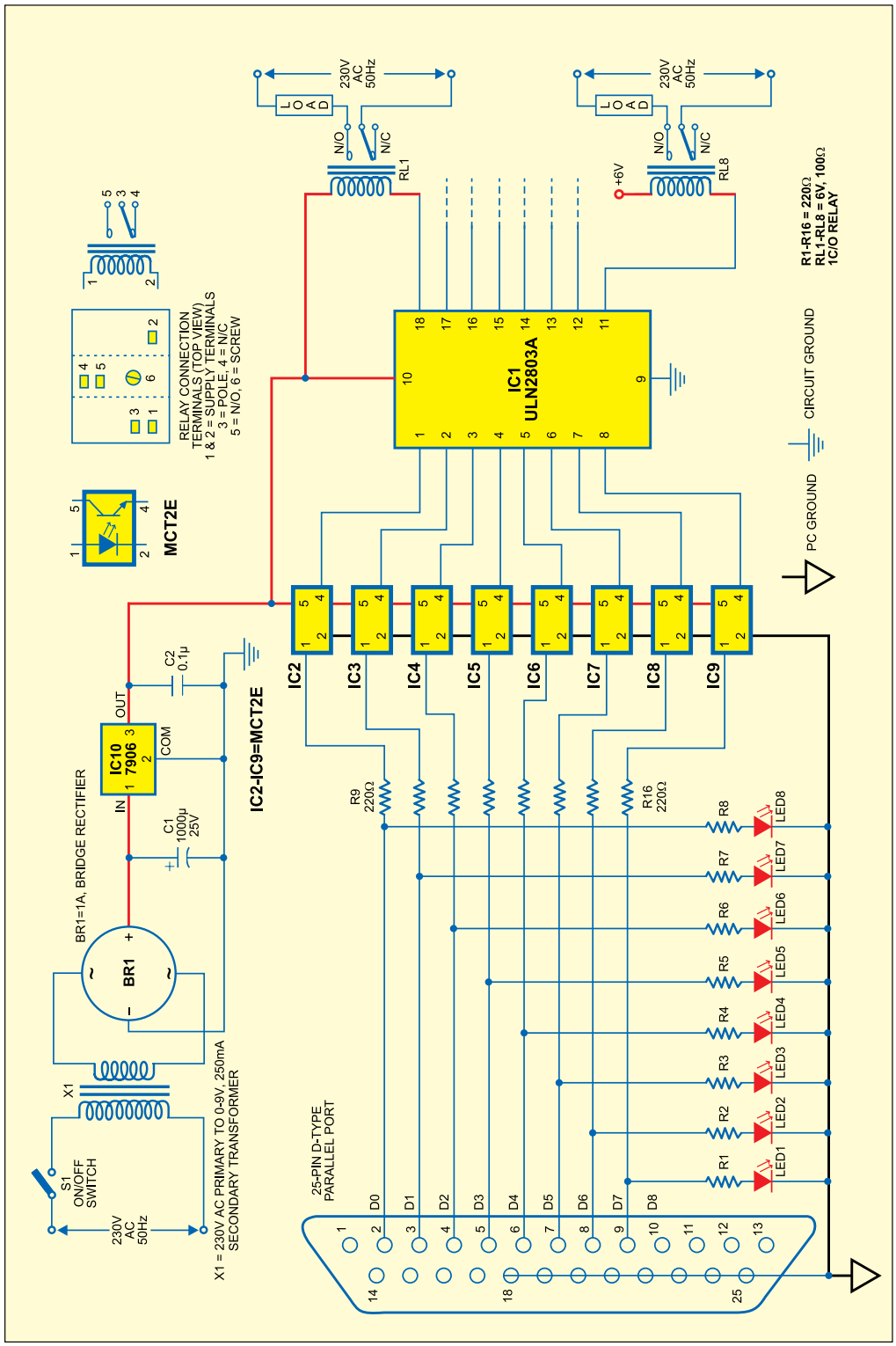


Fig. 2: Circuit for device control through PC's parallel port using Visual Basic

status of the loads. Glowing of any of these LEDs indicates that the device connected to that specific output line is 'on.'

IC ULN2803 (Fig. 3) is a Darlington array relay driver that can drive eight relays. Since IC ULN2803 has an internal freewheeling diode to quench the inductive kick, no external freewheeling diodes are required across the relay coils. The devices are connected through the relay contacts to mains. The relays are used to switch on or off the appliances.

Software program

Before going into details of the program, let us figure out some limitations of Visual Basic programming for interfacing the circuit. Visual Basic cannot directly access the computer hardware to control the external world. All the hardware requests must go through the supported file format of Windows operating system.

So the best way to manipulate the parallel port is the printer object. The printer object allows text and graphics to be printed on the printer through the parallel port of the PC. While all is well with this option, it is useless when you want a direct control of the hardware. In order to control the port directly, we must use something external to our program. A dynamic link library (DLL) file called 'WIN95IO.DLL' is used for that purpose.

The WIN95IO.DLL file is meant for a 32-bit machine, supported by Visual Basic Versions 4, 5 and 6. No matter which version you are using, the DLL file must be in the Windows\system directory of your machine. The interface control software program can be developed

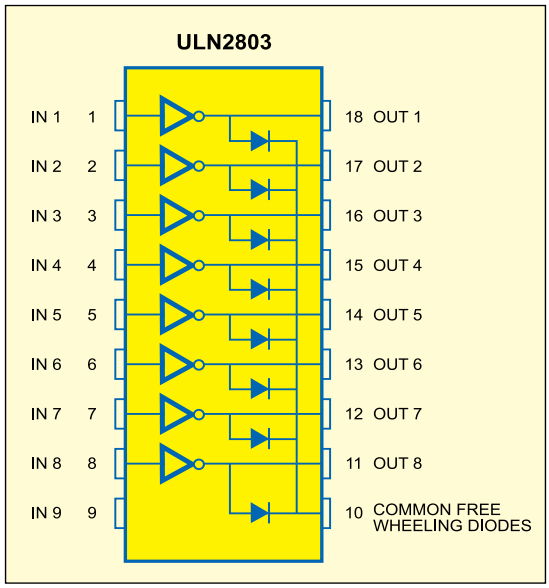


Fig. 3: Pin details of ULN2803

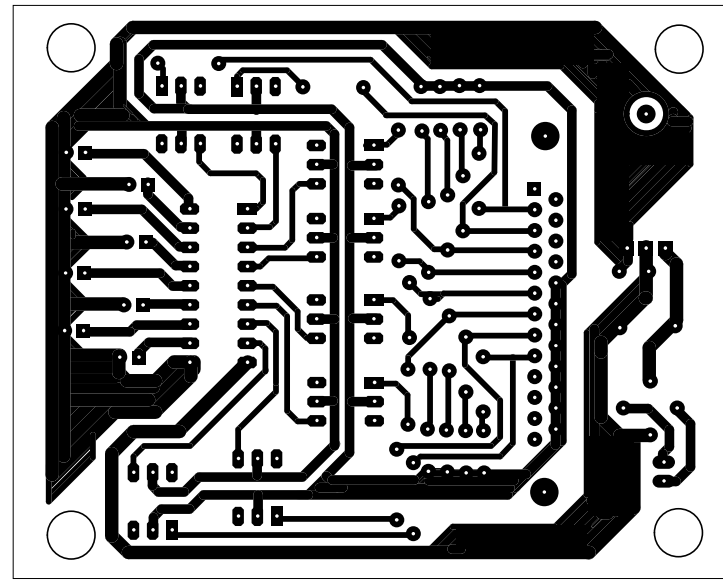


Fig. 5: Actual-size, single-side PCB layout for device control through PC's parallel port using Visual Basic

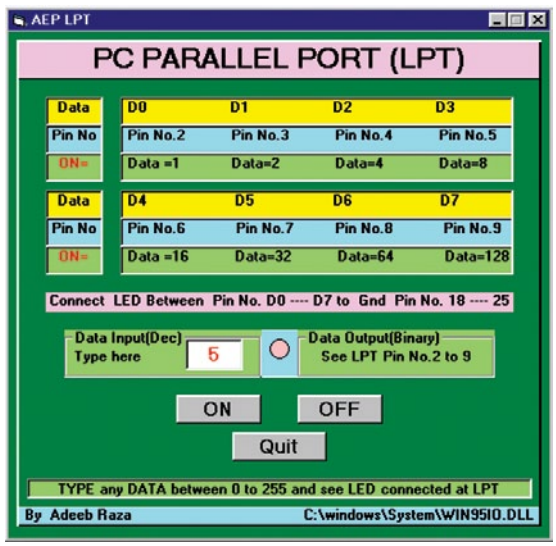


Fig. 4: Screen that appears when program is run

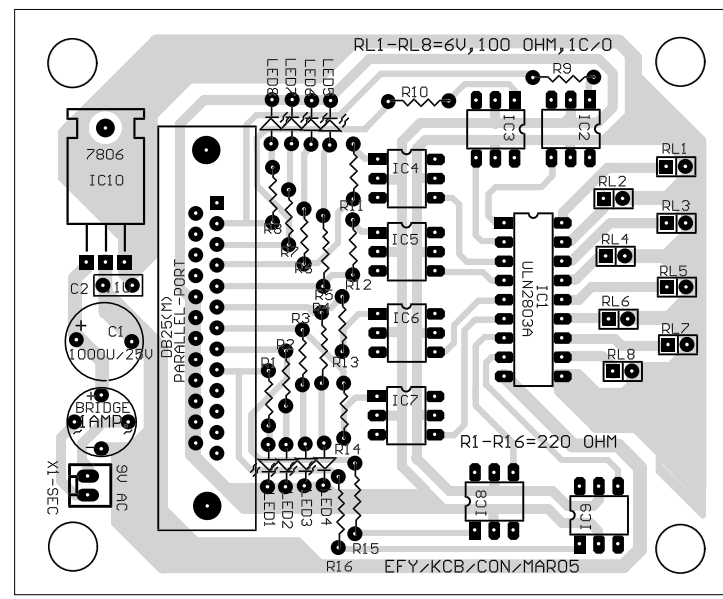


Fig. 6: Component layout for the PCB

thereon. No matter which DLL you use, it won't work under Windows NT due to security reasons.

The program code is given at the end of this article. It is assumed here that Microsoft Visual Basic 6 is installed on your PC and you have the basic programming knowledge.

The program coding is simple and you can write it yourself. Launch Visual Basic from the desktop and open a new project by selecting the 'Standard EXE' option. By default, it will open an empty project window on the screen with 'Form 1' as the file name. The form is one of the supported files of the Visual Basic. Pick the required components as shown in the screenshot (Fig. 4) from the toolbox on the left-hand side of the screen. The properties of each component can be set from the right-hand side of the screen.

The coding starts by declaring 'WIN95IO.DLL' in the first line "Private Declare Sub vbOut Lib 'WIN95IO.DLL' (ByVal AEPPort As Integer, ByVal AEPData as Integer)." The computer port is defined as 'AEPPort.' Its base address is assigned as 378 (in hex) by the program line "AEPPort=&H378." The 'vbOut' statement is used to send a bit to a port, for example, 'vbOut [port],[number]'

When you are done with coding, compile and run the program. You'll get the screen as shown in Fig. 4. Save the project file with '.vbp' extension. Make the executable file from 'File' menu.

EFY note. Form 1 is named as 'Arport' and Project 1 file as 'Arport.vbp.'

Construction

Construct the circuit for device control on any general-purpose PCB. Use eight flexible wires for data bus (D0

through D7) by connecting their one end to the PCB and the other end to the respective data pins of the 25-pin, D-type parallel-port male connector. This male connector connects to the female connector on the PC. An actual-size, single-side PCB for the circuit and its component layout are shown in Figs 5 and 6, respectively.

Testing procedure

1. Install Microsoft Visual Basic 6 on your system.

2. Fabricate or get the PCB shown in Fig. 5.

3. Connect the 8-data line male connector to the female connector on the PC.

4. Launch Visual Basic from the desktop and develop the application as explained in the software program section. Save the project file with extension '.vbp.' Alternatively, you can copy the executable file 'Arport' from the EFY-CD to your system.

5. Open 'Arport' and click 'Input Edit' box. You're prompted to input the data in decimal form. For example, input '5' and click 'On' button using mouse. The indicator on the screen will turn 'red.' Then LED7 and LED5 connected across the parallel port will glow, which corresponds to binary output '00000101.' The appliances connected to the respective output lines will turn on.

6. To turn off the appliances, click 'Off' button on the screen.

7. To exit the application, click 'Quit' button.

Download source code: http://www.efymag.com/admin/issuepdf/Device%20Control.zip

SOURCE CODE (Arport)

```
Private Declare Sub vbOut Lib "WIN95IO.DLL" (ByVal AEPPort As Integer, ByVal AEPData as Integer)
Dim AEPPort As Integer
Dim AEPData As Integer

Sub AEPOut(Data As Integer)
vbOut AEPPort, AEPData
End Sub
Private Sub Form_Load()
Shape1.Visible = False
pat = 0
End Sub
Private Sub Command1_Click()
AEPPort = &H378
pat = Text1.Text
AEPData = Val(pat)
AEPOut (Data)
If pat = "" Then GoTo y Else GoTo x
x:
Shape1.Visible = True
Shape2.Visible = False
y:
End Sub
Private Sub Command2_Click()
AEPPort = &H378
pat = 0
AEPData = Val(pat)
AEPOut (Data)
Shape1.Visible = False
Shape2.Visible = True
End Sub

Private Sub Command3_Click()
AEPPort = &H378
pat = 0
AEPData = Val(pat)
AEPOut (Data)
End
End Sub
```

AUTO CHANGEOVER TO GENERATOR ON MAINS FAILURE

■ **GP CAPT. (RETD) K.C. BHASIN**

An auto-changeover on mains failure (AMF) system comprising mains and standby sources of power supply continuously monitors the incoming mains and in case of its interruption, starts the standby diesel generator (DG) set, monitors its output and then transfers the load to the DG set.

Here is a construction project that utilises off-the-shelf readily available switchgear and integrates it with the indigenously designed logic control circuitry to automatically start the standby supply source on failure of the mains 3-phase supply and stop the DG set on resumption of mains. This system costs about 40 per cent less than the systems supplied by AMF panel designers.

System features

1. The original configuration/operation of the DG set as also its control panel is not disturbed. That means manual start/stop operation of the DG set and its control panel functions of monitoring its 3-phase output are still available.

2. Before changeover either to the DG set or to mains, the selected source is checked for single-phasing, phase reversal, and under- and over-voltage conditions. If the conditions are not fulfilled, changeover to the faulty source is inhibited.

3. Suitable delays have been provided in start and stop control of the DG set.

4. The maximum number of cranking (starting) attempts is presettable by the user.

5. For indicating the mode of operation, selected source of supply, low-battery condition, etc, status-indication LEDs have been provided on the logic control panel.

6. A buzzer warns the operator of low-battery state and over-cranking attempts. It can be reset/disabled by the operator. However, the low-battery indication LED will remain lit as long as the battery voltage remains low.

7. When manual mode is selected, the DG set can be electrically started from the logic panel itself via push-buttons. Latching relays ensure that either the start or the stop operation is performed at a time.

8. Use of the industrial changeover switchgear ensures preferential selection of mains, in case both the DG set supply and mains are available. Mechanical interlocking and tripping before selection arrangements ensure that the two sources are never paralleled.

9. The system is capable of flawless operation under potentially noisy (electrical) environments due to the use of a hardware debounce and feedback circuitry.

10. The logic panel has been designed using discrete ICs, relays and other passive/active devices. Hence understanding the logic is easy and the changes required to meet the peculiarities of the individual standby supply source can be easily implemented.

11. The logic circuit con-

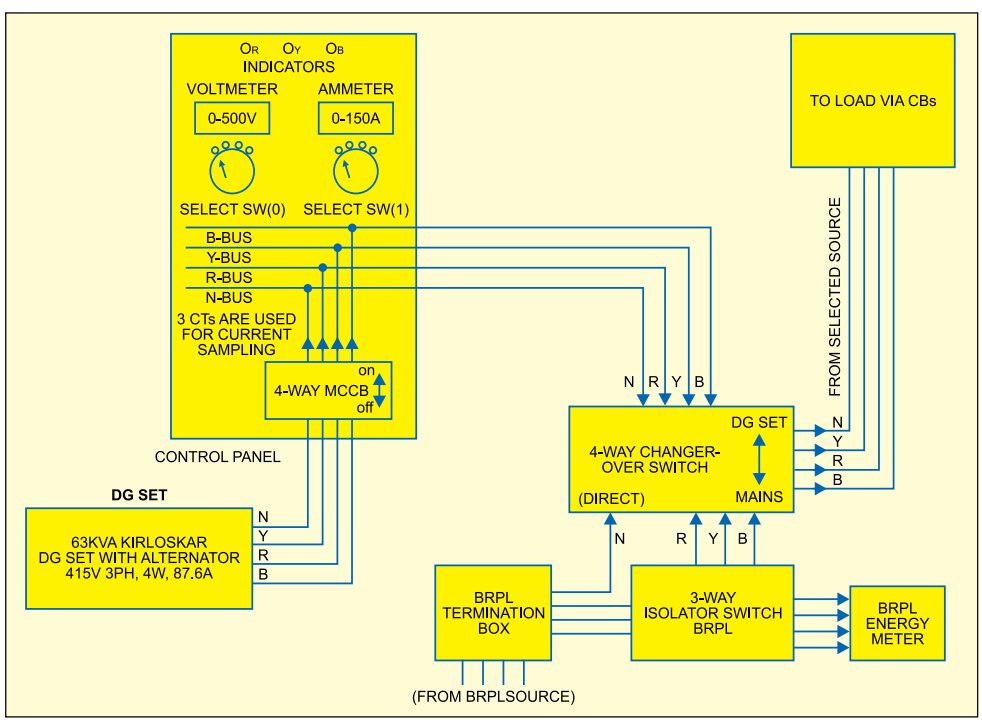


Fig. 1: Line/block diagram of the manual changeover system that existed before changeover to AMF

sumes minimal power, as most of the ICs used are CMOS.

Manual changeover system

Fig. 1 shows the block diagram of the manual changeover system. The 3-phase, 4-wire output of the DG set is terminated on the control panel via the 4-way isolator (moulded-case circuit breakers (MCCBs)). The control panel has the usual voltage and ampere meters with current transformers (CTs) and selector switches for monitoring all the three phases. (Some panels may have a power factor meter as well.) The 3-phase output of the control panel is routed to a 4-way manual changeover switch. The mains 3-phase power is also terminated on the manual changeover switch via an isolator switch and energy meter. The source (mains or DG set output) selected by the manual changeover switch is routed via MCCBs to feed the desired loads.

The AMF system has been designed around a Kirloskar HA series engine with a 3-phase, 4-wire, 415V AC, 50Hz alternator capable of delivering a maximum of 87.6 amperes per phase at a power factor (PF) of 0.8. The alternator uses 300V DC excitation at 4.2A.

The DG set is equipped with:

1. Flywheel with starter ring
2. 12V electric starter
3. Mechanical shutdown lever
4. Battery charging dynamo
5. Engine instrument panel consisting of:

- Off/on/start key
- Lube-oil pressure gauge
- Battery charging ammeter
- Hour meter

Fig. 2 shows the block diagram of the electrical system of the DG set. It differs slightly from the diagram printed on the DG set's instrument panel.

The DG set is shut off by mechanically pulling a lever, which cuts off the fuel supply to the injectors and the engine comes to a halt in eight to ten seconds. The knowledge of functioning of starting circuit/components and charging circuit/components is necessary for proper understanding the design of AMF logic system (to be described later).

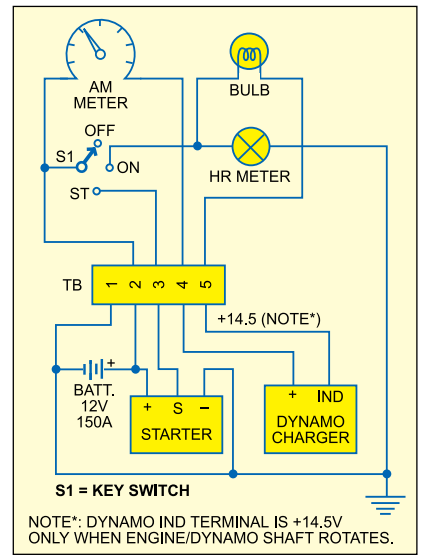


Fig. 2: Schematic block diagram of the DG set's electrical system

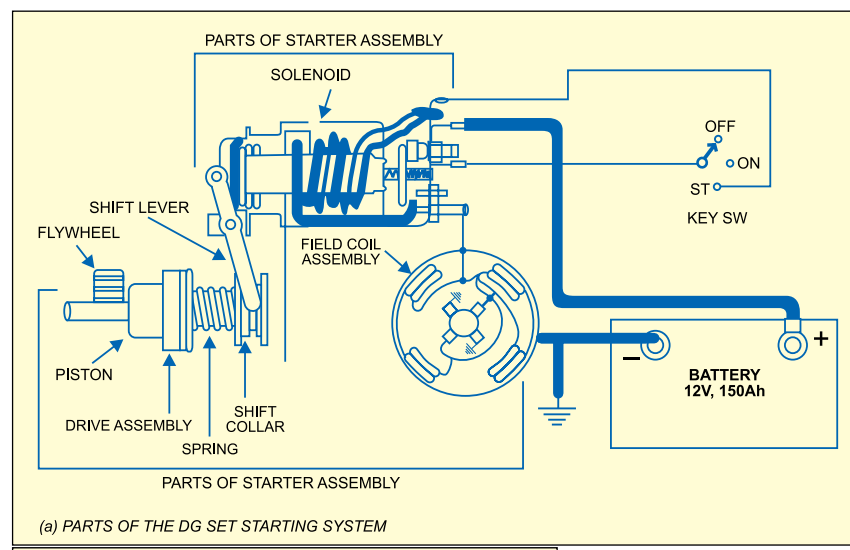


Fig. 3: The DG set starting system (above) and starter motor assembly (below)

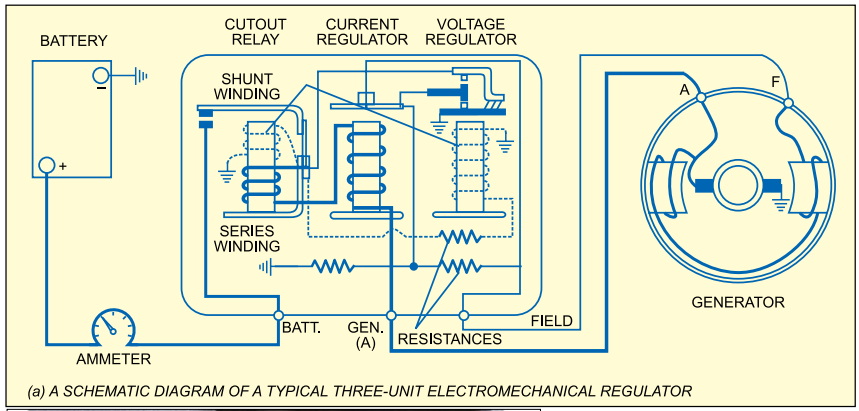
DG set starting/cranking circuit

Fig. 3(a) shows the circuit for starting the DG set. The starter assembly comprises a starter, solenoid assembly as well as shift lever and drive assembly. It is housed inside a metallic body with cut near the drive assembly for engaging its geared pinion with the flywheel ring gear of the DG set when the solenoid is energised. The body of the starter assembly is grounded/connected to the negative terminal of the battery.

Fig. 3(b) shows the complete starter motor assembly. It is similar to the starter assembly fitted on your car.

When the key switch is shifted to 'Start' position, the starter solenoid energises to cause the solenoid plunger to move the shift lever, which engages its pinion with the engine flywheel ring gear. The movement of the plunger also closes the main solenoid contacts, applying +12V battery voltage to the starter motor through solid contacts to allow the starter motor to draw 150-200 amperes of current for overcoming the inertia of the engine. Once the engine starts, the pinion will overrun, protecting the armature from excessive speed and the flywheel from damage. When the key switch is released, the plunger-return spring disengages the pinion.

Caution. Never operate the starter



(a) A SCHEMATIC DIAGRAM OF A TYPICAL THREE-UNIT ELECTROMECHANICAL REGULATOR

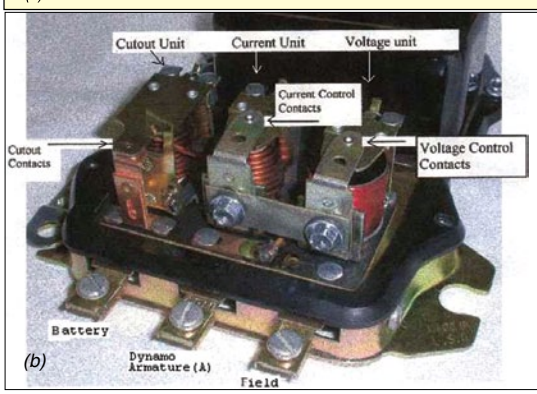


Fig. 4: Schematic diagram of a typical three-unit electromechanical regulator (above) and photograph of a typical 3-unit electromechanical regulator (below)

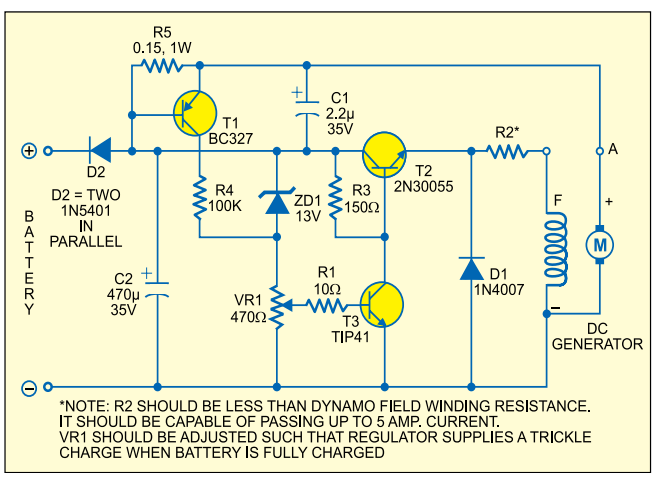


Fig. 5: A typical solidstate electronic regulator with reverse current protection for 12V battery

for more than 15 seconds at a time as excessive cranking can cause overheating of the starter. After each cranking attempt, allow the starter to cool for at least a minute.

Battery charging circuit and components

The charging current for the battery is supplied by the dynamo (also called 'generator'). A generator is like a motor in reverse. Instead of supplying the current to rotate the motor's shaft, we rotate or spin the dynamo's shaft to generate electricity. The dynamo rotor is mechanically coupled to the engine's shaft through a V-belt and pulley arrangement. The current generated in its armature is AC and not DC.

Commutators on its shaft are used to rectify the AC current. Two spring-loaded brushes slide on the commutators. One brush is connected to ground and the other is connected to the main output of the generator (the positive terminal marked 'A' for armature). As the armature/commutator assembly rotates, the brushes touch different contacts on the commutator such that the polarity of the current moving into and out of the armature commutators is always connected to the correct brushes. The net effect of this operation is that the generator output is DC even though the current inside the armature windings is AC.

Three-unit electromechanical regulator. Since the dynamo output is a function of the engine speed, the average DC output may vary. A voltage/current regulator combined with a reverse-current cut-out is used to regulate the output between 13.8V and 14.2V, which is considered to be appropriate for charging a 12V lead-acid battery. The cut-out prevents battery discharge into the generator when its output voltage is below that of the battery.

Fig. 4 shows a typical 3-unit external electromechanical regulator used for the purpose. It comprises three relays. Two of the relays have a shunt and series windings, respectively, while the third (used for cut-out function) employs mixed series and shunt windings. The regulator may also be installed within the dynamo housing itself. A full description of its working principle is given inside the box on the next page.

Solidstate (electronic) regulator. Some newer versions employ solidstate regulators with reverse-current protection. A typical solidstate regulator circuit is shown in Fig. 5. The output voltage of this regulator is held constant by 13V zener diode ZD1 in series with potmeter P1. P1 is adjusted such that when the battery is fully charged to roughly 13.8V, the field current of the generator is adequate to maintain a trickle charge current of 50 to 100 mA (through armature via 0.1-ohm resistor R5) to replenish the battery charge.

Initially, when a battery in discharged state is connected to the circuit, and if the charging current exceeds 4A, transistor T1 conducts to forward bias transistor T3 and transistor T2, in turn, stops conducting, which results in reduced field current of the generator. The net effect is that the output current through the armature and resistor R5 is reduced to maintain the output current from the generator below 4 amperes.

Key-switch operation. Referring back to Fig. 2, when we shift the key switch to 'on' position, the warning bulb glows to indicate that the engine is stationary. One terminal of the warning

bulb is connected to the battery via 'on' position of the key switch and the other terminal is connected to the armature (point 'A'), which is grounded through commutator brushes.

The switch motion from 'on' to 'start' position works against the tension of a spring inside the key-switch. After the engine starts and you release the key switch, it automatically comes back and rests at 'on' position. With the engine running, the armature terminal 'A' builds up a voltage of around 14V DC and as such the bulb stops glowing as the current through the bulb reduces considerably.

The key switch works like the ignition key switch of your car. The warning bulb also works the same way as the battery indicator light on the dashboard of your car. The lighting of the battery indicator while the car is running indicates that your car battery is not charging and hence something is wrong. The same goes for the warning bulb on the DG set. In 'on' position of the key switch, the hour-meter starts working. Any other ancillary equipment that you wish to run with the engine could also be connected to the 'on' terminal of the key-switch.

Once the DG set engine is running at the correct speed and the alternator is working, it generates 3-phase, 415V AC at 50 Hz, which is routed to its control panel for monitoring and its further extension to the changeover switch.

Now, if you are satisfied with manual operation of the DG set but wish only to automate the operation of the changeover switch function, it is a rather simple affair. Automatic changeover switches (also called automatic transfer switches (ATS)) are available from a number of electrical switchgear makers.

The AMF system

Fig. 6 shows the block diagram of the proposed AMF system. The system incorporates automatic switching operation of the DG set and its integration with the automatic transfer switch from Havell. Blocks marked 2, 5 and 6 (existing DG set control panel, mains supply via 3-phase isolator and energy meter, and 3-phase load connections via MCCBs, respectively) have already figured in the manual changeover system.

The DG set has been modified by installing an additional solenoid puller along with a contactor (high-current capacity relay) and another identical contactor for electrically starting the engine by making use of the DG set's solenoid-starter combination. Further explanation of the modification is given under the

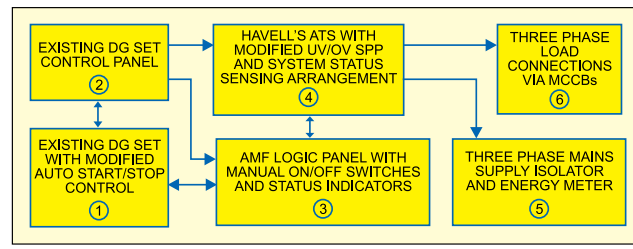


Fig. 6: Block diagram of the proposed AMF system

Three-unit Electromechanical Regulator

Three units control charging. On the left are the cut-out contacts, which connect and disconnect the dynamo armature from the battery. When the output voltage of the generator exceeds 11.8V, the contacts are pulled together and the armature's A terminal is connected via thick wires to the current limiter section. The cut-out section also has a fine wire winding. This winding is connected to ground (also called shunt connected) and provides the magnetic energy to pull the contacts together.

The contacts have a specific air gap and there is a spring trying to pull the contacts open. The spring tension is adjusted to allow the contacts to come together from 11.8 to 13 volts. The thick winding around the outside provides additional pull to the contacts when the current is flowing to the battery to prevent arcing when the voltage output of the dynamo armature is quite close to pull-in voltage.

At the point where the voltage at the armature is below the battery voltage, the current starts flowing from the battery to the armature. This reverse flow of current reverses the polarity of the magnetic field produced by the thick current winding. This magnetic field opposes the field created by the small shunt winding, resulting in a clean release of the contacts.

The centre pole is the current regulator. This section regulates the maximum current that the generator is able to put out without destroying itself. It has a pair of contacts that are normally closed (NC). When the generator voltage starts to flow through the cut-out section, all of the current flows through the current regulator coil.

When the current exceeds a predetermined level (8 to 10 amperes normally), spring tension on the contacts allows the contacts to break. When the contacts open, it removes the hard ground on the dynamo's field (F) terminal. Now, only a parallel path for the field winding to ground is available via a resistor, which causes a reduced-current ground path for the field winding. This reduces the output of the generator.

When the generator output drops, the spring pulls the current contacts back together and bypasses the resistor to ground. The generator again runs to provide the full output and the cycle repeats. If the load is too high, the contacts will be continuously vibrating to limit the current to the preset level. This allows the charging current to be limited to the maximum safe limit.

On extreme right is the third unit forming the voltage control section. It consists of a pair of NC contacts connected in series with the current control contacts to ground and to the field (F) terminal. Under these contacts is a coil of very fine wire wound around a metal pole piece, as the coils on the other two units are. The air gap and the spring tension on these terminals are adjusted to control the voltage output of the dynamo armature from 14 to 14.5 volts.

Since this coil is connected in parallel across the armature terminal to ground, its magnetic field is directly proportional to the armature output voltage. When the voltage reaches the preset level, the contacts break to open the direct ground path for the field current and leave the resistor across the F terminal to ground. As a result, the dynamo armature output voltage drops. The contacts close and the full dynamo armature output becomes available again. It works exactly like the current section, except that it responds only to the voltage output.

There is an additional resistor between the F terminal the armature contact of the cut-out to provide a damping effect when the control contacts open and reduce the arcing of the contacts. It plays no part in the 'controlling' operation of the regulator. The relay contacts are made of tungsten for long life.

description of the modified wiring diagram of the DG set.

Block 3 contains the main logic circuitry for controlling the start/stop of the DG set by making use of sense signals picked up from the DG set as well as from block 4. In addition, it includes audio/visual status and warning indicators, which prompt the operator's intervention during emergency/malfunctioning of equipment. Block 3 also allows you to manually start/stop the DG set, if required.

Block 4 shows use of the industry-standard automatic transfer switch. We've used Havell's ATS due to its relative merits for developing this AMF system. Certain additional circuitry has been incorporated for monitoring both the mains and the DG set supply sources for single-phasing, phase reversal, and under- and over-voltage conditions before permitting changeover of the supply source. The additional circuitry also senses various operations such as tripping of mains and its resumption, as well as the mode of operation of the ATS (auto or manual). These status signals serve as sense signals for the AMF logic panel (block 3) in starting and stopping the DG set appropriately.

The components used for automating the system are detailed below.

Havell's automatic transfer switch

Havell's automatic transfer switch used for this AMF system comprises four symmetrical poles coupled to the main operating mechanism. The switching mechanism is 'quick make, quick break' type. A brief description of its contact mechanism in association with the relay panel (supplied as essential part of the ATS) is given below.

Contact mechanism (Fig. 7(a)). Each pole has two independent sets of moving-contact assemblies for main and standby supply and one fixed-contact assembly for the common outgoing load terminals. Cams, when rotated by the main operating mechanism, mechanically operate the moving-contact assemblies.

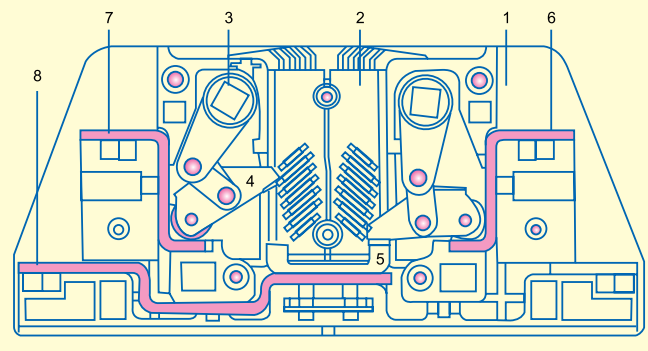
Moving contacts make onto fixed contacts under constant pressure with the back-up spring. Main contacts are made of silver-tungsten to ensure anti-weld characteristics. The Arc Chute plates, placed in the path of contacts, quench the arc and thereby enhance the life of contacts.

The main mechanism independently actuates two sets of cam linkages, which, in turn, operate the two independent moving-contact assemblies.

Fig. 7(b) shows the line diagram of Havell's ATS with essential relays. The contact closing command is effected through solenoid closing coil C supplied with 230V AC. The operating mechanism always responds by closing onto the mains supply side and not to the standby supply side when both supplies are present. Tripping coil TC, when energised, brings the automatic transfer switch to off/neutral position. Closing onto the standby supply side is achieved through selective coil SC. The energisation of selective coil SC disengages the main mechanism and prevents closing onto the mains supply side. The solenoid coil can then close the second set of moving contacts onto the standby supply.

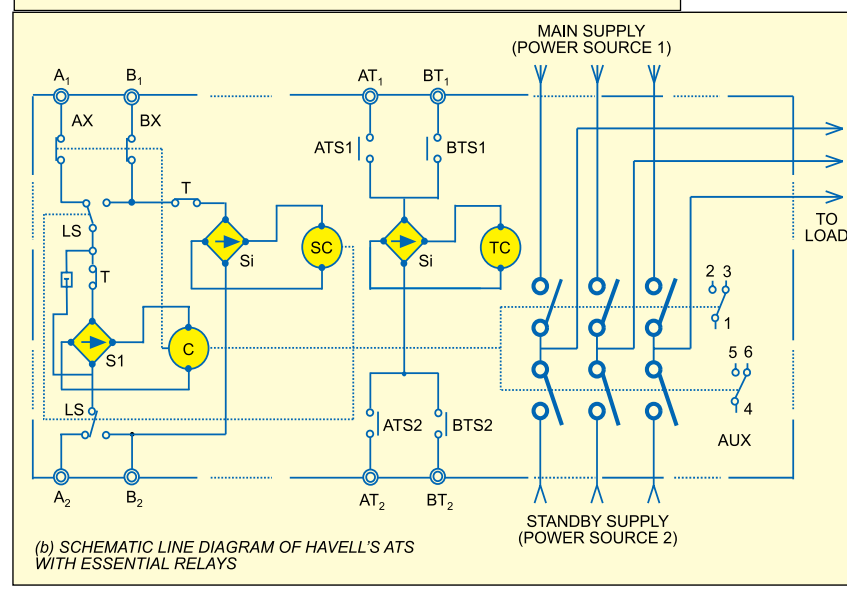
The moving contact mechanisms of mains and standby supplies are inherently mechanically interlocked through a double-throw arrangement that ensures that at no point of time the two supplies are paralleled.

During automatic switch operation, the closure of its contacts towards mains or standby side and tripping of the switch to neutral state are effected by certain mechanically-operated contacts as well as relay-operated contacts. All



1 = FRAME
2 = HOUSING FOR ARC CHUTE
3 = OPERATING SHAFT FOR CONTACTS
4 = MOVING CONTACT
5 = FIXED CONTACT
6 = MAIN SUPPLY - INCOMING TERMINALS
7 = STANDBY SUPPLY - INCOMING TERMINALS
8 = COMMON OUTGOING - LOAD TERMINALS

(a) CONTACT MECHANISM OF HAVELL'S ATS



(b) SCHEMATIC LINE DIAGRAM OF HAVELL'S ATS WITH ESSENTIAL RELAYS

Fig. 7: Contact mechanism of Havell's ATS (above) and schematic line diagram of Havell's ATS with essential relays (below)

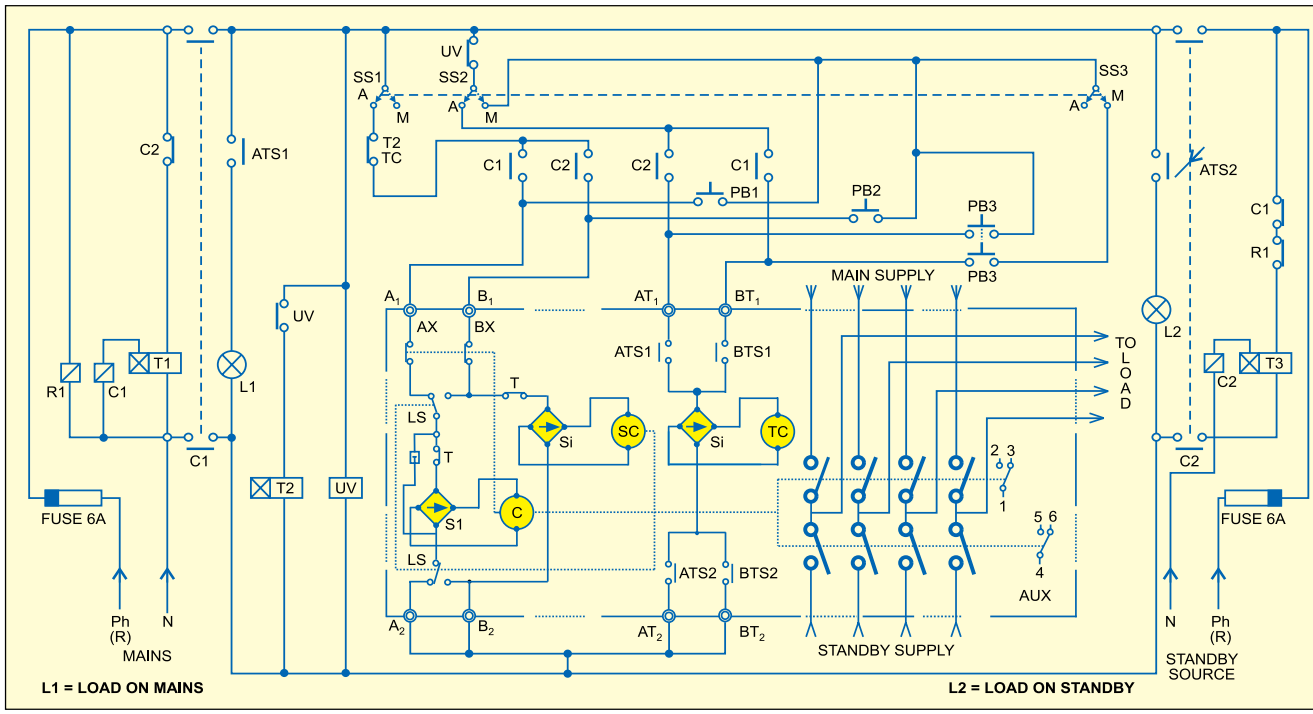


Fig. 8: Line diagram of Havell's integrated ATS

relays/coils operate off 230V AC. For transfer of load onto mains or standby side, 230V AC is applied between terminals A1and A2, or B1 and B2, respectively. For tripping of mains or standby, the 230V AC is applied between terminals AT1 and AT2, or BT1 and BT2, respectively.

Function of some of the relays for operation of the ATS has already been given under the description of its contact mechanism. However, operation of all contacts/relays (coils) used within the ATS, as shown in Fig. 7(b), is summarised below:

1. Auxiliary contacts AX and BX: When ATS switch is in off (tripped/neutral) state, auxiliary mechanical contacts AX and BX are in closed state. When the ATS energises via closing coil C towards mains or standby side by operation of selective coil SC, in association with closing relay C, the respective auxiliary contacts break and the supply to closing coil is cut off.

2. ATS1 and ATS2, and BTS1 and BTS2: These two pairs of mechanical contacts for mains and standby sources, respectively, close when the respective supply has been switched on via the ATS. The application of 230V AC between the affected terminals AT1 and AT2, or BT1 and BT2, will energise trip coil TC to bring the switch to neutral position.

3. TC: Trip coil, when energised, trips the switch to neutral position.

4. SC: Selective coil, when energised, disengages the main mechanism and prevents closing towards the mains supply side by pulling limit switches marked LS (Fig. 7) towards B1 and B2 contacts. SC coil is used along with closing coil C to close towards the standby side.

Emergency operation of ATS. In an emergency, the ATS can be operated manually, as an off-load switch only, as follows:

1. Closing onto mains supply: A manual handle rotates the operating shaft by 45° in anticlockwise direction to achieve closure under off-load conditions.

2. Closing onto standby supply: Closure onto the standby supply side is achieved when 'Selective' mode (through selective command hole) is continuously pressed and the manual handle rotates the operating shaft by 45° in anticlockwise direction.

3. Tripping: Tripping can be achieved manually by pressing 'Trip' button momentarily.

Caution. Before the emergency operation, isolate the load from the ATS.

Integrated ATS operation. The Havell's ATS now comes as an integrated unit. A line diagram of the integrated unit is shown in Fig. 8. Apart from the components shown in Fig. 7(b), it comprises selector switch, pushbuttons, indicator lamps and relays including timers.

The mode-selector switch is a 3-way, double-changeover switch. The three poles in the circuit are marked as SS1, SS2 and SS3. The auto position is marked 'A,' while manual position is marked 'M.' Pushbutton switches PB1, PB2 and PB3 are used for selection of mains and standby (DG set) supplies and tripping of the selected source (mains/standby), respectively, in manual mode of operation. Pushbutton PB3 has two mechanically interlinked sections as shown in the figure. The controlling voltage is derived from Red phase and Neutral of the respective supply sources connected to the circuit via fuses rated at 6 amperes each.

The ATS operates in auto and manual modes as follows:

1. Auto mode. For operation in auto mode, the switch is to be kept in auto (A) position. Let's assume that initially the ATS is in neutral (tripped) position and mains supply is available. Relay R1 energises almost instantaneously, opening its NC contacts marked R1 to ensure that even if standby supply becomes available subsequently, it will not be able to reach timer T3 and contactor C2 (which ultimately controls the SBY source selection). Even if both supplies (mains and standby) become available simultaneously, the standby supply will be cut off from reaching contactor C2 (because of the delay introduced by timer T3), while no timer comes in the path of relay R1.

The mains' red phase becomes available to timer T1, which connects the phase to contactor C1 after the preset delay. Once contactor C1 energises and closes its two contacts, it extends mains' red phase and neutral to A1 and A2 points of the ATS assembly (shown within dotted lines). The phase supply is routed to A1 terminal via SS1 and NC contact T2 of UV (under-voltage relay) initiated timer, which remains closed unless it encounters prolonged under-voltage condition. This sequence ensures switching of load to mains as explained earlier. The mains indicator bulb lights via ATS-1 contact of the ATS. As an additional safety, the NC contact C1 in series with R1 contact also opens during energisation of C1 contactor to break the path for standby supply.

When mains is not available (trips), relay R1 and contactor C1 de-energise. Now, if standby supply from the DG set is made available, its red phase reaches timer T3 via closed R1 and C1 contacts and, after preset delay of T3, contactor C2 energises to open the NC contact in the path of mains' red-phase supply to timer T1. Switching of the standby supply's red phase to control the ATS operation is feasible only after the delay introduced by timer T3.

In case mains resumes before T3's delay period is over, R1 will energise to open the control circuit for standby supply and mains will become available to the load.

However, if mains does not resume within the delay period of timer T3, C2 will energise and it will trip mains from the load via SS2 section of the selector switch (and AT1 and AT2 contacts) before closing towards standby side due to energisation of B1 and B2 contacts via SS1 section of the selector switch and closed contacts of T2. The load is switched to standby supply and L2 indicator bulb glows.

Now, if mains resumes, it will open the path for C2 due to energisation of R1. Once C2 de-energises, C1 will energise via T1. Now, via SS2 section of the mode switch and C1 contacts, the ATS will trip via BT1 and BT2 contacts before switching the load to mains source once again.

2. Manual mode. For manual-mode operation, the selector switch is to be kept in manual (M) position. It is advisable to press the trip button before changing from one source to another in manual mode.

In manual mode, both pairs of trip contacts (AT1 and AT2, and BT1 and BT2) are paralleled via SS3 section of the mode switch when trip button (PB3) is pressed. Pressing of trip button ensures that whatever source is presently connected to the load, it trips the ATS and brings it to neutral position. The red phase of any of the available supplies (mains when both supplies are available) can be used to switch the load towards mains (using pushbutton PB1) or standby supply (using PB2). The automatic operation of the ATS is inhibited by SS1 and SS2 sections of the mode-selector switch. Before changing over the source again, press trip switch PB3 before pressing PB1 or PB2.

We had explained the working of the manual changeover system/components as well as the components utilised for the automatic changeover system. Next section covers the modifications effected in the Havell's automatic transfer switch (ATS) and the Kirloskar 62.5kVA DG set, as well as design of the logic control circuitry for automating the operation of the changeover system.

Modifications in the ATS

Havell's ATS does not monitor all the three phases for under-voltage protection, nor does it have over-voltage or single-phasing or phase-sequence-change prevention arrangement. We have therefore used an industry-standard Minilec make SPP relay (VMR D2) that incorporates all these protections. Its normally-opened (N/O) contacts energise only when all the conditions are satisfied.

We have used a VMR D2 relay for each of the two 3-phase supply sources, i.e., mains supply as well as the DG set supply. The red-phase input from the respective sources is extended to the fuse and neutral point of each control supply to the Havell's ATS when the conditions are fulfilled. The under-voltage limits (80 to 95 per cent) and over-voltage limits (105 to 120 per cent) of the auxiliary supply are adjustable via screwdriver controls.

We have used a 4-way (three N/O and one N/C contacts), 9A rated contactor in conjunction with VMR D2 relays (one each for mains and standby source) to detect the healthy supply status of the available supplies to the logic control panel and to extend the red phase of supplies to Havell's ATS. The connections to VMR D2 relay for 415V AC 3-phase (with 415V AC auxiliary supply), and associated 4-way contactor, are made as under:

1. 415V AC 3-phase supply is connected to terminals marked 1, 2 and 3 (in top row).

2. Auxiliary voltage terminals 7 and 8 (middle row) are connected to any two phases (we used 'R' and 'Y' phases).

3. N/O terminals 13 and 14 (bottom row) are used for energising the 4-way (three N/O and one N/C) contactor. The contacts close only when the 3-phase supply is healthy.

4. Red phase and Neutral of the respective supplies are extended via two N/O contacts to Havell's ATS for control.

5. The remaining two contacts (one N/O and one N/C) are used for conveying the healthy supply status of

the source to the logic control panel.

Further, we have replaced the 3-way, double-changeover mode-selector rotary switch with a similar 4-way, double-changeover rotary switch (Kaycee make 4F46D). The fourth section (SS4) has been used to convey the selected mode information to the logic control panel.

The Havell's integrated ATS with the modified wiring is shown in Fig. 9.

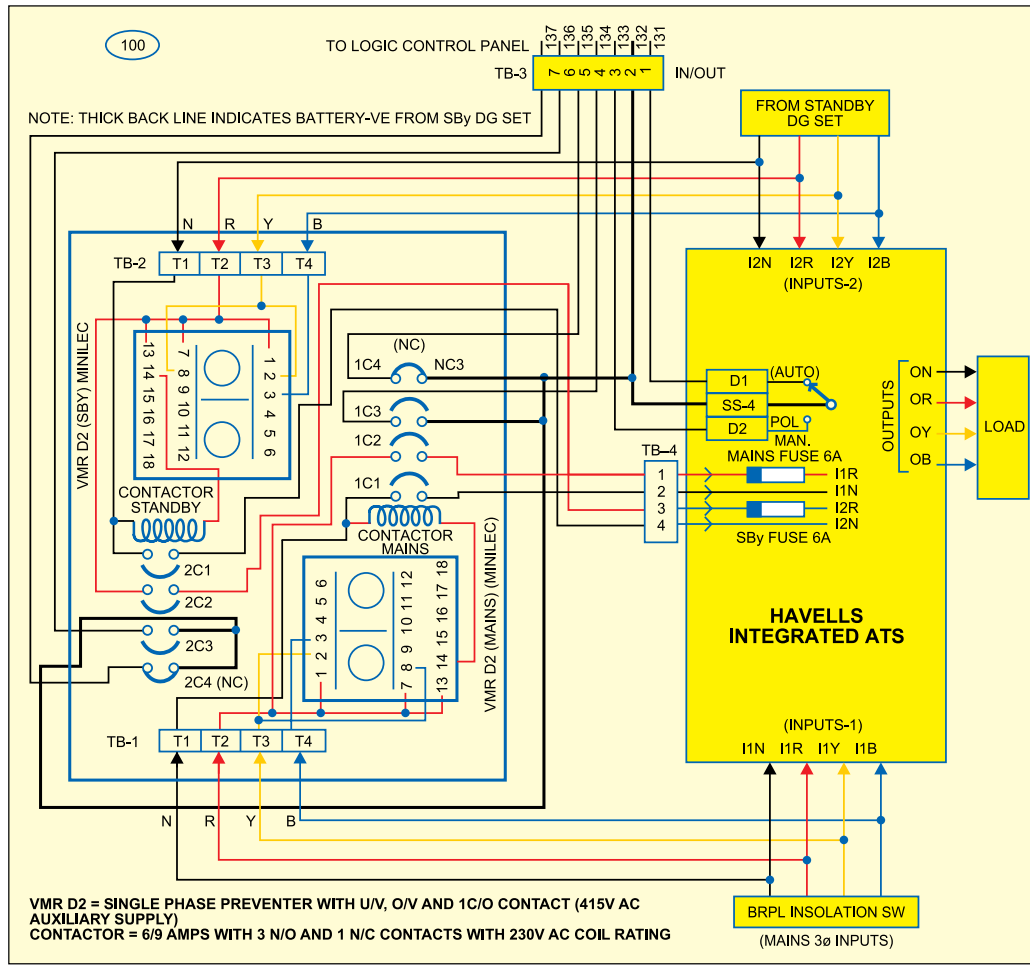


Fig. 9: Havell's integrated ATS showing modifications

Modifications in DG set

Fig. 10 shows the modified wiring diagram of the standby DG set.

For automating the start, stop and run operations of the DG set, we need to sense its present operational status. Two signals are used for knowing the status of the DG set. These are 230V AC alternator output (across R-phase and Neutral) and dynamo-induced voltage of about +14.5V DC. Both these voltages become available after the DG set is running. (***Note:*** For +14.5V to be available, +12V from the DG set battery needs to be extended to 'On' terminal of the key-operated start switch.) A relay in the logic panel was used to extend the +12V supply to 'Run' terminal.

For starting the DG set, we have to switch +12V to the solenoid (forming inbuilt part of the starter) of the starter using a relay with contact rating in excess of 20 amperes. Also, for stopping the DG set, we use a similar relay for energising a solenoid puller.

The solenoid puller (purchased from the local market in Delhi) was secured to rails on which the DG set rests. Its free-moving central rod end was attached to the lever (used for manually stopping the engine) using a thin wire rope, similar to the one used in scooters for clutch/gear operation. As the solenoid arm had no tension, a spring of suitable tension and length was used so that it returned to its original position along with the stop lever once the solenoid was de-energised.

The battery supply and 'IND' terminal of the dynamo are extended to the indigenously designed logic control panel via TB1, while for energising the 'stop' and 'start' relays, and for extending +12V to 'Run' terminal, pin 5 of TB1 is used. The AC sample (R-phase and neutral) from the DG set alternator is extended to the logic control panel via pins 1 and 2 of TB2.

A total of eight lines carrying DC and AC power, and sense and control signals are routed from the DG set to the logic control panel, and another seven lines carrying sense signals are routed from the ATS cubicle to the logic control panel. Fig. 11 shows the interconnection diagram. The arrows indicate the direction of signals.

It is desirable that each wire-end bears proper marking, so that after a disconnection for any fault rectification, the wires are connected back to their original points. For this purpose, numbered plastic ferrules are available from electrical switchgear dealers, which can be inserted in the cable ends to mark the wire numbers. The wiring conventions proposed in Fig. 11 can be used for identification. Use of multicolour leads (with resistor-type colour code) can further eliminate the risk of wrong connections during/after any servicing.

Logic control panel

A summary of signals carried from/to the logic control panel tag blocks (TBs)

to/from the DG set and the ATS cubicle, along with their direction with respect to the logic control panel, is given in Table above. These signals are used to control the start and stop logic for the DG set.

Specifications. The conditions governing the automatic start and stop operation of the DG set and the related timings need to be specified for designing the logic control circuitry. Some of the conditions and timing requirements will vary for individual systems depending on the power rating, battery voltage, and built-in facilities of the DG set. The system design must also take into account the operational requirements/constraints of the system, including whether the DG set is required to feed the load on 24-hour basis (in case of mains failure) or from dawn to dusk, etc. The auto-start and stop logic used by us for our specific system is explained below.

Auto-start logic for the DG set. For the DG set to start, the conditions required are:

1. The mode-selector switch is in 'auto' mode.
2. The DG set is not already running.
3. A transition from 'mains available' to 'mains not-available' has been detected.
4. Mains (healthy) is not available even after 3-second wait, while conditions 1 and 2 are still applicable.

Once conditions 1 through 4 have been satisfied and the engine has gained about 80 per cent speed (as evidenced by the dynamo/alternator output), the starter can be deactivated.

Auto-stop logic for the DG set. For initiating the stop action for the DG set, the following conditions must be met:

1. The mode selector switch is in 'auto' mode.
2. The DG set is running.
3. A transition from 'mains not available' to 'mains available' has been detected.
4. Mains (healthy) is available even after 30-second wait, while conditions 1 and 2 are still applicable. The 30-second is off-load cooling period for the DG set. (***Note:*** A higher cooling period must be taken for engines of higher rating.)

Once conditions 1 through 4 have been satisfied, the engine stopping action is initiated and unless the engine comes to a dead stop, the controlling solenoid should not be released; else, the engine will tend to restart because of inertia.

Engine-cranking logic. This logic functions in association with the auto-start logic for the DG set.

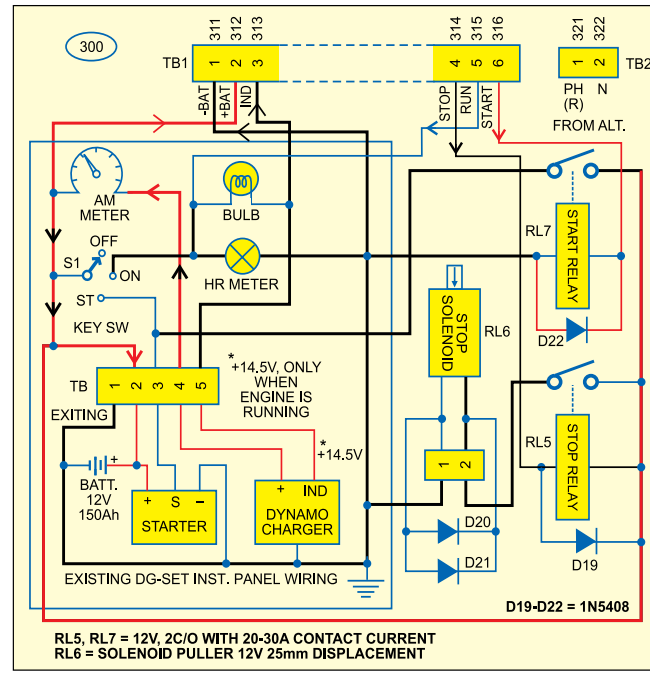


Fig. 10: Modified wiring diagram of the standby DG set

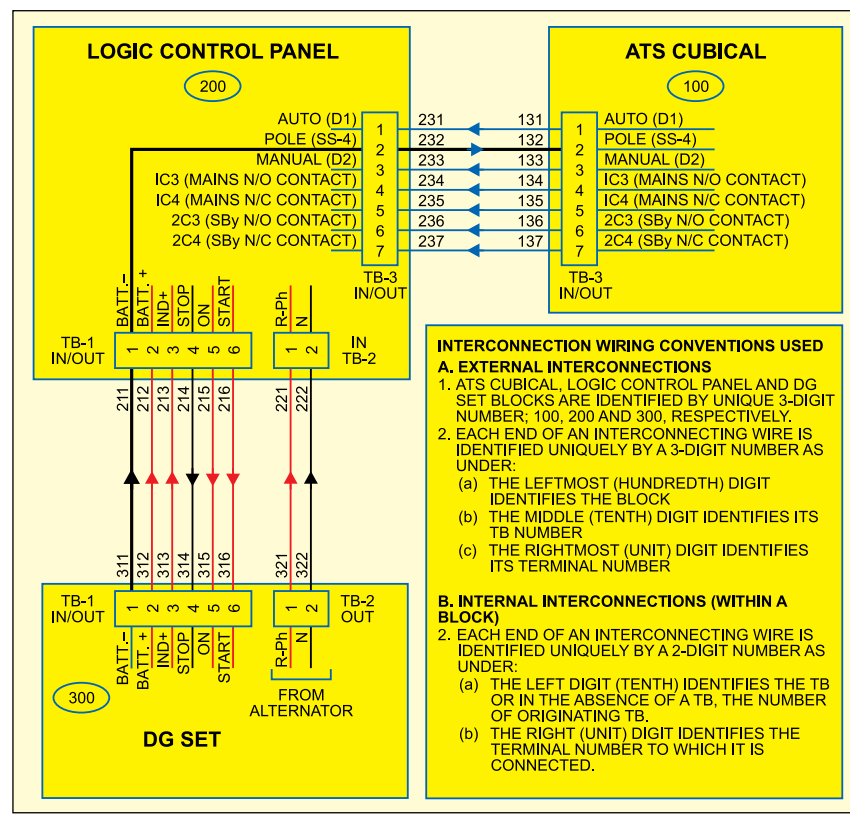


Fig. 11: Interconnections between the logic control panel and the DG set as well as the ATS cubical

Once conditions governing the 'DG set auto-start logic' have been satisfied, and continue to remain satisfied, engine-cranking (starting) is attempted for around twelve seconds. This period is practically adequate for the engine to pick up the speed without excessive draining of the battery.

During winter when the engine remains idle during the whole of night, the

Signals Carried to and from the Logic Control Panel

TB Tag No.	Signal	Direction	Description
TB-1 Tag No.			TB-1 is used for carrying the DC supply and control signals.
1	Bat. -	In	It carries BAT- (Gnd.) from the battery in the DG set.
2	Bat.+	In	It carries +12V from the battery in the DG set.
3	+IND.	In	This wire carries +14.5V generated by the dynamo.
4	Stop	Out	This output is active-low (Gnd) for energising 'Stop' relay RL5 in the DG set (see Fig. 11).
5	On	Out	This output is active-high (+12V). Once the engine starts, it extends +12V to 'On' terminal of the keyswitch.
6	Start	Out	
TB-2 Tag No.			TB-2 is used for carrying the alternator output (red phase and neutral) sample for initiating activation of 'On' relay RL1, which extends +12V to 'On' point of the keyswitch.
1	R.Ph.	In	With the engine running, it carries the red phase from the alternator to energise relay RL1 in the logic control panel.
2	N	In	This terminal extends neutral from the alternator.
TB-3 Tag No.			TB3 brings in active-low (Gnd) signals (via terminals 1, 3 through 7) from the control cubical under different conditions.
1	Auto	In	BAT- (Gnd) return when the mode-selector switch in the ATS cubical is kept in 'auto' position.
2	Pole	Out	BAT- (Gnd) is extended to the ATS cubical via the pole of mode-selector section SS4.
3	Man.	In	BAT- (Gnd) return when the mode-selector in the ATS cubical is in 'manual' position.
4	C3 (N/O Mains)	In	BAT- (Gnd) return when (healthy) mains is available.
5	C4 (N/C Mains)	In	BAT- (Gnd) return when (healthy) mains is not available.
6	C3 (N/O SBy)	In	BAT- (Gnd) return when (healthy) SBy is available.
7	C4 (N/C SBy)	In	BAT-(Gnd) return when (healthy) Sby is not available.

viscosity of lube oil is high, which puts a higher load on the starter and as such the starter may not be able to start the engine at first attempt. Under these circumstances, it is advisable to start the engine manually (after switching off the logic control panel), two/three times so that the starter is able to function smoothly in auto position, in one attempt itself.

If the engine fails to start in the first attempt (of 12 seconds), it waits for 70 seconds before making the next attempt. This period is essential for the battery to accumulate charge for subsequent cranking attempts. The maximum number of cranking attempts can be preset (between two and five) by the operator. A higher number of cranking attempts indicates that the high-viscosity lube oil has been used, the battery terminals/leads to the starter need cleaning/tightening, the starter pinion is not meshing properly with the engine flywheel gear, or the starter itself needs servicing. If the engine fails to start within the preset attempts, audio-visual warning is activated.

Logic control circuit

Apart from implementing the above-mentioned logic, the logic control panel performs quite a few additional functions as well, which will become clear when we go through its circuit details. The schematic diagram of the logic control circuit is shown in Fig. 12.

The connections to/from the DG set are terminated on TB-1 (shown split in two sections) and TB-2, while TB-3 is used for terminating connections to/from the ATS cubicle. The description of the signals on these TBs is given in the table. The ground (battery negative) connection from pin 1 of TB-1 is extended to the modified ATS circuit via pin 2 of TB-3. This ground connection is returned via other pins of TB-2 to indicate the state of mode switch and mains and standby supply sources. The status is displayed through LEDs (LED1 through LED6). Three of these six signals (auto mode 'on,' mains 'on' and mains 'off') are used for logic control.

Detection of the DG set engine status. For detecting the 'on/running' status of the DG set, initially the AC output from its alternator is monitored by the circuit built around MCT2E optocoupler (IC15). The value of resistors R38 (33 kilo-ohms, 0.5W) and R39 (390 ohms, 0.25W) is selected to develop around 3.8V peak across R39 and around 2.5V across capacitor C36 after bridge rectifier BR1.

Considering a voltage drop of 1.25V across the LED of the optocoupler, resistor R36 (120 ohms) is used to limit the current to about 10 mA through the optocoupler LED. The collector of the optocoupler is in low state with the engine/alternator running.

The output of the alternator serves as one of the two inputs to NAND gate IC2B, and is also used for energising 'Run' relay RL1 via NAND gate IC2D (wired as an inverter) and driver ULN2004A (IC8).

On energisation, RL1 contacts extend +12V to 'on' contact of the keyswitch via pin 5 of TB-1. As a result, the hour meter of the DG set starts running and also the dynamo output becomes available for charging the battery of the DG set. The dynamo IND terminal brought to pin 3 of TB-1 goes high (about +14.5V). The output from pin 3 of TB-1 is inverted by NAND gate IC2A and extended to NAND gate IC2B, which acts as a NOR gate (negative logic). Thus when the engine is running, output pin 4 of IC2B is high. Else, it is low.

Two alternate signals have been

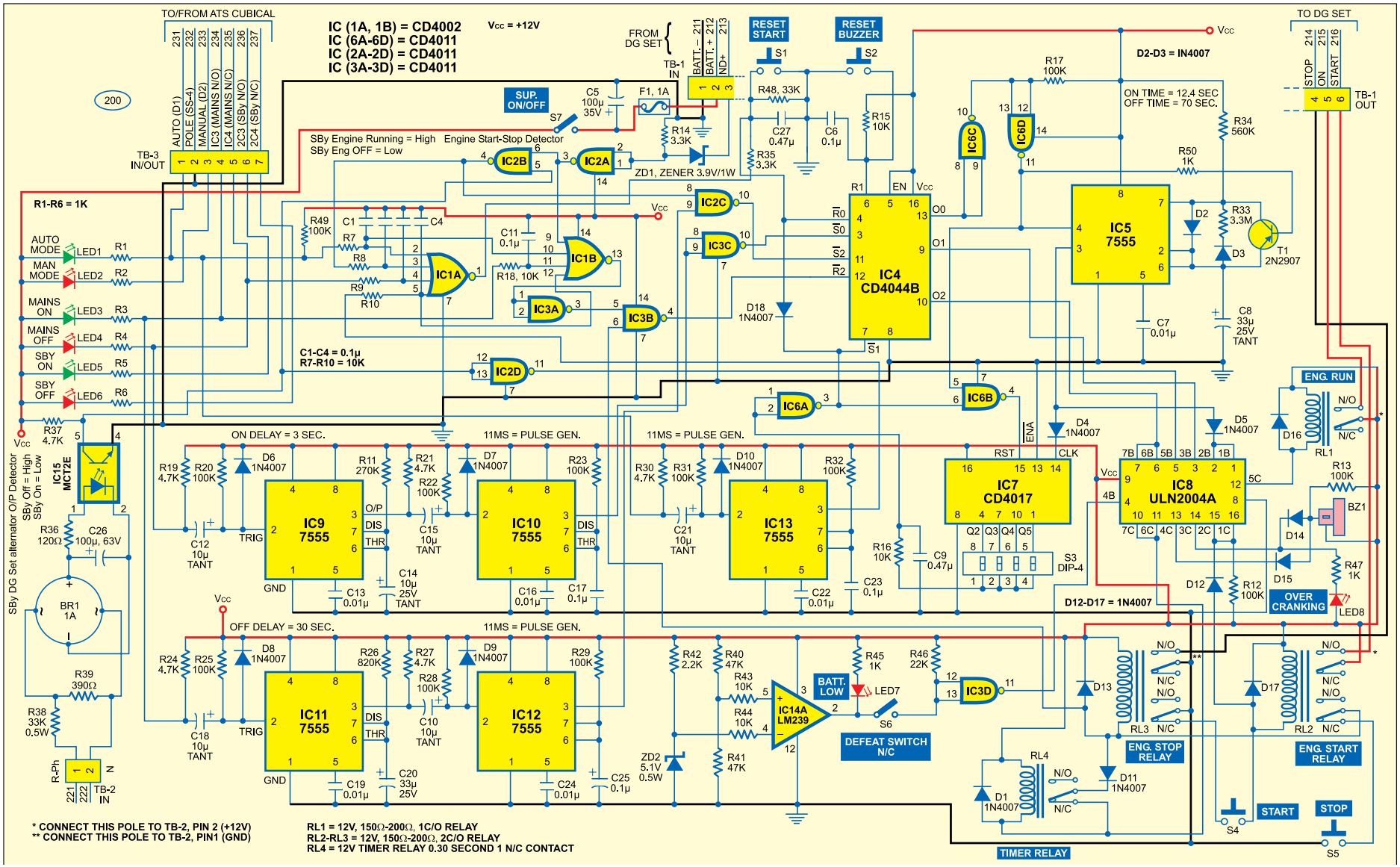


Fig. 12: The logic control circuit

used to indicate the status of the DG set so that if one of the two signals (alternator output or dynamo output) gets disconnected, the other one is available. The output of IC2B is used as the sense signal for 4-input NOR gate IC2A and as a gate (enable signal) for NAND gate IC2C.

IC CD4044B. Quad RS latch CD4044 (IC4) with 3-state outputs has been employed as the core IC for developing the engine start and stop logic. Therefore, before we examine implementation of the logic, let us have a look at the functional block diagram and truth table of CD4044B as shown in Fig. 13. We observe that with Eo pin tied to H (Vcc), the 'on' output will follow:

1. High logic state of Rn only when Sn input pin is held low. Subsequently, even if Sn input goes high, the output 'on' will continue to stay high (latched).

2. Low state of Rn input irrespective of Sn input. This implies that Rn input can be used to reset the 'on' output.

Implementation of the engine start logic. With mode-selector switch in 'auto' position, and mains tripping, we have all the four inputs of NOR gate IC1A low, since:

1. Pin 1 of TB-3 is low when mode switch is in 'auto' position. This is connected to pin 2 of IC1A.

2. When the DG set is not running, the output of NAND gate IC2B is low (since its both inputs are high). This is connected to pin 3 of IC1A.

3. Pin 5 of TB-3 goes from high (when mains is available) to low (when mains becomes unavailable). This is connected to pin 4 of IC1A.

4. The output of monostable IC3 is low in steady state. This is connected to pin 5 of IC1A.

Thus the output of IC1A at pin 1 goes high and the same is connected to R0 input pin 4 of CD4044B (IC4) via resistor R35. The resistor-capacitor combinations at the inputs of IC1A absorb any transients.

Transition detection and delay circuit. As stated earlier, pin 5 of TB-3 undergoes a high-to-low-to-high transition. This transition is detected by a differentiating network comprising capacitor C12, resistor R20 and diode D6. Resistor R19 is used for pulling pin 1 of TB-3 high before transition.

The detected negative-going pulse is coupled to trigger pin 2 of monostable IC9 employing CMOS timer IC 7555, which produces a 3-second wide pulse at its output pin 3. The pulse width can be changed by varying either the value of the resistor between Vcc and pin 7/6 or the capacitor between pin 7/6 and ground as per the following relationship:

Time (seconds) = 1.1R (ohms) × C (Farads).

The trailing end of this pulse triggers the next monostable (IC10), which is wired the same way as IC9 but produces a much shorter (10-12ms) low-to-high going pulse at its output. This pulse is inverted by NAND gate IC3C before application to S0 input (pin 3) of CD4044 (IC4). This results in transferring and latching of the high logic state of R0 (pin 4) to O0 output (pin 13), provided pin 4 continues to be held high even after the 3-second delay. That will be true as long as all the inputs of IC1A remain low, fulfilling the engine-start logic conditions.

Engine start circuit. The engine start circuit comprises IC5 through IC8, start control relay RL3 and high-current start relay RL7 (refer Fig. 10) in the DG set, and associated components. As soon as O0 output (pin 13) of IC4 is latched to high state, the same is buffered by IC6C and IC6D NAND gates (before application to reset pin 4 of IC5), one of the two inputs of NAND gate IC6B as well as the base of pnp switching transistor T1 (2N2907) via 1-kilo-ohm resistor R50.

CMOS timer IC5 is configured as an astable flip-flop controlled by the logic state of its reset pin 4. Its 'on' period is 12.4 seconds and 'off' period is 73 seconds. ('On' period=$0.67(560\times10^{3}\times33\times10^{-6})$ sec.≈12.4 sec. and 'off' period=$0.67\ (3.3\times10^{6}\times33\times10^{-6})$ sec.≈73 sec.) For as long as reset pin 4 is held high, transistor T1 remains cut-off, but once the reset pin goes low, timing capacitor C8 discharges rapidly to keep

PARTS LIST

Semiconductors:

IC1	-	CD4002 dual four-input NOR gate
IC2, IC3, IC6	-	CD4011 quad two-input NAND gate
IC4	-	CD4044 quad NAND R-S latch
IC5, IC9-IC13	-	7555 timer
IC7	-	CD4017 decade counter
IC8	-	ULN2004A relay driver
IC14	-	LM239 quad differential comparator
IC15	-	MCT2E optocoupler
T1	-	2N2907 pnp switching transistor
D1-D18	-	1N4007 rectifier diode,1A
D19-D22	-	1N5408 rectifier diode, 3A
BR1	-	1A bridge rectifier
LED1-LED8	-	Red/green LEDs
ZD1	-	3.9V, 1W zener diode
ZD2	-	5.1V, 0.5W zener diode

Resistors (all ¼-watt, ±5% carbon):

R1-R6, R45, R47, R50	-	1-kilo-ohm
R7-R10, R15, R16, R18, R43, R44	-	10-kilo-ohm
R11	-	270-kilo-ohm
R12, R13, R17, R20, R22, R23, R25, R28, R29, R31, R32, R49	-	100-kilo-ohm
R14, R35	-	3.3-kilo-ohm
R19, R21, R24, R27, R30, R37	-	4.7-kilo-ohm
R26	-	820-kilo-ohm
R33	-	3.3-mega-ohm
R34	-	560-kilo-ohm
R36	-	120-ohm
R38	-	33-kilo-ohm, 0.5W
R39	-	390-ohm
R40, R41	-	47-kilo-ohm
R42	-	2.2-kilo-ohm
R46	-	22-kilo-ohm
R48	-	33-kilo-ohm

Capacitors:

C1-C4, C6, C11, C17, C23, C25	-	0.1µF ceramic disk
C5	-	100µF, 35V electrolytic
C7, C13, C16, C19, C22, C24	-	0.01µF ceramic disk
C8, C20	-	33µF, 25V tantalum
C9, C27	-	0.47µF polyester
C10, C12, C14, C15, C18, C21	-	10µF, 25V tantalum
C26	-	100µF, 63V electrolytic

Miscellaneous:

ATS	-	Havell's 100A ATS with enclosure (220V AC tripping)
VMRD2	-	Minilec VMR D2 single-phase preventer with U/V, O/V and 1C/O contacts (415V AC auxiliary supply)
Contactor	-	6/9A with 3 N/O and 1 N/C contacts, 230V AC coil rating
S1, S2	-	Push-to-on tactile switches
S3	-	4-way DIP switch
S4, S5	-	Push-to-on switch for manual start and stop operations (BCH/Vaishno brand)
S6, S7	-	Toggle switch
BZ1	-	12V piezobuzzer
RL1	-	12V,150- to 200-ohm, 1C/O relay (OEN R series/PLA MPC series or equivalent)
RL2, RL3	-	12V, 150- to 200-ohm, 2C/O relay (OEN R series/PLA MPC series or equivalent)
RL4	-	12V, 0-30 sec. timer relay with N/C contacts
RL5, RL7	-	12V, 2C/O, 20-30A contact rated relay
RL6	-	12V, 25mm displacement solenoid puller
	-	Tag blocks, ferrules, spade/eye ends, connectors, multicolour cable, enclosure for logic control circuit, etc

the timer in complete reset state. In the absence of transistor T1, the output at its pin 3 diminishes very slowly inspite of holding the reset pin low.

The output from pin 3 of astable IC5 is coupled to input pins 1 and 2 of high-current Darlington driver array ULN2804 (IC8) as well as clock pin 14 of decade counter CD4017 (IC7) through isolating diodes D4 and D5, respectively.

During positive clock period of IC4, output pins 15 and 16 of ULN2004A (shorted for augmenting the driving current capacity of IC8 output) energise 'start' relay RL3 to extend +12V battery supply to high-current relay RL7 (in the DG set) via pin 6 of TB-1 to energise the starter. At the same time, the high output of CD4017 (IC7) shifts from Q0 (pin 3) to Q1 (pin 2).

If the DG set starts within 12.4 seconds, it will cause pin 3 of NOR gate IC1A to go high. This, in turn, will cause output pin 1 of the IC to go low to reset O0 output of CD4044 (IC4). The low output at pin 13 of IC4 will reset the astable flip-flop as well as decade counter IC7 via NAND gate IC6B. Also, timer capacitor C8 will quickly discharge into pin 7 of IC4 because of the conduction of transistor T1.

In case the engine doesn't start within the first 12.4 seconds, reset pin 4 of astable flip-flop IC5 stays high, while timing capacitor C8 starts slowly discharging through 3.3-mega-ohm resistor R33 into pin 7 of IC4.

After 73 seconds, output pin 3 of IC4 again goes high for another 12.4 seconds, provided the DIP switch part connected to pin 4 (Q2) of IC7 is not closed. If the switch is closed, the high output at pin 4 of IC7, after passing through the pulse stretcher network comprising resistor R16 and capacitor C9 and inversion by NAND gate IC6A, will cause pin 4 (R0 input) of IC4 to reset its output pin 13 and thus the astable flip-flop (IC5) as well as decade counter IC7 will also be reset.

The negative pulse at the output of NAND IC3A is also directly connected to S1 input (pin 7) of IC4, while its R1 (pin 6) is pulled high through resistor R15. This

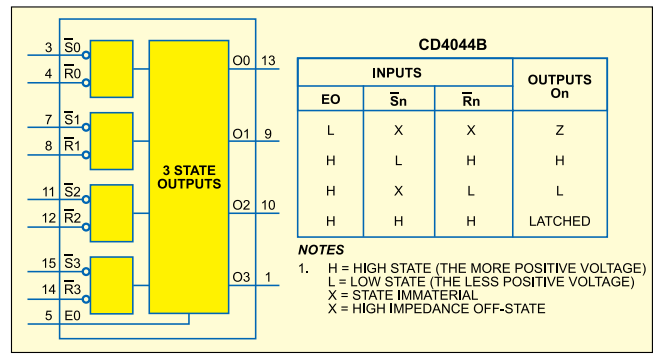


Fig. 13: Functional block diagram and truth table of CD4044B

causes latching of O1 output (pin 9) of IC4 to high state and, in turn, activation of cranking indicator LED8 as well as buzzer BZ1 via ULN2004A. Once the operator has noted the warning, he can reset the LED and the buzzer by pressing 'Reset Buzzer' tactile switch S2, which causes resetting of O1 output (pin 9) to low state.

In case you want to allow up to three cranking attempts, you need to flip the third switch of DIP switch S3 to 'on' position. The audio-visual warning will not activate if the engine starts within the preset attempts.

Implementation of the engine stop logic. Quad NOR gate IC1B is used for detecting some of the stop conditions.

The first conditions of start and stop logic are identical, i.e., the mode-selector switch in the ATS cubical must be in 'auto' position. Therefore pin 2 of IC1A is shorted to pin 10 of IC1B.

The second condition is that the DG set is already running. When the DG set is running, pin 3 of TB-1 is at around +14.5 V and hence the output of NAND IC2A at its pin 3 goes low, which is connected to pin 9 of quad NOR gate IC1B.

The third condition is the availability of healthy mains and its transition from 'not-available' to 'available' status. When mains becomes available, pin 2 of TB-3 goes from 'high' to 'low' (mains 'on' LED4 also lights up). This low output is extended to pin 11 of IC1B via R18.

Once these three conditions are satisfied, the output of IC1B goes high. This high output is applied to R2 input (pin 12) of CD4044 (IC4) after ORing it with another input from the timer circuit. The ORing function is realised using NAND gates IC3A and IC3B.

The mains 'off'-to-'on' transition (occurring at pin 2 of TB-3) detection and 30-second delay for allowing the DG set to cool down (off-load running) are accomplished by the edge-detection and delay mono circuit comprising IC11 (7555).

After 30 seconds, the trailing edge of the output of IC11 triggers IC12 to produce an 11ms pulse, which is passed to pin 9 of NAND gate IC3C, while its pin 10 is enabled (gated) by the engine start/stop sensor output of IC3B. Thus S2 input (pin 11) of CD4044 receives a low-going pulse only if the engine is still running. As a result, O2 output of CD4044 is latched high. This high output is applied to pins 6 and 7 (tied together) of ULN2004A and its output pins 10 and 11 go low to energise timer relay RL4 as well as 'stop' control relay RL3 (via NC contacts of timer relay).

The timer is set for 12 to 15 seconds. This time duration is appropriate for bringing the engine to a dead stop. Energisation of RL3 extends ground for activation of high-current 'stop' relay RL5 in the DG set, which, in turn, activates 'stop' solenoid RL6 in the DG set for the set period.

As mentioned earlier, the engine must come to a dead stop before releasing the engine-stop solenoid. For this, we have to ensure that O2 output at pin 10 of IC4 (CD4044) remains latched (to 'high' state) for the preset delay of the timer. Therefore the low output from pins 10 and 11 of ULN2004 (via N/C contacts of timer relay and diode D11) is applied to pin 6 of NAND gate IC3B. As a result, the output of IC3B or R2 input (pin 12) of CD4044 goes high for the preset period of timer relay RL4.

After the set period is over, the N/C contacts of the timer relay open up and Vcc (via the coil of relay RL3) is extended

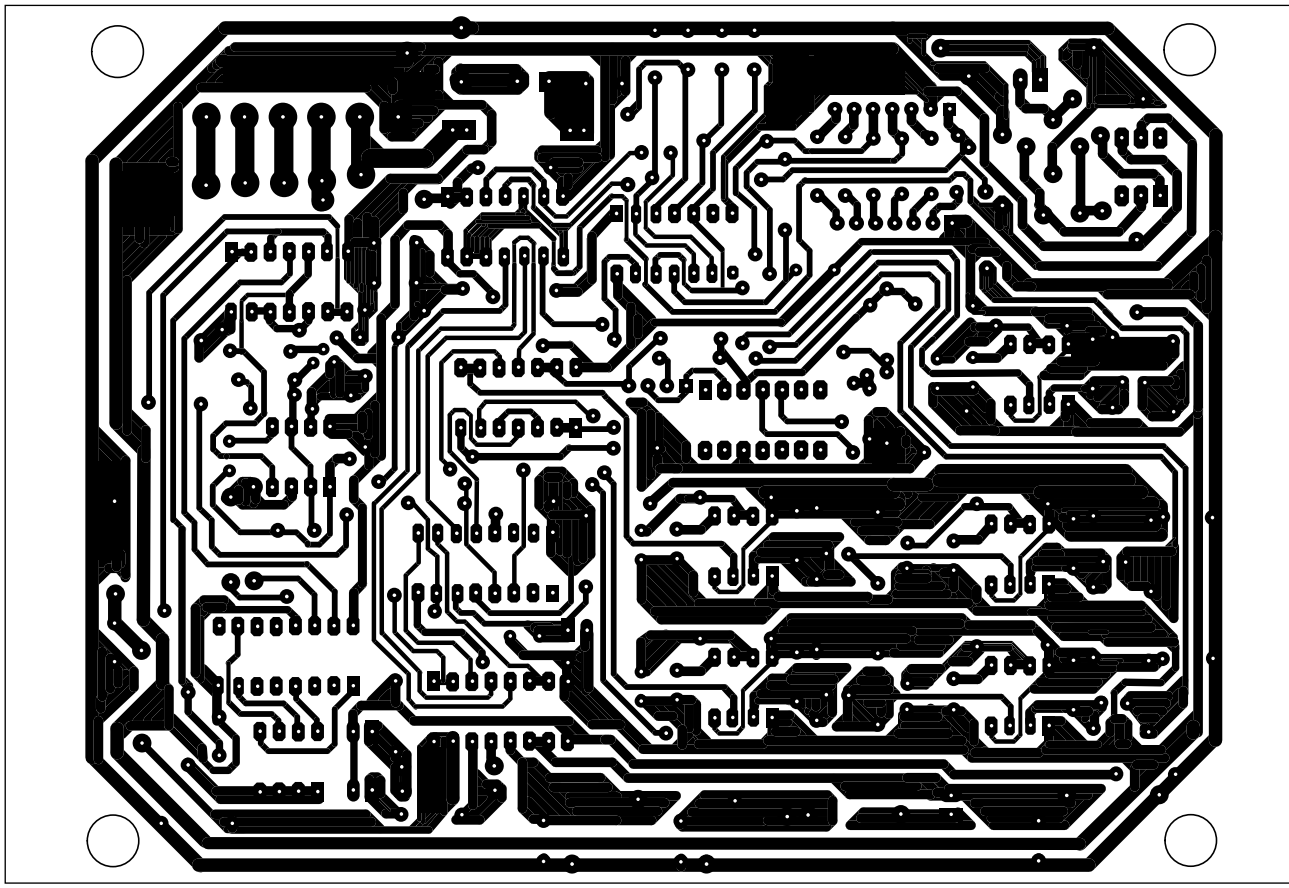


Fig. 14: Single-side, actual-size PCB for the logic control circuit

to pin 6 of IC3B, while pin 5 also goes high since, with the DG set in 'off' condition, pin 9 of quad NOR gate goes high and its output pin 13 goes low. The same output after inversion by NAND gate IC3A is applied to pin 5 of IC3B. As a result, the output of IC3B goes low to reset O2 output of CD4044 and ' engine stop' control signal comes to an end.

Battery-low warning circuit. When the battery voltage falls below 10V, the auto start/stop circuit may not function satisfactorily. It is a sure indication of one or more of the following conditions:

1. Charging circuit is non-functional.
2. The battery is not holding charge due to sulphation or the electrolyte needs to be topped up.
3. Dynamo pulley belt is slipping.
4. The battery has some load when the engine/dynamo is not 'on.'

The warning circuit comprises comparator IC14A (LM239), which compares a 50 per cent sample of the battery voltage against the 5V reference voltage developed across zener ZD2. The output of the comparator is high as long as the battery voltage exceeds 10V.

When the battery voltage falls below 10V, the comparator output goes low to light LED7. Switch S6 is normally kept closed. As a result, when the battery voltage goes below 10V, the output of NAND gate IC3D goes high to sound piezobuzzer BZ1 via ULN2004A and diode D15, which acts as an OR gate here. (In case you need to activate a high-power siren, in place of the buzzer, you can use a 12V relay to connect supply to the siren.) Once the operator has taken cognizance of the warning, he may turn off the buzzer using 'buzzer defeat' switch S6.

Manual start/stop operation. Push-button switches S4 and S5 in conjunction with relays RL2 (start control) and RL3 (stop control) are used to manually control the DG set when the mode switch in the ATS is kept in 'manual' mode. For starting the DG set, 'start' button is kept pressed and released as soon as the engine picks up speed.

Start-control relay energises via the de-energised contacts of stop-control relay RL3. Similarly, energisation of stop-control relay RL3 is possible via the de-energised contacts of relay RL2. This provides a safety against any erroneous pressing of a button while the automatic mode is in operation, and avoids simultaneous operation of 'start' and 'stop' solenoids.

The rest of 'start'/'stop' operation is similar to that of the automatic operation, with the exception of timer relay RL4, which is bypassed during the manual stop operation. ***(Caution:*** Don't release 'Stop' pushbutton until the engine comes to a dead stop. Else, the engine will tend to restart because of its flywheel inertia.)

'Reset start' tactile switch S1 has been provided for resetting the start operation prematurely during circuit testing. For detecting the manual-to-auto mode transition at the ATS, transition-detection mono IC13 has been

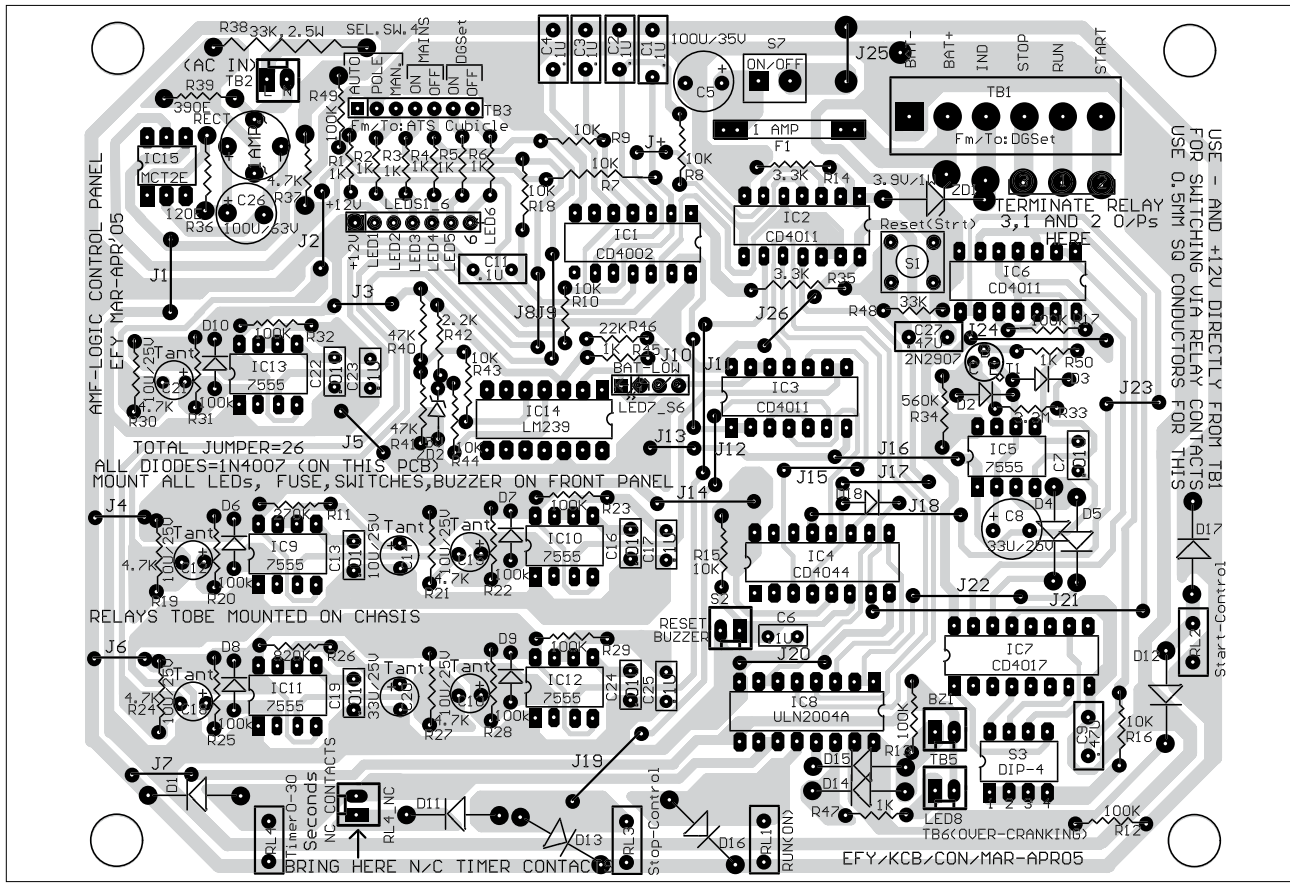


Fig. 15: Component layout for the PCB

used, which provides a reset pulse for start and stop latches of CD4044 (IC4).

PCB and assembly

A single-side, actual-size PCB for the logic circuit (Fig. 12) is shown in Fig. 14 and its component layout in Fig. 15. The front panel of the proposed enclosure for the logic control PCB including all the components (to be mounted externally on its chassis) is shown in Fig. 16.

All the status LEDs, buzzer and switches shown in Fig. 12 (except tactile switch S1) are mounted on the front panel. Suitable Bergstick connectors (male/females) should be used for extending the connections for the LEDs, switches and the buzzer. For the purpose, SIP connectors have been provided on the PCB.

All the cable entries to various TBs should be made from the rear. Use a 1-sq.mm flexible conductor wire for supplying the DC voltage from the DG set to the logic control panel. For the remaining external control signals, a 0.25-sq.mm flexible wiring will suffice. Cartridge fuse holder for fuse F1 and toggle switch S7 may also be mounted on the rear panel, close to TB1.

Mount all the relays (RL1 through RL4) horizontally on the chassis of the panel. DC supply (±12V) to the relay contacts is provided directly from TB-1 terminals using 0.5mm^2 flexible wires. Mark all the external wires using ferrules as shown in the interconnection diagram.

Testing

Test each part of the circuit elaborately by making use of the status-indication lamps and logic explained for various parts of the circuit without extending connections from TB-1 terminals 4, 5 and 6 to the DG set. Use 12V lamps (of 2 to 3 watts) or even LEDs connected to these terminals for testing the circuit operation before connecting them to the corresponding terminals of the DG set. As far as connections to the ATS via TB-2 are concerned, we are merely carrying negative DC supply via its pin 2 for return via other pins of TB-2. Hence there is no danger unless you mix up the wires with 415/230 AC voltages present in the ATS cubicle.

Things to remember

1. Many a times, the power supply (mains 3-phase) maintenance personnel inadvertently interchange the phases during reconnection after servicing. The resulting phase reversal can have very serious repercussions in organisations employing 3-phase motor operated machinery due to reverse rotation of the motors.

The Minilec phase-reversal prevention relay (VMR D2) installed in the ATS cubicle senses this reversal and trips to prevent the control voltage (red phase) from exercising automatic changeover control. Tripping is indicated by an LED on the VMR D2 relay.

Tripping can also be caused by single-phasing or under-/over-voltage of

the 3-phase supply. Hence check that all the three phase voltages are within limits. Once you are satisfied that 3-phase voltages are alright, the most probable cause is loss of phase sequence. If you have a phase-sequence indication meter, you can verify the same.

To correct the phase sequence, first isolate mains and interchange any two of its phase wires going to the ATS—preferably, Y- and B-phase wires since the red-phase wire is used for control in the ATS. In case you are not using any phase-sensitive load, simply interchange the Y and B phase wires on the Minilec relay itself, ensuring (in both cases) that the trip LED goes off when all the three phases of the supply are available. This will enable automatic changeover when mains is available.

2. After any maintenance of the DG set, ensure that all the phase wires are connected correctly. Else, changeover to the DG set supply will not take place for the reasons explained above. Also make sure that the dynamo terminals are properly plugged and battery charging is being indicated by the ammeter on the DG set instrument panel when the engine is running and the keyswitch is in 'on' position. If the filament of the bulb in the charging path is open-circuited, then also the battery charging will not take place.

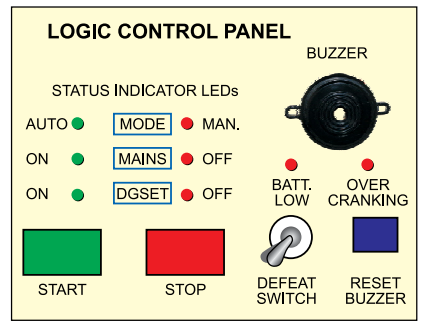


Fig. 16: Front panel of the proposed enclosure for assembling the logic control PCB

3. Use good-quality contactors, relays and timers from companies like L&T, BCH, Siemens, Minilec, Havell, English Electric, PLA, OEN or Omron.

Conclusions

Users may modify the circuits appropriately for meeting the specific requirements of their standby supply sources. It is also possible to reduce the AMF control panel circuitry to just the use of peripheral components by employing a microcontroller to implement the basic logic. So good luck!

PC-BASED SCROLLING MESSAGE DISPLAY

■ SURESH KUMAR

Controlling electronic devices from a PC is fun. Here is a scrolling message display that makes use of the PC's parallel port. The message typed from the keyboard of the PC is displayed on the light-emitting diodes arranged as 5×7 dot-matrix display in moving message format.

LED-based scrolling message displays are increasingly being used at railway stations, public places, colleges, universities, hospitals, general stores, etc for disseminating information. However, most displays lack in storage capacity and cannot display a large number of characters at a time.

This PC-based LED scrolling message display has the following features:

1. The message to be displayed is stored in a file and the message length to be displayed is limited only by free memory space on the hard disk of the computer.

2. The number of characters displayed at a time can be as high as 30.

3. The message stored in the file can be changed using any text editor including Notepad.

4. The running speed of the message displayed can be increased or decreased by pressing a few keys.

Here, the circuit is designed for displaying English characters on a 35 (5×7) LED dot-matrix display.

The PC's parallel port (LPT port) is used to output the display code and the clock signal for the scrolling message display.

The parallel port is terminated into a 25-pin D-type female connector at the back of the PC. IBM PCs usually come with one or two LPT ports. Each parallel port is actually made up of three ports, namely, data port, status port and control port. Here, only data port is used for this scrolling message display.

PARTS LIST

Semiconductors:
IC1 - 7805C 5V regulator
IC2-IC8 - 74174 hex D-type flip-flop
D1-D4 - 1N4007 rectifier diode
LED1-LED42 - Red LED

Resistors (all ¼-watt, ±5% carbon):
R1- R42 - 150-ohm

Capacitors:
C1 - 470µF, 16V electrolytic

Miscellaneous:
X1 - 230V AC primary to 7.5V, 1A secondary transformer

Pins 2 through 9 form the 8-bit data output port. This is purely a write-only port, which means it can only output data. The base address of the first parallel port (LPT1) is '378H' or '888' (decimal).

Parallel-input parallel-output (PIPO) registers are used to shift the signal from right to left. The clock pulse and code signal are generated by the computer program and output from the parallel port (base address 0×378). Theoretically, we can add infinite number of PIPO registers but the maximum number of registers is actually limited to the current triggering value of the clock pulse. To add a large number of PIPO registers, amplify the clock pulse prior to connecting it to the PIPO ICs.

Circuit description

Fig. 1 shows the circuit for the scrolling message display. IC 74174 has been used as PIPO register, which comprises high-speed, hex D-type flip-flops. It is used as a 6-bit edge-triggered storage register. The data on the inputs of the flip-flop is transferred for storage during high-to-low transition of clock. Data lines D0 through D5 of the parallel port are connected to the input pins of the first flip-flop (IC2). The output of IC2 is fed to the next flip-flop IC input as well as LED. Data line D6 is fed to IC8, while data line D7 is connected to the clock inputs of IC2 through IC8. Clock pins of all the flip-flop ICs are connected together. Master reset pin 1 of all the flip-flops is connected to Vcc. Pins 18 through 25 of the parallel port are grounded. As data present on lines D0 through D6 shifts from the first stage to the next stage, and so on, the message appears as scrolling on the dot-matrix LED display.

The present circuit supports a display made of 42 LEDs comprising seven rows and six columns. Up to 30 such units can be added with no change in the circuit. However, to add these units, you need to amplify the clock pulse output. Note that each character is displayed in a matrix of 5 columns and 7 rows (explained later), hence the sixth-column LEDs form part of the next character (column 1).

Fig. 2 shows the power supply circuit. The AC mains is stepped down by transformer X1 to deliver a secondary output of 7.5V AC at 1A. The transformer output is rectified by a full-wave bridge rectifier comprising diodes D1 through D4, filtered by capacitor C1, then regulated by IC 7805C (IC1) to provide regulated 5V DC to the circuit.

An actual-size, single-side PCB for the circuits in Figs 1 and 2 is shown in Fig. 5 and its component layout in Fig. 6.

EFY note. Commercially 7×5 dot-matrix displays with discrete LEDs may not be easily available in the market, therefore a perforated board with holes for the LED leads may be used. The layout of such a board is shown in Fig. 7. The holes are used for passing the LED leads.

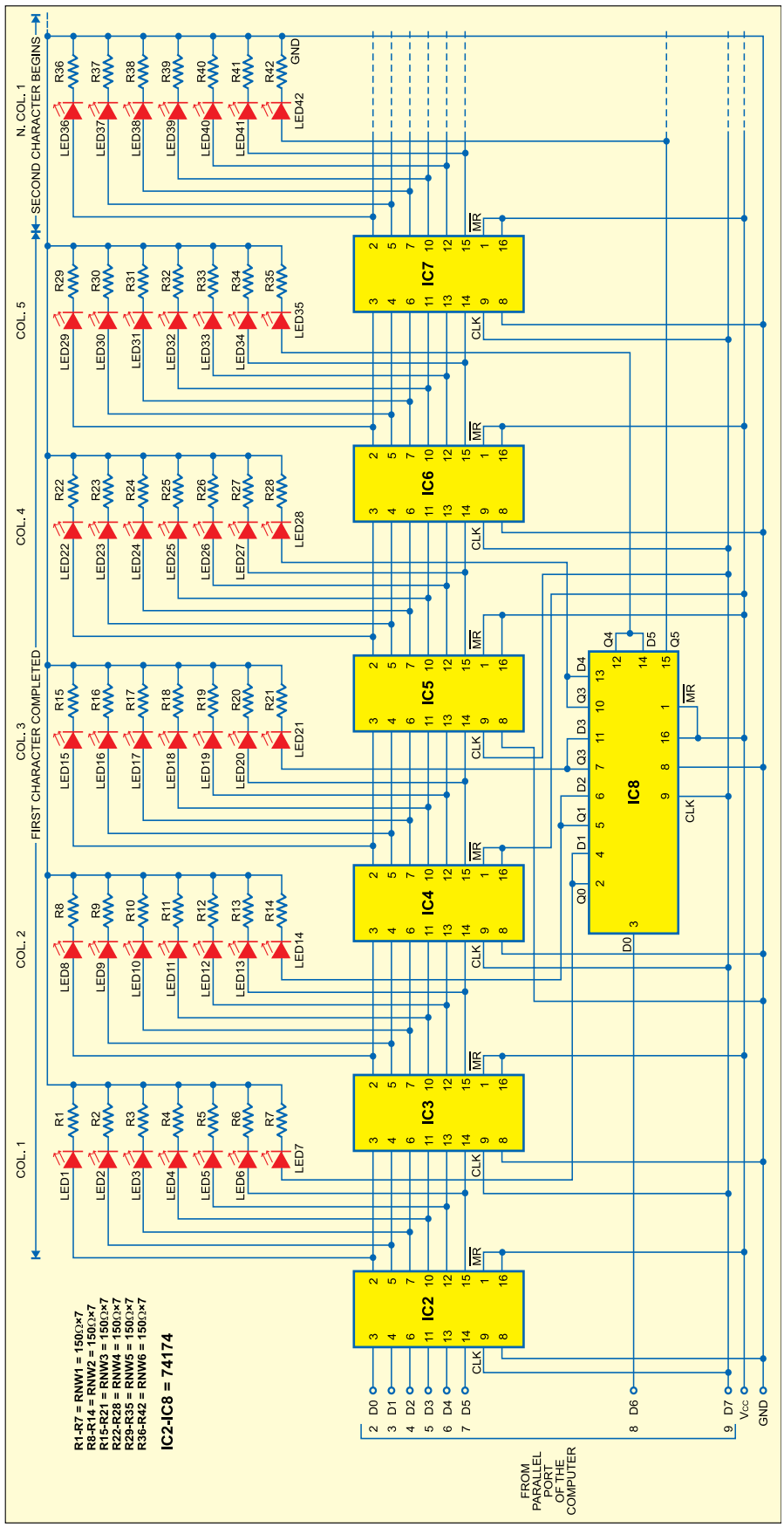


Fig. 1: Circuit of LED-based scrolling message display

The software

The software for the scrolling message display has been developed in 'C' language and compiled in 'Turbo C.' When you run the scroll.exe file, the program tries to open the message.txt file. If this file is not present in the same directory, it creates one with text "Welcome! You are watching running LED display..." and starts sending this message to the circuit via the parallel port for display on 5×7 dot-matrix pattern.

To increase the running speed of the message, press 'I' key, and to decrease the speed, press 'D' key. Press 'R' key for displaying the message from the beginning. When the program reaches the end of the message, it starts from the beginning again. To change the text being displayed, exit the program by pressing 'Esc' and edit the message.txt file using Notepad. After making changes to the message.txt file, save it and execute the scroll.exe file.

The program makes use of the outportb() function, which works perfectly only on Windows 95/98. However, the program may not work with the latest Window versions such as Windows 2000/XP .

When you try to save changes in the message.txt file, the window shows an error saying "Can't save message.txt. It is being used by some other application." This is because the scroll.exe file is running. So exit the program by pressing 'Esc' key, then save your changes made to the message. txt file and run the scroll.exe file. Now you can view your changes in the message being displayed.

The program does not show special characters like '/,' '\,'

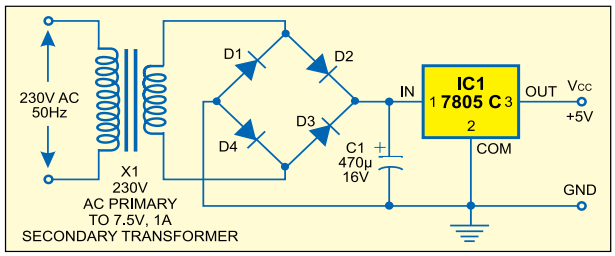


Fig. 2: Power supply

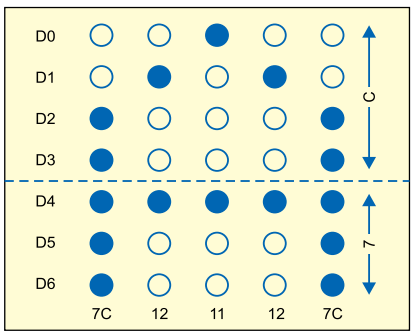


Fig. 3: Design of character 'A'

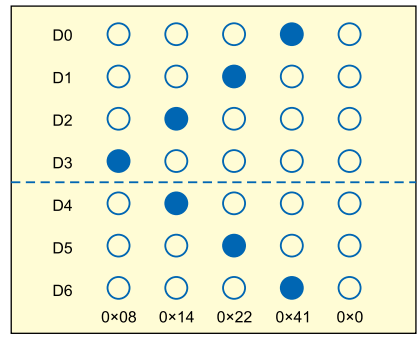


Fig. 4: Design of character '<'

'~,' '@,' '#,' '$,' '%,' '^,' '(,' '),' '{,' '},' and ';.' It has been developed for displaying alphabets ('A' through 'Z'), digits ('0' through '9') and some special characters like '.,' '„"!,' '-,' '+' and '_.'

Other special characters can be added as follows: Suppose you want to display character 'A.' Draw 'A' on the 5×7 LED display as shown in Fig. 3. First, '7CH' data is available at the input of IC2 and the first flip-flop of IC8. When a clock pulse is received, this data (7CH) is output by IC2 and the first flip-flop of IC8 and new data '12H' arrives at the input pin of IC2 and the first flip-flop of IC8. The output data of IC2 and the first

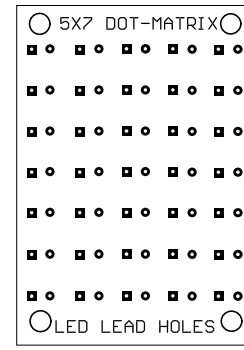


Fig. 7: Perforated board for 5×7 LEDs

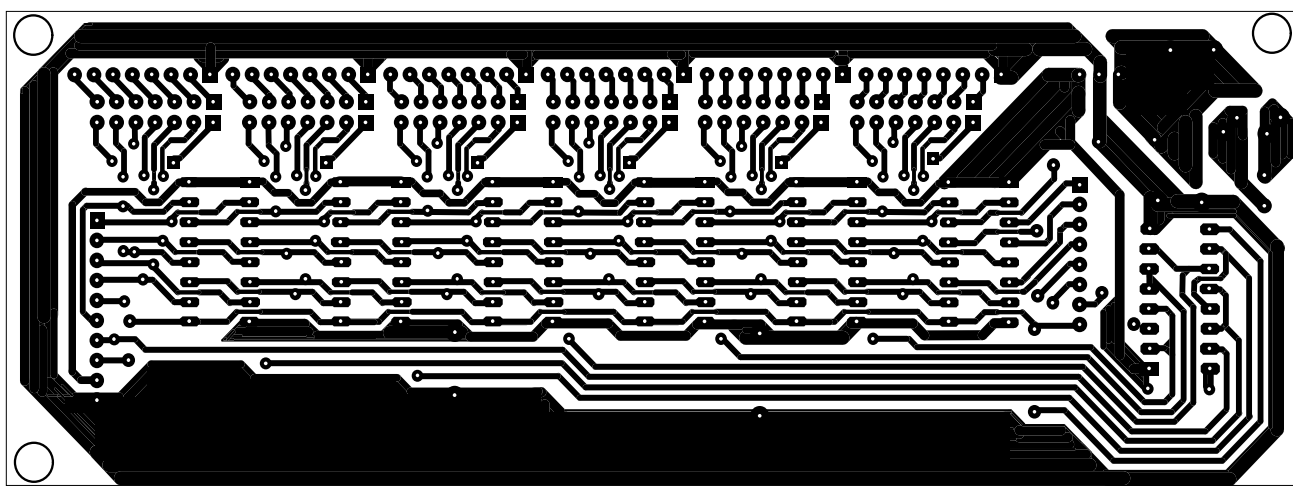


Fig. 5: Actual-size, single-side PCB for the LED-based scrolling message display including power supply

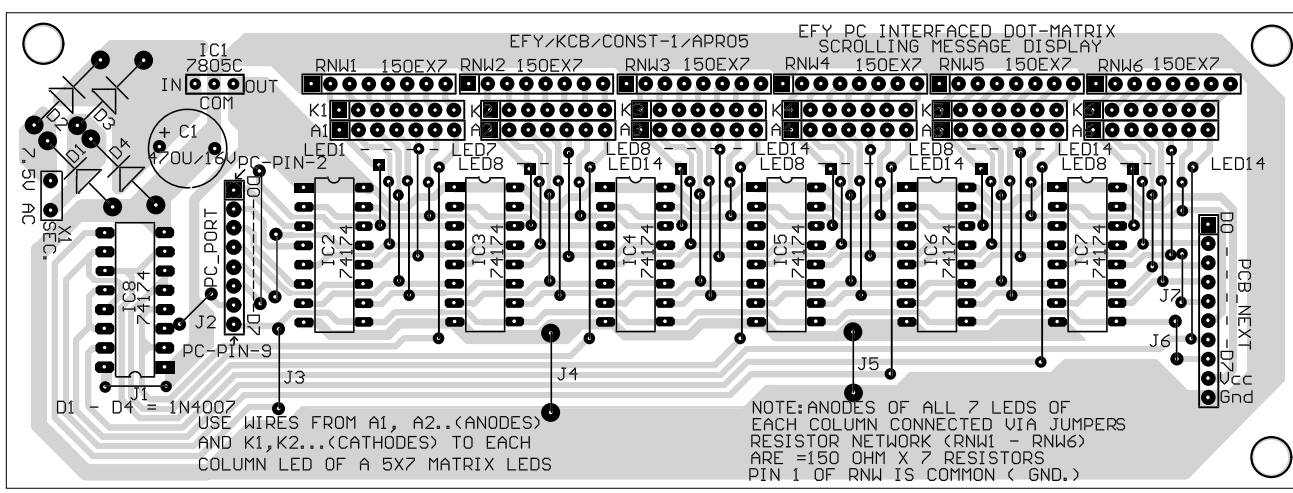


Fig. 6: Component layout for the PCB

flip-flop of IC8 becomes the input for IC3 and the second flip-flop of IC8. When the next clock pulse is received, '7CH' data becomes available at the output of IC3 and output of second flip-flop of IC8, '12H' is available at the output of IC2 and the first flip-flop of IC8 and new data '11H' is available at the input of IC2 and the first flip-flop of IC8. This process continues until the message completes.

Now let's assume that you want to display '<.' For this, first draw this symbol on the 5×7 matrix as shown in Fig. 4. Assuming glowing LED as '1,' convert the binary column sequence into hexadecimal for all the five columns as shown in the figure. Finally, add the following lines in the software program where the comment "Add your codes here" appears:

Case '<' :
str1[0]=0x00;str1[1]=0x41;str1[2]=0x22;str1[3]=0x14;str1[4]=0x8;
break;

Save the file and compile the program again. On executing the program, you can watch '<' being displayed on the message display.

Other special characters can be added in the same way.

Download source code: http://www.efymag.com/admin/issuepdf/PC%20Scroll%20Display.zip

SCROLL.C

```
/*************************************
SCROLLING MESSAGE DISPLAY
DEVELOPED BY : SURESH KUMAR
FINAL YEAR, IITT COLLEGE OF ENGINEERING, PUNJAB
THANX TO ALL TEACHERS AND MY PARENTS
*************************************/
#include<stdio.h>
#include<dos.h>
#include<conio.h>
#include<process.h>
unsigned char str1[5],str2[13],str[5];
int DELAY=100;
void setcode();
void sendcode();
void getcode(char);
void main()
{
        FILE *fp;
        char line[150],ch;
        clrscr();
        fp=fopen("message.txt","r");
        if(fp==NULL)
        {
                fp=fopen("message.txt","w");
                if(fp==NULL)
                {
                        printf("\n\nCAN'T CREATE MESSAGE.TXT CREATE A FILE UNDER NAME MESSAGE.TXT YOURSELF");
                        exit(0);
                }
                fputs(" Welcome! You are watching running LED display... ",fp);
                fclose(fp);
                fp=fopen("message.txt","r");
                if(fp==NULL)
                {
                        printf("\nCAN'T FIND OR OPEN \"message.txt\"");
                        exit(0);
                }
        }
        clrscr();
        startagain:
        while(!kbhit())
        {
                ch=fgetc(fp);
                if(ch==EOF)
                {
                        rewind(fp);
                        continue;
                }
                printf("\nSCROLLING MESSAGE DISPLAY : Sending \'%c\'",ch);
                getcode(ch);
                setcode();
                sendcode();
        }
        ch=getch();
        switch(ch)
        {
                case 'i':
                case 'I':
                if(DELAY>10)
                {
                        DELAY-=5;
                }
                else
                {
                        DELAY-=1;
                }
                if(DELAY<0)
                {
                        DELAY=0;
                }
                printf("\nSCROLLING MESSAGE DISPLAY : Speed Increased");
                break;
                case 'd':
                case 'D':
                DELAY+=10;
                printf("\nSCROLLING MESSAGE DISPLAY : Speed Decreased");
                break;
                case 'r':
                case 'R':
                rewind(fp);
                printf("\nSCROLLING MESSAGE DISPLAY : Started from Begining");
                break;
                case 27:
                clrscr();
                printf("\nSCROLLING MESSAGE DISPLAY : Exiting ");
                fclose(fp);
                delay(1000);
                printf(". ");
                delay(200);
                printf(". ");
                delay(200);
                printf(". ");
                delay(200);
                printf(". ");
                delay(200);
                exit(0);
        }
        goto startagain;
}
void getcode(char ch)
{
        switch(ch)
        {
                case 'a':
                case 'A':
                str1[0]=0x7c;str1[1]=0x12; str1[2]=0x11;str1[3]=0x12;str1[4]=0x7c;
                break;
                case 'b':
                case 'B':
                str1[0]=0x36;str1[1]=0x49; str1[2]=0x49;str1[3]=0x49;str1[4]=0x7F;
                break;
                case 'c':
                case 'C':
                str1[0]=0x22;str1[1]=0x41; str1[2]=0x41;str1[3]=0x41;str1[4]=0x3C;
                break;
                case 'd':
                case 'D':
                str1[0]=0x1C;str1[1]=0x22;str1[2]=0x41; str1[3]=0x41;str1[4]=0x7F;
                break;
                case 'e':
                case 'E':
                str1[0]=0x41;str1[1]=0x41;str1[2]=0x49; str1[3]=0x49;str1[4]=0x7F;
                break;
                case 'f':
                case 'F':
                str1[0]=0x01;str1[1]=0x01;str1[2]=0x09; str1[3]=0x09;str1[4]=0x7F;
                break;
                case 'g':
                case 'G':
                str1[0]=0x3A;str1[1]=0x49;str1[2]=0x41; str1[3]=0x41;str1[4]=0x3E;
                break;
                case 'h':
                case 'H':
                str1[0]=0x7F;str1[1]=0x08;str1[2]=0x08; str1[3]=0x08;str1[4]=0x7F;
                break;
                case 'i':
                case 'I':
                str1[0]=0x41;str1[1]=0x41;str1[2]=0x7F; str1[3]=0x41;str1[4]=0x41;
                break;
                case 'j':
                case 'J':
                str1[0]=0x7F;str1[1]=0x41;str1[2]=0x41; str1[3]=0x41;str1[4]=0x21;
                break;
                case 'k':
                case 'K':
                str1[0]=0x41;str1[1]=0x22;str1[2]=0x14; str1[3]=0x08;str1[4]=0x7F;
                break;
                case 'l':
                case 'L':
                str1[0]=0x40;str1[1]=0x40;str1[2]=0x40; str1[3]=0x40;str1[4]=0x7F;
                break;
                case 'm':
                case 'M':
                str1[0]=0x7F;str1[1]=0x02;str1[2]=0x04; str1[3]=0x02;str1[4]=0x7F;
                break;
                case 'n':
                case 'N':
                str1[0]=0x7F;str1[1]=0x08;str1[2]=0x04; str1[3]=0x02;str1[4]=0x7F;
                break;
                case 'o':
                case 'O':
                str1[0]=0x3E;str1[1]=0x41;str1[2]=0x41;  str1[3]=0x41;str1[4]=0x3E;
                break;
                case 'p':
                case 'P':
                str1[0]=0x06;str1[1]=0x09; str1[2]=0x09;str1[3]=0x09;str1[4]=0x7F;
                break;
                case 'q':
                case 'Q':
                str1[0]=0x3E;str1[1]=0x61; str1[2]=0x51;str1[3]=0x41;str1[4]=0x3E;
                break;
                case 'r':
                case 'R':
                str1[0]=0x46;str1[1]=0x29; str1[2]=0x19;str1[3]=0x09;str1[4]=0x7F;
                break;
                case 's':
                case 'S':
                str1[0]=0x32;str1[1]=0x49; str1[2]=0x49;str1[3]=0x49;str1[4]=0x26;
                break;
                case 't':
                case 'T':
                str1[0]=0x01;str1[1]=0x01; str1[2]=0x7F;str1[3]=0x01;str1[4]=0x01;
                break;
                case 'u':
                case 'U':
                str1[0]=0x3F;str1[1]=0x40; st
```

```
r1[2]=0x40;str1[3]=0x40;str1[4]=0x3F;
                    break;
                    case 'v':
                    case 'V':
                    str1[0]=0x1F;str1[1]=0x20;
str1[2]=0x40;str1[3]=0x20;str1[4]=0x1F;
                    break;
                    case 'w':
                    case 'W':
                    str1[0]=0x7F;str1[1]=0x20;
str1[2]=0x10;str1[3]=0x20;str1[4]=0x7F;
                    break;
                    case 'x':
                    case 'X':
                    str1[0]=0x63;str1[1]=0x14;
str1[2]=0x08;str1[3]=0x14;str1[4]=0x63;
                    break;
                    case 'y':
                    case 'Y':
                    str1[0]=0x03;str1[1]=0x04;
str1[2]=0x78;str1[3]=0x04;str1[4]=0x03;
                    break;
                    case 'z':
                    case 'Z':
                    str1[0]=0x03;str1[1]=0x04;
str1[2]=0x08;str1[3]=0x11;str1[4]=0x61;
                    break;
                    case '0':
                    str1[0]=0x1C;str1[1]=0x22
;str1[2]=0x41; str1[3]=0x22;str1[4]=0x1C;
                    break;
                    case '1':
                    str1[0]=0x40;str1[1]=0x40;
str1[2]=0x7F;str1[3]=0x42;str1[4]=0x44;
                    break;
                    case '2':
                    str1[0]=0x46;str1[1]=0x49;
str1[2]=0x51;str1[3]=0x61;str1[4]=0x42;
                    break;
                    case '3':
                    str1[0]=0x31;str1[1]=0x4B; st
r1[2]=0x45;str1[3]=0x49;str1[4]=0x21;
                    break;
                    case '4':
                    str1[0]=0x10;str1[1]=0x7F; st
r1[2]=0x12;str1[3]=0x14;str1[4]=0x18;
                    break;
                    case '5':
                    str1[0]=0x31;str1[1]=0x49; st
r1[2]=0x49;str1[3]=0x49;str1[4]=0x27;
                    break;
                    case '6':
                    str1[0]=0x32;str1[1]=0x49; st
r1[2]=0x49;str1[3]=0x51;str1[4]=0x3A;
                    break;
                    case '7':
                    str1[0]=0x07;str1[1]=0x79; st
r1[2]=0x01;str1[3]=0x01;str1[4]=0x01;
                    break;
                    case '8':
                    str1[0]=0x36;str1[1]=0x49; st
r1[2]=0x49;str1[3]=0x49;str1[4]=0x36;
                    break;
                    case '9':
                    str1[0]=0x3E;str1[1]=0x49; st
r1[2]=0x49;str1[3]=0x49;str1[4]=0x26;
                    break;
                    case '.':
                    str1[0]=0x60;str1[1]=0x60; st
r1[2]=0x00;str1[3]=0x00;str1[4]=0x00;
                    break;
                    case ' ':
                    str1[0]=0x00;str1[1]=0x00; st
r1[2]=0x00;str1[3]=0x00;str1[4]=0x00;
                    break;
                    case '!':
                    str1[0]=0x67;str1[1]=0x7F; st
r1[2]=0x00;str1[3]=0x00;str1[4]=0x00;
                    break;
                    case '-':
                    str1[0]=0x08;str1[1]=0x08; st
r1[2]=0x08;str1[3]=0x08;str1[4]=0x08;
                    break;
                    case '+':
                    str1[0]=0x08;str1[1]=0x08; st
r1[2]=0x3E;str1[3]=0x08;str1[4]=0x08;
                    break;
                    case '_':
                    str1[0]=0x40;str1[1]=0x40;st
r1[2]=0x40; str1[3]=0x40;str1[4]=0x40;
                    break;
 ////// ADD YOUR CODES HERE////////
                    default:
                    str1[0]=0x0;str1[1]=0x0;str1
[2]=0x0; str1[3]=0x0;str1[4]=0x0;
                    break;
            }
 }
 void setcode()
 {
            int i,k;
            for(i=0,k=0;i<10;i+=2,k++)
            {
                    str2[i]=str1[k];
                    str2[i+1]=str1[k]+128;
            }
                    str2[i]=0;
                    str2[i+1]=128;
 }
 void sendcode()
 {
            int i;
            for(i=0;i<12;i++)
            {
                    outportb(0x0378,str2[i]);
                    delay(DELAY);
            }
 }
```

LOW-COST ENERGY METER USING ADE7757

■ SUNIL KUMAR

Electromechanical energy meters have been the standard for metering the electricity since billing began. But these are now being gradually replaced by digital signal processor (DSP)-based energy meters, or kilowatt-hour (kWh) meters.

More accurate energy measurement and additional features are in fact accelerating the adoption of DSP-based meters. Their additional features include power quality monitoring, recording of current/voltage peaks and voltage sags, registering of digitised waveforms for analysis, and monitoring of active and reactive power and power factor information.

Some metering chips have a serial port interface (SPI) that can be used for establishing communication with a microcontroller-based mobile gadget to control the functionality of the metering chip, perform calibration and transfer the recorded data. Analog Devices offers an extensive range of metering ICs to serve various needs.

Here's an energy meter using Analog Devices' ADE7757 chip for single-phase, 2-wire (phase and neutral) systems used in households. IC ADE7757 is a low-cost, single-chip solution for electrical energy measurement.

The meter is designed based on Analog Devices' application notes. Its salient features are:

1. It can read up to 999,999 units (kWh) with a resolution of 0.01 unit.

2. It is designed for nominal 230V AC, 45-65 Hz and maximum line current of 30 amps. (The metering IC can be used with a maximum current of 120 amps.)

3. The dynamic range is 400 (i.e., 75 mA to 30A).

4. The meter count is 100 impulses/kWh, i.e., 100 impulses will be required to register one unit.

5. The accuracy level is better than Class 2 defined in international standard IEC1036 (1996-09). The maximum error limit for various current values as per this standard is shown in Table I.

IC ADE7757

Fig. 1 shows the functional block diagram of metering IC ADE7757. It is available in 16-lead SOIC narrow-body package. In our PCB layout, it is to be soldered on the conductor side of the PCB. The IC has an on-chip oscillator, so it requires no external crystal or resonator, thus reducing the overall cost of building a watt-hour meter. It operates off a 5V power supply.

In operation, the chip directly interfaces with a shunt resistor (used as the current sensor) and AC analogue voltage sensing input. It has two analogue input channels designated as V1 and V2, respectively. Channel V1 (also called 'current channel') is used for current sensing and channel V2 (also called 'voltage channel') is used for voltage sensing.

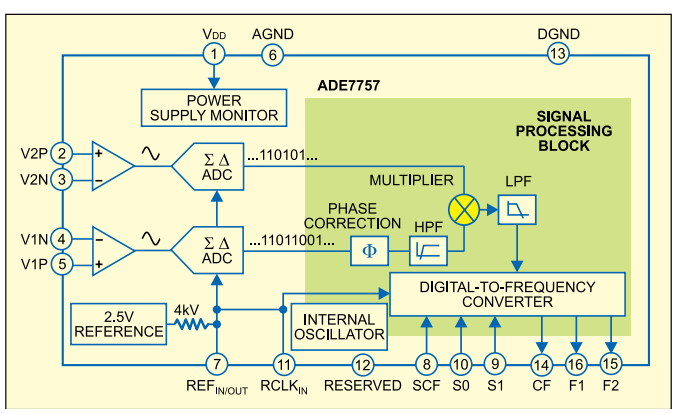


Fig. 1: Functional block diagram of IC ADE7757

PARTS LIST

Semiconductors:

IC1	- ADE7757 metering IC
IC2, IC5	- 7805 5V regulator
IC3, IC4	- MCT2E optocoupler
IC6, IC7	- MM74926 7-segment driver
T1-T8	- BC548 npn transistor
D1-D3	- 1N4007 rectifier diode
ZD1	- 15V, 1W zener diode
BR1	- W04M bridge rectifier
DIS1- DIS8	- LTS543 common cathode, 7-segment display
LED1	- Red LED

Resistors (all ¼-watt, ±5% carbon, unless mentioned otherwise):

R1, R3, R7, R8	- 500-ohm
R2	- 6.2-kilo-ohm
R4	- 470-ohm
R5, R6	- 680-ohm
R9	- 350-micro-ohm (shunt)
R10	- 1.8-mega-ohm
R11	- 2.2-kilo-ohm
R13	- 470-ohm, 1W
R14	- 1-kilo-ohm
R12, R15- R28	- 220-ohm
VR1	- 470-kilo-ohm trimpot

Capacitors:

C1, C3, C7, C8-C10	- 0.1µF ceramic disk
C2, C6	- 10µF, 25V electrolytic
C4, C5, C11, C12	- 0.068µF ceramic disk
C13	- 0.47µF, 630V polyester
C14	- 470µF, 35V electrolytic
C15	- 1000µF, 16V electrolytic

Miscellaneous:

L1, L2	- Ferrite bead inductor
Battery	- 4.5V rechargeable battery
X1	- 230V AC primary to 7.5V, 500mA secondary transformer

The differential output from the current-sensing resistor is connected between V1P and V1N inputs, while the differential output signal proportional to the AC line voltage, obtained through a resistor divider, is connected between pins V2P and V2N.

IC ADE7757 also has a reference

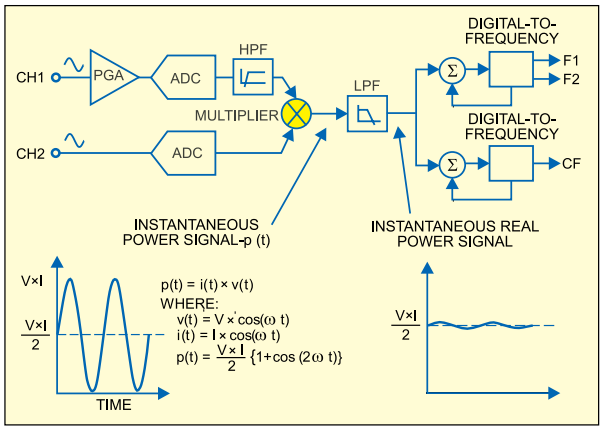


Fig. 2: Block diagram for signal processing along with the waveforms at the output of the multiplier and the low-pass filter (LPF)

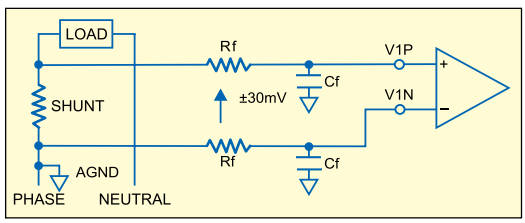


Fig. 3: Typical mains current sampling

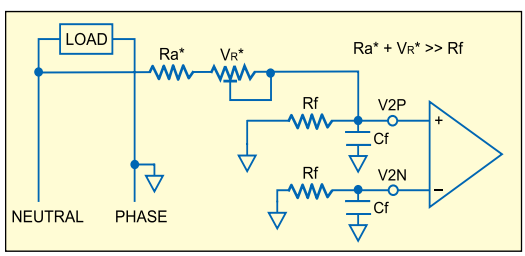


Fig. 4: Typical mains voltage sampling

TABLE I
Accuracy Requirements

Current value[1]	PF[2]	Percentage error limits[3] Class 1	Class 2
0.05 Ib < I < 0.1 Ib	1	±1.5%	±2.5%
0.1 Ib < I < I_{MAX}	1	±1.0%	±2.0%
0.1 Ib < I < 0.2 Ib	0.5 lag	±1.5%	±2.5%
	0.8 lead	±1.5%	
0.2 Ib < I < I_{MAX}	0.5 lag	±1.0%	±2.0%
	0.8 lead	±1.0%	

Notes. 1. The current ranges for the specified accuracy are expressed in terms of the basic current (Ib), which is defined as the value of the current in accordance with which the relevant performance of a direct connection meter is fixed. I_{MAX} is the maximum current at which the accuracy is maintained. At these frequencies we have taken the value of basic current Ib as 5 amp.

2. Power factor (PF) gives the phase relationship between the fundamental voltage (45 to 65 Hz) and current waveforms. Here, it can be simply defined as PF = cos f, where f is the phase angle between the pure sinusoidal current and the voltage.

3. The percentage error =

$$\frac{\text{(Energy registered by the meter–True energy)} \times 100}{\text{True energy}}$$

circuit and a fixed DSP function for calculation of the real power. A highly stable oscillator integrated into the chip provides the necessary clock for the chip. IC ADE7757 supplies the average real-power information on the F1 and F2 low-frequency outputs. These outputs may be used to directly drive a stepper motor-based electromechanical counter or any other suitable counter.

IC ADE7757 also provides a high-frequency output at the calibration frequency (CF) pin for a selected meter constant (here, it is 3200 impulses/kWh). This high-frequency output provides instantaneous real-power information, which is used to speed up the calibration process. It also provides a means for quickly verifying the meter's functionality and accuracy in a production environment.

Theory of operation

The two analogue-to-digital converters (ADCs) used in the chip digitise the output of current and voltage sensors. The ADCs are 16-bit, sigma-delta type with an oversampling rate of 450 kHz. These work with oversampling so that the bandwidth of the input signal is much less than fs/2, where 'fs' is the sampling frequency. In its most basic form, the sigma-delta converter contains a one-bit ADC and DAC. It produces a higher-resolution digital word output by averaging several one-bit samples.

The real power is derived from the instantaneous power signal. The instantaneous power signal is achieved by a direct multiplication of the current and voltage signals. In order to extract the real power component (referred to as the DC component), the instantaneous power signal is low-pass filtered. This scheme correctly calculates the real power for sinusoidal current and voltage waveforms at all power factors. All the signal processing is carried out in the digital domain for superior stability over temperature and time.

Fig. 2 shows the block diagram for signal processing along with the waveforms at the output of the multiplier and after the low-pass filter (LPF). It is observed that this method of extracting the real power information holds good even when the current is not in phase with the voltage. The real power component (DC component) of the instantaneous power for sinusoidal voltage/current waveforms with a power factor of 0.5 (current lagging the voltage by 60°) is:

$$\frac{V \times I}{2} \times \cos 60°$$

The real power calculation holds good even for non-sinusoidal current and voltage waveforms.

Mains current sampling (channel V1). The voltage output from the current sensor (proportional to the load current) is connected to channel V1 of IC ADE7757, which is a fully differential voltage input. The V1P input is positive with respect to V1N. The maximum peak differential signal on channel V1 should be less than ±30 mV (i.e., 21mV rms for a pure sinusoidal signal) with reference to analogue ground (AGND) for the specified operation. Typical sampling connections are shown in Fig. 3.

Mains voltage sampling (channel V2). The output of the line voltage sensor is connected to IC ADE7757 at this analogue input. Channel V2, like channel V1, is a fully differential-voltage input channel with maximum peak differential signal of ±165 mV referenced to analogue ground (AGND). Typical connections for mains voltage sampling channel V2 are shown in Fig.

4. It is quite convenient to adjust the ratios of Ra and V_R for adjusting the gain of the meter.

Phase matching between channels. It is important that the relative phase difference between voltage and current waveforms at the inputs of V1 and V2 channels is not disturbed since any phase mismatch between channels will translate into significant measurement error at low power factors. IC ADE7757 is internally phase-matched over the frequency range of 40 Hz to 1 kHz between the two channels, which ensures that the relative phase relations of the two channels are maintained throughout the useful range of frequencies.

Power supply monitor. The on-chip power supply monitor of IC ADE7757 continuously monitors the power supply (V_{DD}). If the supply is less than 4V, IC ADE7757 is reset. This ensures proper device operation at power-up and power-down. The power supply monitor has built-in hysteresis and filtering that provides a high degree of immunity to false triggering due to noisy supply.

Transfer function. The transfer function refers to the relation between the true power into the load and its representation in terms of the equivalent frequency at F1 and F2 output points. The transfer function of IC ADE7757 is quite linear, and as such, a one-point calibration (at Ib) at unity power factor is all that is needed to calibrate the meter. If precautions are taken at the design stage, no calibration is necessary at power factors as low as 0.5 (i.e., phase difference of 60°).

The output frequency or pulse rate is related to the input voltage signals as follows:

$$\text{Freq} = \frac{515.84 \times V1_{rms} \times V2_{rms} \times F_{1\text{-}4}}{V_{ref}^2}$$

where Freq is the output frequency on F1 and F2 (Hz), $V1_{rms}$ is the differential rms voltage signal on channel V1 (volts), $V2_{rms}$ is the differential rms voltage signal on channel V2 (volts), V_{ref} is the reference voltage (2.5V ± 8%) and $F_{1\text{-}4}$ is one of four possible frequencies selected by using the S0 and S1 logic inputs (see Table II).

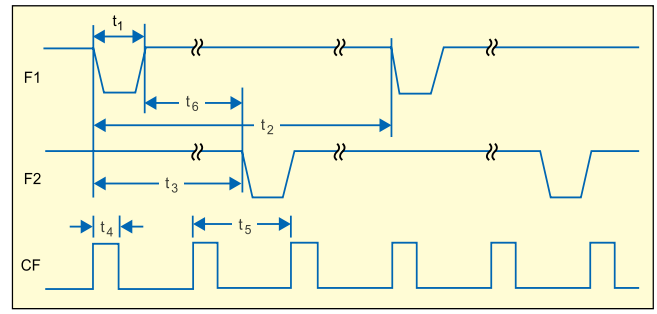


Fig. 5: Output signal timing diagram

TABLE II
$F_{1\text{-}4}$ Frequency Selection

S1	S0	$F_{1\text{-}4}$ (Hz)
0	0	0.85
0	1	1.7
1	0	3.4
1	1	6.8

TABLE III
Output Frequency on CF Pin

SCF	S1	S0	CF Signals (Hz)
1	0	0	128×F1, F2
0	0	0	64×F1, F2
1	0	1	64×F1, F2
0	0	1	32×F1, F2
1	**1**	**0**	**32×F1, F2**
0	1	0	16×F1, F2
1	1	1	16×F1, F2
0	1	1	2048×F1, F2

TABLE IV
Approximate Timings for the Outputs (Fig. 5)

Parameter	Time duration	Units
t_1	550	ms
t_2	1.428 (min.)	sec
t_3	0.714 (min.)	sec
t_4	180	ms
t_5	44.65 (min.)	ms
t_6	165 (min.)	ms

Shunt selection. In order to arrive at the values of $V1_{rms}$ and $V2_{rms}$ we must select the size/power dissipation rating of shunt for developing $V1_{rms}$ (proportional to line current), which is the most critical part of the design. We have chosen the maximum current as 30 amps and the shunt size as given in the application note, i.e., 350 μΩ. At 30 amps, its power dissipation would be $30^2 \times 350 \times 10^{-6}$ watts = 315 mW. This is reasonably low.

The chosen shunt must:

1. Provide the necessary dynamic range (400 to 500).
2. Dissipate less power.
3. Be small-size, so that it can be installed within the meter case to avoid tampering.
4. Have low temperature coefficient. (Manganin has low temperature coefficient.)

For experimental purposes, we used 24mm long 18SWG copper wire, which gave a shunt resistance of around 350 μΩ.

Design example

For designing a meter with 100 pulses/kWh count, proceed as follows:

Step 1. Select the shunt as discussed above. The shunt selected is 350 μΩ. The sense voltage $V1_{rms}$ at constant basic current (Ib) of 5 amps would be 1.75 mV.

Step 2. With nominal mains voltage of 230V AC rms and constant basic current (Ib) of 5 amps, the energy consumed in one hour is 230×5 = 1150 watt-hour or 1.15 kWh.

Since we have selected 100 pulses/kWh, for consumption of 1.15 kWh, the output should be 115 pulses. The equivalent frequency (cycles per sec.) is 115/3600 = 0.0319443 Hz.

Step 3. Assuming S0 = 0 and S1 = 1, read $F_{1\text{-}4}$ value from Table II, which is 3.4 Hz.

Step 4. From the transfer function equation, calculate $V2_{rms}$ by substituting V_{ref}=2.5V and the other values:

$0.0319443 \text{ Hz} = (515.84 \times 1.75 \times 10^{-3} \times V2_{rms} \times 3.4)/2.5^2$

or $V2_{rms}$ = 65 mV

Thus an rms voltage sample of 65 mV measured between V2P and AGND, and V2N and AGND, in

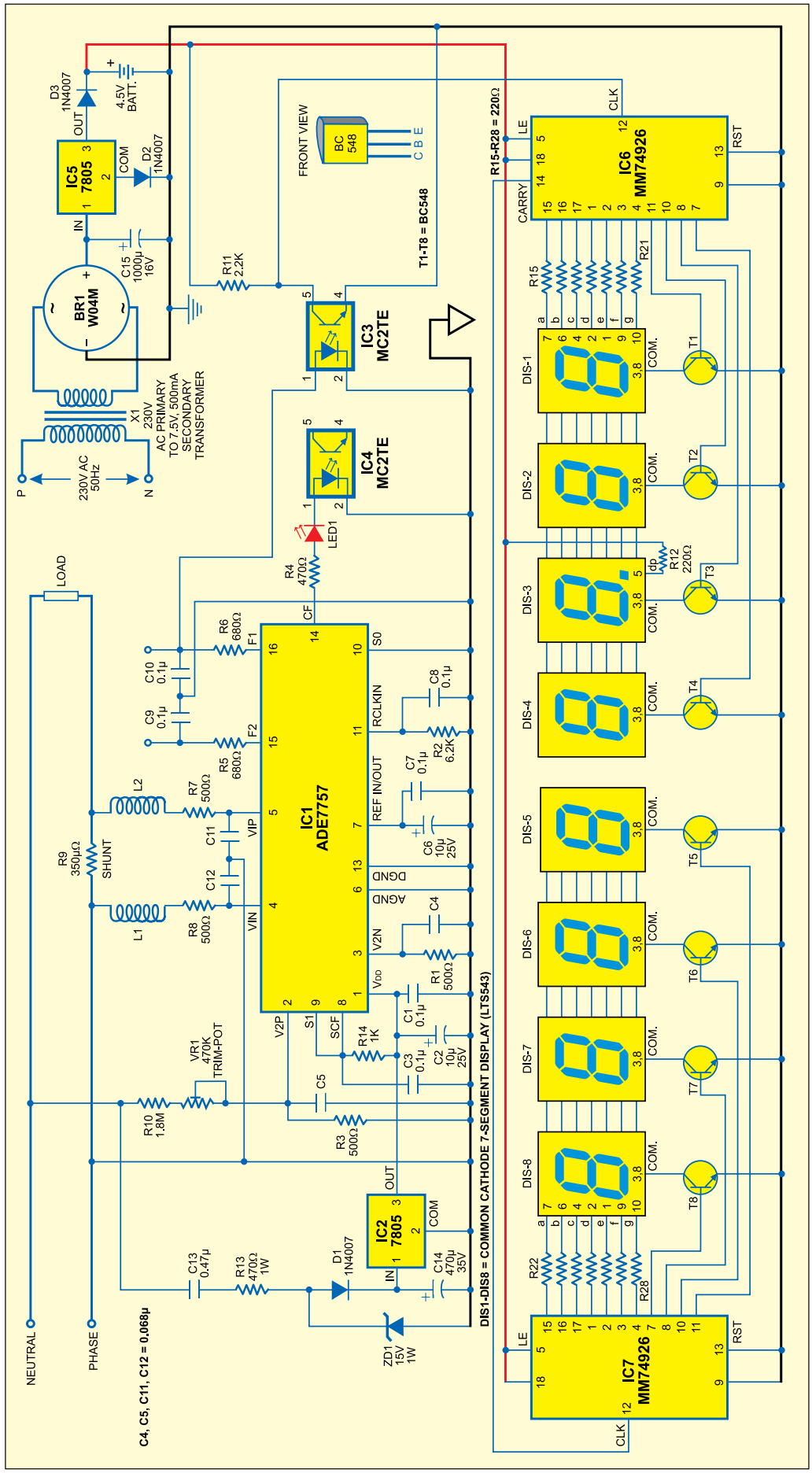


Fig. 6: Circuit of the energy meter using ADE7757

conjunction with rms voltage sample of 1.75 mV (refer step 1) measured between V1P and AGND, and V1N and AGND, should produce 115 pulses per hour, or 0.0319443 Hz (refer Step 2), at F1 and F2 outputs of IC ADE7757. Now, if we set the select calibration frequency (SCF) to logic 1, the output pulses at CF will be 32 times the pulse rate at either F1 or F2 output (see Table III).

As stated earlier, the CF output is used for calibration. Since this frequency is comparatively high, the real power information is accumulated very fast and hence less averaging is carried out for the CF output. That means the meter is more responsive to power fluctuations. The CF output can also be used for interfacing with microcontrollers.

Timing of outputs. The timing diagram for F1, F2 and CF pulse outputs is shown in Fig. 5 and the approximate timings for the selected parameters are given in Table IV.

The circuit

By following the aforementioned design principles, we

arrive at the energy meter circuit shown in Fig. 6. IC ADE7757 (IC1) is at the heart of the energy meter. It directly interfaces with the shunt resistor and operates off the AC input. The only analogue circuitry used in IC ADE7757 is in the sigma-delta ADCs and reference circuit. All the other signal processing is carried out in digital domain.

The power supply for IC ADE7757 is derived directly from mains using the capacitor divider network comprising C13 and C14. Most of the voltage is dropped across C13 (0.47µF polyester capacitor rated for 630V), while resistor R13 (470-ohm, 1W) is used as a current limiter. The output across C14 is limited to 15V DC, which serves as an input to regulator 7805 (IC2). The regulated 5V is fed to IC1 at its V_{DD} pin 1. In this application, the phase line is connected to AGND (pin 6) and DGND (pin 13) and hence to the common terminal of regulator IC2.

Two MM74926 ICs (IC6 and IC7) are cascaded to act as an 8-digit ripple counter, in conjunction with eight 7-segment displays (DIS-1 through DIS-8), which require additional 5V regulated and isolated supply (to avoid extension of live mains to the counter section). A conventional 5V regulator circuit incorporating a bridge rectifier (BR1), smoothing capacitor (C15) and regulator IC 7805 (IC5) has been used for the purpose. A 4.5V rechargeable battery is used to provide back-up so that the counter does not reset when mains fails. Diode D3 prevents battery discharge through the regulator during mains interruption. The voltage drop across diode D3 is compensated by using diode D2 in series with the common terminal of regulator 7805 (IC5).

The F1 output of IC1 is coupled to 8-digit ripple counter IC MM74926 via optocoupler IC3, while LED1 indicates that IC1 is working. CMOS IC MM74926 consists of a 4-digit counter, internal output latch, npn output source drivers for the 7-segment display and internal multiplexing circuitry with four multiplexed outputs. As multiplexing circuit has its own free-running oscillator, it does not require external clock. The counter advances on the negative edge of the clock pulse. The high input at the latch-enable pin displays the counter outputs.

IC6 drives the first four 7-segment displays (DIS1 through DIS4), while IC7 drives the remaining four displays (DIS5 through DIS8). IC6 is cascaded to IC7 by connecting the 'Carry' output of IC6 to the clock input of IC7.

Transistors T1 through T8 drive the respective digit displays DIS1 through DIS8.

Since F1 output comprises 100 pulses for each energy unit (kWh), a decimal point is permanently placed between DIS2 and DIS3. Thus the display can show up to 999999.99 units and then restart from 000000.00.

The meter and the PCB layout must be designed such that the conducted/radiated electromagnetic disturbances and the electrostatic discharge do not damage the meter or disturb its working. Other disturbanc-

Soldering of SMT ICs

- Apply flux to the pads where the IC is to be soldered.
- Add a small amount of solder to one of the corner pin pads.
- Line up the IC with the pads on the PCB. Double check the IC orientation.
- Melt the solder with your iron and move the IC into position with tweezers. Let the solder solidify.
- Solder the diagonally opposite pin. Check under magnification that all pins line up with their respective pads.
- Solder the rest of the pins and check under magnification.

Note. Special techniques may be needed for some packages.

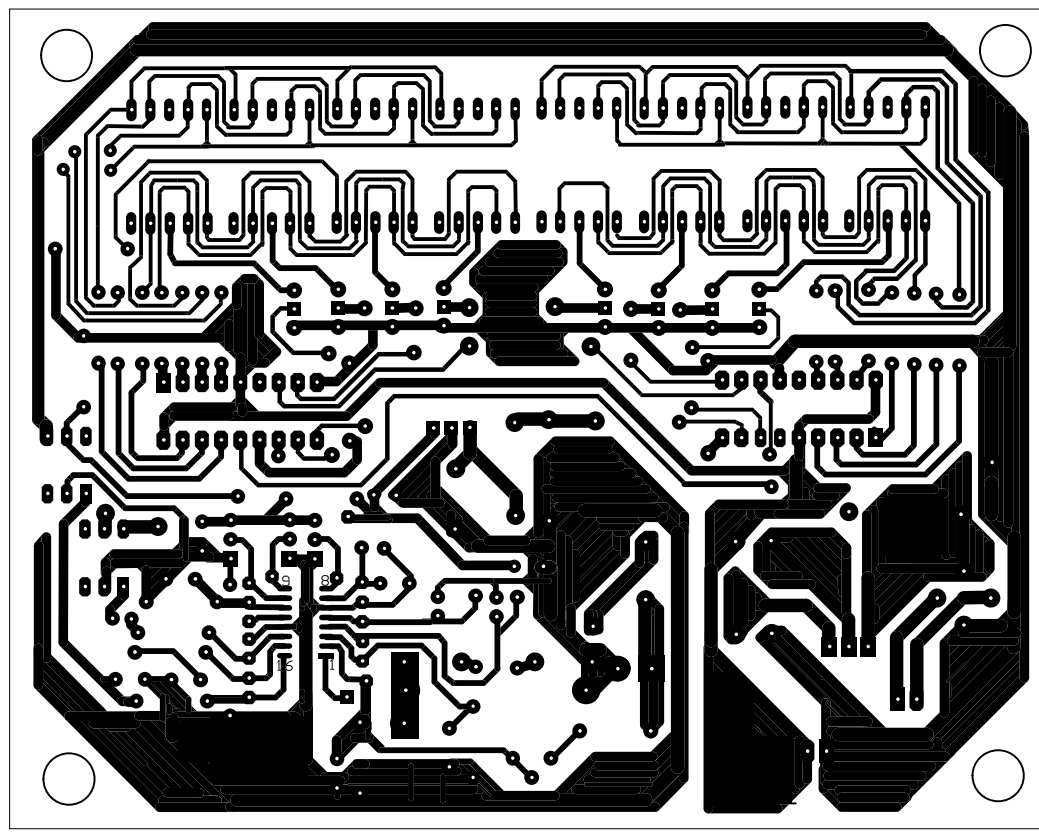


Fig. 7: Actual-size, single-side PCB layout for the energy meter using ADE7757

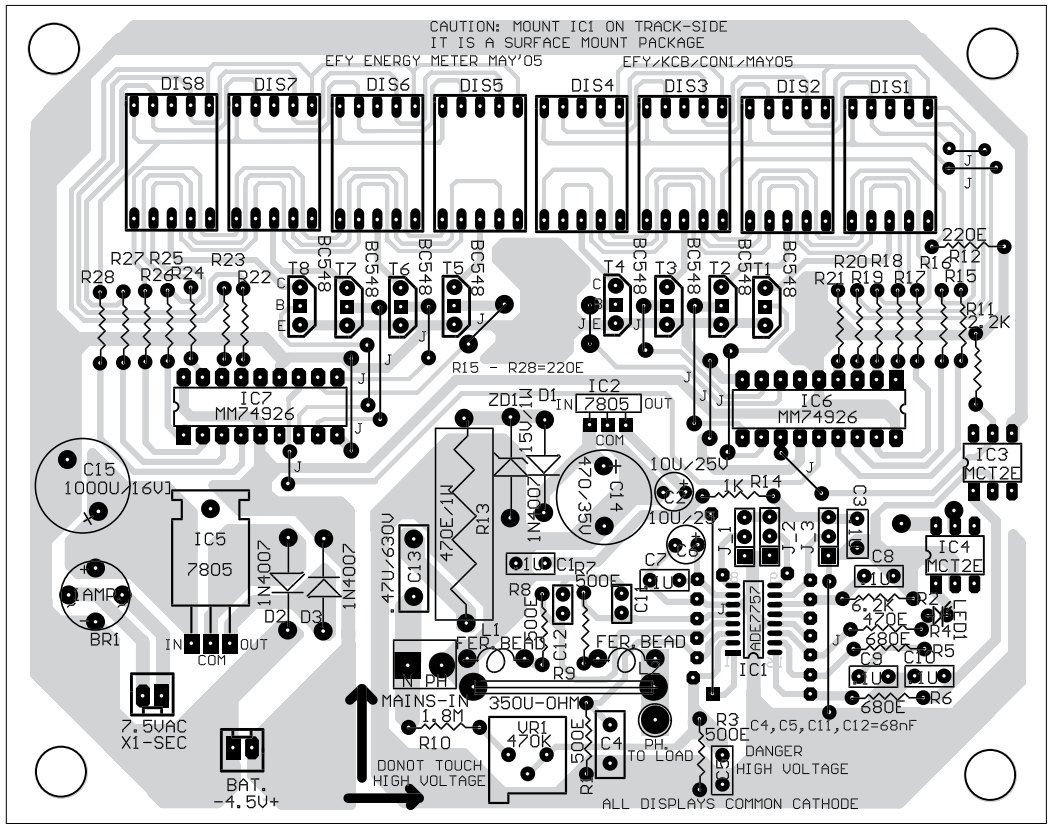


Fig. 8: Component layout for the PCB in Fig. 7

es to be considered are electromagnetic HF fields, fast transience burst and power line surge.

All of the precautionary components and design techniques (ferrite beads, capacitor line filters, large SMD resistors and grounding of the PCB layout) contribute to protect the meter circuitry from all forms of electromagnetic disturbances. Ferrite beads play a more important role against RF and fast transience burst.

This being a non-commercial educational project, all the laid down design principles have not been adhered to. A simple single-side PCB has been used for assembling the energy meter circuit and testing it. The PCB track layout is shown in Fig. 7 and its component layout in Fig. 8. The method for soldering the surface mount ICs is given in the box. ●

TWO-WHEELER SECURITY SYSTEM

■ VINAY UDAY PRABHU

India is the world's largest market for two-wheelers. Newer models with improved fuel efficiency and power ratings keep hitting the market off and on. Sadly, the security aspect of two-wheelers remains neglected. This fallibility inspired us to devise a foolproof yet cost-effective security system to safeguard bikes against theft. Any attempt to move the bike or force the ignition key to start the bike will set off the bike's horn.

The concept

The two-wheeler security system (Fig. 1) comprises a handheld infrared (IR) transmitter, IR receiver/sensor, switching circuit, power supply, turbulence detection unit, alarm and ignition switch. You have to keep the handheld IR transmitter with you and hide the receiver module at a secure place in your bike.

Whenever you leave the bike, switch on the security circuit by pressing the transmitter switch while directing the transmitter towards the sensor module such that the transmitted IR rays fall on it directly. The received signal activates the security circuit to blow the horn when subsequently someone tries to steal your bike by moving the bike or by using a duplicate key. The turbulence-sensing mechanism detects manhandling of the bike to trigger the alarm.

When you return back, switch off the sensor mechanism before starting the bike. Else, you may be caught off guard as this again will trigger the alarm.

In case the transmitter unit is not working or you have misplaced it, you can still activate the security circuit simply by flipping its switch S3 to on position.

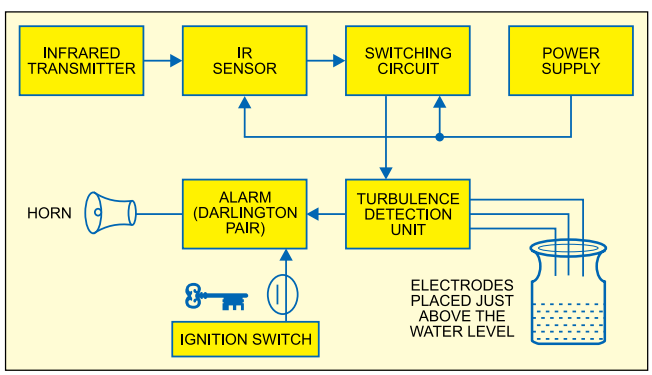


Fig. 1: Block diagram of the two-wheeler security system

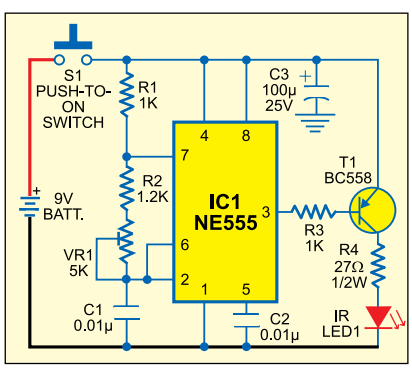


Fig. 2: Transmitter circuit

Circuit description

Transmitter circuit. The transmitter circuit (Fig. 2) works off a 9V battery. It is built around timer IC NE555, which is wired in astable multivibrator mode to generate around 38kHz frequency. The timer output is amplified by pnp transistor BC558 to drive the IR LED (LED1). Resistor R4 limits the current flowing through LED1.

When you press switch S1 momentarily, the astable multivibrator starts oscillating and the 38kHz frequency generated is transmitted through IR LED1. Make sure that IR LED1 is properly oriented towards the IR sensor module of the receiver circuit. Its transmission wavelength of 900 to 1100 nm (near-IR range) lies in the peak receptivity range of TSOP1738 receiver module.

Receiver circuit. The receiver circuit (Fig. 3) comprises power supply, sensor module, switching, turbulence detection and alarm sections.

Power supply. The receiver cir-

PARTS LIST

Semiconductors:

IC1, IC2	- NE555 timer
IC3	- CD4027 dual J-K flip-flop
IC4	- 7805, 5V regulator
IC5	- CD4081 quad two-input AND gate
IC6	- CD4071 quad two-input OR gate
T1	- BC558 pnp transistor
T2, T3, T5	- BC547 npn transistor
T4	- TIP122 npn transistor
D1-D7	- 1N4001
IR LED1	- Infrared LED
IRX1	- TSOP1738 IR receiver module

Resistors (all ¼-watt, ±5% carbon, unless stated otherwise):

R1, R3, R8, R10, R18	- 1-kilo-ohm
R2	- 1.2-kilo-ohm
R4	- 27-ohm, 0.5W
R5, R11	- 100-ohm
R6, R13	- 10-kilo-ohm
R7	- 220-kilo-ohm
R9, R12	- 470-ohm
R14-R16	- 22-kilo-ohm
R17	- 100-kilo-ohm

Capacitors:

C1, C2, C6	- 0.01µF ceramic disk
C3	- 100µF, 25V electrolytic
C4	- 1µF, 25V electrolytic
C5	- 10µF, 25V electrolytic
C7	- 0.1µF ceramic disk
C8	- 100µF, 25V electrolytic

Miscellaneous:

S1	- Push-to-on switch
S2	- Motor bike switch
S3	- On/off-switch
LS1	- 12V horn
E1-E3	- Thin steel rod
	- Plastic bottle
	- 9V battery
RL1	- 12V, 285-ohm, 1C/O relay

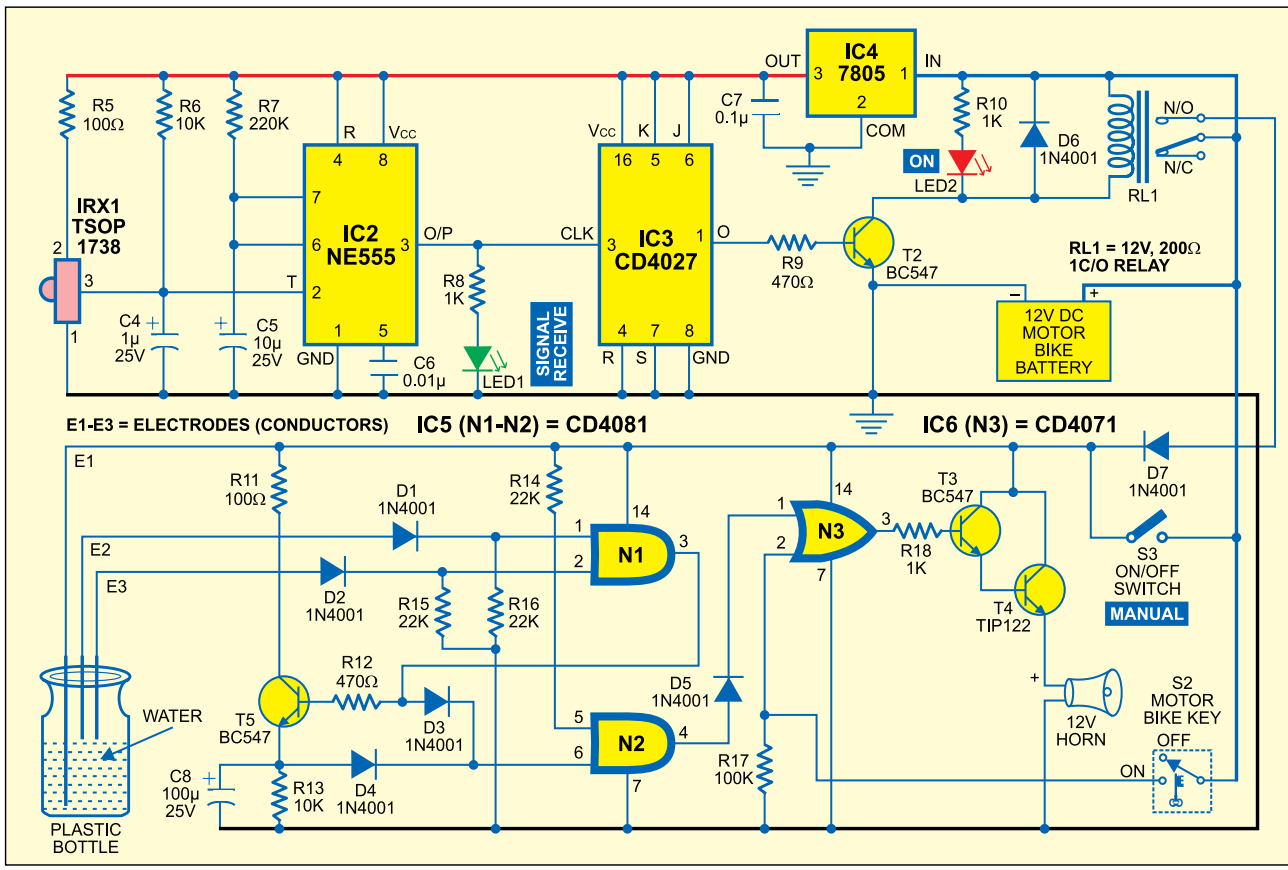


Fig. 3: Circuit of the two-wheeler security system

cuit excluding IR receiver module TSOP1738, timer NE555 (IC2) and J-K flip-flop CD4027 (IC3) works off the 12V battery of the bike. The 12V supply is down-converted to 5V by regulator IC 7805 (IC4) to drive TSOP1738, IC2 and IC3.

Sensor. IR receiver module TSOP1738 is sensitive to the IR radiation modulated at 38 kHz. Its normally high output goes low when any IR radiation is detected.

Switching. The high-to-low transition of the receiver output triggers timer IC2, which is rigged up in the monostable mode, and the green LED (LED1) glows to indicate signal reception and it also generates a positive-going clock pulse for the flip-flop.

The output of IC2 is fed to the clock input of IC3. Here IC3 is wired in toggle mode by connecting its J and K inputs to +5V, and Set and Reset pins to ground. IC3 is triggered by the clock pulse received from IC2 and transistor T2 conducts to energise relay RL1 and the +12V supply energises electrode E1, turbulence-detection section (comprising AND gate CD4081 (IC5) and OR gate CD4071 (IC6)) and the alarm section. The red LED (LED2) glows to indicate enabling of these sections.

When the transmitter switch is pressed again, the relay gets de-energised by the toggling action of the flip-flop and electrode E1, turbulence-detection section (built around IC5 and IC6) and the alarm section (comprising Darlington pair transistors T3 and T4) are disabled. The red LED now goes off. A free-wheeling diode (D6) used in parallel to the relay prevents the transistor from damage when the relay de-energises.

Turbulence detection. The turbulence-detection section detects sudden bike jerks, provided relay RL1 is in energised state. It consists of a small, watertight bottle with three electrodes inserted into it to detect the turbulence caused.

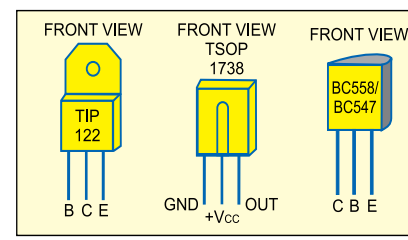


Fig. 4: Pin configurations of TIP122, TSOP1738 and BC558/BC547

When the bike is moved, water inside the bottle shakes to short all the three electrodes inside the bottle together momentarily and the inputs of AND gate N1 go high. The inputs of AND gate N2 also go high and its high output makes output pin 3 of OR gate N3 high. This causes forward biasing of Darlington pair of transistors T3 and T4 and the horn blows.

At the same time, npn transistor T5 gets forward biased to charge capacitor C8. As a result, pin 6 of AND gate N2 remains high for some time even after the electrodes are no longer shorted by the splash of water in the bottle. Hence,

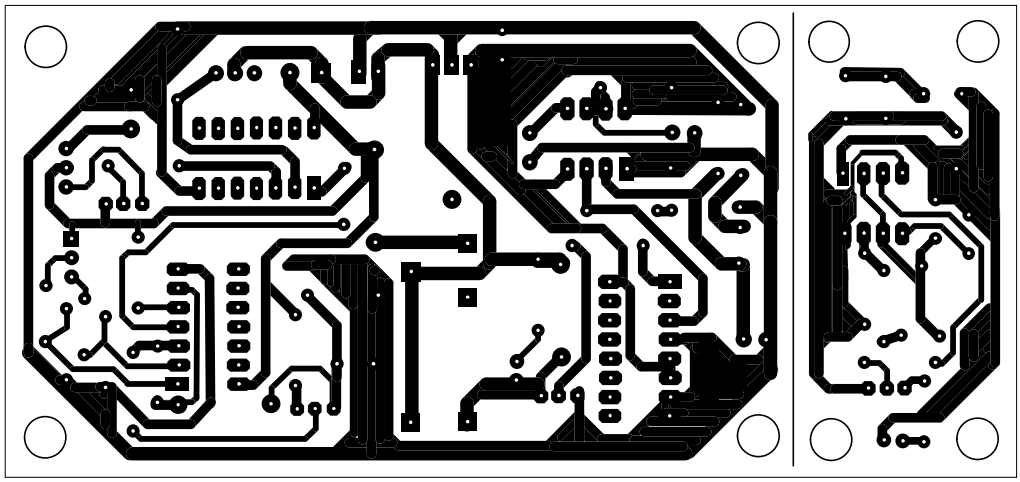


Fig. 5: Actual-size, single-side PCB layout for two-wheeler security system

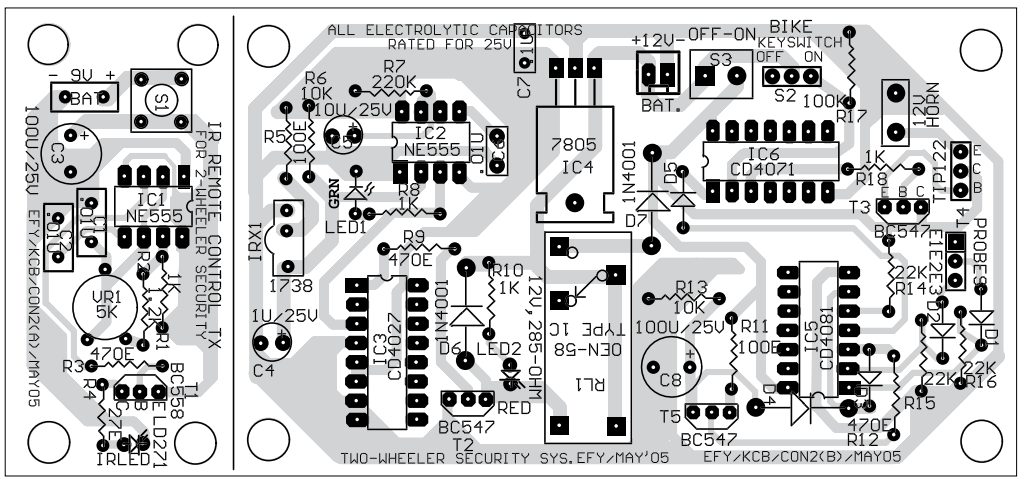


Fig. 6: Component layout for the PCB

the horn blows until capacitor C1 discharges below the threshold level of CMOS gate via resistor R13.

The basic aim of the turbulence detection module is to activate the horn when the bike is moved or ignition key is forced to start the bike. This is achieved by using two-input OR gate N3 (IC CD4071), which is connected to:

1. The output of AND gate N2, which is virtually the output of the turbulence detection module

2. A lead from the 'on' terminal of the bike key

When either or both the inputs of N3 are high, its output goes high to provide sufficient current to drive the Darlington pair transistors of the warning-indication section and the horn blows.

Alarm. The alarm section comprises a high-current gain Darlington pair of transistors pumping current into the bike's horn. The emitter of transistor T4 is directly connected to the positive terminal of 12V horn. A heat-sink is used to dissipate the excessive heat generated by npn power transistor TIP122.

Fabrication

Assemble the transmitter and receiver circuits on separate general-purpose PCBs as you have to carry the transmitter unit with you and install the receiver unit in the bike. The lid of the water-filled plastic bottle should be tight enough so that the water does not leak out.

Fig. 4 shows pin configurations of power transistor TIP122, IR sensor module TSOP1738 and npn transistor BC558 (or BC547).

The combined actual-size, single-side PCB for the transmitter and receiver circuits (Figs 2 and 3) is shown in Fig. 5 and its component layout in Fig. 6. You can separate the two PCBs by cutting along the vertical line of the PCB. Make sure that manual switch S3 in the PCB is easily accessible, so that you may use it to switch on/off the security circuit in case the transmitter is not working or you've lost it.

Demerits

1. Two logic gates of CD4081 and three logic gates of 4071 are left unutilised.

2. TSOP1738 may get triggered by a TV remote.

3. Short range of the transmitter (4 to 5 metres).

Precautions

1. Make sure that electrode E1 is always dipped into the water.

2. Hang the two electrodes (E2 and E3) to rest just above the water level so that they easily get shorted by the shaking water when the bike is moved.

3. Make sure the bike's battery is fully charged.

4. Keep the circuit protected from water and high temperature.

Other applications

1. Turbulence detection.

2. The circuit can be modified to function as an overflow indicator in water tanks.

3. The receiver unit itself can be used as an infrared toggle switch. ●

MEDIUM-POWER LOW-COST INVERTER

■ T.K. HAREENDRAN

This medium-power inverter is capable of generating approximately 300VA power. You can power the inverter from your car battery to generate 50Hz AC supply. The inverter provides enough back-up power to light up up to three 100W bulbs for up to two hours, provided the car battery is fully charged.

Fig. 1 shows the block diagram of the medium-power inverter. The power house comprises car battery, power supply, oscillator-cum-divider, driver, inverter transformer, power amplifier, buzzer and battery-level indicator sections. To keep the cost low, the charger circuit has not been included here. The car battery can be charged through the car battery charger circuit whenever it discharges.

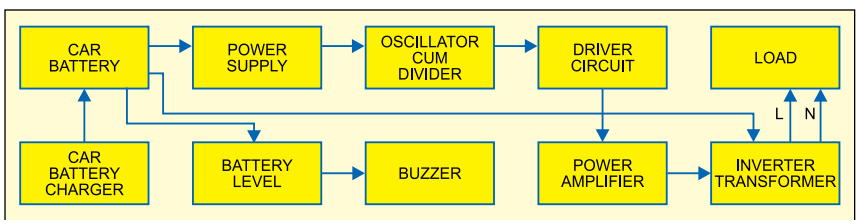


Fig. 1: Block diagram of medium-power inverter

The circuit

Connect the car battery to the circuit using crocodile clips. The red clip should be connected to the positive terminal of the battery and the black clip should be connected to the negative terminal of the battery. If crocodile clips are connected to the wrong terminals of the battery, LED1 glows to alert you.

Now flip switch S1 towards 'on' position to enable the circuit. LED3 glows to indicate power-'on' and 12V DC reaches regulator IC 7805 (IC1). The regulated output is fed to the oscillator-cum-divider and driver while the centre terminal of the inverter transformer primary is connected to the positive terminal of the car battery through high-currrent carrying wires. Capacitor C1 functions as a reservoir capacitor.

Low-battery indicator. For long life of the battery, it should not be allowed to discharge to a voltage below 10V. Even a single event of deep discharge can reduce the charge-holding capacity of the battery permanently.

For audio-visual indication of the low-battery level, a dual operational amplifier IC LM358 has been used. A fixed reference voltage of 5.1V is applied to its positive input, while the sensing voltage is applied to its negative input. Set preset VR1 such that the piezobuzzer sounds when the on-load battery voltage falls below 10V DC.

When the battery voltage drops below 10V, the sense input voltage drops below 5.1V and output pin 1 of IC4 goes high to sound the buzzer and light up LED2.

Oscillator-cum-divider. The oscillator-cum-divider section is built around timer IC LM555 (IC2) and dual J-K flip-flop 7473 (IC3). Only one flip-flop of the dual JK flip-flop is used here.

Timer LM555 is wired as an astable multivibrator, whose time period is decided by resistors R7 and R8 and capacitor C5. It produces 100 Hz at output pin 3, which is given to pin 5 of the J-K flip-flop to produce 50 Hz with 50% duty cycle. When the inverter is switched on using switch S1, IC2 starts producing 100 Hz, while the J-K flip-flop produces 50 Hz at its output pins 8 and 9. The output of timer IC2 can be checked using the oscilloscope at test point (TP).

Driver circuit. The flip-flop output is fed to MOSFET driver transistors T1 and T2 via a diode-resistor combination. At any instant, if the voltage of pin 8 of IC3 is +5V, the voltage at its pin 9 will be 0V, and vice versa. Therefore when transistor T1 conducts, transistor T2 is cut off, and vice versa. Whenever output pin 8 of IC3 goes high, npn transistor T1 conducts and the corresponding set of MOSFETs (T3

PARTS LIST

Semiconductors:
IC1 – 7805 5V regulator
IC2 – LM555 timer
IC3 – 7473 dual J-K flip-flop
IC4 – LM358 dual operational amplifier
T1, T2 – BD139 npn transistor
T3-T8 – IRFZ44 power MOSFET
D1, D2 – 1N4148 diode
LED1, LED2 – 5mm red LED
LED3 – 5mm green LED
ZD1, ZD2 – 5.1V zener diode

Resistors (all ¼-watt, ±5% carbon, unless mentioned otherwise):
R1-R3, R5, R6
R9-R12 – 1-kilo-ohm
R4 – 1-ohm, 0.5W
R7 – 220-ohm
R8 – 15-kilo-ohm
VR1 – 470-kilo-ohm preset

Capacitors:
C1, C3 – 0.1µF ceramic disk
C2 – 1000µF, 35V electrolytic
C4 – 100µF, 25V electrolytic
C5 – 0.47µF ceramic disk
C6 – 0.01µF ceramic disk

Miscellaneous:
S1 – On/off switch
PZ1 – Piezobuzzer
X1 – 12V-0-12V primary to 300VA inverter transformer
- Crocodile clips (red and black)
- Multistrand high-current carrying wires

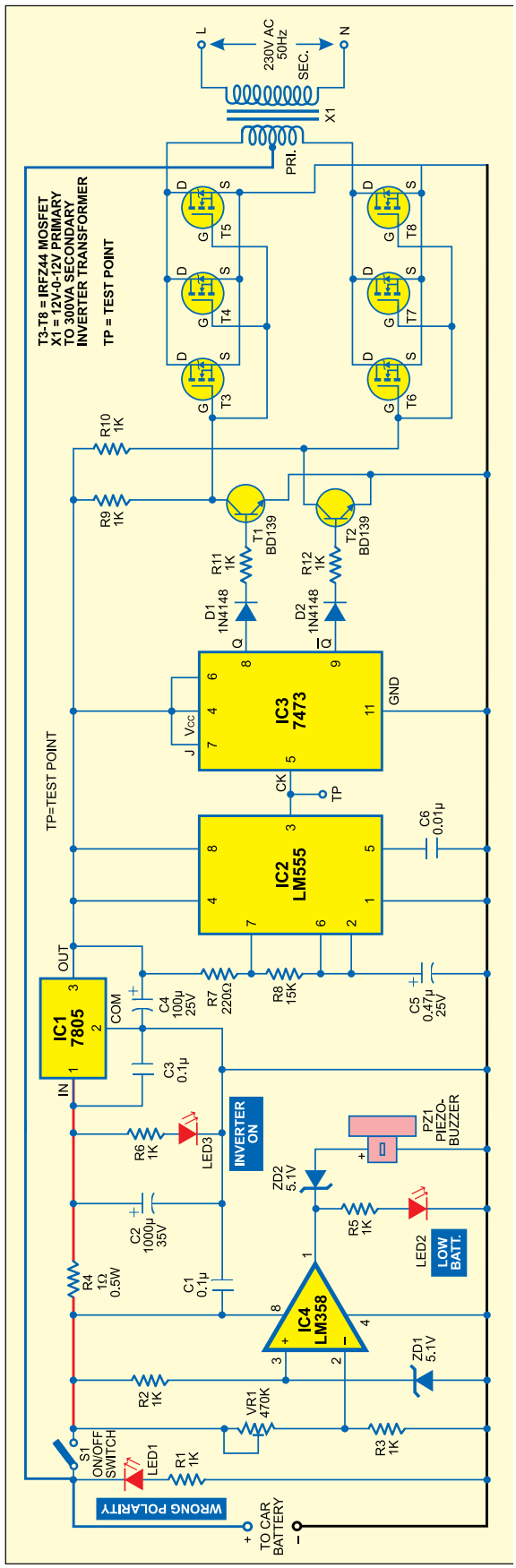


Fig. 2: Circuit of medium-power inverter

through T5) remains cut off while the collector of transistor T2 is at 5V. Thus current flows through half of the inverter transformer's primary winding. Similarly, when output pin 9 of IC3 goes high, npn transistor T2 conducts and the corresponding set of MOSFETs (T6 through T8) remains cut off while the collector of transistor T1 is at 5V. Thus current flows through the inverter transformer's primary winding.

Power amplifier. The power amplifier section comprises two sets of three power MOSFETs (IRFZ44) connected in parallel for operation of the inverter. The output of IC3 drives the MOSFETs (T3 through T5, and T6 through T8) via transistors T1 and T2 to generate 50Hz, 230V AC at the output of inverter transformer X1.

Fabrication

You can assemble the circuit on any general-purpose PCB. However, an actual-size, single-side PCB for the medium-power inverter circuit is shown in Fig. 3 and its component layout in Fig. 4. Pin configurations of MOSFET IRFZ44, regulator IC 7805 and npn transistor BD139 are shown in Fig. 5.

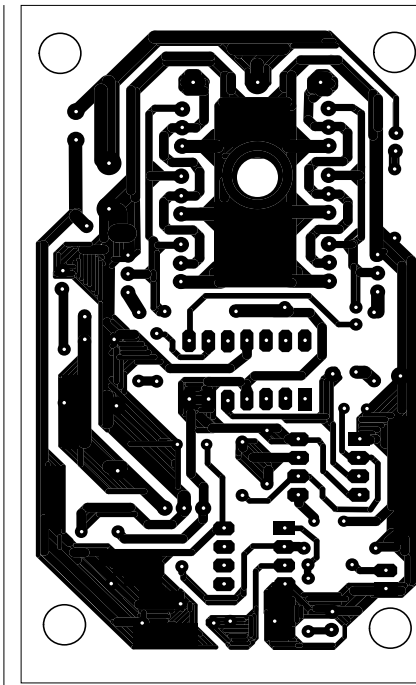


Fig. 3: Actual-size, single-side PCB for medium-power inverter

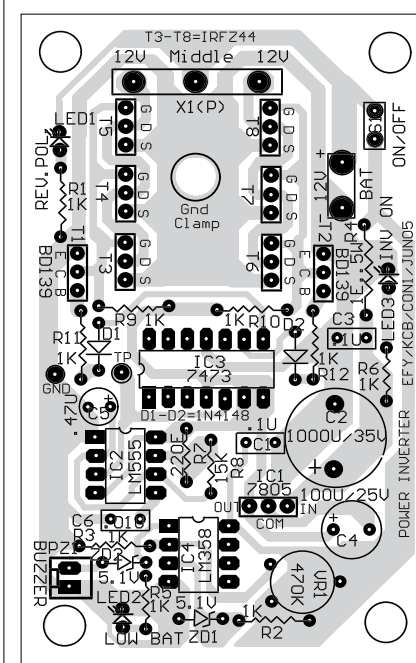


Fig. 4: Component layout for the PCB

After construction, enclose the entire circuit in a portable box (see Fig. 6). The first MOSFET set comprises T3, T4 and T5, and the second MOSFET set comprises T3, T4 and T5. Use separate heat-sinks for each MOSFET set. Since MOSFETs T3 through T5, and T6 through T8, are connected in parallel, connect the drains of the

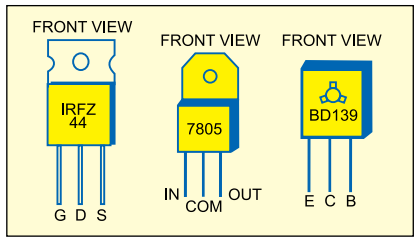


Fig. 5: Pin configurations of MOSFET IRFZ44, regulator IC 7805 and npn transistor BD139

MOSFETs (internally connected to the backplate having a through hole) using a copper/brass nut/bolt onto the respective common heat-sink. Mount all the status LEDs, piezobuzzer and switches on the front panel. Use heavy-gauge, multistrand battery wires (2.5 sq. mm or more) for the following DC connections:

1. From positive battery terminal to the middle of transformer X1 primary.

2. From the negative terminal of the battery to the common ground on the PCB using copper/brass nut and bolt (provision for the same is made on the PCB).

3. From each heat-sink set (where the drains of the MOSFETs have been connected using nuts and bolts) to the respective primary terminals of transformer X1.

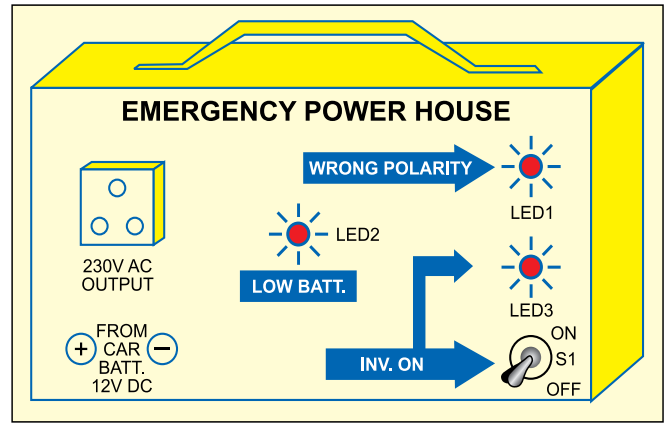


Fig. 6: Proposed portable box

EFY. Following additional precautions may also be taken:

1. On full load, the current drawn from the battery could be as high as 30 amperes. Therefore the ground track around source terminals of the MOSFETs on the PCB may be strengthened by depositing additional solder.

2. You may add a resistor (0.5-ohm rated at 20W) in series with positive battery lead going to the middle terminal of transformer X1 primary. In the case of excessive current being drawn by any of the MOSFETs (due to shorting, etc), only the resistor will burn, which can be easily replaced. ●

PROGRAMMABLE TIMER BASED ON AT90S4433 AVR

■ **D.S. OBEROI AND HARINDER DHINGRA**

A programmable timer finds numerous applications in the industry for efficient operation of machines in the desired sequence and for precise time durations. Such timers form part and parcel of all programmable logic controllers (PLCs), which control the switching on and switching off functions of appliances like fans/blowers, heaters and airconditioners, as desired.

Here's an AVR AT90S4433 microcontroller-based programmable timer that can switch on/off an appliance after a certain time. This time can be set through the pushbuttons provided in the circuit. After the set time elapses, the microcontroller-based logic generates the desired on/off signal to activate/deactivate the appliance. With this timer, you can program time periods from one minute to 99 hours and 59 minutes.

The circuit uses four 7-segment displays for displaying the time during normal operation of the timer and certain alphanumeric characters during setting/programming of the timers. These operations have been covered in the software part and the accompanying flow-charts. After the programmer has been set and is running, it can be immediately reset through switch S1 (Mode/End) in the case of an emergency.

The circuit uses the internal timer of AT90S4433 microcontroller for generating an interrupt every second, which, in turn, is used for computing the 'on'/'off' time period.

The hardware

Fig. 1 shows the block diagram of the programmable timer, while the circuit is shown in Fig. 2. Here AT90S4433 microcontroller is being operated at 4.000MHz frequency and this serves as the basis for all the timings, i.e., for generating an interrupt every second and controlling the LED display, etc. The circuit uses four 7-segment displays (DIS1 through DIS4). During normal operation of the timer, DIS1 and DIS2 display hours, while DIS3 and DIS4 display minutes. However, during setting/programming operation of the timers, they display certain alphanumeric characters as stated earlier. The seven segments and the decimal point of all the displays are wired in parallel and fed via port D (PD0 through PD7) outputs, while their common-anode terminals are separately driven by four transistors (T1 through T4), which are interfaced to Port B (PB0 through PB3) outputs of the microcontroller.

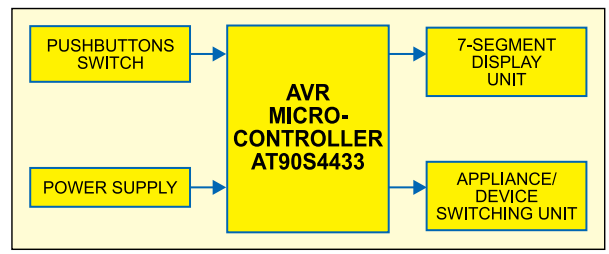


Fig. 1: Block diagram of programmable timer

Multiplexing of displays reduces not only the number of input/output (I/O) lines required but also the mean current required for operating the display units. Whenever some data is to be displayed, it is first latched on Port D and then that particular display unit is enabled by switching on the respective controlling transistor via Port B. The data is displayed without flickering with the help of the software. The SIGNAL(SIG_OVERFLOW0) interrupt service routine (ISR) is used to control the display units in different operational modes of the timer.

For manual setting and control of the timer, three dual-function pushbutton switches S1 through S3 labeled as Mode/End, On/Minute and Off/Hour, respectively, have been provided. Their functions have been explained under the software subheading. The pushbuttons are interfaced to the microcontroller via Port C. Their status is repeatedly checked by the getKeyStatus() routine, which returns information about the pushbutton pressed and then an appropriate action is initiated by the software. The pushbutton functionality is dual in nature, i.e., each button will perform a different function in a different mode of timer operation.

For activating/deactivating an appliance, an output interface signal marked DEVICE_CTRL (in Fig. 2) is available from Port B (PB4). The software will control the logic of this signal to switch on or switch off an

PARTS LIST

Semiconductors:
IC1 - AT90S4433 AVR microcontroller
T1-T5 - BC548 npn transistor
D1 - 1N4007 rectifier diode
DIS1-DIS4 - LTS542 common-anode, 7-segment display
LED1 - Red LED

Resistors (all ¼-watt, ±5% carbon, unless stated otherwise):
R1-R8, R14 - 220-ohm
R9-R12 - 1-kilo-ohm
R13 - 10-kilo-ohm
R15-R17 - 150-ohm
R18-R21 - 4.7-kilo-ohm

Capacitors:
C1, C2 - 27pF ceramic disk
C3 - 0.1µF ceramic disk
C4 - 47µF, 16V electrolytic

Miscellaneous:
X_{TAL} - 4.000 MHz crystal
RL1 - 5V, 150Ω, 1C/O relay
S1-S3 - Push-to-on switch

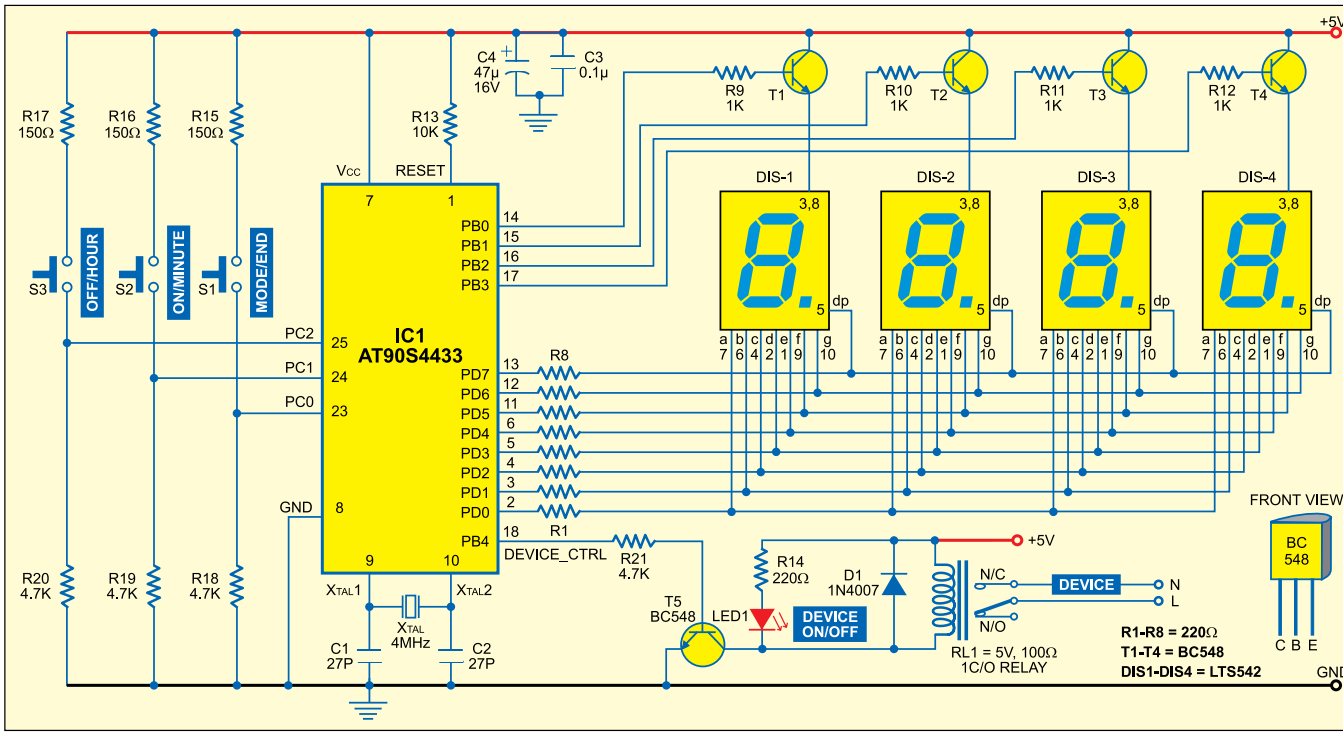


Fig. 2: Circuit of programmable timer using AT90S4433

appliance through energisation or de-energisation of relay RL1 via relay driver transistor T5.

The ports are configured as follows:

1. Port-B: It is configured in output mode and used for activating/deactivating an external appliance and controlling the transistors of the display units.

2. Port-C: It is configured in input mode and used for interfacing the pushbutton switches.

3. Port D: It is configured in output mode. All the segments of the display units are interfaced to this port.

The software

Two separate software are provided. The first software (labeled Timer-A) provides cyclic on/off functionality, while the second software (labeled Timer-B) provides switch-on functionality after a preset elapsed time. The following explanation relates to Timer-A, however the setting and operation of Timer-B have been briefly explained later in the article.

The desired functionality is achieved by controlling the hardware through software. The AVR-GCC 3.2 software (which can be downloaded from 'www.avrfreaks.net/AVRGCC' for Windows OS and from 'ftp://gatekeeper.dec.com/pub/GNU/' for Linux OS free of cost), along with AVR STUDIO version 3.0, is used for programming (in 'C' language syntax), debugging and generating the compatible hex code for AT90S4433 microcontroller.

The software uses both the 8- and 16-bit internal timers of the microcontroller. The 8-bit timer (timer 0) is used for displaying the information without flickering and the 16-bit timer (timer 1) is used for generating an interrupt every second, which is used for computing the elapsed time during both the 'on' and 'off' periods of the cycle.

The SIGNAL (SIG_OVERFLOW0) ISR of timer 0 is used for flicker-free display of the information. Timer-0 is initially configured to operate in timer mode with prescaling factor of '256' and preloaded with a desired value. Whenever timer 0 overflows, its ISR is initiated to latch the desired display data on Port D and also switch on the corresponding transistor of that particular display unit.

All the display units have to be refreshed with the respective data fast enough so that a flicker-free display is achieved, and for this reason, timer 0 is preloaded with an appropriate value. In this ISR, the operation status is checked by checking the value of DISPLAY_MODE and then the desired data is latched to Port D. After the ISR completes its operation, timer 0 is once again loaded with the desired refresh value (REFRESH_VAL).

The particular display unit that is to be made active with the desired data will depend upon the value of the DISPLAY_ON variable. The two left-most display units are used for displaying the hours, while the remaining two display units are used for displaying minutes (in countdown mode of operation). This functionality is achieved by the showNormalDisplay() subroutine.

In countdown mode, the displayed information changes in last 60 minutes of the operation. Now the two left-most display units are used to display minutes and the two right-most display units are used to display seconds. This is achieved with the help of the showLast60MinutesDisplay() subroutine.

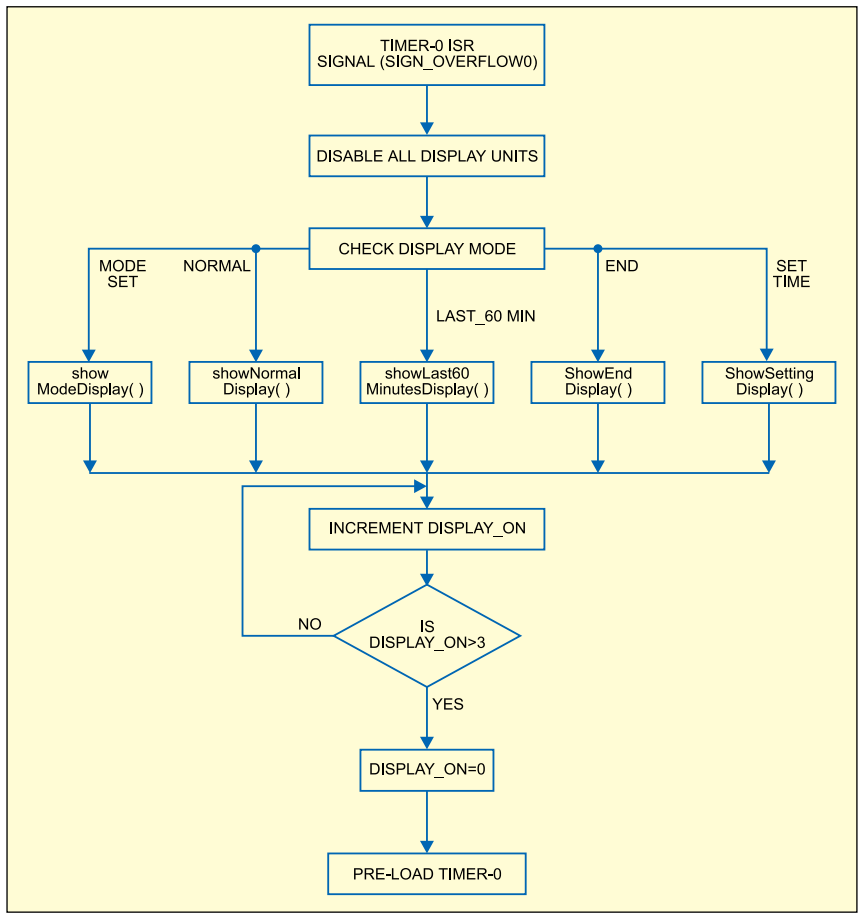


Fig. 3: Flow-chart of timer-0 ISR

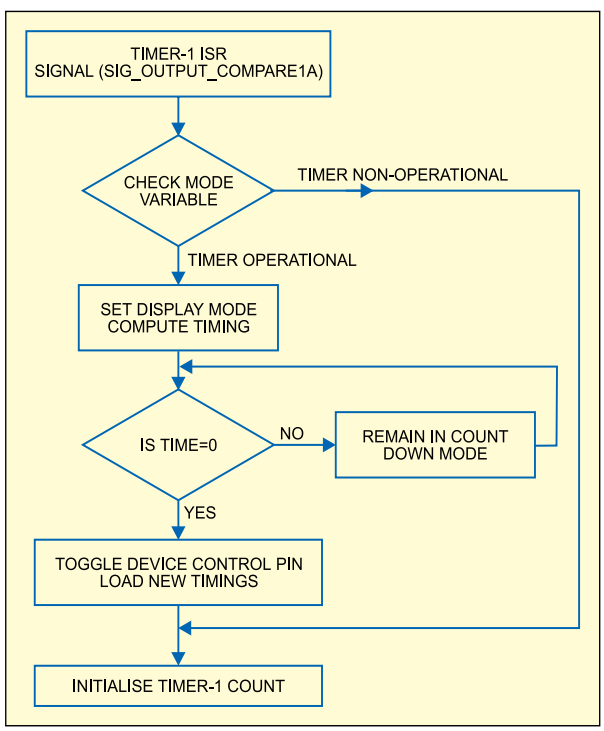


Fig. 4: Flow-chart of timer-1 ISR

Similarly, the displays for mode setting and the end message are achieved with the help of showMode Display() and show EndModeDsiplay() subroutines, respectively. While setting the 'on' and 'off' time periods, the show SettingTime() subroutine is used for displaying the desired information. Fig. 3 shows the flow-chart of timer-0 ISR.

The 16-bit timer 1 is used for time keeping. It is configured to work in compare mode with prescaling factor of '256' in conjunction with two compare registers of the microcontroller, which are preloaded with the desired value of 0x3D09. While in operation, as soon as the contents of timer-1 equal that of the compare register, a compare interrupt (SIGNAL(SIG_OUTOUT_COMPARE1A)) is generated. This ISR computes the elapsed time and controls the output signal (DEVICE_CTRL) at the end of 'on'/'off' time periods. It is also used to switch from 'on' period to 'off' period, and vice versa, after the lapse of the respective time period. The ISR sets the values of the DISPLAY_MODE variable, which, in turn, is used by timer-0 ISR for displaying the correct information (particularly in normal mode and last 60 minutes of the timer operation). Fig. 4 shows the flow-chart of timer-1 ISR.

Once the ports and internal timers of the microcontroller have been configured by the software, the main operational loop of the software is initiated, which basically checks the status of pushbutton switches with the help of the getKeyStatus() subroutine. Depending upon the mode of the operation, each pushbutton switch has a specific function. The status of the pushbutton switches is checked in conjunction with the value of the DISPLAY_MODE variable. The DISPLAY_MODE variable at any given moment indicates the status of operation and its value is used for displaying the correct information with the help of timer-0 ISR. The pushbutton switch functioning will also depend upon the value of this variable.

In the time period setting mode, the hour_minSettingChange() subroutine is used to set on_hour, off_hour, on_min and off_min values. This subroutine also checks for the maximum limits of hours (99) and minutes (59). Once the settings have been made, the modeSettingChange() subroutine is initiated from this main operational loop, which will now start the circuit's operation and the countdown process.

In operational mode, the main loop mainly checks the status of Mode pushbutton switch (S1). When Mode pushbutton switch is pressed, the out-

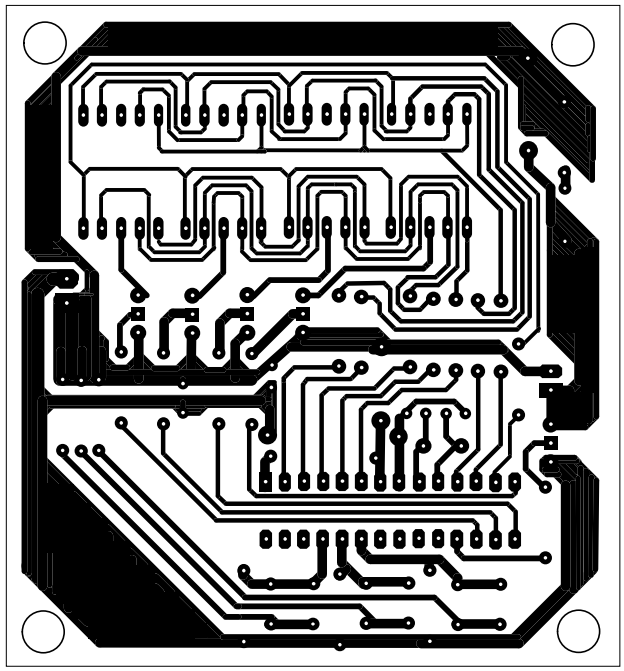


Fig. 5: Actual-size, single-side PCB for programmable timer

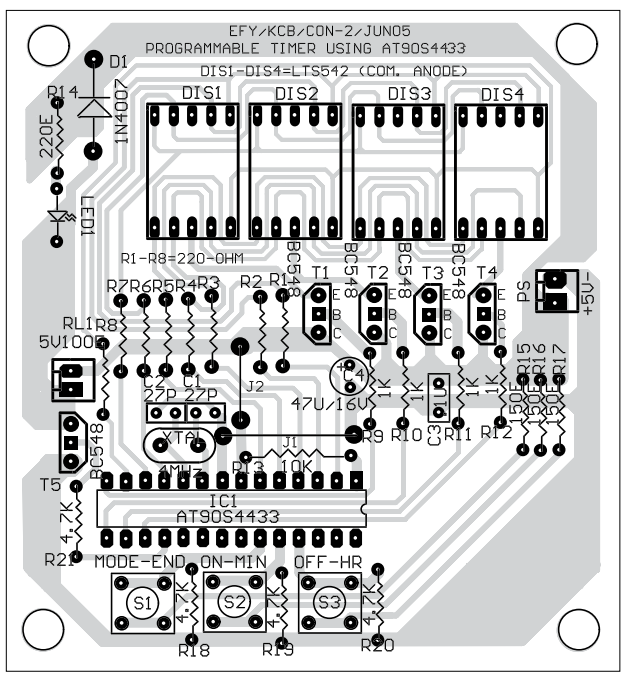


Fig. 6: Component layout for the PCB

put control signal immediately deactivates the appliance and End message is displayed. To resume the normal operation, the Mode pushbutton switch has to be pressed once again, which will take the user to mode setting.

Settings and operation of timer-A are as follows:

1. Immediately after power-on or reset, the following message is displayed:

nodE

which indicates that none of the 'on' and 'off' time periods has been set.

2. On pressing the Mode/End pushbutton, there is no change in the display status and software operation.

3. Now by pressing the On/Min pushbutton, you can set/change the 'on' time period. By continuously pressing the Hour or Min pushbutton, you can set any value from '0' to '99' for hours and '0' to '59' for minutes. Release the respective buttons after the desired value is displayed.

4. The value can be changed by pressing the respective pushbutton once again.

5. Press Mode/End pushbutton to set the 'on' time period.

6. Now the following information is displayed:

n.o.dE

The decimal point in the two leftmost display units indicates that the 'on' time period has been set.

7. Now by pressing the On/Min pushbutton, you can again change the 'on' time period. In this state, the already set value is displayed first and the change made will be in count-up mode from this time onwards. On pressing the Mode/End button, the new value of 'on' time period becomes active and the message of Step 6 is displayed.

8. By pressing the Off/Hour pushbutton, you can set the 'off' time period in the same manner as for 'on' time period in Steps 3, 4, 5 and 7.

9. Now the following information is displayed:

n.o.d.E.

The decimal point in the two rightmost display units indicates that 'off' time period has been set.

10. Once the settings have been made, the time period settings can be changed again by repeating the aforementioned steps.

11. In Step 3, if Off/Hour pushbutton is pressed first for setting the 'off' time period, the display will be:

nod.E.

In the setting mode, any of the 'on' and 'off' time periods can be set first.

12. To start the timer, press Mode/End pushbutton for around 1.5 seconds. The countdown starts from 'on' time period and the DEVICE_CTRL output control signal becomes high. The circuit operation is indicated by blinking of the decimal point on DIS2 every second. The output signal will change with the respective active period ('on' or 'off') one after the other.

In case you need to switch off the external appliance immediately, stop the working of the timer by pressing Mode/End pushbutton for one second. The output control signal goes low and the following message is displayed:

End

which indicates that the process has been aborted.

The circuit will remain in this state until you set new values. For setting the new values, follow the procedure from Step 1 onwards.

Setting and operation of timer-B are given below:

1. '0.0.0.0.' is displayed at start-up, with decimal points flashing.

2. Press Min and Hour keys to set minutes and hours for the activity.

3. Press Mode key to start the timer.

4. Now display the countdown mode.

5. At the end of the countdown mode, 'End' will be displayed and device will switch off.

6. At any time if Mode key is pressed, End message will be displayed and by pressing Mode key, the system will be at Step 1 of the operations.

The actual-size, single-side PCB for the programmable timer using AT90S4433 is shown in Fig. 5 and its component layout in Fig. 6.

Software Timer-A is used for the cyclic 'on'/'off' functionality, while software Timer-B is used for switching on the appliance after a set time period has elapsed.

Further improvements

The software can be modified for:

1. Changing the sequence of 'on' and 'off' time period activation.

2. Operation of 'on' and 'off' time periods for a definite number of times.

3. Synchronisation with external activity.

4. Controlling multiple appliances. ●

Download source code: http://www.efymag.com/admin/issuepdf/Timer.zip

Ms Dhingra, B.E., has worked with Punjab Communications Ltd and is now a lecturer at Govt. College of Engineering & Technology, Jammu. Mr Oberoi, M.Tech, has worked with Electronics System Punjab Ltd and is now principal design engineer at DOEACC Centre in Jammu/Srinagar

MANUAL AT89C51 PROGRAMMER

■ R. JEYARAMAN AND R. LAKSHMANAN

For hand coding of AT89C51 microcontroller, here is a simple programmer to program binary data into the microcontroller. It doesn't require use of a computer and all the control signals and data are entered manually. The programmer can erase, read and write data into the flash memory of the microcontroller. It can also read the signature byte of the microcontroller.

Circuit description

Fig. 1 shows the circuit of the manual programmer. It uses timer NE555 (IC1) wired as a monostable to increment the address location one by one in conjunction with counter CD4040.

When switch S1 is pressed, IC1 generates a clock pulse, which is given to clock pin 10 of 12-bit binary counter CD4040 (IC2). The 12-bit counter is used for generating the 12-bit binary address for AT89C51 microcontroller IC. The microcontroller to be programmed is inserted into the ZIF socket for programming, reading and erasing. During flash programming and verification, port 1 receives low-order address bytes, while port 2 receives high-order address bits and some control signals.

Q0 through Q11 outputs of IC2 are connected to pins 1 through 7 of port 1 and pins 21 through 24 of port 2 of the microcontroller. These pins act as the address pins (A0 through A11) of the microcontroller. The outputs of IC2 are also connected to LED1 through LED12 via current-limiting resistors R4 through R15, respectively. LED1 through LED12 indicate address location for the microcontroller IC. Switch S2 is used to reset the counter (IC2).

Port 0 receives code bytes during flash programming and outputs them during program verification. Port-0 pins 39 down to 32 are used as data pins for the microcontroller, and connected to pins 1 through 8 of DIP switch SW1 as well as LED13 through LED20 via current-limiting resistors R16 through R23. The LEDs indicate data at/for the addressed location of the microcontroller. Port pins 16, 17, 27 and 28 are controlled by slide switches S6, S7, S8 and S9, respectively, as per the table given below.

Programming pulse (ALE/$\overline{PROG}$) and programming voltage ($\overline{EA}$/Vpp) are derived with the help of IC3 and transistors T1 through T3. Timer IC3 generates and determines the programming and erasing pulse time. It is configured as a monostable whose time period is decided by resistor R24 and capacitors C6 and C7. Capacitors C6 and C7 are selected by slide switch S4 for programming and erasing, respectively. The triggering pulse is applied through a high-pass R-C network (comprising C5 and R28) and diode D5.

Output pin 3 of IC3 is connected to the base of transistor T1 via resistor R27. The high pulse output of IC3 drives transistor T1 and provides low pulse to pin 30 of the microcontroller. LED21 glows to indicate application

PARTS LIST

Semiconductors:

IC1, IC3	– NE555 timer
IC2	– CD4040 12-bit binary counter
IC4	– 7812 12V regulator
IC5	– 7805 5V regulator
IC6	– AT89C51 microcontroller
T1-T3	– BC548 npn transistor
D1-D4	– 1N4007 rectifier diode
D5	– 1N4148 switching diode
LED1-LED12, LED22-LED29	– Red LED
LED13-LED20	– Green LED
LED21	– Yellow LED

Resistors (all ¼-watt, ±5% carbon, unless mentioned otherwise):

R1-R3, R31-R33	– 10-kilo-ohm
R4-R23	– RNW1, RNW2, RNW3
R35-R42	– 220-ohm
R24, R26	– 1-kilo-ohm
R27, R29	– 2.7-kilo-ohm
R28	– 15-kilo-ohm
R30	– 100-ohm
R34	– 4.7-kilo-ohm
R25	– 12-kilo-ohm
RNW4	– 4.7-kilo-ohm×8 (SIP9)

Capacitors:

C1	– 0.47µF, 16V electrolytic
C2, C8	– 0.01µF ceramic disk
C3, C4, C12, C13	– 33pF ceramic disk
C5, C9, C10	– 0.1µF ceramic disk
C6	– 10µF, 16V electrolytic
C7	– 1µF, 16V electrolytic
C11	– 1000µF, 35V electrolytic

Miscellaneous:

X1	– 230V AC to 15V, 300mA secondary transformer
X_{TAL}	– 4MHz crystal
S1-S3	– Push-to-on switch
S4-S9	– Slide switch
SW1	– 8-way DIP switch
	– 40-pin ZIF socket

Flash Programming Modes

Mode	Pin 9	Pin 29	Pin 30	Pin 31	Pin 27	Pin 28	Pin 16 WR	Pin 17 RD
	RST	$\overline{PSEN}$	ALE/$\overline{PROG}$	$\overline{EA}$/V_{PP}	P2.6	P2.7	P3.6	P3.7
Write code data	H	L	⌒	H/12V	L	H	H	H
Read code data	H	L	H	H	L	L	H	H
Chip erase	H	L	⌒ (1)	H/12V	H	L	L	L
Read signature byte	H	L	H	H	L	L	L	L

Note: 1. Chip erase requires a 10ms PROG pulse

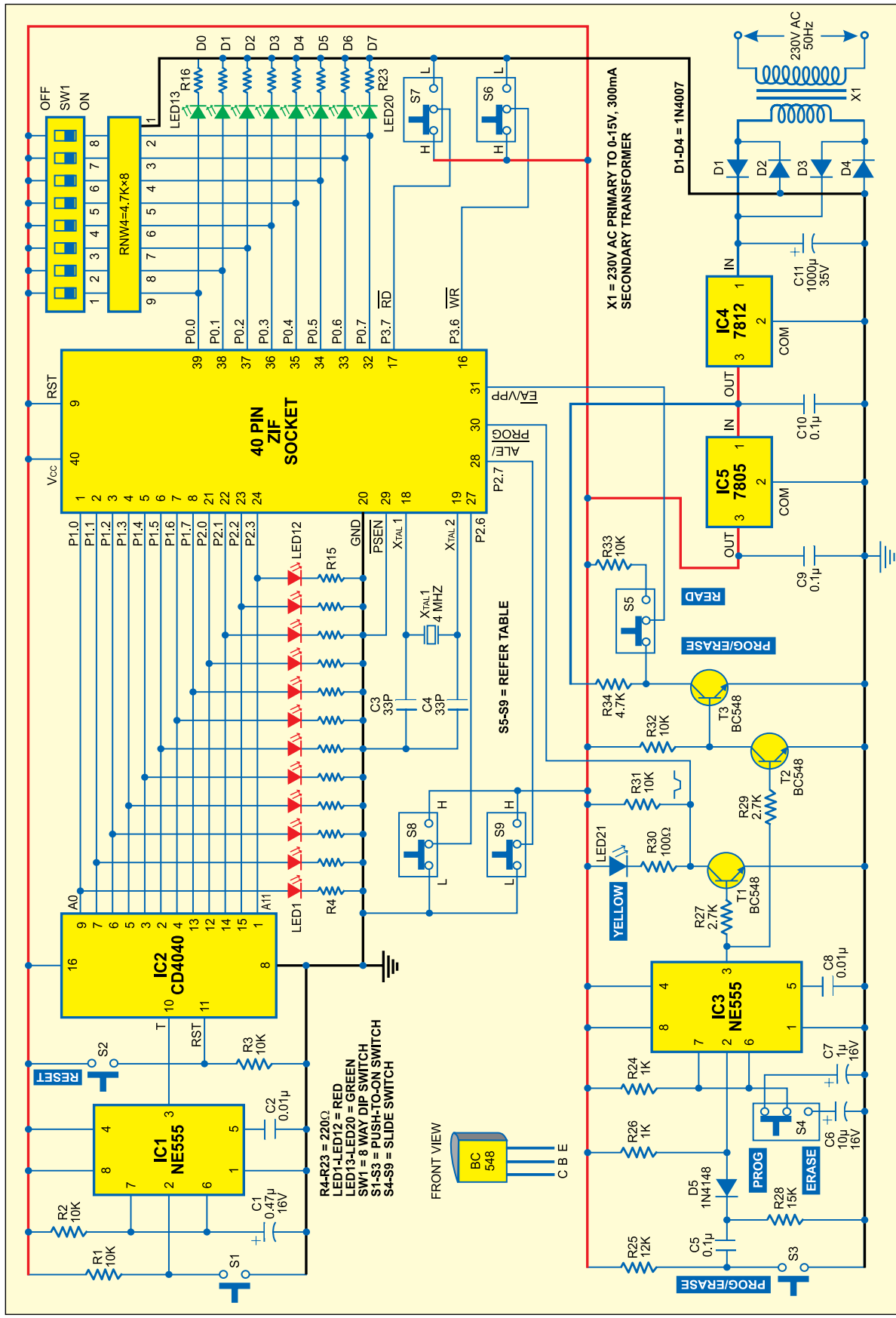


Fig. 1: Circuit diagram of manual AT89C51 programmer

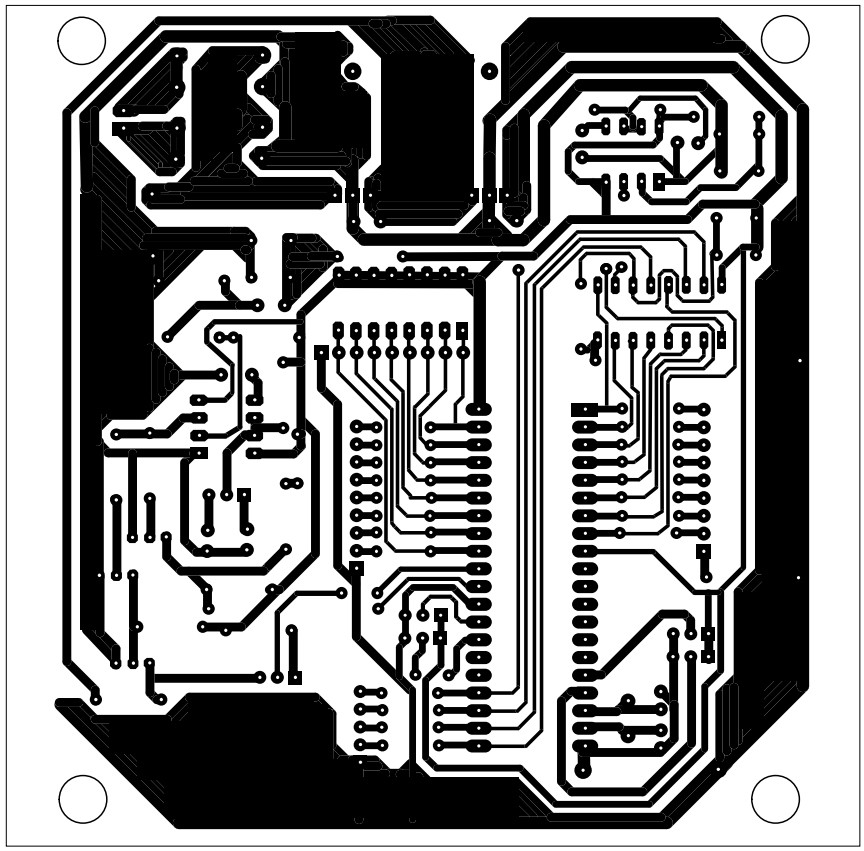


Fig. 2: Actual-size, single-side PCB layout for manual AT89C51 programmer of Fig. 1

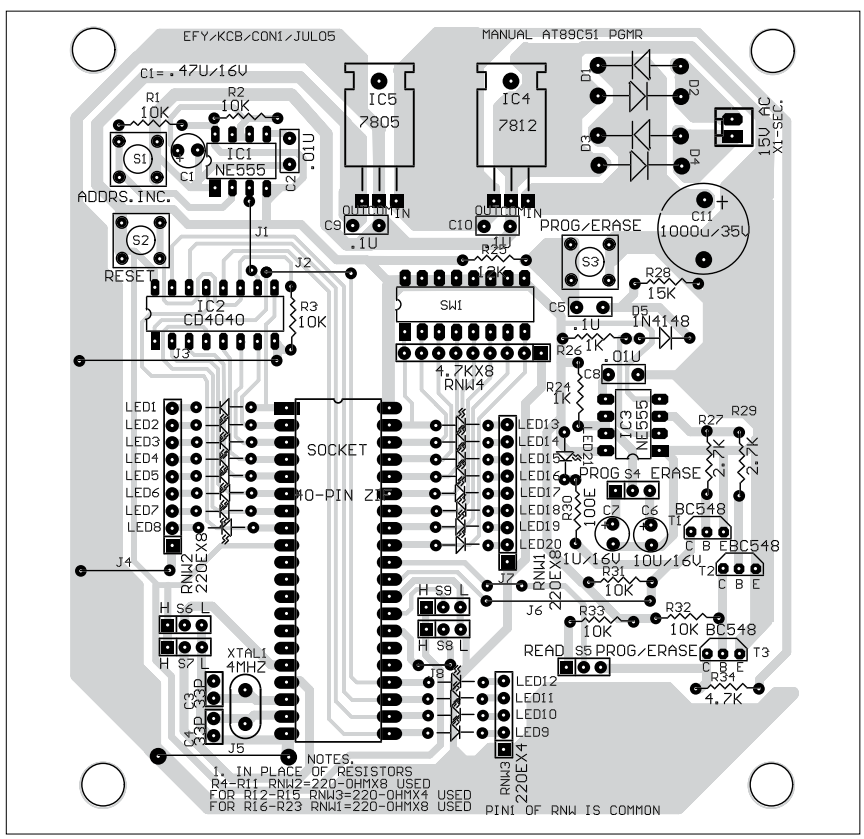


Fig. 3: Component layout for the PCB in Fig. 2

of the programming/erasing pulse. IC3 also determines the time period for application of the programming/erasing voltage (12V) to the microcontroller. When its output goes high, npn transistor T2 conducts and its collector goes low. As a result, transistor T3 is cut off and its collector voltage rises to around 12V. Slide switch S5 is used for selecting 12V or 5V as required.

The AC mains is stepped down by transformer X1 to deliver a secondary output of 15V at 300 mA. The transformer output is rectified by a full-wave bridge rectifier comprising diodes D1 through D4, filtered by capacitor C11, and regulated by IC4 and IC5 to provide regulated 12V and 5V supplies, respectively. Capacitors C9 and C10 bypass any ripple in the regulated outputs.

Programming

The AT89C51 microcontroller is programmed byte by byte. Before programming, the flash memory must be erased completely and control signals should be set according to the table for erasing, programming and reading the microcontroller as follows.

Chip erase. Move slide switch S4 towards Erase position. The high output at pin 3 of IC3 provides a low pulse of around 10 milliseconds (ms) available at pin 30 of the microcontroller (ALE/PROG). Now move slide switch S5 towards Prog/Erase position for programming voltage (12V). The entire flash array can be erased electrically by setting pins 28, 16 and 17 to low level using switches S9, S6 and S7, respectively, and pin 27 at high level using switch S8. Insert the microcontroller into the ZIF socket and press switch S3 (LED21 glows to indicate application of the programming pulse) to erase the flash array (reset to all 1's).

Write code. Press switch S2 to reset the address counter to address '0000H.' In this state, LED1 through LED12 are off. Move slide switch S4 towards Prog position and switch S5 towards Prog/Erase position. Make control signal

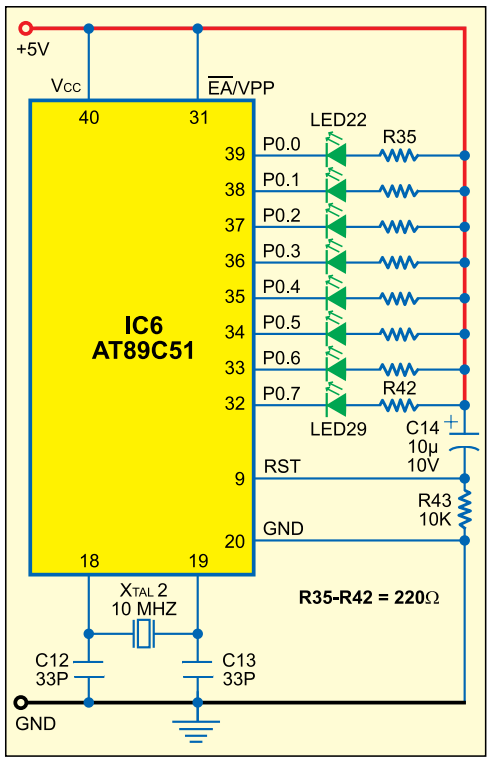


Fig. 4: Circuit of the example ring counter

press switch S1 momentarily to increment the address (indicated by LEDs). Set the code data using DIP switch SW1 and program it by pressing switch S3. Repeat this process until the program code completes.

Read/verification. Move slide switch S5 towards Read position and DIP switch SW1 towards Off position. Make control signal pins 27 and 28 low using switches S8 and S9, and pins 16 and 17 high using switches S6 and S7, respectively. Reset the counter using switch S2 and read/verify data at location '0000H.' Increment the address counter using switch S1. The data at the incremented location (say, 0001H) can be seen on LED13 through LED20. Again press switch S1 and see the data at the incremented address on the LEDs.

Reading the signature bytes. Signature bytes can be read using the same procedure as used for normal verification of locations 030H, 031H and 032H, except that all the control pins (pins 27, 28, 16 and 17) must be pulled low using switches S8, S9, S6 and S7, respectively. The values returned would be:

(030H) = 1EH indicates 'manufactured by Atmel'

(031H) = 51H indicates '89C51'

(032H) = FFH indicates '12V programming'

(032H) = 05H indicates '5V programming'

Note. The present circuit is meant for 12V programming.

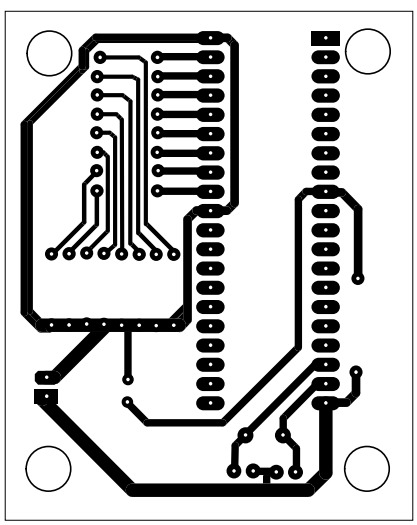


Fig. 5: Actual-size, single-side PCB layout for the ring counter

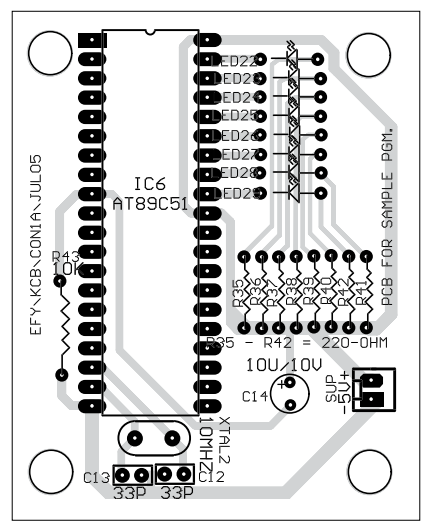


Fig. 6: Component layout for the PCB in Fig. 5

pins 28, 16 and 17 high using switches S9, S6 and S7, respectively, and pin 27 low using switch S8. Set the code data using DIP switch SW1. The data (D0 through D7) being set is indicated by LED13 through LED20. Program the code data by pressing switch S3. (LED21 glows to indicate application of the programming pulse.)

For programming the next data,

RING.LST

Addr.	OPCode	Line	Mnemonics
		1	$MOD52
0000		2	ORG 0000H
0000	747F	3	MOV A , #07FH
0002	F580	4	MOV 80H,A
0004	1140	5	ACALL DELAY
0006	74BF	6	MOV A , #0BFH
0008	F580	7	MOV 80H,A
000A	1140	8	ACALL DELAY
000C	74DF	9	MOV A , #0DFH
000E	F580	10	MOV 80H,A
0010	1140	11	ACALL DELAY
0012	74EF	12	MOV A , #0EFH
0014	F580	13	MOV 80H,A
0016	1140	14	ACALL DELAY
0018	74F7	15	MOV A , #0F7H
001A	F580	16	MOV 80H,A
001C	1140	17	ACALL DELAY
001E	74FB	18	MOV A, #0FBH
0020	F580	19	MOV 80H,A
0022	1140	20	ACALL DELAY
0024	74FD	21	MOV A, #0FDH
0026	F580	22	MOV 80H,A
0028	1140	23	ACALL DELAY
002A	74FE	24	MOV A, #0FEH
002C	F580	25	MOV 80H,A
002E	1140	26	ACALL DELAY
0030	0100	27	AJMP 0000H
		28	
0040		29	ORG 0040H
0040	7FFF	30	DELAY: MOV R7, #0FFH
0042	7DFF	31	LOOP1: MOV R5, #0FFH
0044	1D	32	LOOP: DEC R5
0045	ED	33	MOV A, R5
0046	70FC	34	JNZ LOOP
0048	1F	35	DEC R7
0049	EF	36	MOV A, R7
004A	70F6	37	JNZ LOOP1
004C	22	38	RET
		39	END

VERSION 1.2k ASSEMBLY COMPLETE, 0 ERRORS FOUND

Construction

A single-side PCB layout for the manual programmer (Fig. 1) is shown in Fig. 2 and its component layout in Fig. 3.

Sample program

To test the working of the programmer kit, here's a program (RING.LST) for a continuously running ring counter:

The circuit of the ring counter is shown in Fig. 4. Its PCB layout is shown in Fig. 5 and component layout in Fig. 6.

Download source code: http://www.efymag.com/admin/issuepdf/Manual%2089c51%20Programmer.zip ●

COMPUTERISED ELECTRICAL EQUIPMENT CONTROL

■ V. MARIYAPPAN

Controlling electrical devices from a PC is great fun. Here is a Windows-based program written in 'C' language for controlling up to eight devices from the PC's parallel port termed as printer port (LPT). The program accepts the input in decimal numbers and outputs at the data output pins of the PC's parallel port for controlling the connected devices.

PC's parallel port

The parallel port is made up of three ports, namely, data port, status port and control port. It is found on the back of the PC as a D-type, 25-pin female connector. Here, we are concerned only with data lines D0 through D7 terminated at pins 2 through 9.

The data port is a write-only port, which means it can be used only to output data. Pins 18 through 25 of the connector are grounded. Control port is read/write capable, which means it can be used both for outputting and inputting some data to/from the external hardware. Status port is a read-only port, which means it can be used only to read data from the external hardware.

Table below shows pin details of the standard parallel port (SPP) and their traditional usage. The base address of the first parallel port (LPT1) is 378 (hex) or 888 (decimal). The data port of the parallel port can be accessed at its base address. The status port can be accessed at base address + 1, i.e., 0379 hex (or 889 decimal). The control port can be accessed at base address + 2, i.e., 037A hex (or 890 decimal). In case you are using LPT2 port, then substitute the base address of LPT2 as 0278 (hex) in place of 0378 (hex).

PARTS LIST

Semiconductor:

IC1-IC4	- CD4013 D-type flip-flop
IC5-IC12	- MCT2E optocoupler
IC13	- 7805 5V regulator
T1-T8	- BC548 npn transistor
D1-D8	- 1N4007 rectifier diode
BR1	- 1A, bridge rectifier

Resistors (all ¼-watt, ±5% carbon, unless stated otherwise):

R1-R8	- 100-ohm
R9-R17	- 10-kilo-ohm

Capacitors:

C1	- 1000µF, 25V electrolytic
C2	- 0.1µF ceramic

Miscellaneous:

RL1-RL8	- 5V, 100-ohm, 1C/O relay
S1	- Push-to-on switch
X1	- 230V AC primary to 9V, 250mA secondary transformer
	- 25-pin D-type parallel-port male connector

Pin Details of the Parallel Port

Pin number	Traditional use	Port name	Read/Write	Port address	Port bit
2-4	Data out	Data port	W	Base	D0-D2
5-9	Data out	—	W	Base	D3-D7
1	Strobe	Control port	R/W	Base+2	C0
14	Auto feed	—	R/W	Base+2	C1
16	Initialise	—	R/W	Base+2	C2
17	Select input	—	R/W	Base+2	C3
15	Error	Status port	R	Base+1	S3
13	Select	—	R	Base+1	S4
12	Paper end	—	R	Base+1	S5
10	ACK	—	R	Base+1	S6
11	Busy	—	R	Base+1	S7

Fig. 1 shows the circuit for interfacing the PC's parallel port to the devices to be controlled. The parallel port outputs the control signals generated by the software. The control signals are not continuous but a single clock pulse. For every 'on' or 'off' control, only a single clock pulse is sent from the parallel port to the circuit.

Data pins D0 through D7 of the parallel port are connected to pin 1 of optocouplers IC5 through IC12 via resistors R1 through R8, respectively. Optocouplers ensure complete isolation of the parallel port's data pins from the relay driver circuit.

Each optocoupler consists of an infrared light-emitting diode (LED) and an npn phototransistor. When a high going pulse is available on the data pin, the internal LED drives the phototransistor of optocoupler MCT2E and it provides a clock pulse to the corresponding flip-flop (IC CD4013) section.

IC CD4013 is a dual D-type flip-flop with independent set, clear and clock inputs and a single output. It accepts data when its clock pin is low and transfers it to the output on the positive-going edge of the clock. The high Q output of the flip-flop drives the corresponding transistor to energise the relay and switch on/off the device.

The flip-flops are set up for toggle mode by connecting their D inputs to $\overline{Q}$ outputs. Set inputs of all the flip-flops are grounded. Switch S1 is used to reset the flip-flops manually.

Fig. 2 shows the circuit of the power supply. The AC mains is stepped down by transformer X1 to deliver secondary output of 9V at 250 mA. The transformer output is rectified by full-wave bridge rectifier BR1, filtered by capacitor C1 and regulated by IC13 to

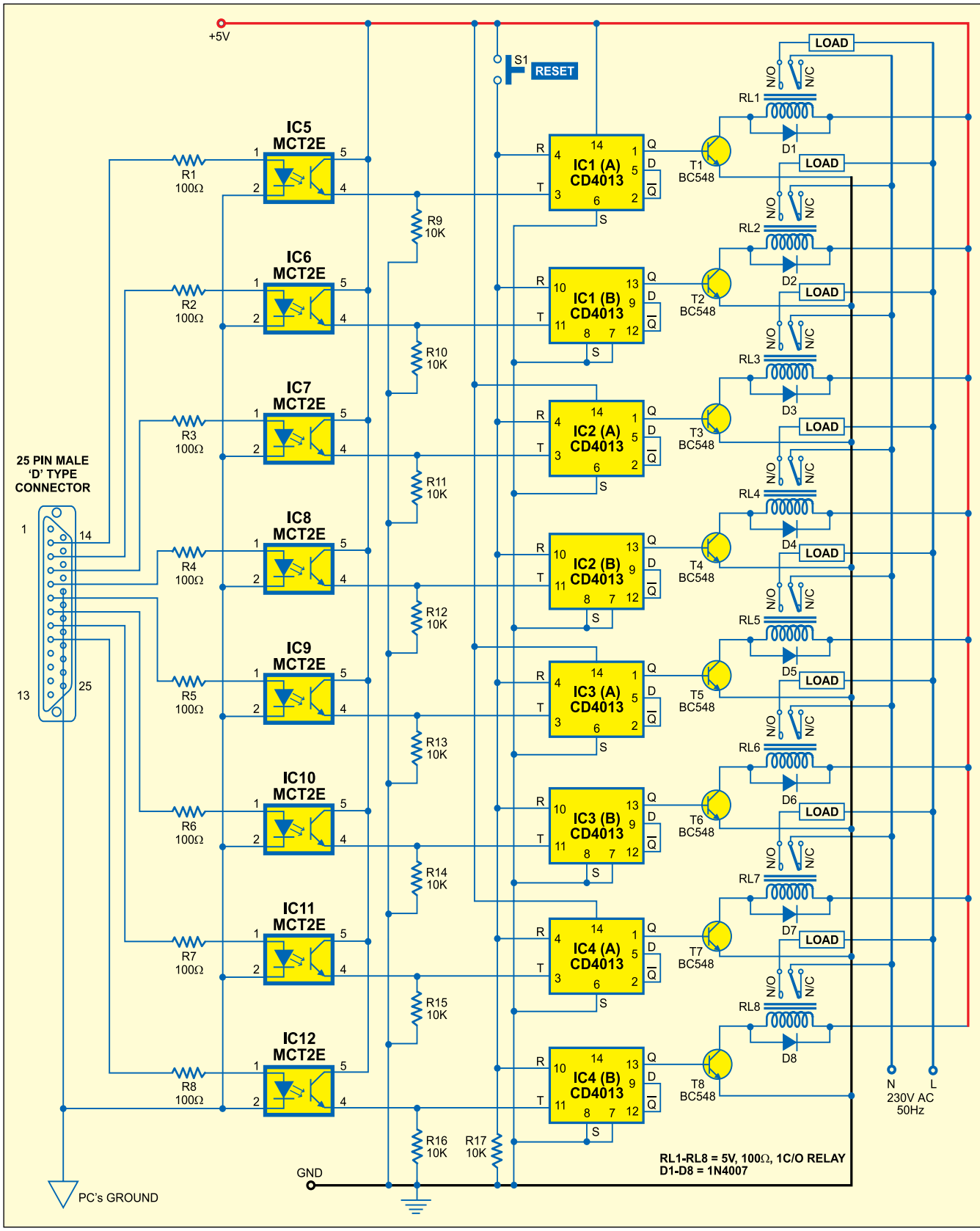


Fig. 1: Circuit diagram for computerised electrical equipment control

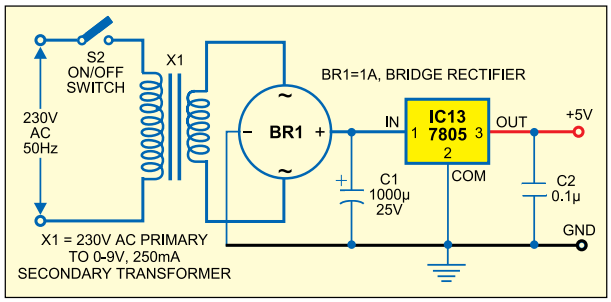


Fig. 2: Power supply circuit

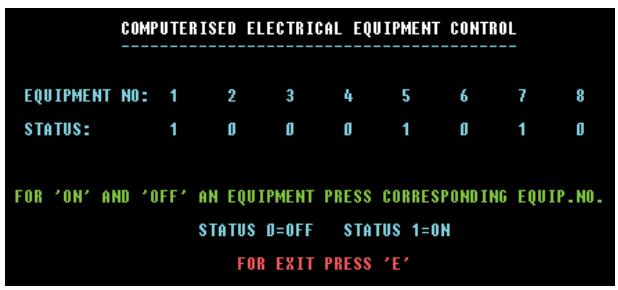


Fig. 3: Screenshot of the main screen for computerised electrical equipment control

provide regulated 5V supply. Capacitor C2 bypasses any ripple in the regulated output.

Before attempting to control the devices, the program (Control.exe) is to be run. A screen as shown in Fig. 3 appears which shows the current status of each device (on=1, off=0). To switch on a device, press the corresponding device number on the keyboard. This makes a high control pulse available on the data pin of the respective parallel-port pin and the toggle flip-flop switches from 'off' state to 'on' state to energise the relay and switch on the device. The status of the corresponding device on screen changes automatically.

Similarly, to turn the device 'off,' press the device number on the keyboard. This again makes a high control pulse available on the data pin and the corresponding toggle flip-flop switches from 'on' state to 'off' state to de-energise the relay and turn the device 'off.'

The program can be terminated simply by pressing the 'E' key, but before that you should turn all the devices 'off.' If you try to terminate the program without shutting down any of the devices, the message "Please shutdown all the equipment" will appear on the screen for a short period followed by the main screen.

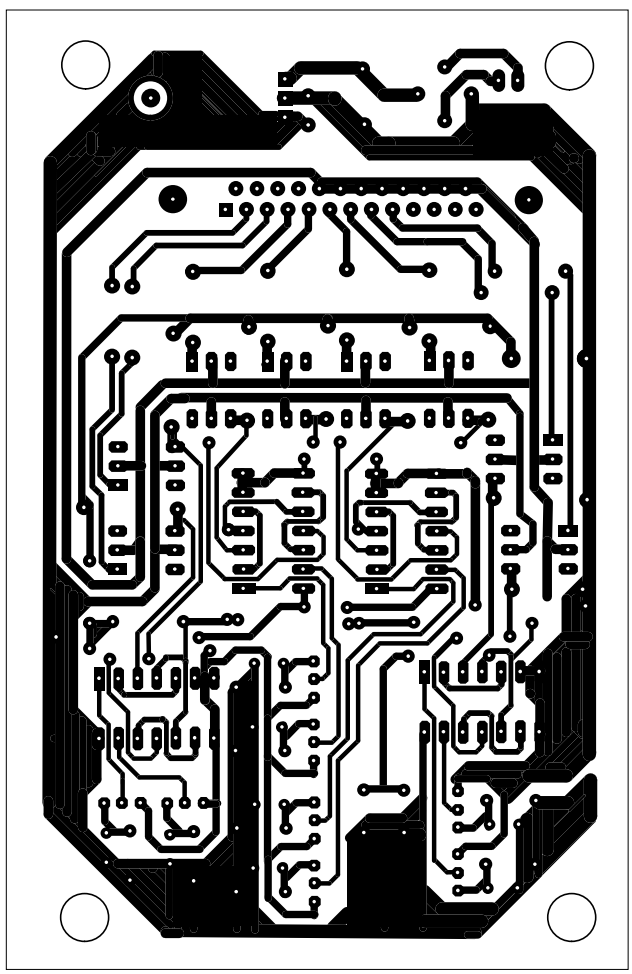


Fig. 4: Combined actual-size, single-side PCB layout for computerised electrical equipment control and power supply circuits

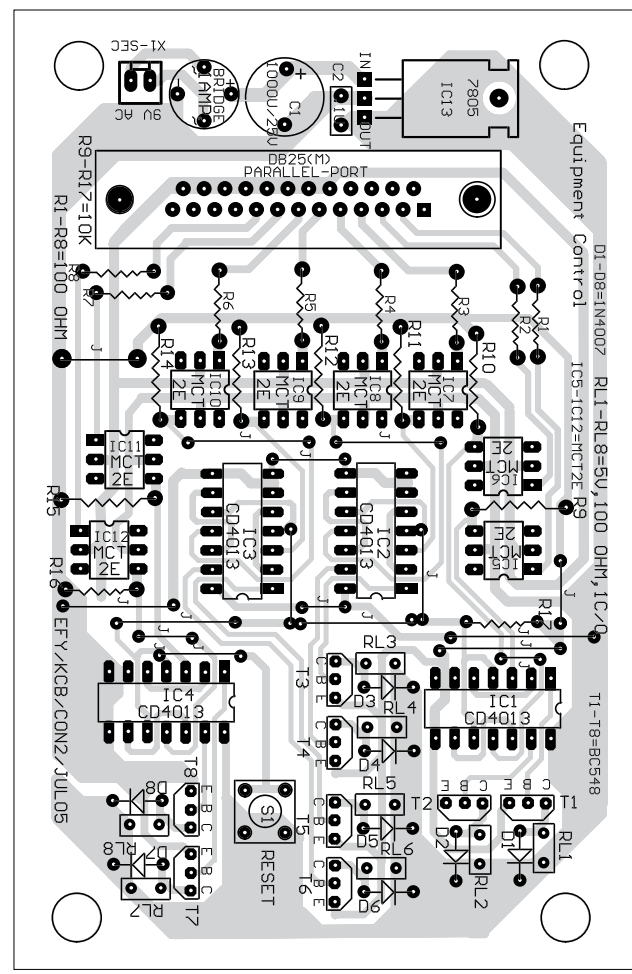


Fig. 5: Component layout for the PCB

In case you press any invalid key (not used in the software), the message "Invalid key pressed wait 2 seconds" is displayed followed by the main screen.

The combined single-side PCB layout for the equipment control and power supply circuits is shown in Fig. 4 and and its component layout in Fig. 5.

Download source code: http://www.efymag.com/admin/issuepdf/Equipment%20Control.zip

CONTROL.C

```
/*COMPUTERISED ELECTRICAL EQUIPMENT CONTROL*/
#include<stdio.h>
#include<conio.h>
#include<dos.h>
void main()
{
void tone(void);
int p=0x0378;
char ex[23]={"Created by V.MARIYAPPAN"};
int j;
char ex1[34]={"For Further Details & Improvements"};
int k;
char ex2[40]={"Contact: Email-marietech2003@yahoo.co.in"};
int l;
char ex3[23]={"Programming Language: C"};
int m;
int u[10];
int i;
static a,b,c,d,e,f,g,h;
char no;
clrscr();
textcolor(15);gotoxy(20,6);
cprintf("COMPUTERISED ELECTRICAL EQUIPMENT CONTROL");
textcolor(11);gotoxy(20,7);
cprintf("----------");
textcolor(11);gotoxy(10,10);
cprintf("EQUIPMENT NO: 1 2 3 4 5 6 7 8");
textcolor(11);gotoxy(10,12);
cprintf("STATUS: %d %d %d %d %d %d %d %d",a,b,c,d,e,f,g,h);
textcolor(10);gotoxy(9,16);
cprintf("FOR 'ON' AND 'OFF' AN EQUIPMENT PRESS CORRESPONDING EQUIP.NO.");
textcolor(11);gotoxy(28,18);
cprintf("STATUS 0=OFF STATUS 1=ON");
textcolor(12);gotoxy(32,20);
cprintf("FOR EXIT PRESS 'E'\n");
no=getch();
switch(no)
 {
 case '1':
 a=!a;
 tone();
 outportb(p,1);
 delay(500);
 outportb(p,0);
 break;

 case '2':
 b=!b;
 tone();
 outportb(p,2);
 delay(500);
 outportb(p,0);
 break;

 case '3':
 c=!c;
 tone();
 outportb(p,4);
 delay(500);
 outportb(p,0);
 break;

 case '4':
 d=!d;
 tone();
 outportb(p,8);
 delay(500);
 outportb(p,0);
 break;

 case '5':
 e=!e;
 tone();
 outportb(p,16);
 delay(500);
 outportb(p,0);
 break;

 case '6':
 f=!f;
 tone();
 outportb(p,32);
 delay(500);
 outportb(p,0);
 break;

 case '7':
 g=!g;
 tone();
 outportb(p,64);
 delay(500);
 outportb(p,0);
 break;

 case '8':
 h=!h;
 tone();
 outportb(p,128);
 delay(500);
 outportb(p,0);
 break;

 case 'e':
 if((a|b|c|d|e|f|g|h)==1)
 {
 clrscr();
 textcolor(10);gotoxy(20,12);
  cprintf("PLEASE SHUT DOWN ALL THE EQUIPMENTS");
 sound(200);
 delay(500);
 nosound();
 delay(3000);
 break;
 }
 else
 {
 clrscr();
 for(j=0;j<23;j++)
 {
 textcolor(10);gotoxy(20+j,12);
 cprintf("%c",ex[j]);
 sound(3000+j);
 delay(30);
 nosound();
 }
 for(m=0;m<23;m++)
 {
 textcolor(10);gotoxy(20+m,13);
 cprintf("%c",ex3[m]);
 sound(1800+m);
 delay(30);
 nosound();
 }
 for(k=0;k<34;k++)
 {
 textcolor(10);gotoxy(20+k,14);
 cprintf("%c",ex1[k]);
 sound(2000+k);
 delay(30);
 nosound();
 }
 for(l=0;l<40;l++)
 {
 textcolor(10);gotoxy(20+l,15);
 cprintf("%c",ex2[l]);
 sound(2500+l);
 delay(30);
 nosound();
 }
 printf("\n\n\n\nPress any key");
 getch();
 outportb(p,0);exit(0);
 }

 case 'E':
 if((a|b|c|d|e|f|g|h)==1)
 {
 clrscr();
 textcolor(10);gotoxy(20,12);
  cprintf("PLEASE SHUT DOWN ALL THE EQUIPMENTS");
 sound(200);
 delay(500);
 nosound();
 delay(3000);
 break;
 }
 else
 {
 clrscr();
 for(j=0;j<23;j++)
 {
 textcolor(10);gotoxy(20+j,12);
 cprintf("%c",ex[j]);
 sound(2500+j);
 delay(30);
 nosound();
 }
 for(m=0;m<23;m++)
 {
 textcolor(10);gotoxy(20+m,13);
 cprintf("%c",ex3[m]);
 sound(3500+m);
 delay(30);
 nosound();
 }
 for(k=0;k<34;k++)
 {
 textcolor(10);gotoxy(20+k,14);
 cprintf("%c",ex1[k]);
 sound(3000+k);
 delay(30);
 nosound();
 }
 for(l=0;l<40;l++)
 {
 textcolor(10);gotoxy(20+l,15);
 cprintf("%c",ex2[l]);
 sound(3500+l);
 delay(30);
 nosound();
 }
 printf("\n\n\n\nPress any key");
 getch();
 outportb(p,0);exit(0);

 }
 default:
 clrscr();
 sound(500);
 delay(100);
 nosound();
 textcolor(11);gotoxy(30,12);
 cprintf("INVALID KEY PRESSED");
 textcolor(11);gotoxy(33,14);
 cprintf("WAIT 2 SECONDS");
 delay(3000);
 break;
 }
 main();
 }
 void tone(void)
 {
 sound(1000);
 delay(100);
 nosound();
 }
```

REMOTE-CONTROLLED STEPPER MOTOR

■ N.C. RAGHAV ANAND

Here is a stepper motor system wherein the direction of rotation of the stepper motor (in clockwise and anticlockwise directions) can be controlled remotely. Besides, the speed can also be controlled locally.

Stepper motor basics

A stepper motor converts electrical pulses into specific rotational movements. The movement created by each pulse is precise and repeatable.

Stepper motors have teeth on both the rotor and the stator. Torque is generated by alternately magnetising the stator teeth electrically, and the permanent magnet rotor teeth try to align up with the stator teeth.

The coils are arranged around the circumference of the stator in such a way that if they are driven with square waves which have a quadrature phase relationship between them, the motor will rotate. A transition of either square wave causes the rotor to move by a small angular 'step,' hence the name 'stepper motor.'

The size of this angular step is dependent on the teeth arrangement of the motor, but a common value is 1.8 degrees, or 200 steps per revolution. Speed control is achieved by simply varying the frequency of the square waves.

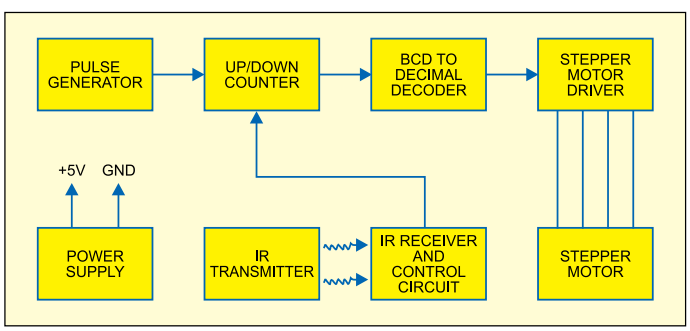


Fig. 1: Block diagram of IR remote control system for stepper motor

System overview

Fig. 1 shows the block diagram of the IR remote control system for the stepper motor. The pulse generator provides clock pulse to the up/down counter. The four parallel BCD outputs of the counter are converted into one-of-ten active-high outputs by the BCD-to-decimal decoder. The decoded outputs are fed to the stepper motor driver to drive the stepper motor.

The 38kHz infrared signal transmitted by the IR transmitter is received by the IR receiver to control the direction of rotation of the stepper motor. The pulse generator can control the speed of the motor.

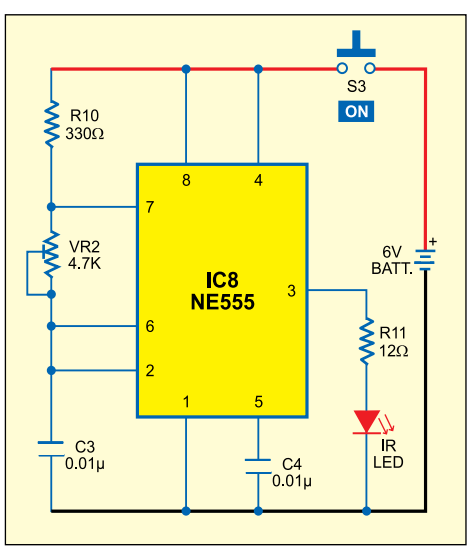


Fig. 2: IR transmitter

Circuit description

IR transmitter. Fig. 2 shows the circuit of the IR transmitter. The transmitter circuit, powered by a 6V battery, is built around timer NE555 (IC8), which is wired as an astable multivibrator having a frequency of around 38 kHz.

The frequency of the astable is decided by resistor R10, preset VR2 and capacitor C3. Preset VR2 is used to set the frequency to 38 kHz. The output of IC8 is fed to an infrared LED via current-limiting resistor R11. When switch S3 is pressed, the IR LED trans-

PARTS LIST

Semiconductors:

IC1	-	TSOP1738 IR receiver module
IC2	-	CD4013 dual D-type flip-flop
IC3, IC8	-	NE555 timer
IC4	-	CD4029 up/down counter
IC5	-	CD4028 BCD-to-decimal decoder
IC6	-	ULN2803 Darlington pair driver
IC7	-	CD40106 NOT gate
IC9	-	7805C 5V regulator
T1	-	BC547 npn transistor
D1-D10	-	1N4148 switching diode
BR1	-	500mA bridge rectifier
LED1	-	Red LED
LED2	-	Green LED
	-	IR LED

Resistors (all ¼-watt, ±5% carbon):

R1, R6, R7, R10	-	330-ohm
R2	-	1-kilo-ohm
R3-R5	-	10-kilo-ohm
R8	-	3.3-kilo-ohm
R9	-	5.6-kilo-ohm
R11	-	12-ohm
VR1	-	100-kilo-ohm preset
VR2	-	4.7-kilo-ohm preset

Capacitors:

C1	-	1µF, 16V electrolytic
C2-C4	-	0.01µF ceramic disk
C5	-	1000µF, 16V electrolytic
C6	-	0.1µF ceramic disk

Miscellaneous:

X1	-	230V AC primary to 3V-0-3V, 350mA secondary transformer
S1, S2	-	Push-to-on switch
Battery	-	6V battery
	-	Stepper motor

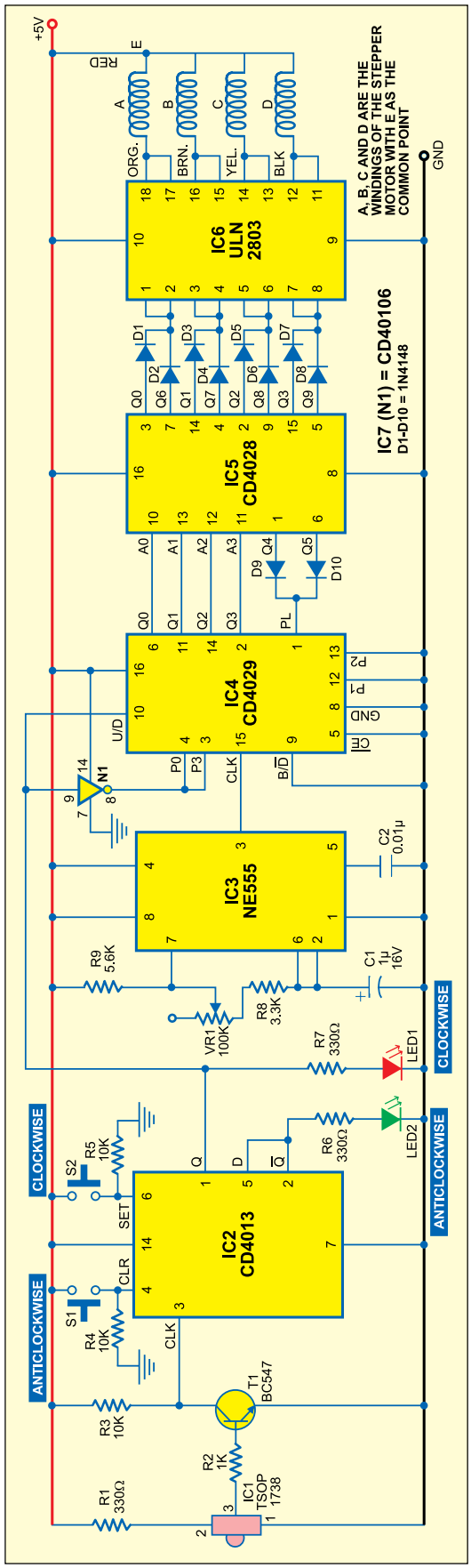


Fig. 3: IR receiver-cum-stepper motor driver circuit

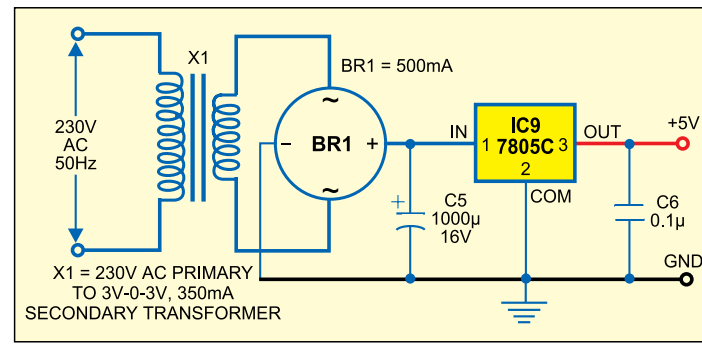


Fig. 4: Power supply

mits 38kHz modulated infrared signal.

Receiver-cum-stepper motor driver. Fig. 3 shows the circuit of the receiver-cum-stepper motor driver. The transmitted 38kHz modulated signal is received by infrared receiver module TSOP1738 (IC1) and it outputs a clock pulse to D-type flip-flop CD4013 (IC2) via transistor T1.

IC2 accepts data when its clock input is low and transfers it to the output on the positive-going edge of the clock. IC2 is configured as a toggle flip-flop as its Q output is connected to 'D' input. Thus when TSOP1738 receives a signal from the transmitter, it clocks IC2 and its Q output changes from high to low and vice versa on alternate clock inputs. Consequently, the state of Q output of IC2 controls the direction of the stepper motor. When Q output is high the stepper motor rotates in clockwise direction, and when Q output is low the stepper motor rotates in anti-clockwise direction. The direction of rotation can also be controlled manually by setting Q output high or low with the help of switches S2 and S1, respectively.

Set and Reset pins (pins 4 and 6) are normally pulled down by resistors R4 and R5, respectively. These are also connected to switches S1 and S2, respectively. LED1 and LED2 indicate the clockwise and anticlockwise direction of rotation of the stepper motor. The Q output of IC2 controls the up/down pin and parallel input pins P0 and P3 of IC4 (CD4029).

Timer NE555 (IC3) is configured as an astable multivibrator whose frequency is determined by resistors R8 and R9, preset VR1 and capacitor C1. Preset VR1 is used to vary the frequency and consequently the speed of the stepper motor.

IC3 provides clock input to up/down counter IC CD4029 (IC4). IC CD4029 is a presettable up/down counter that counts in either binary or decade mode depending on the voltage level applied at its B/$\overline{D}$ pin. The B/$\overline{D}$ input (pin 9) of IC4 is grounded to configure it as decade counter.

Counter IC4 advances one count for low-to-high transition of the clock pulse when its CE and PL pins are low. It counts up when its up/down input is high, and vice versa. The count-enable input (pin 5) and preset inputs P1 and P2 are grounded, while the parallel-load input pin (PL) is controlled by Q4 and Q5 output pins of IC5. Q0 through Q3 outputs of IC4 are connected to A0 through A3 inputs of IC5 (CD4028).

IC CD4028 is a 4-bit BCD-to-one-of-ten active-high output decoder. BCD inputs A0 through A3 make the selected output high, while the other nine outputs remain low.

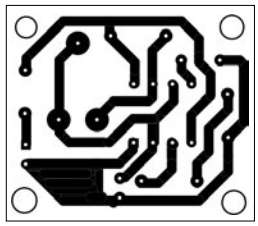


Fig. 5: Actual-size, single-side PCB for IR transmitter

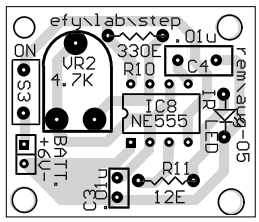


Fig. 6: Component layout for the transmitter PCB

When the Q output of IC2 goes high, the counter (IC4) is enabled for up counting with parallel inputs P0 and P3 going low. Decoder (CD4028) outputs Q0 through Q3 go high one after another according to IC4 outputs. When Q4 output of IC5 goes high, the 0000H parallel input data at P0 through P3 pins is loaded into the CD4028 (IC4). The counter starts counting afresh in up mode.

When Q output of IC2 goes low, the counter (IC4) is configured for down counting and its parallel inputs P0 and P3 become high. In down counting mode, Q9 down through Q6 outputs of the decoder (IC5) go high one after another. As soon as Q5 output of the decoder goes high, the 1001H parallel input data is loaded into the counter and it again starts counting down from Q9 through Q6.

The four outputs Q0 through Q3, and Q6 through Q9, of IC5 are ORed via diodes D1 through D8 driving the stepper motor in clockwise or anti-clockwise direction. The four ORed outputs of IC5 are connected to eight input pins (two each in parallel) of IC ULN 2803. The combined waveforms for clockwise and anti-clockwise rotation are shown in Fig. 9.

IC ULN2803 consists of eight Darlington-pair driver transistors. It is basically an inverter that when fed with positive input generates negative output. Stepper motor coils A, B, C and D are connected to output pins 17-18, 15-16, 13-14 and 11-12 of ULN2803, respectively, with their common terminal E connected to the 5V power supply.

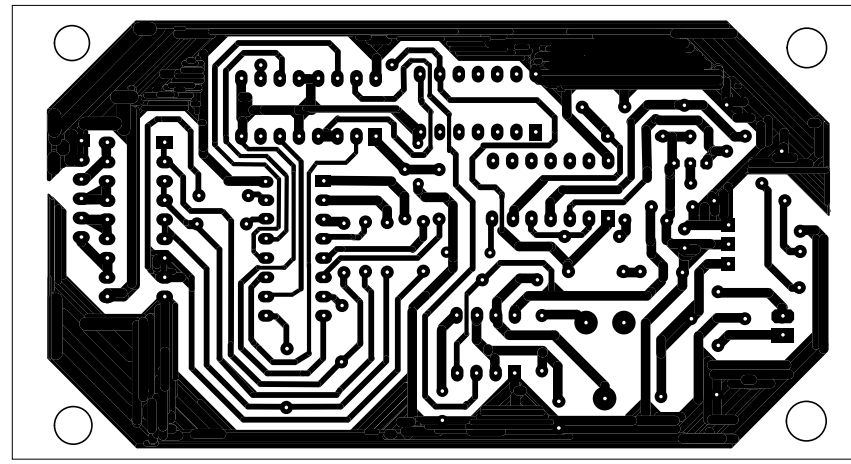


Fig. 7: Actual-size, single-side PCB for IR receiver-cum-stepper motor driver circuit

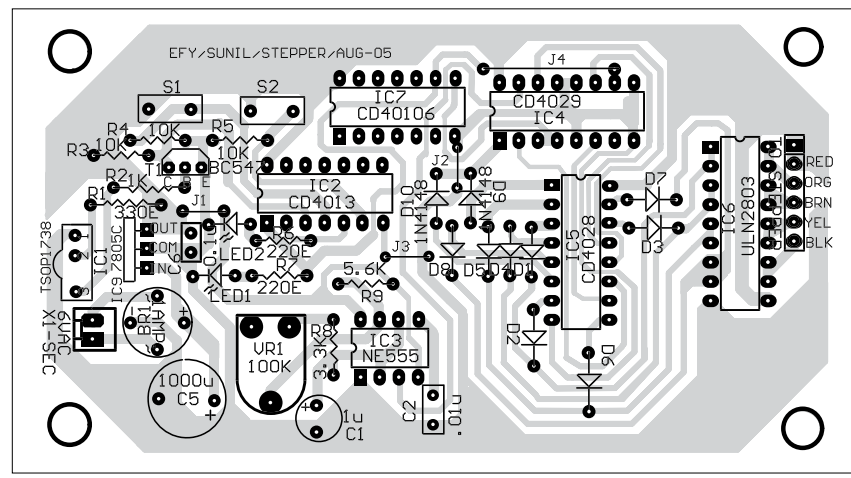


Fig. 8: Component layout for the receiver-cum-driver PCB

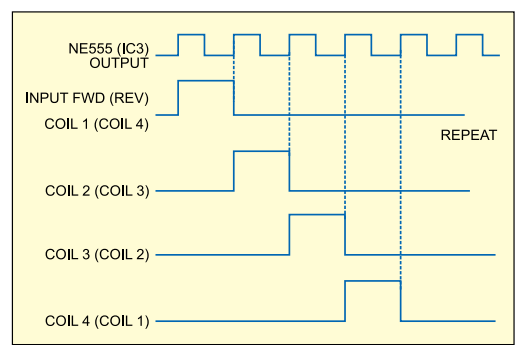


Fig. 9: Combined waveforms for clockwise and anti-clockwise rotation

The low output of IC ULN2803 provides path for the current and the coils energise one by one to rotate the stepper motor in clockwise/anticlockwise direction.

Power supply. Fig. 4 shows the circuit of the power supply. The AC mains is stepped down by transformer X1 to deliver a secondary output of 3V-0-3V, 350 mA. The transformer output is rectified by bridge rectifier BR1, filtered by capacitor C5 and regulated by IC9 to provide 5V regulated supply. Capacitor C6 bypasses any ripple in the regulated output.

Construction

Actual-size, single-side PCB layouts for the IR transmitter and receiver-cum-stepper motor driver circuits (Figs 2 and 3) are shown in Figs 5 and 7, and their component layouts in Figs 6 and 8, respectively.

Mount bases for the ICs on the PCB so that these can be removed easily when required. Normally, six wires of different colours are available for connection to the stepper motor. The colour code for connecting the stepper motor coils to the driver outputs is shown in Fig. 3. ●

DIGITAL STOPWATCH

■ **RAJ K. GORKHALI**

This digital stopwatch can count up to 99.9 seconds with a resolution of 0.1 second (or in steps of 0.1 second) or up to 999 seconds with a resolution of 1 second. This has been made possible by employing clock frequency options of 10 Hz and 1 Hz, respectively.

The stopwatch can be used to accurately measure short time intervals. It is portable and operates off four 1.5V rechargeable Ni-Cd cells.

Circuit description

Fig. 1 shows the block diagram of the digital stopwatch. It comprises clock generator IC MM5369, quad 2-input NAND gate IC CD4011B, presettable divide-by-'n' counter IC CD4018B, binary coded decimal (BCD) up-/down-counter IC 4510B, BCD-to-7 segment latch/decoder/driver IC 4511B and common-cathode display IC LTS543.

Fig. 2 shows internal connections and block diagram of IC MM5369, which is commonly used for generating clock pulses in digital timepieces. The MM5369 is a CMOS IC with 17 binary divider stages that can be used to generate a precise reference from commonly available high-frequency quartz crystals of 3.579545 MHz. An internal pulse is generated by mask programming the combinations of stages 1 through 4, 16 and 17 to set or reset the individual stages.

IC MM5369 advances by one count on the positive transition of each clock pulse. Two buffered outputs are available: the crystal frequency for tuning purposes and the 17th-stage output. The IC is available in an 8-lead as well as 14-lead dual-in-line epoxy package and features:

1. Crystal oscillator
2. High speed (4 MHz at 10V Vdd)
3. Wide supply range of 3V to 15V
4. Low power
5. Fully static operation
6. Low current

Fig. 3 shows the circuit of the digital stopwatch. IC MM5369 (IC1) along with resistor R1, capacitors C1

PARTS LIST

Semiconductor:

IC1	-	MM5369 17-stage oscillator/divider
IC2	-	CD4011B quad 2-input NAND gate
IC3, IC4	-	CD4018B presettable divide-by-'n' counter
IC5, IC6, IC7	-	CD4510B BCD up-/down-counter
IC8, IC9, IC10	-	CD4511B BCD-to-7 segment latch/decoder/driver
DIS1-DIS3	-	LTS543 common-cathode, 7-segment display
LED1	-	Red LED

Resistors (all ¼-watt, ±5% carbon):

R1	-	2.2-mega-ohm
R2, R27	-	1-kilo-ohm
R3, R4	-	22-kilo-ohm
R5-R25	-	330-ohm
R26, R28	-	680-ohm

Capacitors:

C1, C2	-	33pF ceramic disk
C3	-	0.0022µF ceramic disk

Miscellaneous:

X_{TAL}	-	3.579545MHz crystal
S1-S3	-	Miniature pushbutton microswitches
S4	-	SPDT switch
S5, S6	-	On/off switch
	-	14- and 16-pin IC bases
	-	Nickel-cadmium cells 1.5V (4 Nos)
	-	Multistrand wires
	-	Solder metal
	-	Suitable mounting cabinet

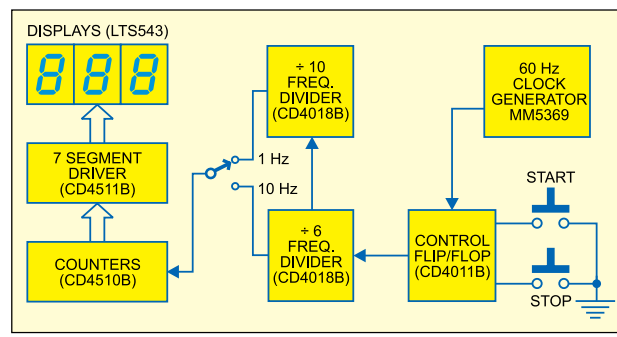


Fig. 1: Block diagram of digital stop watch

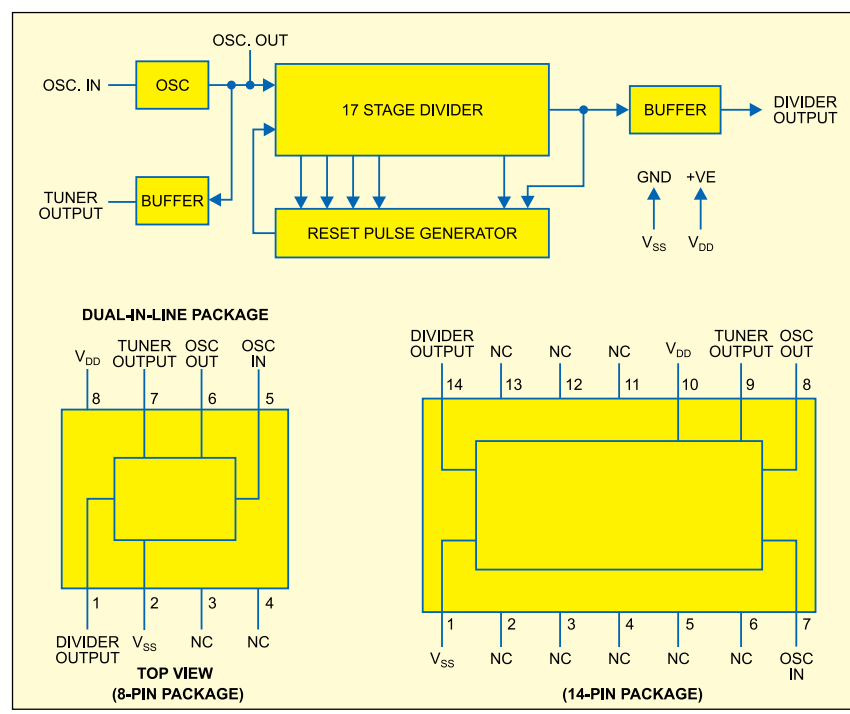


Fig. 2: Internal connections and block diagram of IC MM5369

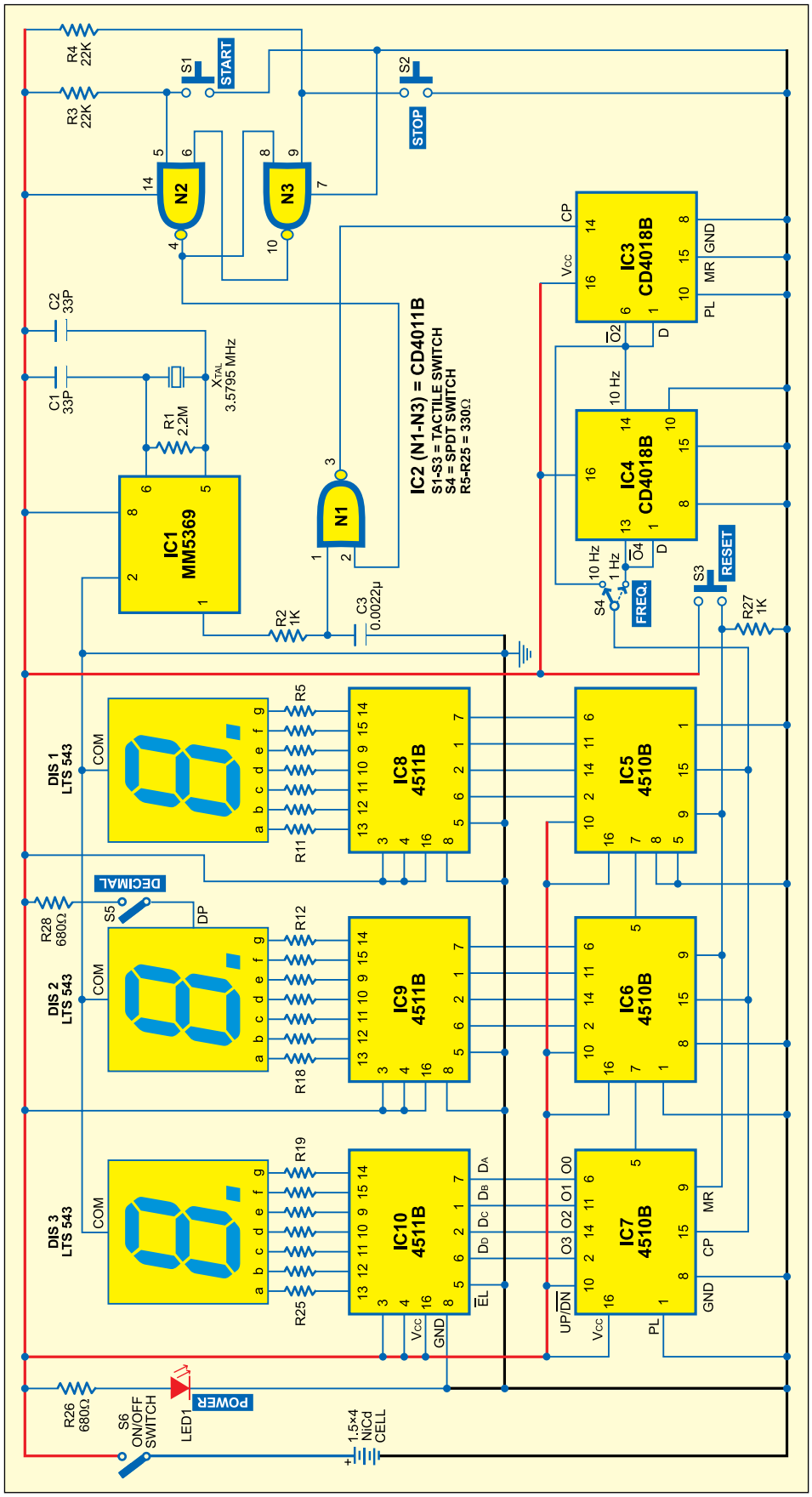


Fig. 3: Circuit of the digital stop watch

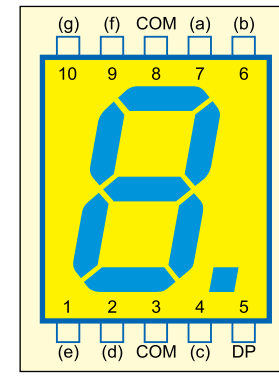


Fig. 4: Pin configuration of IC LTS543

and C2, and 3.579545MHz crystal generates a 60Hz clock signal at its output pin 1. This 60Hz clock signal is fed to IC CD4018B (IC3) via an arrangement of NAND gates.

Whenever the start switch is pressed momentarily, pin 4 and therefore pin 2 of IC CD4011B (IC2) goes high to enable NAND gate N1 and the clock signal is passed onto IC CD4018 (IC3). When stop switch S2 is pressed momentarily, NAND gate N1 is disabled due to pin 4 of IC2 going low and the clock pulses don't reach IC3.

IC3 and IC4 (each CD4018B) are presettable divide-by-'n' counters, where 'n' could be 2, 3, 4, 5, 6, 7, 8, 9 or 10. IC3 has been wired as a divide-by-6 counter by feeding its O2 output back to data input pin 1. IC4 has been wired as a divide-by-10 counter by feeding its O4 output back to data input pin 1.

For 60Hz clock input, IC3 outputs a 10Hz clock signal. This clock signal is fed to IC4 and it outputs a 1Hz clock signal. Switch S4 is used to select the required frequency.

IC5, IC6 and IC7

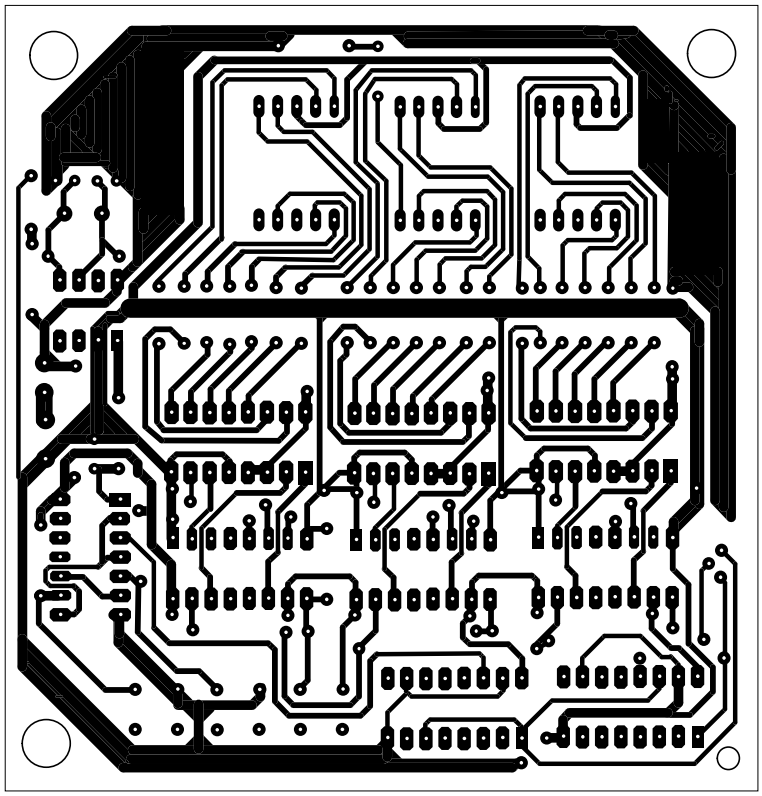


Fig. 5: Actual-size, single-side PCB layout for the digital stop watch

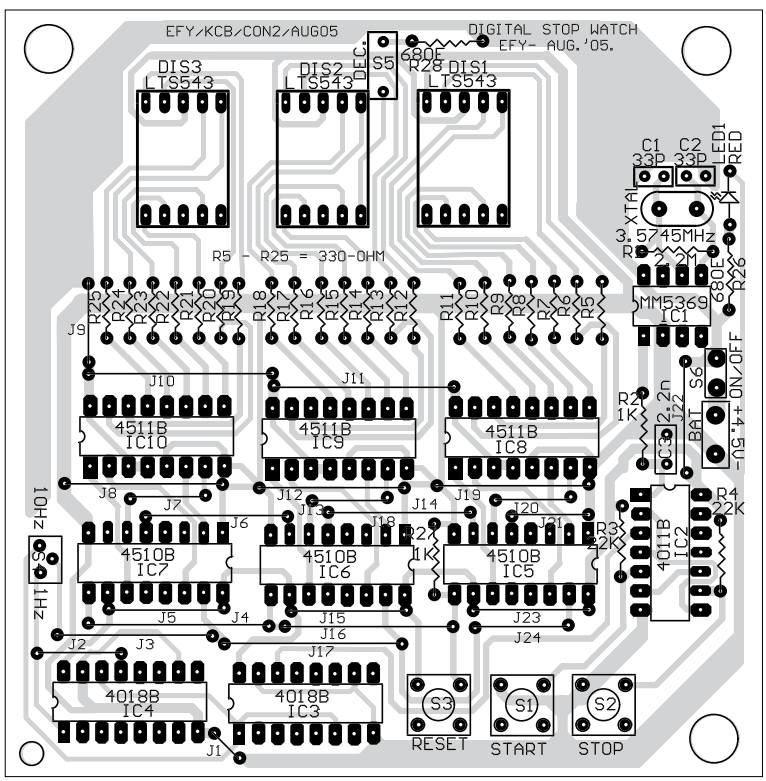


Fig. 6: Components layout for the PCB

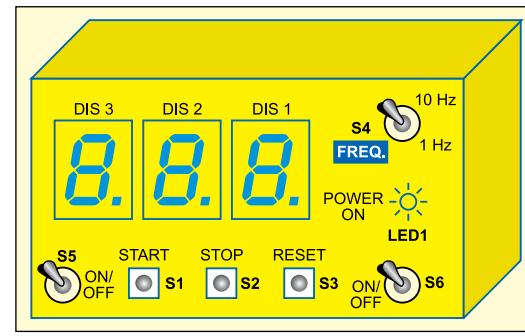


Fig. 7: Proposed display panel box

(each CD4510B) are presettable up/down BCD decade counters. These counters are connected in a cascade arrangement and can count up to 999 clock pulses. Therefore, when fed with a 1Hz clock, the maximum time delay achieved between start and stop operations is 999 seconds. Similarly, when the clock frequency is chosen to be 10 Hz and decimal-point switch S5 is flipped to 'on' position, the maximum attainable time delay is 99.9 seconds.

Reset switch S3 can be used to reset the counter to all zeros before start. Usually, the reset terminal is connected to ground via resistor R27. It is connected to +5V when you momentarily press reset switch S3 to reset the counter.

IC8, IC9 and IC10 (each CD4511B) are BCD-to-7-segment latch/decoder/driver ICs. These can directly drive 7-segment, common-cathode LED displays.

DIS1, DIS2 and DIS3 (each IC LTS543) are 7-segment displays of common-cathode type. Pin configuration of IC LTS543 is shown in Fig. 4. R5 through R25 are current-limiting resistors.

Construction

A single-side, actual-size PCB for the digital stopwatch is shown in Fig. 5 and its component layout in Fig. 6. The proposed display panel box is shown in Fig. 7. All the 7-segment displays, LED1 and switches are mounted on the front panel. ●

INFRARED INTERRUPTION COUNTER

■ **PRADEEP G.**

Most optical interruption counters make use of a light bulb with light-dependent resistor (LDR) or ordinary phototransistor as the sensor. These interruption counters work satisfactorily in darkness only and cannot be used outdoors because of the chances of false counting due to light sensed from other light sources like sun, light bulb, etc.

The interruption counter described here uses an infrared (IR) sensor that can sense a particular modulated frequency of infrared beam. A small transmitter circuit employing an IR LED is used to emit modulated IR signals.

The block diagram of the infrared interruption counter providing an overview of the system is shown in Fig. 1. The astable multivibrator produces 36kHz frequency and npn transistor BC547 drives the IR LED to transmit the modulated infrared signal. The transmitted IR signal continuously falls on the IR sensor (receiver).

When somebody crosses the path of the IR beam falling on the sensor, the triggering circuit activates to trigger the monostable multivibrator. The output of the monostable advances the count of the 4-digit counter-cum-display driver to display the count on 7-segment, common-cathode displays.

Circuit description

The infrared interruption counter circuit consists of power supply, transmitter and infrared interruption counter stages.

Power supply. Fig. 2 shows the power supply circuit. The AC mains is stepped down by transformer X1 to deliver secondary output of 9V at 500 mA. The transformer output is rectified by a full-wave bridge rectifier comprising diodes D1 through D4, filtered by capacitors C3 and C4, and regulated by IC 7805 (IC1) to provide regulated 5V supply for the transmitter and infrared receiver-cum-counter stages. Capacitor C5 bypasses any ripple in the regulated

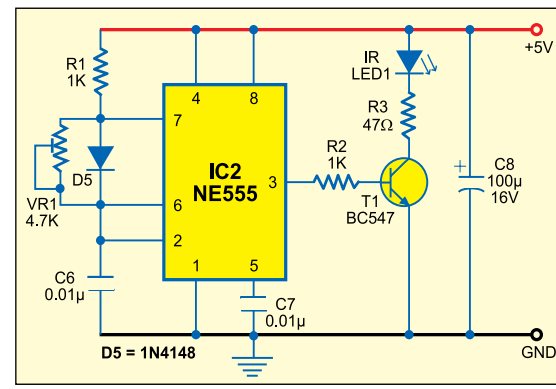


Fig. 3: IR transmitter circuit

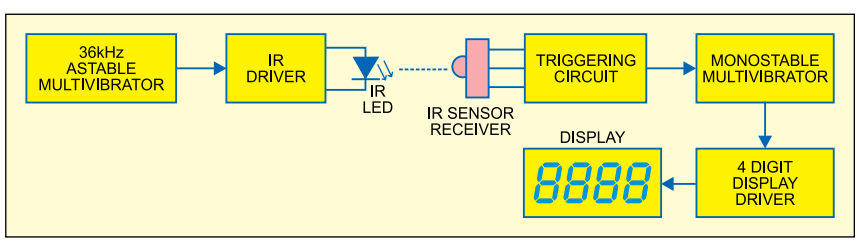


Fig. 1: Block diagram of infrared interruption counter

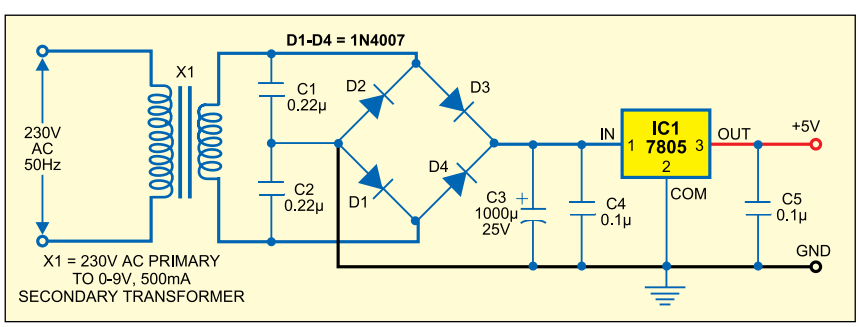


Fig. 2: Power supply circuit

PARTS LIST

Semiconductors:

IC1	- 7805 5V regulator
IC2, IC3	- NE555 timer
IC4	- 74C926 display driver
IR LED1	- IR transmitting LED
IRX1	- TSOP1736 IR receiver
T1-T7	- BC547 npn transistor
D1-D4	- 1N4007 rectifier diode
D5	- 1N4148 switching diode

Resistors (all ¼-watt, ±5% carbon):

R1-R2	- 1-kilo-ohm
R3-R4	- 47-ohm
R5	- 2.2-kilo-ohm
R6, R8, R22	- 10-kilo-ohm
R7, R9, R10	- 100-kilo-ohm
R11-R17	- 1-kilo-ohm
R18-R21	- 1.5-kilo-ohm
VR1	- 4.7-kilo-ohm

Capacitors:

C1, C2	- 0.22µF ceramic disk
C3	- 1000µF, 25V electrolytic
C4, C5, C10, C11	- 0.1µF ceramic disk
C6, C7, C12	- 0.01µF ceramic disk
C9	- 47µF, 16V electrolytic

Miscellaneous:

X1	- 230V AC primary to 0-9V, 500mA secondary transformer
S1	- Push-to-on switch
	- IC bases
	- Wires

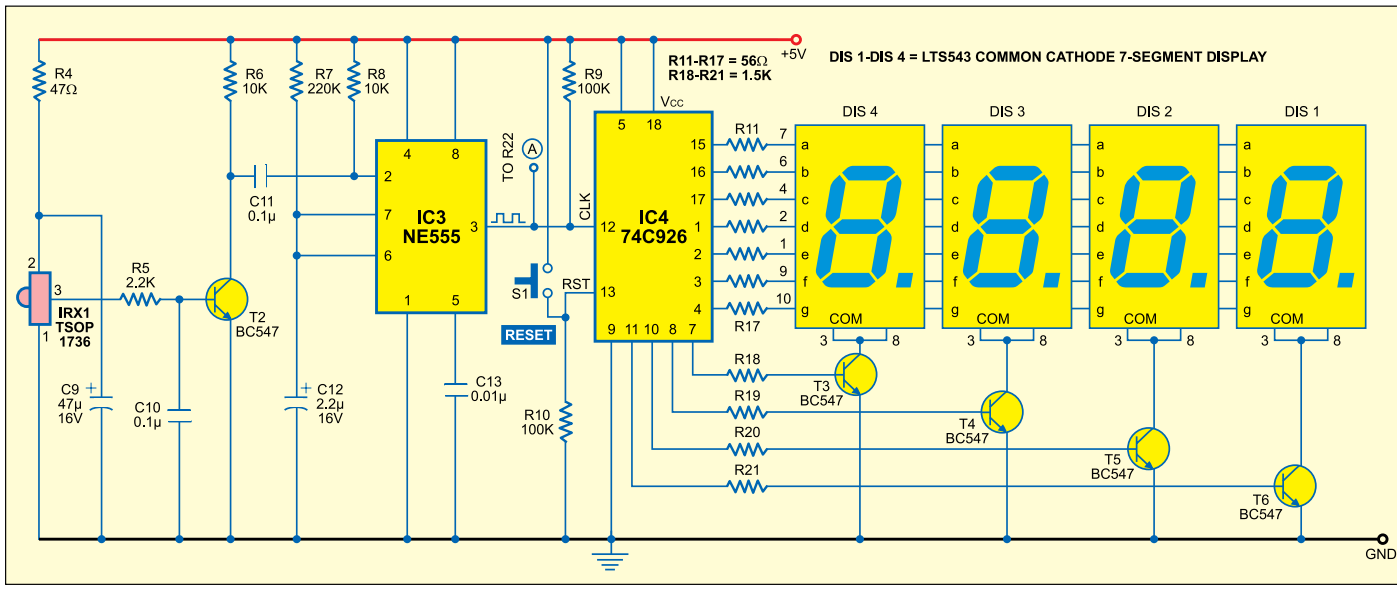


Fig. 4: Circuit of infrared interruption counter

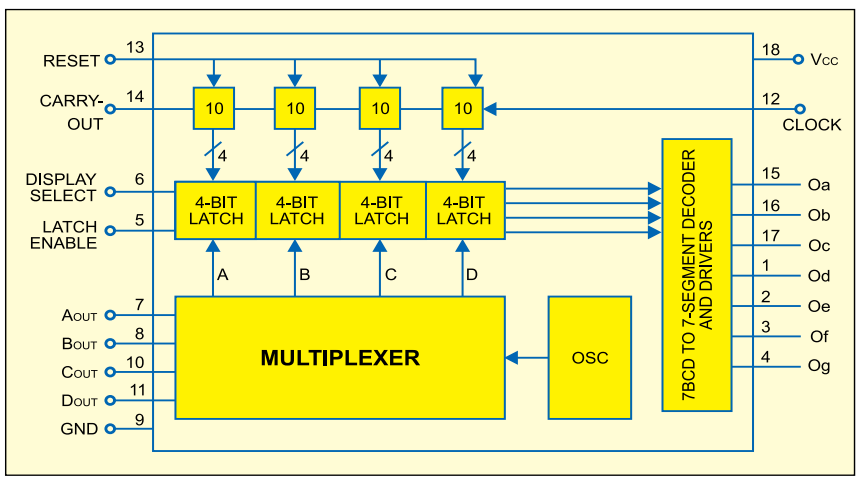


Fig. 5: Internal pin configuration of IC 74C926

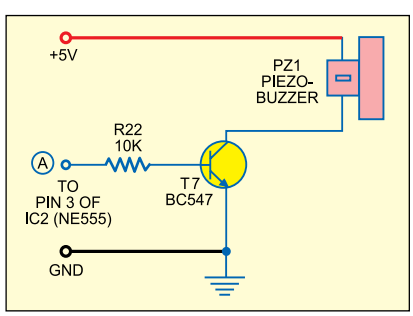


Fig. 6: Audible beeper (optional)

output.

Transmitter stage. The transmitter circuit (see Fig. 3) works off 5V regulated supply. It is built around timer NE555 (IC2), npn transistor BC547, IR LED1 and some resistors and capacitors.

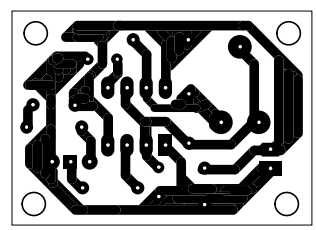


Fig. 7: Actual-size, single-side PCB for IR transmitter

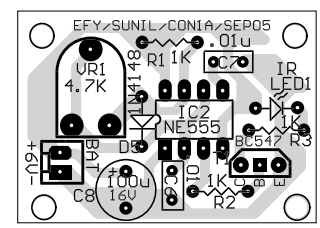


Fig. 8: Component layout for the PCB in Fig. 7

Timer NE555 is wired as an astable multivibrator whose frequency is set at 36 kHz by adjusting preset VR1. The npn transistor (T1) is used to drive IR LED1, which can transmit modulated IR signals up to around 7 metres without any lense arrangement.

Infrared interruption counter stage. The IR interruption counter circuit (Fig. 4) is built around IR receiver TSOP1736 (IRX1) utilising timer NE555 (IC3), 4-digit counter-cum-display driver IC 74C926 with multiplexed 7-segment output drivers (IC4), 7-segment common-cathode displays DIS1 through DIS4, BC547 npn transistors and some discrete components.

IR sensor TSOP1736 is readily available in the market. It is commonly used in TV sets as a miniaturised receiver for IR remote control systems. Fig. 5 shows the internal functional block diagram and pin configuration of IC 74C926.

IR receiver module TSOP 1736 is meant for pulsed operation. When it is exposed to continuous 36kHz modulated IR beam, its output remains high and the collector of transistor T2 is held low. During a brief interruption of the IR beam, a low-to-high-to-low pulse appears at the collector of transistor T2 to trigger the monostable formed by IC3. The monostable multivibrator is set for a time delay of nearly half second. The 4-digit counter with multiplexed 7-segment output drivers (IC 74C926) advances by one digit for every clock pulse received from

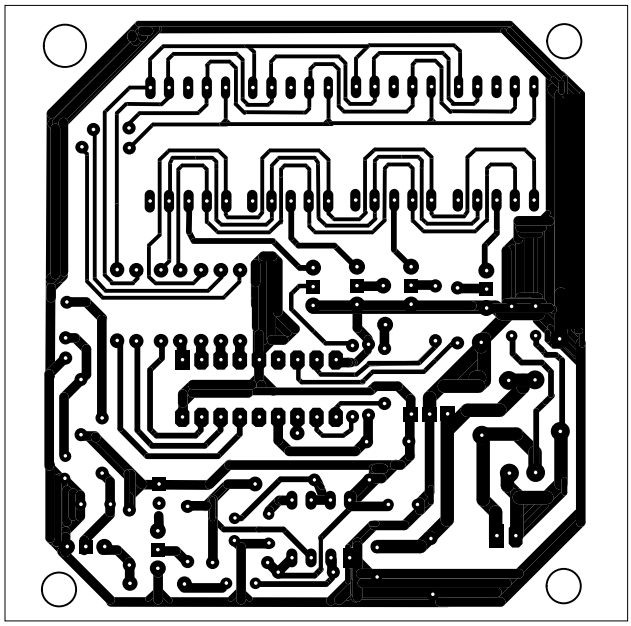


Fig. 9: Actual-size, single-side combined PCB layout for infrared interruption counter, power supply and beeper

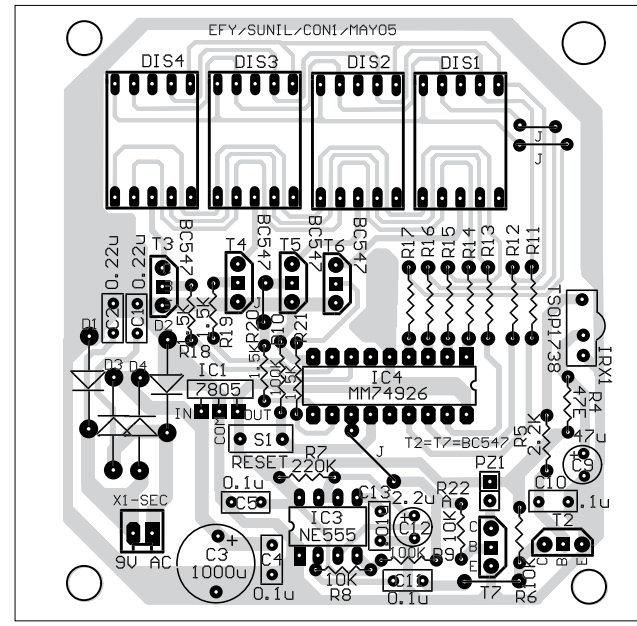


Fig. 10: Components layout for the PCB in Fig. 9

the multivibrator. It can count up to '9999.' The counter can be reset to zero at any time by pressing the reset microswitch.

Thus the display, at any time, shows the count of interruption of the IR beam since the last reset. The interruption counter can be employed as visitor counter or object counter in industrial applications.

You can add a beeper circuit, as shown in Fig. 6, to provide an audible indication of each interruption. The output pulse from monostable IC3, generated during an interruption, will activate transistor T7 to drive the piezobuzzer for duration of the monostable pulse.

Construction

An actual-size, single-side PCB layout for the IR transmitter (Fig. 3) is shown in Fig. 7 and its components layout in Fig. 8, while the combined PCB layout for the interruption counter (Fig. 4), power supply (Fig. 2) and beeper (Fig. 6) is shown in Fig. 9 with its components layout in Fig. 10.

For easy servicing, use IC bases to mount the ICs on the PCB. After assembling the PCB, place it near the entry gate. Use long wires for connections to the IR transmitter LED and IR receiver TSOP1736 so that these can be taken out of the PCB and mounted on the opposite pillars of the entry gate. The transmitter should be oriented such that the transmitted IR ray directly falls on the receiver module. ●

AUDIO MIXER WITH MULTIPLE CONTROLS

■ MALAY BANERJEE

When recording sound from several orchestral instruments being played by different musicians using a single microphone, the only way to adjust the sound balance is to change the position of the musicians relative to the microphone. When recording direct to stereo master tape, it's crucial to make sure that all the voices and instruments sound right before you hit the record button.

Here is an eight-input audio mixer circuit with bass, treble, volume and balance controls, which you can use to balance sounds from all the sources until you have the desired mix. For capturing the sound from various sources, the audio mixer employs up to eight microphones.

Circuit description

Fig. 1 shows the block diagram of the audio mixing system along with the audio power amplifier, while the circuit of the audio mixer along with tone controller is shown in Fig. 2. The power supply and audio power amplifier circuits are shown in Figs 3 and 4, respectively.

Here, dual operational amplifier IC 747 (IC3) is used for mixing several inputs without any mutual interaction. The two internal amplifiers share a common bias network and power supply. The IC has short-circuit protection and wide common-mode and differential voltage ranges.

In this application, +12V and –12V regulated DC supplies are used for operation of IC 747. The microphone output signals M1 through M4, after their individual level adjustments, are mixed and applied across the differential input terminals (pins 1 and 2). Similarly, microphone outputs M5 through M8 are applied across the differential input terminals (pins 7 and 6) of the second amplifier inside op-amp IC 747 after their individual level adjustments.

For level adjustment, logarithmic variable resistors VR1 through VR4 and VR5 through VR8, respectively, are employed while feeding the output from respective microphones to the input of the two amplifiers inside IC 747. The outputs of the two amplifiers taken from pins 12 and 10, respectively, are combined at the junction of resistors R9 and R10 before feeding to the next stage (tone controller) via capacitor C12 (10 µF).

The overall gain of individual amplifiers can be adjusted with the help of

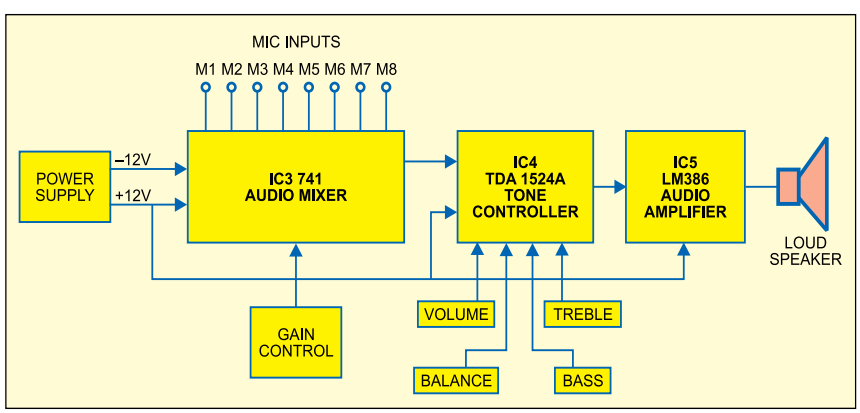


Fig. 1: Block diagram of the audio mixer with bass, treble, volume and balance controls

PARTS LIST

Semiconductor:

IC1	-	7812 +12V DC regulator
IC2	-	7912 –12V DC regulator
IC3	-	747 dual operational amplifier
IC4	-	TDA1524A stereo tone controller
IC5	-	LM386 low-power audio amplifier
D1-D4	-	1N4001 rectifier diode
LED1, LED2	-	5mm light-emitting diode

Resistors (all ¼-watt, ±5% carbon):

R1-R8, R18, R19	-	1-kilo-ohm
R9-R11, R14	-	10-kilo-ohm
R12, R15	-	33-kilo-ohm
R13, R16	-	4.7-kilo-ohm
R17	-	2.2-kilo-ohm
R20	-	10-ohm
VR1-VR8, VR15	-	10-kilo-ohm log potmeter
VR9, VR10	-	10-mega-ohm linear potmeter
VR11-VR14	-	47-kilo-ohm linear potmeter

Capacitors:

C1-C8, C22-C25, C31, C33-C35, C38	-	0.1µF ceramic disk
C9-C12, C41	-	10µF, 25V electrolytic
C13, C14	-	15nF ceramic disk
C15, C39	-	100µF, 25V electrolytic
C16, C19, C20, C21	-	2.2µF, 25V electrolytic
C17, C18	-	56nF ceramic disk
C26-C29	-	47nF ceramic disk
C36	-	220µF, 25V electrolytic
C37	-	3.3µF, 25V electrolytic
C30, C32	-	1000µF, 25V electrolytic
C40	-	33nF ceramic disk

Miscellaneous:

X1	-	230V AC primary to 12V-0-12V, 1A secondary transformer
S1-S3	-	SPST on/off switch
Loud-speaker	-	8Ω, 1W loudspeaker
M1-M8, J1	-	Audio input jack
RCA1, RCA2	-	Audio output RCA connector

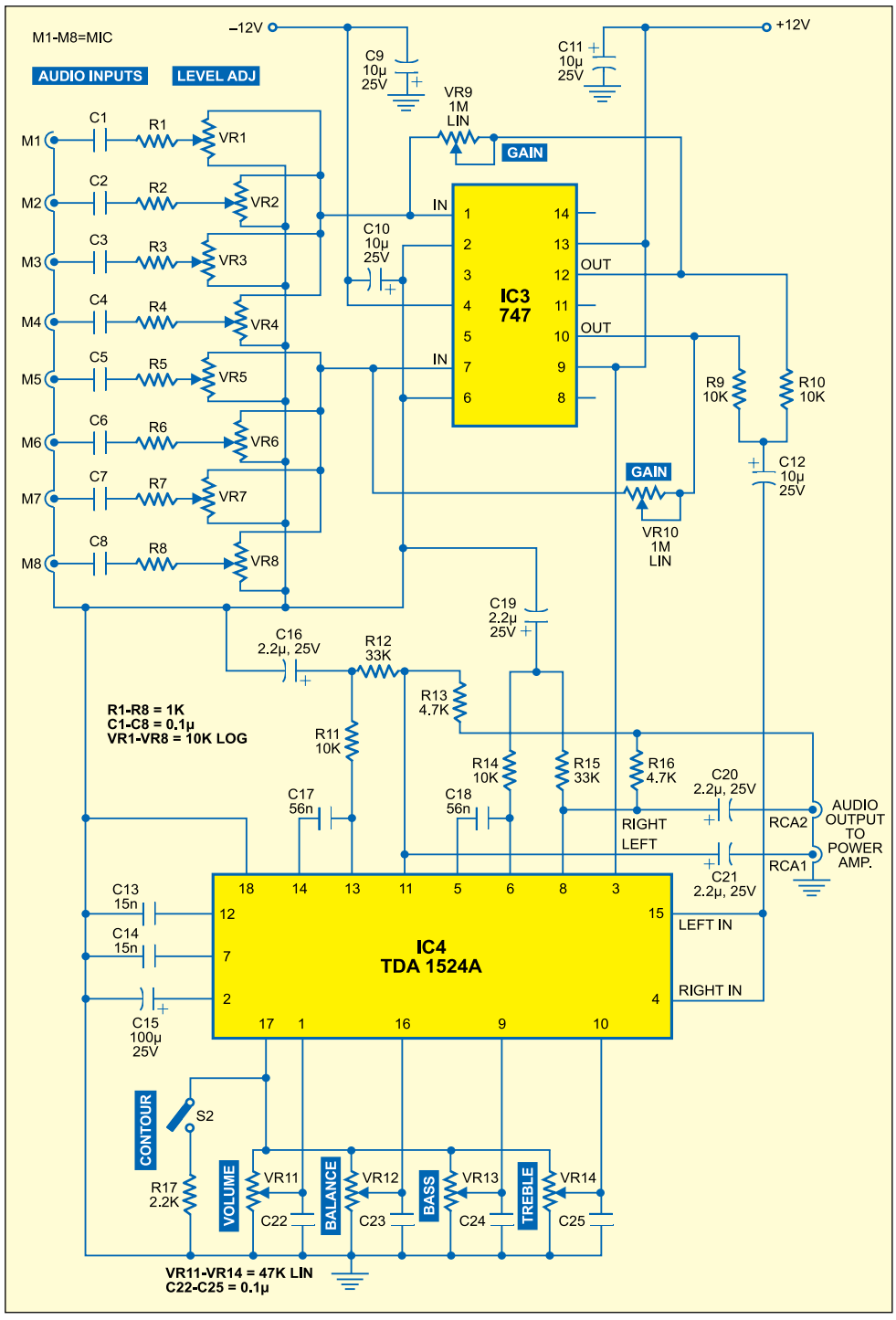


Fig. 2: Circuit of audio mixer with bass, treble, volume and balance control

potmeters VR9 and VR10, respectively.

The amplified mixed signal output of IC 747 is applied to shorted input pins 15 and 4 of stereo tone controller IC TDA1524A (IC4). TDA1524A is designed as an active stereo-tone/volume control for car radios, TV receivers and mains-fed equipment. It includes functions for bass and treble control, volume control with built-in contour (can be switched off) and balance. All these functions can be controlled by DC voltages or by single linear potentiometers. This IC serves as an efficient tone controller. Although it may work reasonably well with 9V DC supply, for better bass response, a 12V supply can be used. A good heat-sink is necessary for longer life and better performance of the IC.

Features of TDA1524A are:

1. Simple construction
2. Low noise and distortion
3. Switchable contour (for quick changing of the tonal response)
4. Its output can drive most power amplifiers.
5. Bass emphasis can be increased by incorporating a double-pole, low-pass filter
6. Wide power supply voltage range

General specifications are:

1. DC input: 12V (typical)
2. DC battery: 35 mA
3. Maximum output: 3V RMS
4. Maximum input: 2.5V
5. Maximum gain: 21.5 dB
6. Volume control range: –80 to +121.5 dB
7. THD at 1 kHz: 0.3%
8. Ripple rejection at 100 Hz: 50 dB

Potmeters VR11, VR12, VR13 and VR14 are meant for adjustment of volume, balance, bass and treble, respectively. Switch S2 is contour switch, which can be used to change the tonal response of the of the IC. The outputs are available at pins 8 and 11 for right and left channel, respectively. (***EFY note.*** Since both the left- and right-channel input pins 15 and 4 have been shorted in this application, the IC acts as a mono volume/tone control circuit.)

Audio power amplifier. The audio amplifier circuit shown in Fig. 4 is optional. One can use much higher-power audio amplifier along with the audio mixer circuit.

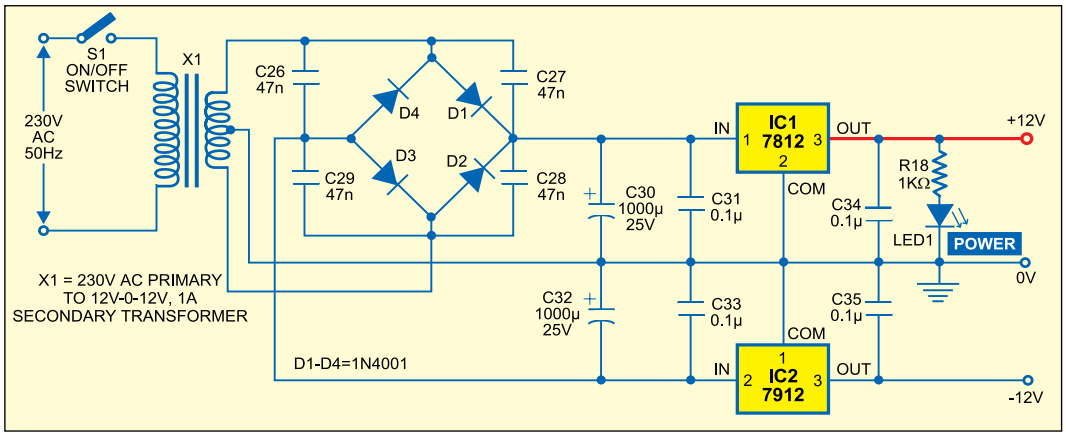


Fig. 3: Power supply circuit

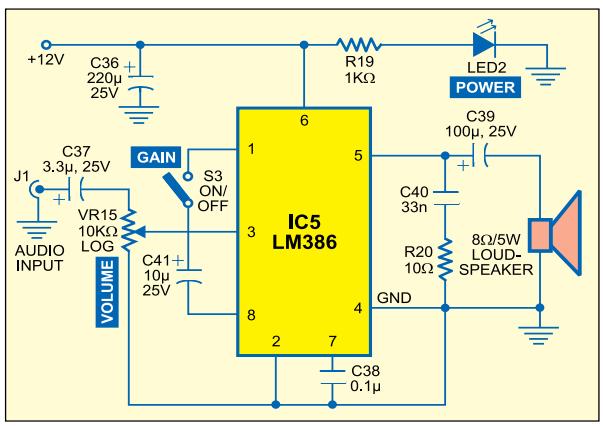


Fig. 4: Low-power audio amplifier circuit

nation of a resistor and a capacitor. If only a capacitor is put between pins 1 and 8 using switch S3 as shown in Fig. 4, the gain would increase to 200 (46 dB). The amplified output is taken from pin 5 and fed to the loudspeaker through electrolytic capacitor C39 (100 µF). The higher the value of C39, the higher the pitch of the audio frequency response in the speaker.

Power supply. The power supply section for the circuit is shown in Fig. 3. It consists of a step-down transformer (230V AC primary to 12V-0-12V, 1A secondary), bridge rectifier, filter network and regulator ICs 7812 and 7912 to provide +12V and –12V regulated DC outputs, respectively. When switch S1 is closed, the presence of power is indicated by the glowing of LED1.

Construction

Assemble the circuit on any general-purpose PCB. Mount IC bases on the PCB. There is no soldering method that is ideal for all IC packages. The use of

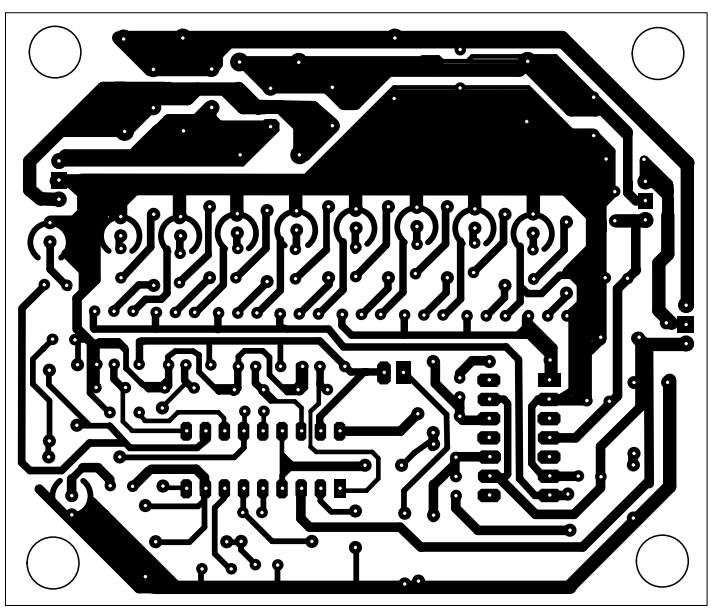


Fig. 5: Combined actual-size, single-side PCB for audio mixer and power supply circuits

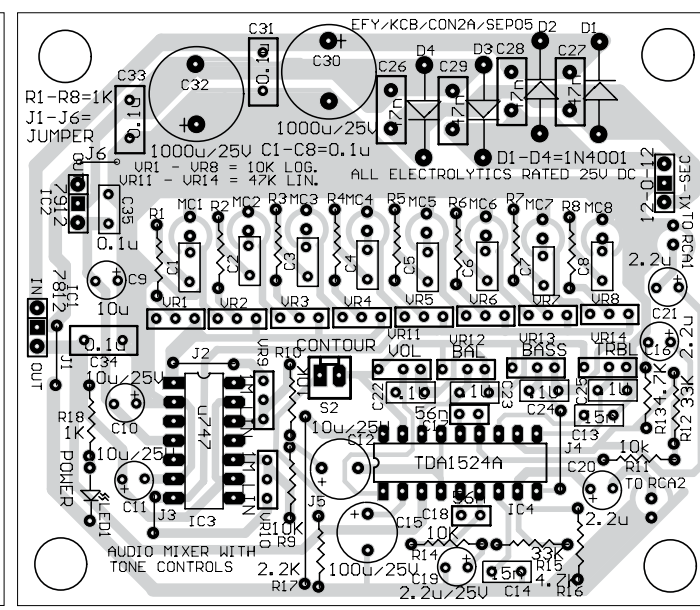


Fig. 6: Components layout for the PCB in Fig. 5

The low-power audio amplifier employing IC LM386 (IC5) shown in Fig. 4 can output a maximum audio power of 1 watt. It gets +12V DC supply at its pin 6. The audio input from sources like Walkman and audio mixer can be fed to pin 3 of IC5 through volume control VR15.

The gain of LM386 is internally set to 20 to keep the external part count low. However, to make LM386 a more versatile amplifier, pins 1 and 8 are provided for setting the gain—externally to any value between 20 and 200—by using an appropriate combi-

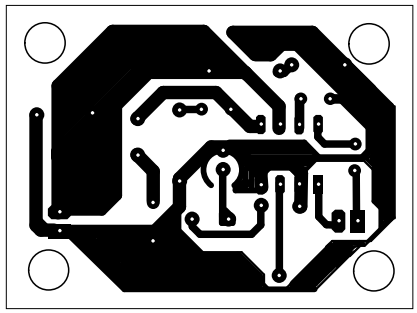


Fig. 7: Solder-side PCB layout for the audio amplifier circuit

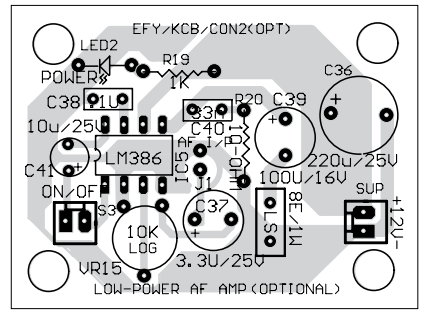


Fig. 8: Components layout for the PCB in Fig. 7

IC bases prevents damage to the ICs while soldering and also makes it easy to replace them. Use audio input jack connectors for M1 through M8 input points. Also use audio output connectors at the outputs of IC4.

A combined actual-size, single-side PCB layout for Figs 2 and 3 is shown in Fig. 5 and its components layout in Fig. 6. The solder-side PCB layout for Fig. 4 is shown in Fig. 7 and its components layout in Fig. 8.

Note. If you are not using IC base for TDA1524A, the maximum permissible temperature of the solder is 260 °C; solder at this temperature must not be in contact with the joint for more than five seconds. The total contact time of successive solder waves must not exceed five seconds while using wave soldering.

Testing procedure

1. After assembling the PCB, check the circuit connections before switching on the power supply.

2. Use a standard microphone at the first input point M1 and then keep it near an audio source. You can use the power amplifier circuit given here for testing or another higher-output power amplifier.

3. Vary VR1 slowly until a clear and distortion-free amplified output is obtained.

4. If the sound output is not clear and VR1 does not help, vary gain control VR9.

5. If the problem still persists, check volume, balance, bass and treble controls.

6. Check the various controls in your audio power amplifier section.

7. Repeat steps 2 through 5 for the rest of the inputs. Having checked all the inputs, now the audio mixer is ready for use. ●

NOISE-MUTING FM RECEIVER

■ TURJASU PYNE

The tuning of a frequency-modulated (FM) receiver to an FM radio station frequency involves a lot of 'hissing' noise in between the stations, which is very irritating for the operator and as such undesirable.

Digital FM receivers are free from this problem, because in them the output is automatically clipped during off-station gaps. However, digital FM receivers are comparatively much costlier than their analogue counterparts.

Analogue FM receivers employ an LC tuning system that generates 'hiss' noise at the output, which remains unclipped during off-station tuning. However, when the received signal level is adequate (i.e., when the receiver frequency is very close to an FM transmitting station frequency), the limiter circuit preceding the ratio detector circuit in an analogue FM receiver will clip/limit any noise riding the FM carrier.

The simple FM muting circuit described here eliminates this 'hissing' noise from the output of an analogue FM receiver circuit when it is not resonating to any FM transmiting station's frequency. In other words, this circuit mutes the output from an analogue FM receiver when its tuner circuit is sweeping in between FM stations.

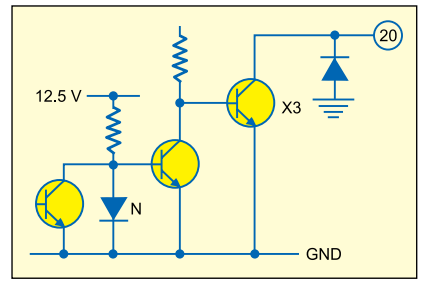


Fig. 2: Internal circuit schematic around pin 20 of CXA1619S

Circuit description

The FM muting circuit has been configured around a Sony CXA1619S AM/FM receiver chip, which is available in a 30-pin PDIP package. Sony FM receiver chips are known for their superior features and cost-effectiveness. The functional block diagram of the chip is shown in Fig. 1.

A complete circuit of the FM receiver including the muting circuit and power supply is shown in Fig. 3. Most of the FM kits available in the market also make use of Sony CXA1619S, which is a complete AM/FM receiver IC with very few external components. It can operate at voltages as low as 3V. Here, we have used a 6V supply so that we can feed the output of the receiver directly to a 32-ohm headphone or to a low-power AF amplifier built around IC LM386. A 3-pin jumper has been provided to enable the desired selection.

The muting circuit makes use of the tuning meter output at pin 20 of CXA1619S. The internal circuit around pin 20 of the chip is shown in Fig. 2.

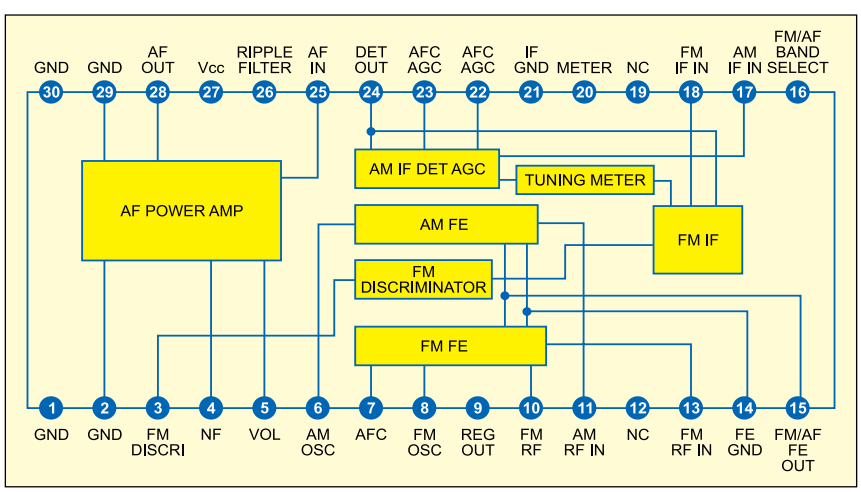


Fig. 1: Functional block diagram of Sony CXA1619S

PARTS LIST

Semiconductors:

IC1	- CXA1619S FM receiver
IC2	- 7806 +6V regulator
IC3	- CD40106 hex inverter
IC4	- LM386 audio power amplifier
T1	- BC548 npn transistor
BR1	- 1A bridge rectifier
D1	- 1N4148 switching diode
D2- D4	- 1N4001 rectifier diode

Resistors (all ¼-watt, ±5% carbon):

R1	- 150-ohm
R2, R3	- 330-ohm
R4	- 100-kilo-ohm
R5	- 1-kilo-ohm
R6	- 33-kilo-ohm
R7	- 68-kilo-ohm
R8	- 4.7-kilo-ohm
R9	- 4.7-ohm

Capacitors:

C1	- 4.7µF, 16V electolytic
C2, C8, C9	- 22pF ceramic disk
C3	- 0.047µF ceramic disk
C4	- 47nF ceramic disk
C5, C6	- 10µF, 16V electolytic
C7	- 3.3pF ceramic disk
C10, C11	- 0.02µF ceramic disk
C12	- 100µF, 25V electolytic
C13, C15, C17	- 0.1µF ceramic disk
C14	- 1000µF, 16V electolytic
C16	- 0.04µF ceramic disk
C18	- 1000µF, 35V electolytic
VC1, VC2	- 77pF trimmer

Miscellaneous:

X1	- 230V AC to 6V-0-6V, 500mA secondary transformer
RL1	- 5V, 500-ohm, open-type reed relay (PLA make)
S1	- On/Off switch
CR1	- 10.7MHz, 2-pin ceramic resonator
CR2	- 10.7MHz, 3-pin ceramic resonator
LS1	- 8-ohm, 1W loudspeaker
	- Headphone
	- Battery 1.5V×4
J1	- Shorting jumper

Normally, this pin is used for tuning indication using a light-emitting diode (LED). The cathode of the tuning indicator LED is connected to pin 20, while the anode of the LED is connected to positive supply through a current-limiting resistor.

When the receiver is tuned to the centre of FM carrier, pin 20 of CXA1619 goes very low and the LED lights up the brightest.

Instead of connecting an LED, we have coupled pin 20 of CXA1619S to the input of a CMOS inverter gate via diode D1. In 'off' tuning condition (when the receiver is not tuned to any FM station), input pin 1 of CMOS inverter gate N1 (1/6 CD40106) is biased to Vcc using a potmeter arrangement comprising resistors R6 (33 kilo-ohms) and R7 (68 kilo-ohms). Thus, in between the FM stations, output pin 2 of the inverter gate remains low and therefore transistor T1 (BC548) remains cut off and the 5V inverter relay connected at its collector remains de-energised.

However, when the receiver is tuned to an FM station, its output pin 20, as also input pin 1 of the CMOS inverter gate, go low. As a result, the output of the inverter gate goes high to energise the relay via transistor T1.

The audio output from CXA1619S, available at its pin 28, is passed via 100µF capacitor C12 and the contacts of reed relay to either the headphones or the AF amplifier circuit only when the relay is energised, which happens only when the receiver is tuned to the carrier frequency of an FM station. Thus the annoying noise output of the receiver during 'off' station tuning is not heard in the headphones or the loudspeaker. In other words, the circuit configured around the CMOS inverter gate acts to mute the noise output when the receiver is not tuned to any FM transmitting station.

Power supply. The supply voltage for inverter gate N1 (IC3), transistor T1 and the FM receiver chip/kit has been derived from the output of IC 7806 so that a regulated voltage is maintained across the entire circuit. Alternatively, in the event of mains failure, and for portable applications, four 1.5V cells may be used. Diode D3 in the power supply circuit ensures that when mains is available, the battery output remains isolated/not used.

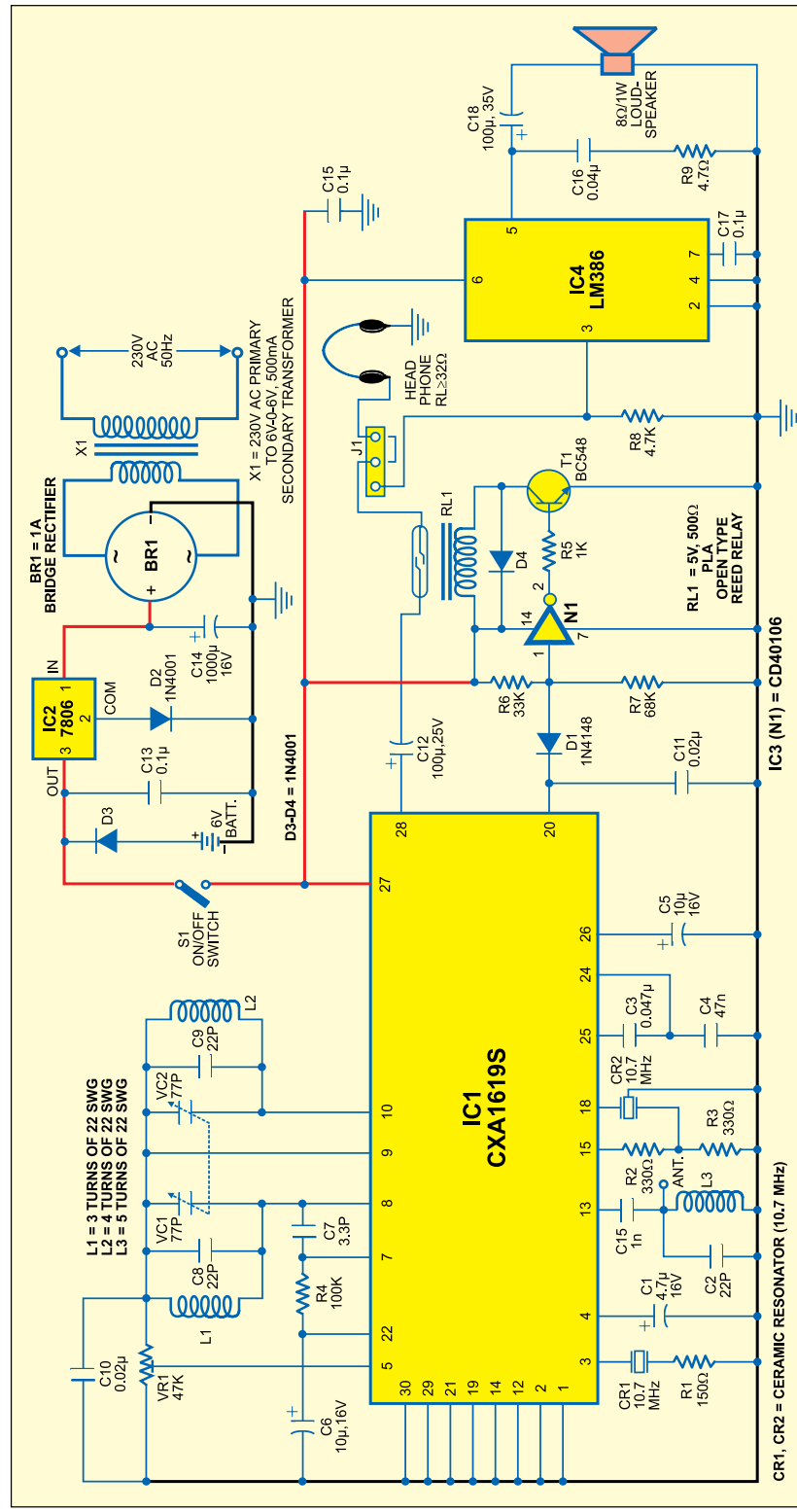


Fig. 3: Complete circuit of the FM receiver including the muting circuit and power supply

Antenna. For antenna, you may use either a 75cm telescopic antenna or simply a wire of similar length.

Circuit operation

When no FM station is being received, diode D1 does not conduct and as

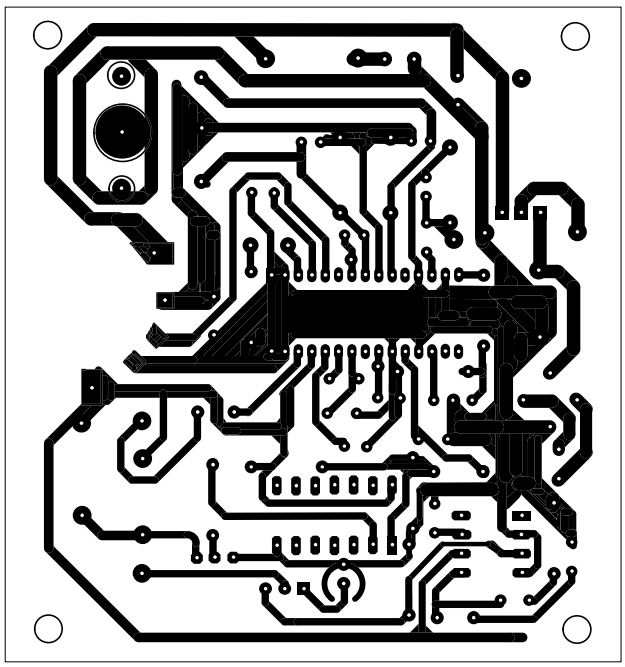


Fig. 4: Actual-size, single-side PCB of noise-muting FM receiver

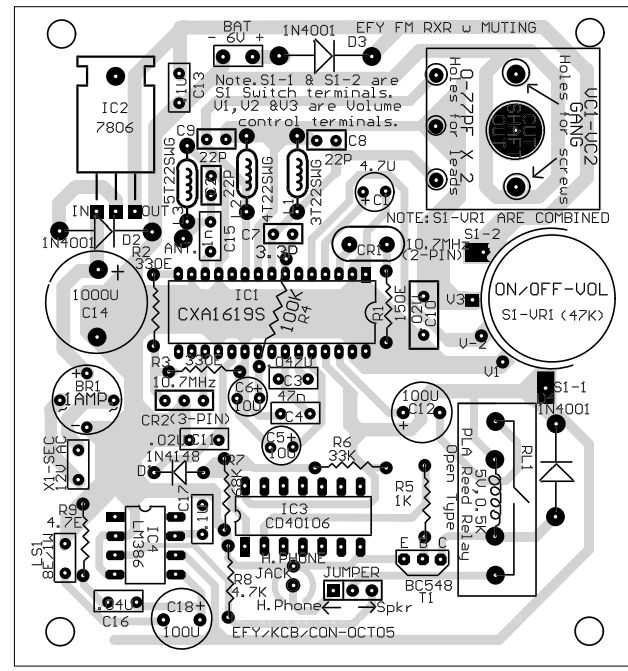


Fig. 5: Component layout for the PCB

such, pin 1 of inverter gate N1 is held high while the output at pin 2 of the inverter remains low. Thus transistor T1 does not conduct and relay RL1 remains cut off. So, whether the load is a headphone or an audio power amplifier stage driving a loudspeaker as shown in Fig. 3, no audio will be heard in the headphones/speaker. In other words, when the receiver is not tuned to an FM station's frequency, the audio output from CXA1619S is muted.

This muting circuit may be incorporated into any readily available FM receiver kit as long as the FM receiver chip used in the kit has a metering pin for connecting an LED. The muting circuit needs no adjustments as the inverter gate has been kept biased to

Vcc, and when a station is being tuned into the output at pin 20 of CXA1619S starts falling, and when the input of the CMOS gate goes below ½Vcc the logic state of the CMOS changes abruptly and the FM station can be heard via headphones or speaker of the power amplifier.

In case the receiver circuit uses only a headphone as load, you can do away with the audio power amplifier circuit including the loudspeaker. Potmeter VR1 serves as volume control for the headphones or the AF power amplifier, whichever is in use.

Construction

An actual-size, single-side PCB layout for the noise muting circuit for FM receiver is shown in Fig. 4 and its component layout in Fig. 5. Provision for a jumper has been made in the PCB so that either a headphone or AF amplifier circuit with speaker may be used as desired.

This PCB can be fitted with a Philips make miniature gang condenser or equivalent for VC1 and VC2. For power 'on'/'off' switch S1 and volume control VR1, a switch-cum-volume control is used in the PCB. The terminals of this switch-cum-volume control need to be directly soldered over the corresponding pads on the PCB. ●

PC-BASED STEPPER MOTOR CONTROLLER

■ ASHUTOSH M. BHATT

This stepper motor controller is perhaps the cheapest, smallest and simplest. A pair of H-bridges with a software program written in 'C++' is used to control the bipolar stepper motor with a step resolution of 18 degree per pulse.

The controller is a combination of driver and switching circuits. The driver is the actual circuit that drives the stepper motor and the switching circuit decides how the motor should be driven. So, it is basically the switching circuit that controls the motor. The transistors (T1 through T8) act as switches. The switching of these transistors is controlled by the software via data pins D0 through D7.

You can control three parameters of the stepper motor: speed, direction and number of steps. To vary the speed of the motor, you have to vary the pulse repetition frequency (PRF). To change the direction of the motor, you have to change the sequence of pulses applied to its coils. By limiting the number of applied pulses, you can restrict the motor to complete the desired number of steps.

Specifications of the stepper motor

Stepper motors of various ratings/specifications are available in the market for different applications. Here, the stepper motor is taken from an 8.9cm (3.5-inch) floppy drive. It's a bipolar stepper motor rated at 5V DC with step resolution of 18° per pulse. The motor has two coils inside and four terminals (colour-coded, but not always) for external connections. Stepper motors rated at 5V and up to 1 ampere of current and different step size (e.g., 1.8° per pulse) may also be used with this circuit and control software.

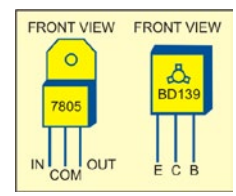


Fig. 2: Pin details of BD139 transistor and regulator IC 7805

Circuit description

H-Bridge driver. H-Bridge is a standard, well-known circuit widely used as stepper motor driver. It is a bridge connection of four transistors (see Fig. 1). Because there are two coils in the bipolar stepper motor, two H-bridge circuits, one for each coil, have been used. One H-bridge is formed by transistors T1 through T4 and the other bridge is formed by transistors T5 through T8.

Transistors T1 through T8 are BD139 type and should be used with heat-sinks. Pin details of BD139 and regulator IC 7805 are shown in Fig. 2. The bases of all the eight transistors are connected to data pins (D0 through D7) of the 25-pin, D-type male connector through 1-kilo-ohm current limiting resistors R1 through R8.

The bases of transistors T1 and T4 are connected to parallel-port pins 2 (D0) and 3 (D1) through resistors R1 and R2, respectively, and the bases of transistors T2 and T3 are connected to parallel-port pins 4 (D2) and 5 (D3) through resistors R3 and R4, respectively. The red and orange terminals of the first coil (COIL1) are connected to the first H-bridge section as shown in Fig. 1.

The bases of transistors T5 and T8 are connected to pins 6 (D4) and 7 (D5) through resistors R5 and R6, respectively, and the bases of transistors T6 and T7 are connected to pins 8 (D6) and 9 (D7) through resistors R7 and R8, respectively. The yellow and green terminals of the second coil (COIL2) are connected to the second H-bridge section as shown in Fig. 1.

Power supply. The power supply section is shown in Fig. 3. It consists of a 230V AC to 9V AC, 1A secondary transformer (X1), filter, bridge rectifiers and 5V DC regulator 7805 (IC1). The regulated 5V DC is connected to the H-bridge circuits. The circuit ground is shorted to pins 18 through 25 of the D-type parallel-port connector. When switch S1 is closed, LED1 glows

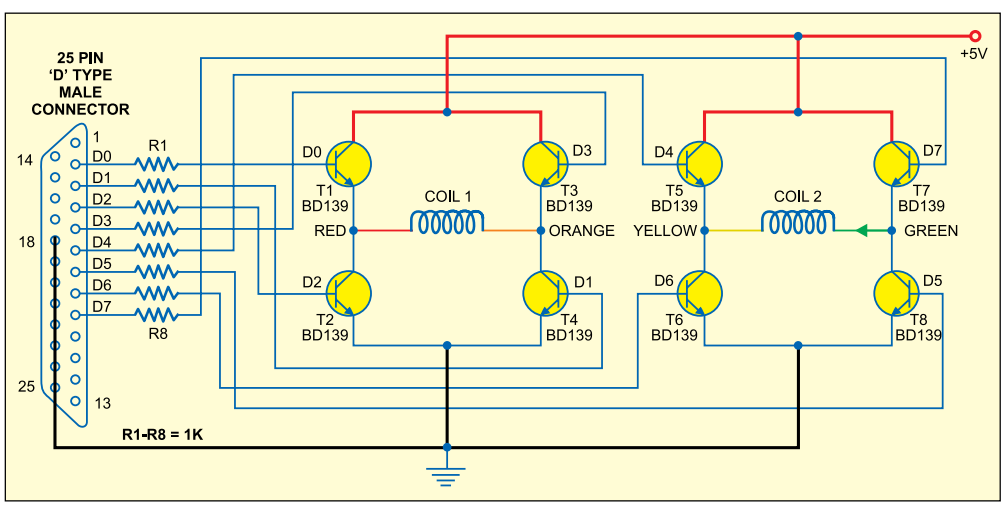


Fig. 1: Circuit of PC-based stepper motor controller

to indicate the presence of power in the circuit.

Operation

Specific sequence of pulses are given to the red and orange terminals of COIL1 and yellow and green terminals of COIL2 to rotate the motor either in clockwise or anticlockwise direction as explained in the following paragraph.

Direction control. In Tables I and II, '0' indicates low logic and '1' indicates high logic. We know that the current flows from high to low. Changing the direction of rotation is nothing but changing the direction of current that flows through the coils.

Speed control. To vary the speed, you have to vary the pulse repetition frequency (PRF). The PRF of 20 Hz means 20 pulses will be given to the stepper motor in one second. Since the step resolution of the motor is 18°/pulse, the motor will rotate 20x18° = 360° (one complete revolution) in one second. So the speed of the motor is one revolution per second (RPS) or 60 RPM. Now if you increase the PRF from 20 Hz to 40 Hz, the RPS will also double to 2 RPS (120 RPM).

Number of rotations. The step resolution of 18°/pulse means if you apply only one pulse, the motor will rotate by 18°. If you apply 10 pulses sequentially, the motor will rotate 180° (half of a revolution). So if you limit the number of pulses applied to the motor, you can stop it at any angular position (multiple of 18°) after completing the desired number of full revolutions. Thus if you apply only 25 pulses, the motor will complete one full revolution and rotate further by 90° (¼ revolution) and stop.

H-bridge. The transistors in the circuit act as switches. When high logic (3.49V) is applied to any data pin of the port, the transistor connected to it conducts and acts as a closed switch, allowing the current to pass through it. When low logic (0.09V) is applied, the transistor stops conducting and acts as an open switch, so the current cannot pass through it.

The pulse sequences to be given to switch the transistors are shown in tables. The current will flow into/out of the coils through the four terminals of the motor (red, orange, green and yellow).

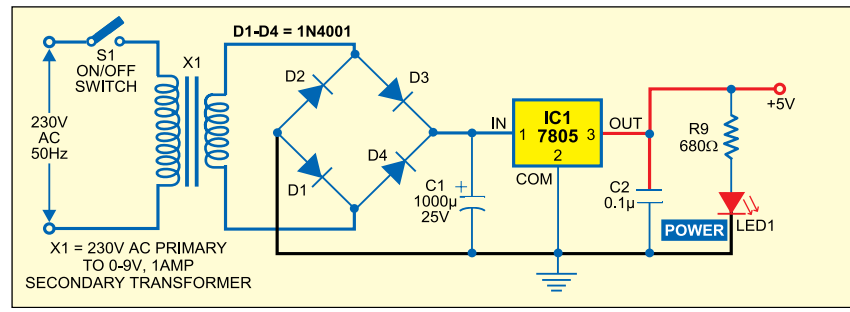


Fig. 3: Power supply for the circuit

TABLE I
Pulse Sequence to Rotate the Motor Clockwise

Data bit / Port pin	D7 9	D6 8	D5 7	D4 6	D3 5	D2 4	D1 3	D0 2
Phase 1	1	1	0	0	1	1	0	0
Phase 2	0	0	1	1	1	1	0	0
Phase 3	0	0	1	1	0	0	1	1
Phase 4	1	1	0	0	0	0	1	1

1= High, 0 = low

TABLE II
Pulse Sequence to Rotate the Motor Anti-clockwise

Data bit / Port pin	D7 9	D6 8	D5 7	D4 6	D3 5	D2 4	D1 3	D0 2
Phase 1	1	1	0	0	1	1	0	0
Phase 2	1	1	0	0	0	0	1	1
Phase 3	0	0	1	1	0	0	1	1
Phase 4	0	0	1	1	1	1	0	0

1= High, 0 = low

Clockwise rotation. In the first phase, orange and green terminals should be high and red and yellow terminals should be low. To achieve this, out of the eight transistors, four transistors (T2, T3, T6 and T7) should conduct. For this, you have to output hex data word 'CC' (1100 1100) from the LPT port.

In the second phase, red and yellow terminals should be high, while orange and green terminals should be low. To achieve this, only transistors T2, T3, T5 and T8 should conduct. For this, you have to output hex data word '3C' (0011 1100) from the LPT port.

In the next phase, red and green terminals should be high, while orange and yellow terminals should be low. To achieve this, transistors T1, T4, T5 and T8 should conduct. For this, hex data word will be '33' (0011 0011).

In the next phase, red and yellow terminals should be low, while orange and green terminals should be high. To achieve this, transistors T1, T4, T6 and T7 should conduct. For the purpose, hex data word 'C3' (1100 0011) has to be output from the LPT port.

Thus the data sequence to be fed to the port for clockwise rotation of the motor is CC-3C-33-C3.

Anticlockwise rotation. To rotate the motor in anti-clockwise direction, the sequence of hex data to be output from the LPT port will be CC-C3-33-3C.

The software

All the controlling actions are performed by the software program. The program is written in 'C++' language and compiled in Turbo C++ Version 3. The complete software program (STEPCNT.CPP) is given at the end of this article along with necessary comments. You require the egavga.bgi graphic file to be in the same directory as the application program to run the program. The output of the program is shown in Fig. 4.

The main functions of the software are:

1. Change the direction of rotation of the stepper motor by switching the eight transistors in proper manner.

2. Vary the RPM of the stepper motor accurately.

3. Stop the motor at a given angular position after the desired number of complete rotations

The software is divided into three parts: graphics, stepper motor control and mouse interfacing.

Graphics. The graphics part generates complete view of the control panel. It draws buttons like clockwise, anticlockwise and RPM increase/decrease, displays instructions, draws borderline, writes text like 'RPM,' 'rotations,' 'number of rotations,' etc. Graphic functions are used to make the program output screen visually appealing.

Stepper motor control. To change the direction of rotation of the motor, the program generates the desired pulse sequence, either CC-3C-33-C3 (to rotate the motor in clockwise direction) or CC-C3-33-3C (to rotate the motor in anticlockwise direction), on the parallel port with appropriate delay. The delay adjustment is done depending upon the RPM.

To vary the RPM, the program varies the PRF. First, the current RPM (S) is converted into RPS (S1) by dividing it by '60' as follows:

$$S1 = \frac{S}{60}$$

Now for one complete revolution, you have to apply 20 pulses. So the RPS factor (S1) multiplied by '20' will give you the desired PRF.

The delay (d) between the pulse sequences is given by:

$$d = \frac{1000}{20 \times S1} = \frac{50}{S1} \text{ milliseconds}$$

When RPM is greater than '10,' you can increase or decrease the RPM by a factor of ±10. For RPM less than '10,' you can increase or decrease it by ±1 only. There is no limit on the maximum RPM but the minimum limit is 1 RPM.

As stated earlier, 20 sequential pulses are required for a complete revolution of the stepper motor. Since a sequence of four pulses is repeated (for clockwise or anticlockwise movement), we may say that a revolution of the stepper motor involves five identical sequences of four pulses. You can increase or decrease the number of rotations linearly by ±1. For 'N' below '1,' you can decrease or increase 'N' by a factor of '0.5.' The minimum limit is 0.25 (quarter revolution), but there is no maximum limit.

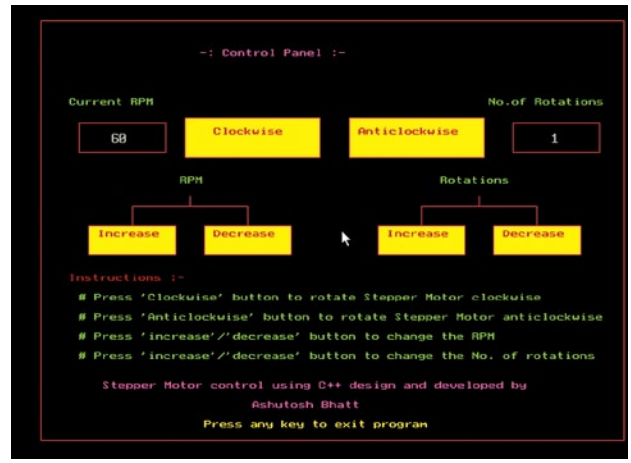


Fig. 4: Screenshot of the program output

Mouse interfacing. This is the most interesting part of the program. It enables you to perform a task at a click of mouse. To understand how the mouse is interfaced, you have to go through the entire theory of hardware interfacing using 'C++.' Here, only some references have been made. For details, refer to the 'Mouse Interfacing' chapter of 'Let Us C' book by Kanitkar.

In this program, the functions that handle the mouse event are initmouse( ), resmptr(int p, int q, int r, int s), showmptr( ) and getmpos(int *t, int *u, int *v).

The initmouse( ) function loads mouse driver into the program. You've to pass '0' value through input union REGS to the int86( ) function. This function will return some non-zero value through output union REGS to the main program. If this function returns '0,' it means the mouse driver is not loaded. So the program displays the message "mouse driver is not loaded" and shuts the screen off using the exit( ) function.

The resmptr(int p, int q, int r, int s) function restricts mouse movement within the boundary specified by the four variables passed to it. Pass all these boundary limits through input union REGS to the int86( ) function. So the resmptr(int p, int q, int r, int s) function will restrict mouse movement out of this boundary.

The showmptr( ) function displays mouse pointer on the program screen. For this, you just have to pass value '1' through input union REGS to the int86( ) function. The showmptr( ) function will now show mouse pointer on the screen.

The getmpos(int *t, int *u, int *v) function performs two tasks: determines whether the mouse button is pressed or not, and captures the current mouse pointer position from the screen. You have to pass value '3' through input union REGS to the int86( ) function. This function will return 'x' and 'y' coordinates of the mouse pointer and also return value '1' if the mouse button (left) is pressed or '0' if the button is not pressed.

How the program works?

The program output screen includes the control panel for speed, direction and number of rotations of the stepper motor.

The program continuously checks for mouse-click event. Whenever there is a mouse click, the getmpos( ) function instantly captures 'x' and 'y' coordinates of the mouse pointer and passes them to the main program.

The main program decides on which position the click event has occured. If the click event occured on any button (clockwise, anti-clockwise, etc), it performs the desired task. For example, if you click the 'RPM increase' button, the program gets the coordinates and directly switches them to 'if' loop, increases the RPM and also displays it on the screen.

Construction and operation

Construct the hardware on a breadboard or on a PCB. An actual-size, single-side PCB layout is shown in

Fig. 5 and its component layout in Fig. 6. Connect the bases of the transistors to the respective data pins of the port DB25 (25-pin, D type male connector) as shown in Fig. 1. Insert DB25 into the PC's LPT-port female connector. Connect all the coil terminals (red, orange, green and yellow) to respective points as shown in the schematic.

Apply 5V DC supply to the circuit and connect the 5V stepper motor with its terminals as shown in Fig. 1. Now run the program on a computer powered by Windows 95/98 operating system. You will see the control panel on the computer screen. Switch on the 5V supply and LED1 will glow. Move the mouse pointer to any of the buttons as desired.

To rotate the motor clockwise, press and hold 'clockwise' button with left mouse button on the control panel. Similarly, for anticlockwise rotation, press and hold 'anticlockwise' button. The motor will rotate in the desired direction along with the beep sound as long as the button is kept pressed. When you release the button, the beep sound as well as the motor will stop.

If the motor rotates in anticlockwise direction when you press 'clockwise' button, just reverse any pair of terminals of the motor coils.

Run the stepcnt.exe file on your computer. You will see the control panel for the stepper motor controller on your screen. Default RPM and number of rotations are 60 RPM and 1, respectively. If you press 'clockwise' or 'anticlockwise' button, the motor will rotate until the desired rotations complete. You can increase or decrease the RPM or even the number of rotations by simply left-clicking that button once. Pressing these buttons more than once will increase/decrease the RPM/number of rotations by the same amount.

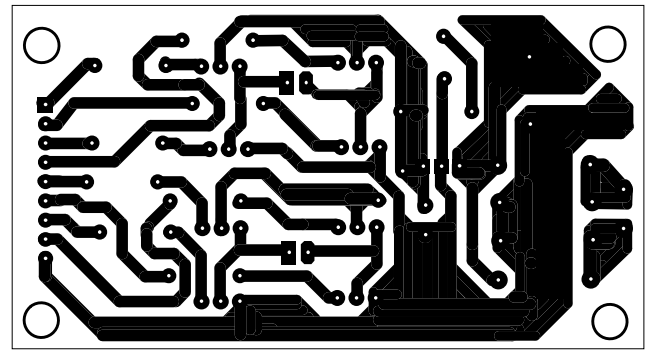


Fig. 5: Actual-size, single-side PCB layout for PC-based stepper motor controller

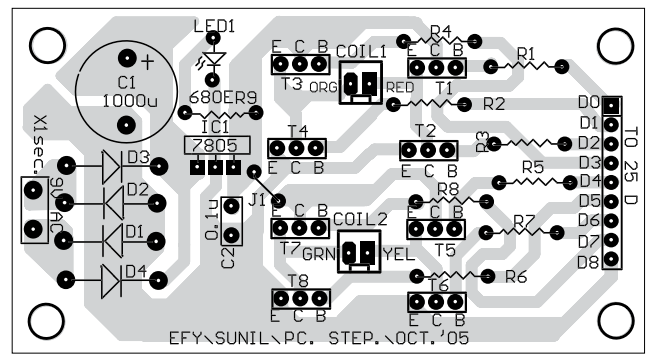


Fig. 6: Component layout for the PCB

The RPM and the number of rotations are perfectly calibrated for this particular stepper motor and you will get the accurate result for the RPM. That means if you choose 1 RPM, the motor will complete one revolution in one minute exactly. For the number of rotations also, if you choose one rotation, the motor will complete only one rotation. If you choose 0.25, the motor will complete only quarter revolution (90°).

Note. The stepper motor you choose should have the same specifications as given in this project. In the program, first enter RPM as '60' and then the number of rotations as '1.' Select clockwise or anticlockwise direction and then press 'Enter' key. The program will now display the PRF (=20 Hz) and the current RPM (=60) with a melodious sound output.

If the motor doesn't rotate, it means you have connected its four terminals wrongly. You can correct this using trial-and-error method. If the motor runs in a direction opposite to that you have selected, just reverse one pair of the coil terminals.

Because you have entered one rotation with 60 RPM, the motor will complete only one rotation in one second. If it doesn't, your stepper motor has some different specification. To check the specifications of the stepper motor, in the sample program select RPM as '5' and change the number of rotations (like 1.25, 1.5, etc) to get the step resolution. The delay factor (d) can be changed in the software. When you are satisfied with the result, switch to main program 'stepcnt.exe'

In case you wish to use the stepper motor with a step size of 1.8° per pulse, the software program for the same will differ. The same is included in the source code download folder.

Download source code: http://www.efymag.com/admin/issuepdf/Stepper%20Motor%20Control.zip

STEPCNT.CPP

```
#include<graphics.h>
#include<conio.h>
#include<dos.h>
#include<process.h>
#include<iostream.h>
union REGS i,o;
void main()
  {
    int driver,mode,x,y,but;      //intialitions of all variables and functions
    driver = DETECT;
    int initmouse();          // to load mouse driver
      int resmptr(int p,int q,int r,int s); //restric mouse pointer within boundry
    int showmptr();                  // shows mouse pointer
    int getmpos(int *t,int *u, int *v);  // captures the current position of mouse pointer
      int text(int e,int f);   // changes the size and color of text
    float s1,d=50,s=60;   // default RPM=60 and no. of rotations = 1
    float r=1,n=5;
    initgraph(&driver, &mode, "C:\\tc\\bgi"); //initialize graphics mode
    outport(0x0378,0x00);          // clear parallel port
    if(initmouse() == 0)
    {                        // load mouse driver if not
            closegraph();                        // exit the program
            restorecrtmode();
```

```
            cout<<"\nMouse driver not loaded";
            exit(1);
    }
    gotoxy(14,10);
    cout<<s;                    // display current RPM
    gotoxy(71,10);          // and no. of rotations
    cout<<r;
    showmptr();
    resmptr(30,30,635,460);
    setcolor(LIGHTRED);
    rectangle(30,30,635,460);      //Border line
    rectangle(70,135,160,165);     //RPM Box
    rectangle(520,135,610,165);    //No.of Rotation box
    setfillstyle(SOLID_FILL,YELLOW);
    rectangle(180,130,320,170);
    floodfill(202,132,LIGHTRED);     //clockwise button
    rectangle(80,240,170,270);
    floodfill(82,242,LIGHTRED);     //RPM inc button
    rectangle(350,130,490,170);
       floodfill(352,132,LIGHTRED);        //anticlockwise
button
    rectangle(200,240,290,270);
    floodfill(202,242,LIGHTRED);    //RPM dec button
    rectangle(380,240,470,270);
    floodfill(382,242,LIGHTRED);    //rotation inc button
    rectangle(500,240,590,270);
    floodfill(502,242,LIGHTRED);    //rotation dec button
    line(125,220,245,220);
    line(425,220,545,220);
    line(185,220,185,210);
    line(485,220,485,210);
    line(245,220,245,240);
    line(545,220,545,240);
    line(125,220,125,240);
    line(425,220,425,240);
    text(8,13);
    outtextxy(195,60,"-: Control Panel :-");
    text(6,4);
    outtextxy(60,290,"Instructions :-");
    outtextxy(210,140,"Clockwise");
    outtextxy(360,140,"Anticlockwise");
    outtextxy(90,245,"Increase");
    outtextxy(210,245,"Decrease");
    outtextxy(390,245,"Increase");
    outtextxy(510,245,"Decrease");
    setcolor(10);
    outtextxy(175,190,"RPM");
    outtextxy(445,190,"Rotations");
    outtextxy(60,110,"Current RPM");
    outtextxy(495,110,"No.of Rotations");
    text(5,10);
    outtextxy(70,310,"# Press 'Clockwise' button to rotate
Stepper Motor clockwise");
       outtextxy(70,330,"# Press 'Anticlockwise' button to
rotate Stepper Motor anticlockwise");
    outtextxy(70,350,"# Press 'increase'/'decrease' button
to change the RPM");
    outtextxy(70,370,"# Press 'increase'/'decrease' button
to change the No. of rotations");
    setcolor(13);
      outtextxy(95,400,"Stepper Motor control using C++
design and developed by");
    outtextxy(250,420,"Ashutosh Bhatt");
    setcolor(YELLOW);
    outtextxy(200,440,"Press any key to exit program");
    while(!kbhit())          // loop until any key is pressed
    {
      getmpos(&but,&x,&y);                        // capture
current pointer position when click event happens
     if(x>=200 && x<=300 && y>=130 && y<=170 && (but
& 1) == 1)   // and switch to that if loop
                {
                   text(6,13);
                   outtextxy(210,140,"Clockwise");
                   for(int i=1;i<=n;i++)
                              {
                                sound(500);
                                outport(0x0378,0xcc);
                                delay(d);
                                outport(0x0378,0x3c);
                                delay(d);
                                outport(0x0378,0x33);
                                delay(d);
                                outport(0x0378,0xc3);
                                delay(d);
                                nosound();
                              }                  //for loop ends
                   text(6,4);
                   outtextxy(210,140,"Clockwise");
                }                      // first if ends

       else if(x>=350 && x<=490 && y>=130 && y<=170
&& (but & 1) == 1)
                {
                   text(6,13);
                   outtextxy(360,140,"Anticlockwise");
                   for(int i=1;i<=n;i++)
                              {
                                sound(750);
                                outport(0x0378,0xcc);
                                delay(d);
                                outport(0x0378,0xc3);
                                delay(d);
                                outport(0x0378,0x33);
                                delay(d);
                                outport(0x0378,0x3c);
                                delay(d);
                                nosound();
                              }            // for loop ends
                             text(6,4);
                             outtextxy(360,140,"Anticlockw
ise");
                 }            //second if ends

     else if(x>=80 && x<=170 && y>=240 && y<=270 &&
(but & 1) == 1)
                {
                   gotoxy(10,10);
                   cout<<"       ";
                   text(6,2);
                   outtextxy(90,245,"Increase");
                   sound(1000);
                   delay(200);
                   nosound();
                   if(s>=10) s=s+10;
                   else s++;                  // when this button
is pressed
                   s1 = s/60;                 // increase current
RPM and also
                   d = 50/s1;                 // change delay
                   gotoxy(14,10);
                   cout<<s;
                   text(6,4);
                   outtextxy(90,245,"Increase");
                }                // third if ends

              else if(x>=200 && x<=290 && y>=240 &&
y<=270 && (but & 1) == 1)
                {
                   gotoxy(10,10);
                   cout<<"       ";
                   text(6,2);
                   outtextxy(210,245,"Decrease");
                   sound(1000);
                   delay(200);
                   nosound();
                   if(s>10)
                              {
                                s=s-10;                  // when
this button is pressed
                                 gotoxy(14,10);                //
decrease it till s>1
                                 cout<<s;                    // if
s<=1 stop decreasing
                              }                             // and
display massage
                   else
                              {
                                if(s>1)
                                 {
                                   s--;
                                   gotoxy(14,10);
                                   cout<<s;
                                 }
                                else
                                 {
                                   gotoxy(11,10);
                                   cout<<"min limit";
                                 }
                              }
                   s1 = s/60;
                   d = 50/s1;
                   text(6,4);
                   outtextxy(210,245,"Decrease");
                }                    // forth if ends

              else if(x>=380 && x<=470 && y>=240 &&
y<=270 && (but & 1) == 1)
                {
                   gotoxy(67,10);
                   cout<<"       ";
                   text(6,2);
                   outtextxy(390,245,"Increase");
                   sound(1000);
                   delay(200);
                   nosound();
                   if(r<1) r=r*2;
// when this button is pressed
                   else r++;            // increase no. of rota-
tion
                   gotoxy(71,10);        // if rotations are <
1 then
                    cout<<r;                 // double it every
time
                   n=r*5;               // otherwise increase it
linearly
                   text(6,4);
                   outtextxy(390,245,"Increase");
                }                // fifth if ends

              else if(x>=500 && x<=590 && y>=240 &&
y<=270 && (but & 1) == 1)
                {
                   gotoxy(67,10);
                   cout<<"       ";
                   text(6,2);
                   outtextxy(510,245,"Decrease");
                   sound(1000);
                   delay(200);
                   nosound();
                   if(r>1)
                              {
                                r--;      // when this button
is pressed
                   gotoxy(71,10);                // decrease No. of
rotations
                    cout<<r;                 // till r=0.25 if r<0.25
stop decreasing
                              }   //and display a massage
                   else
                              {
                                if(r>0.25)
                                 {
                                                   r=r/2;
                                                   gotoxy(71,10);
                                                   cout<<r;
                                 }
                                else
                                 {
                                                   gotoxy(67,10);

cout<<"Ooppps...";
                                 }
                              }
                   n=r*5;
                   text(6,4);
                   outtextxy(510,245,"Decrease");
                }        // last if ends
    }     // while loop ends
  }       // main ends

  getmpos(int *but,int *x,int *y)
              {
                i.x.ax = 3;
                int86(0x33,&i,&o);
                *but = o.x.bx;
                *x = o.x.cx;
                *y = o.x.dx;
              }
   initmouse()
    {
              i.x.ax = 0;
              int86(0x33,&i,&o);
              return(o.x.ax);
              }
   showmptr()
              {
               i.x.ax = 1;
               int86(0x33,&i,&o);
              }
   resmptr(int a,int b,int c,int d)
              {
                i.x.ax = 7;
                i.x.cx = a;
                i.x.dx = c;
                int86(0x33,&i,&o);
                i.x.ax = 8;
                i.x.cx = b;
                i.x.dx = d;
                int86(0x33,&i,&o);
              }
              text(int e,int f)
              {
               setcolor(f);
               settextstyle(SMALL_FONT,HORIZ_DIR,e);
              }
```

AUTOMATIC 3-PHASE INDUCTION MOTOR STARTER

■ R. LAKSHMANAN

Starters for 3-phase squirrel-cage induction motors often use star-to-delta converters. The stator coils of the motor are connected in star configuration at the time of power-on and switched to delta configuration when the motor reaches 3/4th of its full speed, after the stator coils have developed sufficient back electromagnetic force (emf).

The starter circuit presented here offers two main advantages: single-phase prevention and automatic star-to-delta conversion. It can be used only with those motors which are rated for connection in delta configuration at the given line voltage and which have both ends of each of the three stator windings available individually.

At start, the line voltage is applied to one end of each of the three windings, with the other ends bridged together, effectively connecting the windings in star configuration. Under this connection, the voltage across the windings is $1/\sqrt{3}$ of line-to-line supply voltage and so the current flowing through each winding is also reduced by this factor. Compared to delta connection, the resultant current flowing from the supply, as also the torque, is reduced by a factor of 1/3 in star configuration. The relevant equations for star and delta connections are given in the box.

As soon as the moment of inertia is overcome, and sufficient back emf is induced in the stator windings, the star connection is opened and the ends of the windings are connected to the 3-phase supply in a fashion to create a delta connection.

Induction motor basics

The AC induction motor, also called the squirrel cage motor, comprises a simple cage-like rotor and a stator containing three windings. The changing field produced by the AC line current in the stator induces a current in the rotor, which interacts with the field and causes the motor to rotate.

The base speed of the AC motor is determined by the number of poles built into the stator windings and the frequency of the AC input voltage. A load on the motor causes the motor to slip in proportion to the load.

Circuit description

Fig. 1 shows the circuit of the automatic star-to-delta converter comprising a single-phase preventer and a timer.

Three single-phase transformers are used to step-down the 3-phase supply separately. Phases R, Y and B are stepped down by transformers X1, X2 and X3 to deliver the secondary output of 12V at 300 mA. The transformer output is rectified by a full-wave rectifier and filtered by a capacitor.

The three 12V DC supplies drive relays RL1, RL2 and RL3, respectively. When all the three phases are present, the 12V DC supply derived from the R

PARTS LIST

Semiconductors:

IC1	-	NE555 timer
T1	-	BC557 pnp transistor
T2	-	BC547 npn transistor
D1- D16	-	1N4007 rectifier diode
D17	-	1N4148 switching diode
LED1	-	Green LED
LED2	-	Red LED

Resistors (all ¼-watt, ±5% carbon):

R1	-	56-kilo-ohm
R2-R5	-	1-kilo-ohm
VR1	-	470-kilo-ohm preset

Capacitors:

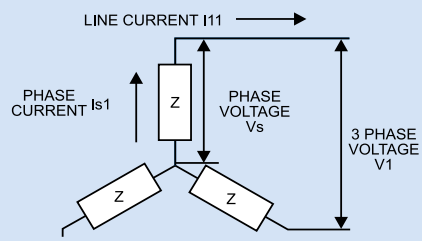
C1	-	1000µF, 25V electrolytic
C2, C3	-	10µF, 25V electrolytic
C4	-	470µF, 25V electrolytic
C5	-	0.01µF ceramic disk

Miscellaneous:

X1-X3	-	230V AC primary to 12V, 300mA secondary transformer
RL1, Rl2	-	12V, 200-ohm, 1c/o relay
RL3, RL4	-	12V, 250-ohm, 3c/o, 30A relay

Equations for Star & Delta Connections

STAR CONNECTION

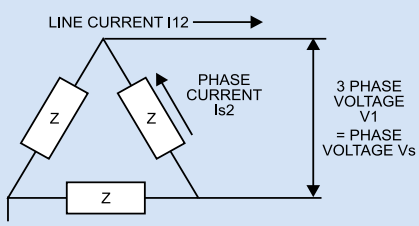


$$\text{PHASE VOLTAGE Vs} = \text{3 PHASE VOLTAGE } V1 \times \frac{1}{\sqrt{3}}$$

$$\text{PHASE CURRENT IS1} = \frac{\text{PHASE VOLTAGE Vs}}{Z} = \frac{\sqrt{3} \times V1}{3Z}$$

$$\text{LINE CURRENT I11} = \text{PHASE CURRENT IS1} = \frac{\sqrt{3} \times V1}{3Z}$$

DELTA CONNECTION



$$\text{PHASE VOLTAGE Vs} = \text{3 PHASE VOLTAGE } V1$$

$$\text{PHASE CURRENT IS2} = \frac{\text{PHASE VOLTAGE Vs}}{Z} = \frac{V1}{Z}$$

$$\text{LINE CURRENT I12} = \sqrt{3} \times \text{PHASE CURRENT IS2} = \frac{\sqrt{3} \times V1}{3Z}$$

CONCLUSION

$$\frac{\text{LINE CURRENT I11}}{\text{LINE CURRENT I12}} = \frac{\sqrt{3} \times V1}{3Z} \times \frac{Z}{\sqrt{3} \times V1} = \frac{1}{3}$$

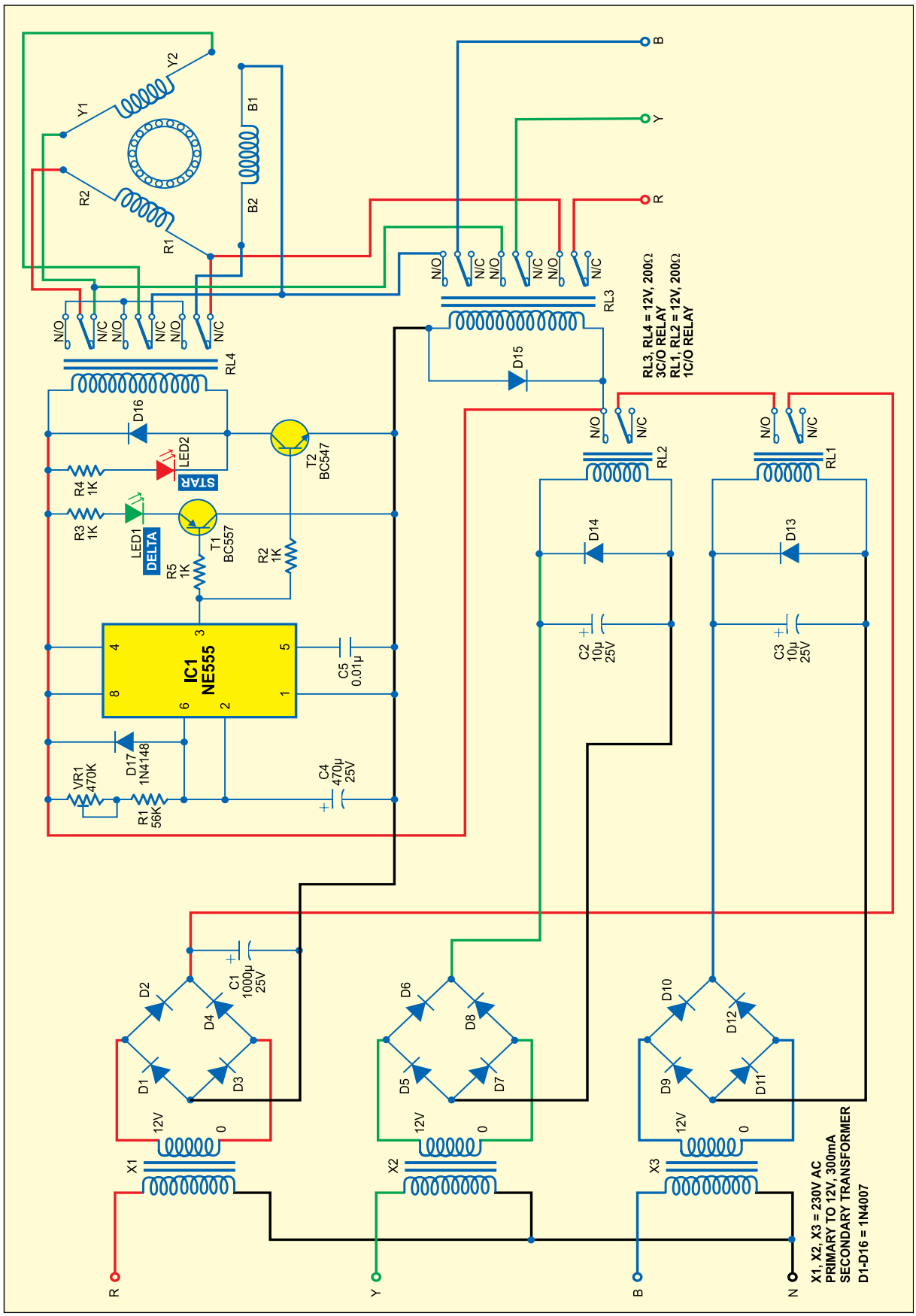


Fig. 1: Circuit of automatic 3-phase induction motor starter

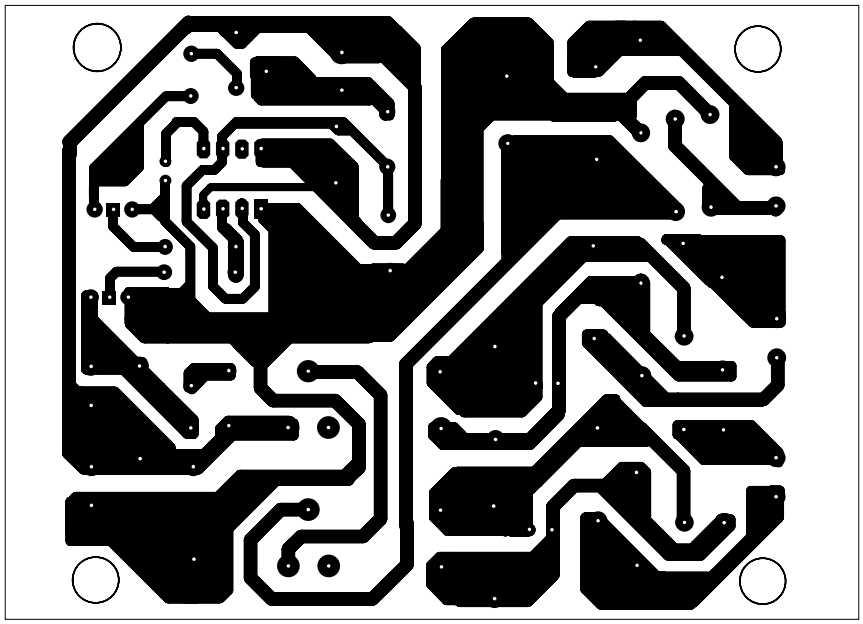


Fig. 2: Actual-size, single-side PCB layout of Fig. 1

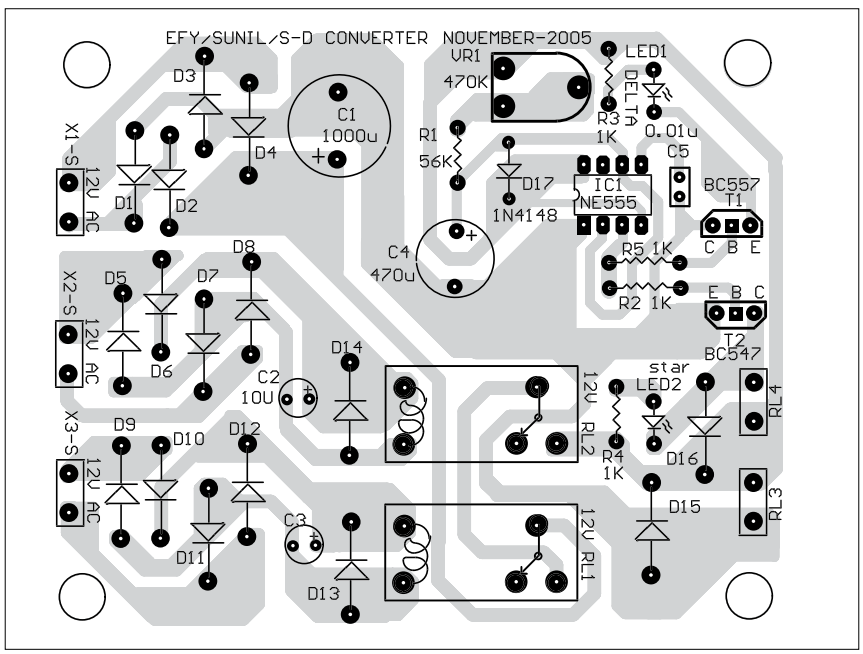


Fig. 3: Component layout for the PCB

phase is fed to the coil of relay RL3 and the timer circuit through the contacts of relays RL1 and RL2. As a result, relay RL3 energises.

Simultaneously, timer NE555 (IC1), which is configured as a monostable multivibrator, is also triggered. Its time period is determined by capacitor C4, resistor R1 and preset VR1. Preset VR1 is used to set the time period required to reach 3/4th of the full speed of the motor. The negative triggering pulse for IC1 is provided by the combination of resistor VR1, R1 and capacitor C4. The timer output at pin 3 is connected to the base of transistor T2 via resistor R2. As a result, transistor T2 is driven to saturation and relay RL4 energises (indicated by glowing of LED2). Thus at power-on, relay RL3, as also RL4, energises (if all three phases are present) to connect the stator windings in star configuration. On tracing the connections you will observe that R phase is connected to R1 end of R windings, Y phase is connected to Y1 end of Y windings and B phase is connected to B1 terminal of B stator windings. The other ends of all the stator windings (i.e., R2, Y2 and B2) get bridged together to form star connection.

After the specified delay, which is provided for the speed of the motor to 3/4th of its full speed value, the monostable output goes low to cut off transistor T2 and de-energise relay RL4. The motor stator coils now switch to delta configuration. Now you will observe that R phase gets connected to the junction of R1 and B2 terminals, Y phase is connected to Y1 and R2 terminals and B phase is connected to B1 and Y2 terminals of the stator winding. This connection conforms to delta configuration. Since the output of IC1 is low in this state, pnp transistor T1 is forward biased to light up LED1 and indicate delta configuration.

Relay ratings. RL1 and RL2 are normal control relays, which are used to energise relay RL3. PCB-mounted OEN Type 57 relays rated for 12 volts (or equivalent) may be used for the purpose. RL3 and RL4 are required to support the complete line current during star as well as delta configurations. Hence the contacts must be rated to withstand full line-to-line voltage and expected full current of the motor in delta configuration. Thus heavy-duty power relays of appropriate voltage and current rating for 12V coils must be used. The relays are to be mounted outside the PCB on the chassis of a suitable metal cabinet, which must be earthed properly to avoid any risk of shock.

Before connecting the motor to the circuit, proper operation of the relays must be checked. The windings must be connected as shown in the circuit diagram.

Construction

An actual-size, single-side PCB for the automatic 3-phase induction motor starter circuit is shown in Fig. 2 and its component layout in Fig. 3. ●

USING AVR MICROCONTROLLERS FOR PROJECTS

■ **K. PADMANABHAN, P. SWAMINATHAN & S. ANANTHI**

The AVR 8535 microcontroller and its new version ATmega8535 are versatile, high-performance but low-cost chips. This article series covers typical applications of this processor illustrating its power and cost-effectiveness in an embedded system.

The AVR family comprises several chips, all with almost the same instruction set. Of them, the 90S8515, 90S8535 and ATmega8535 chips are low-cost and readily available with the complete set of port pins. The ATmega8535-16 is more powerful and available for around Rs 250. Capable of running at 16 MHz and achieving almost 16 million instructions per second (MIPS), it is one of the fastest devices available in the market today.

Using ATmega8535, you can build a microcontroller-based project with following features:

1. Four ports, of which one of them has eight analogue-to-digital converter (ADC) channels

2. ADC conversion time is as little as 60 microseconds. Imagine adding an external ADC to 8051 or any other microcontroller chip—that would have taken the cost to over four digits. And mind you, it is a 10-bit ADC, not just 8-bit.

3. If an 8MHz crystal is connected, each instruction executes in 1/8th of a microsecond. The 89C51 at 12MHz clock had its internal division by twelve, so it ran at just one microsecond. Thus, ATmega8535 chip is eight times faster with an 8MHz crystal. However, you can also use a higher-frequency crystal. The chip is basically a RISC processor that executes most instructions in one clock cycle itself.

4. The chip has RS-232 transmit and receive terminals much like the 8051 family, but it can support even higher baud rates.

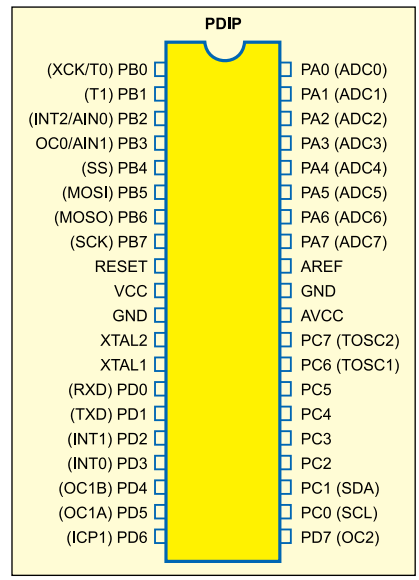


Fig. 1: Pin configuration of ATmega8535

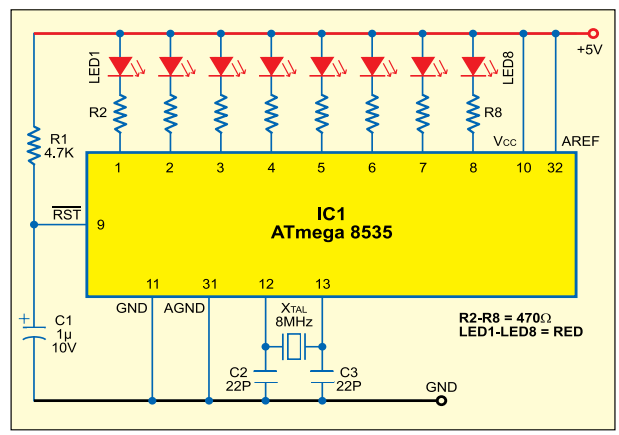


Fig. 2: A simple LED display circuit using ATmega8535

5. It has quite a few internal registers, RAM, EEPROM and CODE memory (flash memory in excess of 4kB).

6. The instruction set is versatile, complete with several arithmetic, logic and transfer instructions and related jump instructions, etc.

7. An analogue comparator pin, which can compare an external analogue voltage and take control action.

8. Reset is possible through the software, and a watchdog is provided. Power-down or sleep modes are available.

9. An additional serial interface, known as the SPI bus, with three wires: data (2) and clock (1). These pins can be used for programming or loading the code from a PC through the printer port or serial port. For programming the internal flash memory locations, just 5V supply is enough.

10. Two PWM output pins, which are useful for power control applications.

11. Several timers as in other members of the 8051 family, but with much better time resolution.

12. Additional features like input capture and output compare.

Here, we shall delve into the chip's operations with typical programs and circuits. All the development tools including 'C' compiler are available for free from the Internet.

The features of ATmega8535 make it the right candidate for various embedded control applications. Even a digital filter can be implemented on the device, provided you are fully conversant with its hardware and software features. You can download the databook of ATmega8535 from the 'ATMEL.com' Website to understand its features and work out simple applications.

The sample programs given here can be used to yield a powerful con-

troller for many applications like a filter or motor controller.

Programming the chip

The AVR source code file with '.asm' extension can be written using either the EDIT, Wordpad or notebook programs.

As with all microprocessor or microcontroller programs, for the source code, one has to enter the program by mnemonics and assembler directives and then convert the same into a code list for the program. (Directives are assembler commands used to control the input, output and data allocation of the assembler. These are, however, not translated into op-codes directly.) This is done using the cross-assembler software 'avrasm.exe.'

To describe the modus of writing of an Assembly language program, a simple program (LED.ASM) for AVR processors is given below:

LED.ASM

```
.NOLIST
.INCLUDE "m8535def.inc"
.LIST
.DEF      mp = R16
.org $0000 ; Reset address
rjmp      main
main:
ldi R16,low(RAMEND); Load low byte
         address of the end of the RAM
         into register R16
out SPL,R16 ; Initialise stack
         pointer to the end of the
         internal RAM
ldi R16,high(RAMEND) ; Load high byte
         address of the end of the RAM
         into register R16
out       SPH, R16  ; Initialise high
          byte of stack pointer ; to the
          end of the internal RAM
ldi       mp,0b11111111
          out       DDRB,mp
loop: ldi mp,0x00
          out       PORTB,mp
          Rcall delay
          ldi       mp,0xFF
          out       PORTB,mp
          Rcall delay
          ldi       mp,0xFF
          out       PORTB,mp
          rjmp      loop
delay: clr r19
          ldi r17,$ff
          loop1: inc r17
          brne loop1
          inc r19
          brne r19,loop1
          ret
```

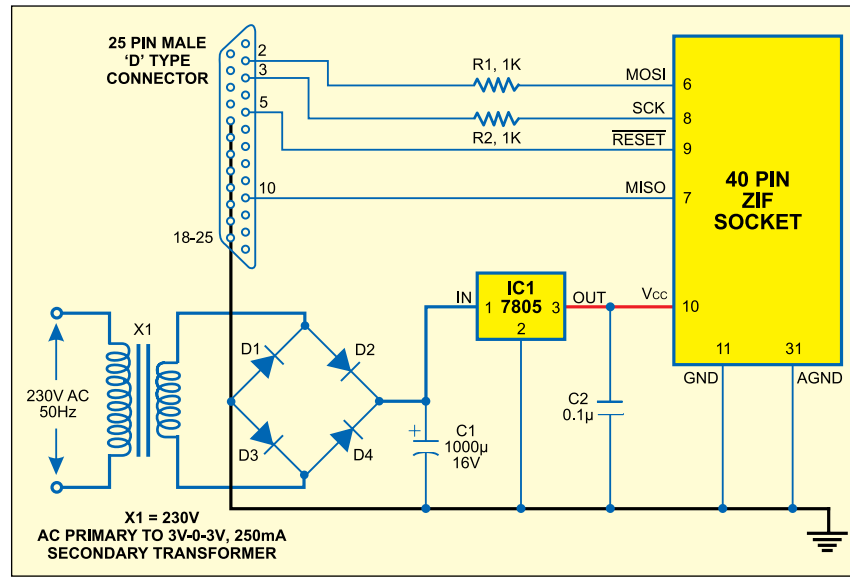


Fig. 3: Circuit diagram of AVR programmer (Pod)

This program helps you understand:

1. Access to the output port (here port B, where LEDs are connected)
2. The different parts of a typical assembler program
3. Different conventions like use of semicolon, upper-/lower-case letters, etc

Explanatory notes for LED.ASM

1. In Assembly language, all the text on a line after a semicolon (;) is treated by the cross-assembler as comments and it does not use it for code formation.

2. Including the m8535def.inc processor-specific file in Assembly program means all the I/O register names, I/O register bit names, etc appearing in the datasheet can be used. Failure to include this file may result in a number of error messages. Ensure that this file is placed in the same directory as your source code file (LED.asm in this case). Else, give complete path for the m8535def.inc file.

3. Following conventions have been used in the program:

(a) Words in upper-case letters are used for command directive words of the Assembly language or predefined ports of the processor.

PARTS LIST

Parts list for LED display circuit (Fig. 2)

Semiconductors:
IC1 - ATmega8535
LED1-LED8 - Red LED

Resistors (all ¼-watt, ±5% carbon):
R1 - 4.7-kilo-ohm
R2-R8 - 470-ohm

Capacitors:
C1 - 1µF, 10V electrolytic
C2, C3 - 22pF ceramic disk

Miscellaneous:
X_{TAL} - 8MHz

Parts list for AVR Programmer (Fig. 3)

Semiconductors:
IC1 - 7805 5V regulator
D1-D4 - 1N4007 rectifier diode

Resistors (all ¼-watt, ±5% carbon):
R1, R2 - 1-kilo-ohm

Capacitors:
C1 - 1000µF, 16V electrolytic
C2 - 0.1µF ceramic disk

Miscellaneous:
X1 - 230V AC primary to 6V, 250mA secondary transformer
- 40-pin ZIF socket
- 25-pin D-type male connector

Parts list for message display on the LCD (Fig. 6)

Semiconductors:
IC1 - ATmega8535

Resistors (all ¼-watt, ±5% carbon):
R1 - 4.7-kilo-ohm
VR1 - 10-kilo-ohm preset

Capacitors:
C1 - 1µF, 10V electrolytic
C2, C2 - 22pF ceramic disk

Miscellaneous:
X_{TAL} - 8MHz
- 16x1-character Hitachi make LCD or 16×2-character LCD

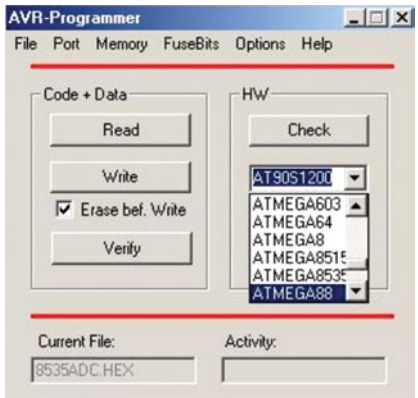


Fig. 4: Screenshot of AVR-Programmer

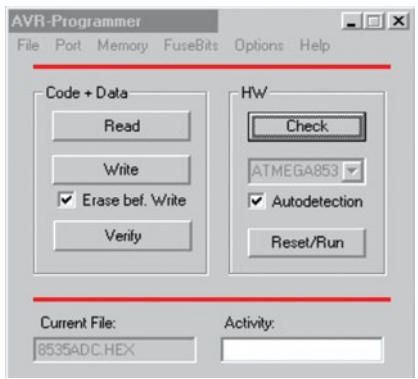


Fig. 5: Screenshot of AVR-Programmer showing activity window

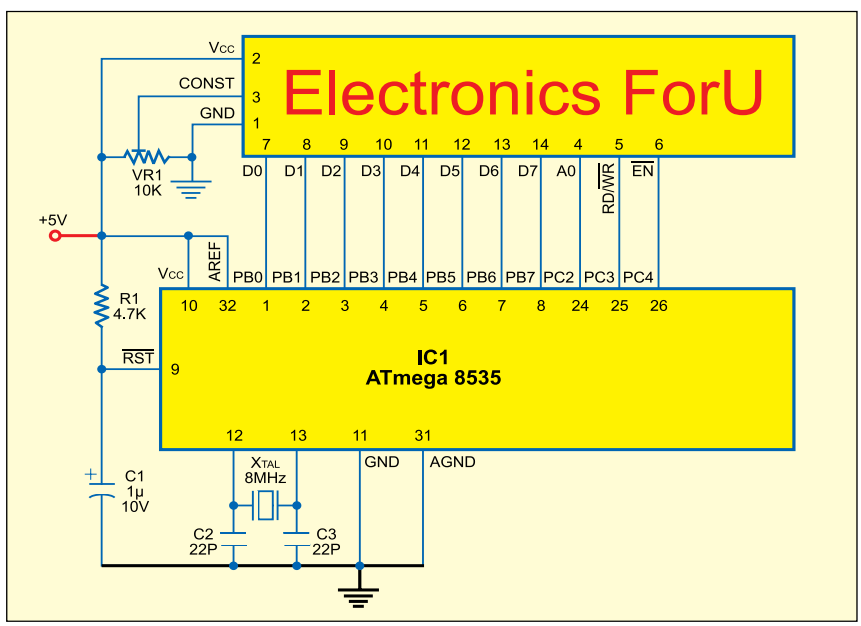


Fig. 6: Circuit for message display on the LCD

(b) Mnemonic words are written in lower case.

4. LIST directive turns on the listing output if it had been previously turned off. Similarly, NOLIST directive, if used, will turn off the listing output.

5. DEF directive is used to define a text-substitution label for a string. A label/name is easy to remember. Here, register R16 is replaced with 'mp' name. Thus whenever 'mp' is encountered in the source code, it will be automatically replaced with 'R16.'

6. '.org $0000' defines the reset address. When power is switched on, the program starts from this location. A restart from the reset address can be activated by resetting the respective hardware pin of the chip (pin 9) or upon watchdog timer reaching its zero count. A relative jump command (rjump) at this reset location directs the program execution to label (main)—as long as the label is within 2k locations from the reset address (0000). Incidentally, 'rjmp main' is the first code-generating instruction.

7. It is essential to set up the stack pointer before being able to call any subroutine, since stack is required for saving the return address, where the next program execution is to start from. The program lines starting with 'ldi R16,low(RAMEND)' and ending with 'out SPH, R16' do just that.

8. The 'ldi mp, 0b11111111' and 'out DDRB, mp' lines set port-B pins as the output. The first line, interpreted as 'load immediate (ldi) into register 'mp'', loads binary value '11111111' into the 'mp' register. The second line transfers the contents of 'mp' (11111111) to the data direction register of port B (DDRB). DDRB is already defined in the m8535def.inc file. (If you want to set port-B pins as input, load binary '00000000' into 'mp' and output it to DDRB.) Incidentally, '0b' precedes a binary number. Similarly '0x' precedes a hex number. Numbers without these prefixes denote decimal numbers by default. Hence you may replace '0b11111111' with either '0xFF' or simply '255' to achieve the same results.

9. The rest of the program starting at label 'loop:' and ending with 'rjmp loop' achieves switching on and off of the LEDs with a delay. The delay subroutine starting at label 'delay:' and ending with return instruction 'ret' is called from within the loop.

Initially, 'mp' is loaded with hex value '00' and output through port-B pins, making them low. Since the cathodes of all the eight LEDs are connected to these port pins via current-limiting resistors, the LEDs light up. Thereafter, the delay subroutine (Rcall delay) is called and 'mp' is loaded with hex value 'FF' and transferred to the port-B output to turn off the LEDs. The loop is repeated as long as the power is switched on.

10. The internal R-C clock of ATmega8535 is 1 MHz by default. In the absence of 'Rcall delay' instruction, each of 'ldi' and 'out' instructions requires 1000 ns, while 'rjmp' instruction requires 2000 ns. Thus loop execution would take 4000 ns. This amounts to LED switching rate of 250 kHz.

Introduction of delay between switching on and off reduces this frequency to around 0.5 Hz by decrementing registers 'r19' and 'r17' from '255' to '0,' thereby making the elapsed time slower by 256×256 (which works out to around 0.5Hz rate).

After assembling the LED.asm source file, the program will have eight words. The LED.LST file stores the result of the assembly process in the form of a listing.

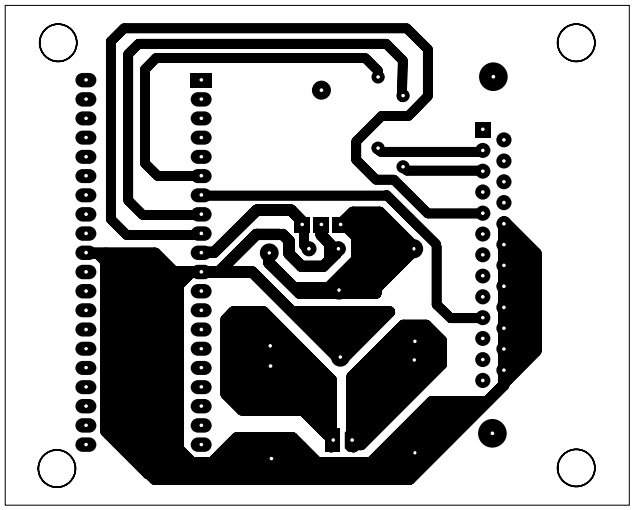


Fig. 7: Actual-size, single-side PCB layout for AVR programmer (Pod)

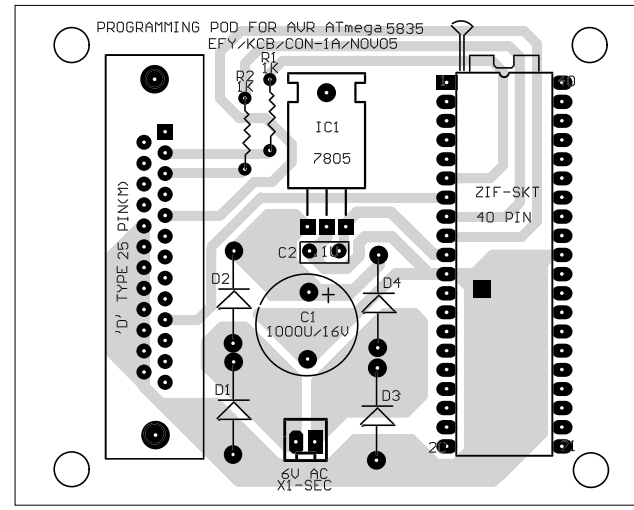


Fig. 8: Component layout for the PCB in Fig. 7

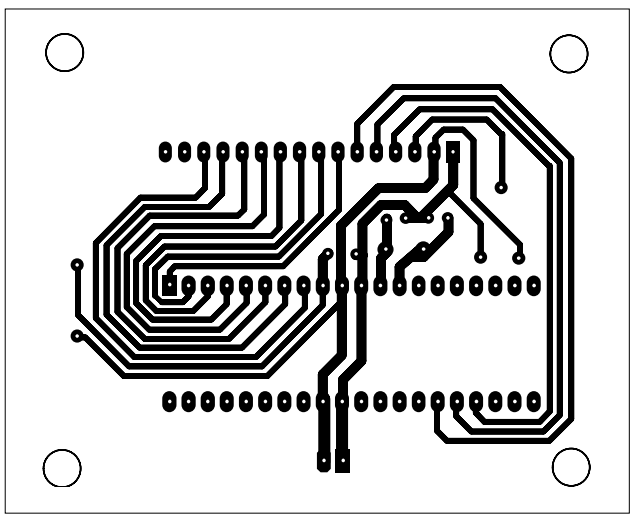


Fig. 9: Actual-size, single-side PCB layout for message display on LCD

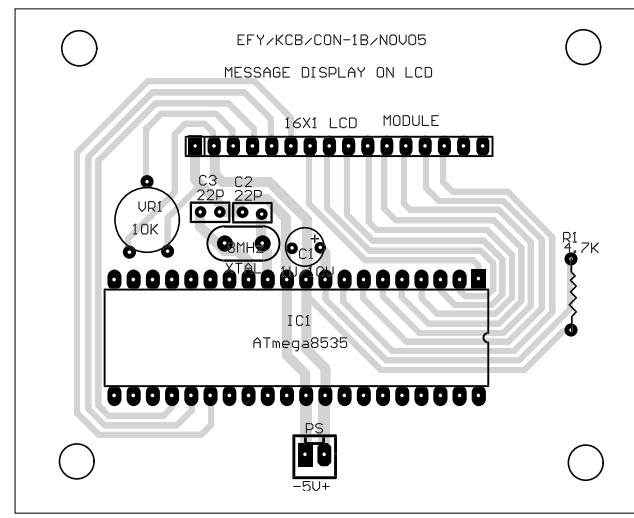


Fig. 10: Component layout for the PCB in Fig. 9

Once a program has been written using any editor, wordpad or notepad, it is assembled using the avrasm.exe AVR assembler, available on the download link given at the end of this article. Of course, the AVRSTUDIO 4.0 integrated development environment (IDE) is more versatile and user-friendly software for development, but the avrasm.exe assembler is simpler and direct.

Simply typing 'avrasm -i LED.asm LED.lst LED.hex' under the DOS prompt makes the cross-assembler generate code for the LED.hex file and also provide a text file giving both the code and the program together in LED.lst. Thus, you get the LED.lst listing file and the LED.hex Intel hex code file.

Alternatively, you can prepare a batch file as follows:

Upon DOS prompt, enter 'copy con avr.bat.' In the following line, type 'Avrasm -i %1.asm %1.lst %1.hex.' Pressing 'F6' key in the following line displays 'Control-Z.' Now pressing the 'Enter' key displays "1 file copied."

Now the avr.bat file has been prepared. This simple batch file is invoked to assemble this (or any) program by typing 'Avr LED' upon the DOS prompt and pressing the 'Enter' key.

This assembles the program, and forms both the list file (that contains the code-cum-Assembly listing) and the hex file (the actual Intel-format hex file for use by the programmer).

Likewise, any other assembly program 'xxx.asm' can be coded into the hex file by simply typing 'avr xxx' on DOS prompt. 'xxx' denotes the name of the program. The '.asm' is not to be typed.

In our LED.asm program, we have included the m8535def.inc file. This file is required along with the avrasm.exe cross-assembler. For other AVR processors like 90S8515, 90S8535 and at-Tiny 26, the files to be included are 8515def.inc, 8535def.inc and tn26def.inc, respectively.

The next task is to burn the code into the chip. Note that a chip previously programmed or erased is automatically erased when a new program is burnt into it using the device programmer as described below.

The AVR device programmer

The AT-PROG programmer software is used for programming ATmega8535.

This menu-drive programming software is simple to use and invoked from command prompt.

The software uses a simple pod connected to the printer port of a computer. The circuit of the pod (shown in Fig. 3) is very simple. It just connects the IC to be programmed to the pins of the PC's printer port.

This circuit is assembled on a small PCB with a D25 male-female plug at one end. The IC base is a 40-pin zero-insertion-force socket (ZIF). This enables easy insertion and removal of the IC to be programmed.

The AT_PROG.exe is a simple programming software that can be run under DOS prompt by typing AT-PROG. The files At-prog-hlp.htm, At-prog.exe, At-prog.cfg and At-prog.ini should be placed in one directory before running the AT-PROG. These files have been included in this month's EFY-CD as part of this article.

The menu-driven window of the AT-PROG programmer has the following menu items:

1. File menu. This menu is used to select or open the LED.hex file, or whatever, which is to be programmed into the device.

Pull down the menu by clicking it. Under 'Open' option, enter the file name as 'LED.hex' and press 'Enter.' The IC to be programmed is selected from the AVR-Programmer window by clicking the edge of the small rectangular window and choosing the IC as shown in Fig. 4. Now connect the printer-port connector to the programming pod, whose circuit is shown in Fig. 3.

2. ***Write menu.*** On clicking the 'Write' menu, the 'Activity' window at the bottom whitens and shows 'Connecting' (refer Fig. 5). Then, the data is transferred to the IC and verified after programming, showing 'ok' in the same window.

3. Check menu. This menu is used to find out whether the IC is inserted in the socket and whether the connector connections are okay. It will indicate an error if the IC is not there or not responding.

In this mode of programming, the serial-peripheral interface (SPI) of the AVR chip is used. This interface has three wire connections:

(i) Master output and slave input (MOSI)

(ii) Master input and slave output (MISO)

(iii) Serial clock (SCLK)

Using these wires, the SPI interface does the serial transfer of data (i.e., our program codes) into the chip, which is configured as a slave. The data and clock are connected via MOSI and SCLK pins of the chip, respectively. Upon reception of each byte, the chip acknowledges it by sending a byte (53hex).

In 'Check' mode, the IC is enquired about its name by the computer (Master), which it replies with its signature code embedded in the chip memory by the manufacturer. Each IC has its specific signature code. Thus, by noting the code itself, what IC is being programmed will be known to the computer. So the small window under the device-select rectangular window can be clicked to show 'autodetect' the IC.

4. Options menu. In this menu, the speed of the clock used for transferring data from the computer can be selected as 'slow,' 'normal' or 'fast.' With present high-speed PCs, choose 'normal' or 'slow.' In the same menu, the 'read signature bytes' option is to be enabled and it is so by default.

5. Port menu. The port menu, which is next to the file menu, is useful if a different printer port is available. The program automatically selects the available printer port.

When the 'Activity' window shows 'ok' after clicking the 'Write' menu, remove the programmed chip from the programmer circuit board and fix it onto the target circuit for the LED.asm program (shown in Fig. 2). Now apply 5V and press the switch connected to Reset pin, if needed. (The circuit resets at power-on.) The LEDs start blinking fast and the waveform can be observed on the CRO for any of the pins at the output to the LEDs. It will be around 600 Hz.

Message display on the LCD module

Method I. Given below is the source code for message display on the LCD module along with suitable comments wherever needed.

LCD_CHAR.ASM

```
;*************************************
; * This program writes a message on to
the LCD *
;*************************************
.NOLIST
.INCLUDE "m8535def.inc"
;device =ATmega8535
.LIST
;
; Constants
;
; Used registers
;
.DEF rmpr = R16
.DEF temp = R14
.DEF result=R12
.DEF mpr =R16
; Code starts here
;
.CSEG
.ORG $0000
;
; Reset-vector
rjmp Start ; Reset-vector

;******** various subroutines for LCD
display********
;cmd is the LCD module's command entry
subroutine. Command value in R16
         cmd:  cbi portc,2
                    cbi portc,3
                    cbi portc,4
                    out portb,r16
                    sbi portc,4
                    nop
          nop
          nop
          nop
          nop
          cbi portc,4
          rcall delay1
          ret
;lcdwr is the LCD module's data entry
subroutine.
ASCI code value in R16
         lcdwr:  cbi portc,2
                      cbi portc,3
                      cbi portc,4
                      sbi portc,2
                      out portb,r16
```

```
                    sbi portc,4
                    nop
                    nop
                    nop
                    nop
                    nop
                    cbi portc,4
                    rcall delay1
                    ret
;init_lcd is the LCD module's initialise
LCD routine for cursor, etc.
init_lcd:
ldi R16,$38 ;function_set command for
8-bit data
         drive
rcall cmd ;write this in the command
register for LCD
rcall delay1 ; wait since this takes some
         milliseconds for LCD
rcall delay1
ldi R16,$0e ; command for entry mode set
rcall cmd ; cursor active, display on,
          no blink.
rcall delay1
ldi R16,6 ; command for cursor shift
         right after each write
rcall cmd
ldi r16,1 ;command for clear display
rcall cmd
rcall delay1
ret

delay1:clr result
ldi R16,$a0 ; a suitable number for the
           required delay
loop2: inc R16 ; increments from 160
          ($a0) to 256 brne loop2
inc result ; increments result register
          from 0 to 255 brne loop2
ret ;got 256 times 95 for loop

; ********* End of the subroutine section
********
;
; ********** Main program **********
;
; Main program routine starts here
;
Start:    ldi       R16,low(RAMEND)
; Load the low byte address of the end
   of the RAM into register R16
         out       SPL,R16
; Initialise the stack pointer to the end
of the internal RAM
         ldi       R16,high(RAMEND)
; Load the high byte address of the end
  of the RAM into register R16
         out       SPH, R16
; Initialise the high byte address of the
  stack pointer to the end of the
  internal RAM

LCD:
 ldi r16,$ff
 out ddrb,r16 ;ff makes all bits as the
 output on port B
 out ddrc,r16 ;PORT C BITS USED FOR
            LCD WIRING
 ldi r16,$55
 out portb,r16;then alternate bits are
 low and high
 ($01010101)
          ;the above is a test which one
          can find if the program works!

 rcall init_lcd
 ldi R16,$80
 rcall cmd
; simply observe pins 1-8 for alternate
  high and low outputs!
 ldi R16,$45 ;"E"

 rcall lcdwr

 ldi R16,$6c ;"l"
 rcall lcdwr
 ldi R16,$65 ;"e"
 rcall lcdwr
 ldi R16,$63 ;"c"
 rcall lcdwr
 ldi R16,$74 ;"t"
 rcall lcdwr
 ldi R16,$72 ;"r"
 rcall lcdwr

 ldi R16,$6f ;"o"
 rcall lcdwr
 ldi R16,$6e ;"n"
 rcall lcdwr

 ldi R16,$c0 ; this command is to set to
            the next half of the LCD
 rcall cmd ;because 8 characters have
          filled the first half
;omit the above two lines if a two-row
         LCD display or a Hitachi 1-row
; display is used.
 ldi R16,$69 ;'i'
 rcall lcdwr
 ldi R16,$63 ;"c"

 rcall lcdwr

 ldi R16,$73 ;"s"
 rcall lcdwr
 ldi R16,$20 ;" "
 rcall lcdwr
 ldi R16,$46 ;"F"
 rcall lcdwr
 ldi R16,$6f ;"o"
 rcall lcdwr
 ldi R16,$72 ;"r"
 rcall lcdwr

 ldi R16,$55 ;"U"
 rcall lcdwr
here: RJMP HERE; Test of the serial
interface
```

This program displays 'Electronics ForU' on the LCD module (Fig. 6). The message may be displayed on the LCD in a single or two rows depending on the LCD module. In some LCD modules, the first eight characters are written consecutively, while for display of the next eight characters, the program needs to restart the cursor at address $C0. But Hitachi-make single-row types do not need to restart the cursor's address after the eighth entry; the characters can be written consecutively up to '16,' i.e., in a single row.

The program is named as 'LCD_CHAR.asm' and assembled into the '.hex' file by typing 'avr lcd_char' and invoking the cross-assembler AVR. Now the lcd_char.hex file is generated. The AT-PROG programmer burns this code into the flash memory of the ATmega8535.

Note that while assembling this program using 'avr lcd_char' command, the definition file for IC ATmega8535 (m8535def.inc) should be in the same directory.

Method II. This message display program uses look-up table. In the message display program described in Method I, 'Call lcdwr' instruction was written for each character. Here, instead, if we enter all the bytes for 'Electronics ForU' in a table, they can be picked up one by one until the end and shown on the LCD screen. For the

purpose, there is an instruction called load program memory (LPM).

The table, as also the name, is stored in the program memory. Here is the program along with necessary comments.

LCD TABLE.ASM

```
;————————————————————
;
.INCLUDE "m8535def.inc"
 ;device =ATmega8535
.LIST
;
; Constants
;
; Used registers
;
.DEF rmpr = R16
.DEF temp = R14
.DEF result=R12
.DEF mpr =R16
; Code starts here
;
.CSEG
.ORG $0000
;
; Reset-vector
rjmp Start ; Reset-vector

;********* various subroutines for LCD
display*********
;cmd is the LCD module's command entry
;subroutine.Command Value in R16
 cmd:     cbi portc,2
          cbi portc,3
          cbi portc,4

          out portb,r16
          sbi portc,4
          nop
          nop
          nop
          nop
          nop
          cbi portc,4
          rcall delay1
          ret
;lcdwr is the LCD module's data entry
subroutine.
Asci codeValue in R16
lcdwr:    cbi portc,2
          cbi portc,3
          cbi portc,4
          sbi portc,2

          out portb,r16
          sbi portc,4
          nop
          nop
          nop
          nop
          nop
          cbi portc,4
          rcall delay1
          ret
;init_lcd is the LCD module's initialize
LCD routine for cursor etc.
init_lcd:
ldi R16,$38 ;function_set command for
8 bit data drive
rcall cmd ;write this in the command
register for LCD
rcall delay1 ; wait since this takes some
          milliseconds for LCD
rcall delay1
ldi R16,$0e ; command for entry mode set
rcall cmd ; cursor active, display on,
no blink.
rcall delay1
ldi R16,6 ; command for cursor shift
           right after each write
 rcall cmd
 ldi r16,1 ;command for clear display
 rcall cmd
 rcall delay1
 ret

 delay1:
 clr result
 ldi R16,$a0 ; a suitable number for the
               required delay

loop2: inc R16 ; increments from 160
          ($a0) to 256
           brne loop2
           inc result ; increments result
           register from 0 to
           255
           brne loop2
           ret ;got 256 times 95 for loop

;************ End of the subroutine
section ***********
;
; ********** Main program ************
;
; Main program routine starts here
;
Start:    ldi       R16,low(RAMEND)

; Load low byte address of end of RAM
       into register R16
         out       SPL,R16
; Initialize stack pointer to end of
  internal RAM
         ldi       R16,high(RAMEND)
; Load high byte address of end of RAM
  into register R16
         out       SPH, R16
; Initialize high byte of stack pointer
  to end of internal RAM
 LDI R16,$FF
 OUT DDRB,R16
 OUT DDRC,R16 ; MAKE PORTS B AND C AS
OUTPUT PORTS (WIRED TO lcd)
 rcall init_lcd
 clr r17
 clr r18 ; required in the table fetch
 routine LCD:
 ldi ZH, high(table*2) ; Set up Z to
  point to the beginning of table
          ldi ZL, low(table*2)
          add ZL, r17
          ; Offset Z by r18:r17
          adc ZH, r18
          lpm
          ; Load

 mov r16,r0 ;get loaded value into r16
 cpi r16,$ff ;table end?
 breq idle
 rcall lcdwr ;write on lcd display
 inc r17
 rjmp LCD
; use Hitachi LCD display module if 1
row type is used; or else use ; ;any
two-row type LCD.
Otherwise, this above program sequence
will ;work only for 8 characters. The
rest will not be seen:
"Electron " ;only will be visible.

idle:
          ldi r16, (1<<SE) ; Enable sleep
          out MCUCR, r16
          sleep
          rjmp idle
table:
.db $45,$6c,$65,$63,$74,$72,$6F,
$6e,$69,$63,$73,$72,$6f,$72,$55,$
```

The actual-size PCB for programming and LCD message display are given in Figs 7 and 9, while their component layouts are shown in Figs 8 and 10, respectively.

In the first part of this article, we had described the main features of the AVR microcontroller and the hardware/software required for an AT-PROG programmer board interfaced to the printer port of a PC. Further, we explained the methods for message display on a liquid crystal display (LCD).

This part dwells on the architecture of ATmega8535 along with application programs exploiting its important features for embedded control.

Architecture of ATmega8535

Pin configuration of ATmega8535 was shown in Fig. 1 of Part 1. The device has ports for input/output, interrupts, serial communication and various others functions. There are a total of 32 pins, which are arranged as 'A,' 'B,' 'C' and 'D' ports for various functions as shown in Table I.

A crystal of maximum 16MHz or 8MHz frequency can be connected across pins 12 and 13 of ATmega8535 or its low-voltage version ATmega8535(L), respectively. Pin 9 serves as the active-low reset pin.

The non-volatile program and data memories built into ATmega8535 are:

1. 8 kB of self-programmable flash for storing the software code of the application program.

2. 512 bytes of SRAM, which is a read/write memory.

3. 512 bytes of EEPROM for storing the data. Unlike the flash memory, it can be accessed in a program for writing and reading.

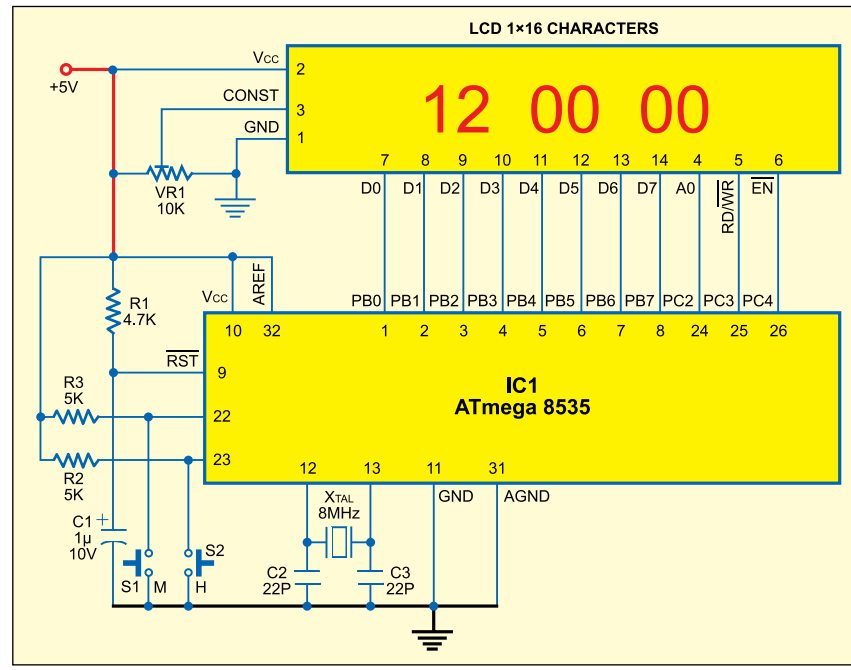


Fig. 11: Circuit diagarm of real-time clock

Programming the on-chip code/program memory

The on-chip flash memory is programmed by pulling up the reset pin and sending data through pins 6 (MOSI) and 7 (MISO), and pin 8 (SCK), which is used for clocking the the code data into the flash memory. This is accomplished by the host computer by sending appropriate instructions and the code bytes; data verification is done by reading the flash memory and comparing it with the original code data. Writing the lock bits to prevent reading of the code in the chip is accomplished through the instructions and the relevant data.

For using the AVR device, these instructions are built into the AT-PROG program (explained in Part I), which is run on the host PC.

Selection of clock. There are some additional fuse bits, which can be programmed for some extra operational functions. Note that the AVR device, as shipped, is preset to work at 1 MHz with its internal oscillator. If you want to use an external crystal, say, of 8MHz frequency, you have to exercise this option by programming the fuse bits accordingly. A fuse bit is just like a flash code memory location.

The CKSEL fuse bits can be programmed to select the desired crystal. The device clocking options are selectable by Flash Fuse bits as shown in Table II. The clock from the selected source is input to the AVR clock generator and routed to the appropriate modules.

Since the default oscillator is 1MHz, unless we set the CKSEL bits to an appropriate value, the external crystal on pins 12 and 13 will not function for ATmega8535.

Programming the fuse bits. The fuse bit programming option is available on the screen when the AT-PROG is run on the PC. When this option is

TABLE I
Port Description

Port description	Pin Nos.	Usage
Port A (PA0-PA7)	40 to 33	Bidirectional I/O pins with 20mA sink capability and active internal pull-ups; alternately used as ADC input as well as data lines to external RAM
Port B (PB0-PB7)	1 to 8	Input or output port, also used for additional functions as T0, T1, AIN0, AIN1, SS, MOSI, MISO and SCK pins
Port C (PC0-PC7)	22 to 29	Used for address output if external RAM is attached; four pins are alternately used as SCL, SDA for I^2C, TOSC1 and TOSC2, respectively
Port D (PD0-PD7)	14 to 21	Bidirectional, as for port A. Also serve as pins for serial communication, interrupts 0 and 1, and PWM 1 and 2 output comparison, etc.

TABLE II
Device Clocking Options Select*

Device clocking option	CKSEL3.0
External crystal/ceramic resonator	1111-1010
External low-frequency crystal	1001
External RC oscillator	1000-0101
Calibrated internal RC oscillator	0100-001
External clock	0000

**For all fuses, '1' means unprogrammed, while '0' means programmed*

Instruction set for ATmega8535

The instruction set comprises several arithmetic, logical, branch and bit-test type instructions. You can download a 150-page user manual for the AVR instruction set from Atmel's site 'www.atmel.com/dyn/resources/prod_documents/doc0856.pdf.' A summary of the instruction set is given on pages 299 through 301 of the Atmega8535(L) datasheet.

Some of the important instructions are given in Table IV.

Points to be noted

1. When the relative call or jump instruction is executed, the entire memory address space can be accessed.

2. During interrupts and subroutine calls, the return address value is stored in the stack space, which is to be defined by the user at the beginning of every program in SRAM space.

3. The 16-bit stack pointer is read-/write-accessible in the I/O space.

4. The 512-byte data RAM is easily accessed through five different addressing modes supported.

5. A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the status register. Every interrupt has a separate address for vectoring, where the instruction causing it to jump to the memory area of that particular interrupt has to be kept stored by the programmer.

There are many interrupts available in ATmega8535. In order to use any interrupt, you need to place the address of the program of the respective interrupt service routine at the vector address.

From location '001H' to '014H,' there are 20 such interrupt vector locations in the order of their priority. Address '000H' is used for the reset vector. A reset may be caused by power-on reset, brownout reset and watchdog reset, or externally by making pin 9 low. Table 19 on page 45 of the datasheet lists the details of reset and interrupt vectors.

Note that here we are dealing with word addresses, so each location is actually two bytes long. In this two-byte location, if you place a RETI (return from interrupt) instruction, nothing will be done upon that interrupt. For example, if you place a jump instruction to the required routine, you can

TABLE III
Some Registers in the I/O Space of ATmega8535

DDRB data direction reg. of port B $37	DDRA $3A	DDRC $34	DDRD $31	UDR UART data reg. $2C
PINB input reg. of port B $36	PINA $39	PINC $33	PIND $30	UCSR UART control reg. A=$2B B=$2A
PORTB output reg. port B $38	PORTA#3B	PORTC $35	PORTD$32	UBRR UART baud rate reg.
ADMUX ADC channel sel. $27	ADCSRA ADC control/ status register* $26	ADCH ADC value high reg. $25	ADCL: ADC value low-byte reg. $24	ACSR analogue comparator control/ status reg. ($28)

**ADC in ATmega8535 is named 'ADCSRA'*

Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fig. 12: Bit description for status register; I-global interrupt-enable bit, T-T bit copy storage, H-half carry flag, S-sign bit, V-overflow in 2's complement arithmetic, N-negative number flag (2's complement arith.), Z-zero flag, C-carry flag

selected, it pops up a menu of its own. On this menu, you can write the necessary code for CKSEL programming.

Internal registers. Six of the 32 registers can be used as three 16-bit indirect address register pointers for data space addressing, enabling efficient calculations. One of three address pointers (X, Y and Z registers, described under 'register operations' section) is also used for table look-up.

The I/O memory space contains 64 addresses for CPU peripheral functions like control registers, timers/counters and analogue-to-digital converter (ADC). It can be accessed directly or as the data space locations following those of the register files, i.e., after '20H' and up to '5FH.'

The memory space contains important registers for use in interrupt selection, timer control, UART, SPI interface, watchdog and reset selection modes, etc. Table III shows the exact addresses of these I/O registers.

The bit description for the status register (SREG) is shown in Fig. 12.

PARTS LIST

Parts list for real-time clock (Fig. 11)

Semiconductors:
IC1 - ATmega8535

Resistors (all ¼-watt, ±5% carbon):
R1 - 4.7-kilo-ohm
R2, R3 - 5-kilo-ohm
VR1 - 10-kilo-ohm preset

Capacitors:
C1 - 1µF, 10V electrolytic
C2, C3 - 22pF ceramic disk

Miscellaneous:
X_{TAL} - 8MHz
S1, S2 - Push-to-on switch
- 16x1-character Hitachi make LCD or 16×2-character LCD

TABLE IV
Instruction Set for ATmega8535

1. Arithmetic and logical instructions

ADD rd, rs	SUB (rd-rs)	AND rd, rs	OR rd, rs	EOR rd, rs
ADC Add with carry	SBC Subtract with carry	ANDI AND with immediate data	ORI rd, K	COM rd One's complement
ADIW (Word) Add immediate to word	SBCI rd=rd-Carry-K	INC rd Increment reg.	CLR rd Clear register	NEG rd ; 2's complement
	SBIW Subtract immediate from word	DEC rd decrement reg.	SER rd Sets register	SBR rd,n ; sets *nth* bit in rd

Note: *K-immediate data, rd-destination register, rs-source register*

2. Data movement instructions

MOV rd,rs	LDI rd, K Immediate load	ST X, rs Stores rs into [X]	LPM R0 ← [Z] Load from prog. mem.	PUSH rs Pushes to stack the value of register Rs
	LD rd, X* Load indirectly	ST X+, rs ; store indirectly and incr. X	IN rd, port# As PORTB for reg.	POP rs Pops into Rs from stack
	LD rd, X+ Indirect, post incr. address	ST –X, rs Decr. after storing	Out Port, rd	
	LD rd, -X Dec. X by 1 and then read indirectly	STD Z+disp, rs Stores indirect with added displacement	STS k, rs Stores direct to SRAM addr. k	

**X denotes register pair R26-R27. Likewise, Y and Z are also usable for these instructions. # Port B should be entered as 'PINB' for inputs for the assembler; The notation PORTB used for output.*

3. Boolean logic BIT based instructions

SBI P, b Sets bit P=addr, bit no.	CBI P, b Clear bit in I/O register	LSL rd Logical left shift	ROL rd Rotate left through carry	SEC Sets carry flag
		LSR rd Logical right shift	ROR rd Rotate right through carry	CLC Clears carry
	CLI Interrupts disabled.	ASR Arithmetic shift right	SWAP rd Swaps nibbles in Rd	SEZ, CLZ Set/clear zero flag
	SEI Global interrupt enable		Bst/BLd reg, b Stores loads a bit in SREG(b) (status register)	NOP No operation

Note. *Bits can be in any I/O register or bits of any register Rd. P means an I/O address register.*

4. Frequently used test and skip as well as jump and call instructions

RJMP k Jumps k (-2047 to +2048)	RCALL k Calls relative	CP Rd, Rs Compares Rd-Rs	BREQ K Branches to K relative if zero flag is set	
LJMP PC ← Z	ICALL Calls to [Z] indirectly	CPSE Rd, Rs Compares, skips the next instruction if equal	BRNE K Does the opposite of the above	
RET Subroutine return		CPI Rd, K Immediate data compared	BRCS k Branch if carry is set	
RETI Interrupt service return			BRCC k Branch if carry not set	

Note. Most of these instructions are relative branching, except some, which need a full address of destination for jumping.

5. Additional multiply instructions

MUL	Rd, Rr	Multiply unsigned	R1:R0←Rd×Rr
MULS	Rd, Rr	Multiply signed	R1:R0←Rd×Rr
MULSU	Rd, Rr	Multiply signed with unsigned	R1:R0←Rd×Rr
FMUL	Rd, Rr	Fractional multiply unsigned	R1:R0←(Rd×Rr)<<1
FMULS	Rd, Rr	Fractional multiply signed	R1:R0←(Rd×Rr)<<1
FMULSU	Rd, Rr	Fractional multiply signed with unsigned	R1:R0←(Rd×Rr)<<1

write:

```
rjmp timer_routine
```

Then you can use that interrupt to jump to the timer_routine.

As mentioned above, the interrupt vectors follow the reset address at '000H,' wherein a jump instruction to the corresponding actual memory addresses, defined by the labels, is placed. For example, the instruction:

```
RJMP Int0
```

It means that the external interrupt

Bit	7	6	5	4	3	2	1	0
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Fig. 13: Bit details for TCCR0 register; bits 0, 1 and 2 are defined in Table IV reproduced from the original datasheet

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fig. 14: Bit details for TIMSK register

TABLE V
Clock-Select Bit Description

CS02	CS01	CS00	Description
0	0	0	No clock source (timer/counter stopped)
0	0	1	$clk_{I/O}$/(no prescalling)
0	1	0	$clk_{I/O}$/8 (from prescaler)
0	1	1	$clk_{I/O}$/64 (from prescaler)
1	0	0	$clk_{I/O}$/256 (from prescaler)
1	0	1	$clk_{I/O}$/1024 (from prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge
1	1	1	External clock source on T0 pin. Clock on rising edge

routine has the label 'Int0,' to which the processor jumps upon pin 17 getting a high logic signal. Also, at the label 'Int0,' if a simple return instruction is entered as:

```
Int0:   reti
```

This instruction simply ignores such an interrupt and returns to the main program. In case you need to process the interrupt, enter the necessary code starting at label 'Int0.'

After the interrupt processing instructions, the various subroutines are entered. Then comes the main program. In the main program, the first thing to write is the stack initialisation instructions. Here, the stack pointer is set to the highest end of the internal RAM, for which a temporary register is used, and the high and low addresses are written to the stack pointer using an instruction at the Ext_Int0 vector address (0x001) such as:

```
ldi temp,low(RAMEND)
out spl,temp
ldi temp,high(RAMEND)
out sph,temp
```

Register operations. Each register is assigned a data memory address, mapping it directly into the first 32 locations of the data space. Register pairs R26-R27, R28-R29 and R30-R31 serve as 16-bit registers, which are used for indirect addressing of the data memory space. These three 16-bit registers are known as 'X' (R27:R26), 'Y' (R29:R28) and 'Z' (R31:R30) registers, respectively. The last 16 registers in the register file (R16 through R31) cannot be used with the first 16 registers (R0 through R15).

The operating instructions for registers have direct and single-cycle access to the registers. The following instructions—constant arithmetic instructions—use the second half of the registers in the register file and cannot be used with the first half:

```
sbci, subi, cpi, andi, ori and ldi
```

The following general instructions that use two registers or only a single register can use the entire register file:

```
Sbc, sub, cp and & or
```

Embedded control functions and their applications

Here we'll use the following four functions of ATMega8535 for typical control applications:

1. Timers; two 8-bit and one 16-bit with add-on features
2. Pulse-width modulated output
3. Analogue-to-digital converter
4. Serial RS-232 interface

Timers and their applications. Both timer 0 and timer 1 are 8-bit timers, while timer 1 is a 16-bit timer. The clock inputs to the timers can have a variety of selections. The CPU clock itself, divided by a prescaling divider with divisors of 8, 64, 256 and 1024, can be chosen. Further, it can also count an externally applied clock at T1 pin (for timer 1).

We shall use these timers for developing a real-time clock with time display on the LCD (see Fig. 1). For the purpose, the registers to be used in timer 0 are:

1. TCCR0: Timer counter control register 0
2. TIMSK: Timer interrupt mask register

TCCR0 register bits. Fig. 13 shows the bit details for TCCR0 register. Bits WGM01, COM00, COM01, WGM00 and FOC0 (bits 3 through 7) of TCCRO register are used with the timer-based comparators for waveform and pulsewidth-modulated output generation. Since these bits are not required for the normal timing operation of the timer, they have not been used here.

The timer clock is selected by using the remaining three bits (CS00, CS01 and CS02). We will set these bits to '0,' '1' and '1,' respectively, for dividing the 1MHz default internal clock of ATmega8535 k (with no external crystal) by '64.' This division gives 65 microseconds per clock. Then we accumulate the counts for getting one second and divide it by '60' to get minutes and again by '60' to get hours, which are counted up to '12' and the process is repeated.

TIMSK register bits. Fig. 14 shows the bit details for TIMSK register. Bit 0 refers to 'timer-overflow interrupt enable.' It must be set to enable the interrupt action on overflow. The TIMO_OVF interrupt ($0009 address) is used to direct a vector at this ad-

dress to the respective interrupt service routine, where we will perform the relevant action that is needed upon timer-0 overflowing, i.e., when the number in its TCNT0 register (timer counter 0) crosses '255' (decimal). So in the software program for real-time clock, we initialise TIMSK to '01.'

Software program for real-time clock (8535clk.asm). The 8535clk.asm program with suitable explanations and comments is given at the end of this article. The programmed IC can be fixed to the RTC circuit board to show real-time clock on the LCD.

8535CLK.ASM

```
; Things to learn here:
; - Timer in interrupt mode
; - Interrupts, Interrupt-vector
; - BCD-arithmetic
.LIST
.NOLIST
.INCLUDE "m8535def.inc"
;device =ATMEga8535
.LIST
; Universal register definition
.DEF        mp = R16
.DEF    result=R18
; Counter for timer timeouts, MSB timer driven by software
.DEF        z1 = R0
; Working register for the Interrupt-Service-Routine
; Note that any registers used during an interrupt, including the status-register with all the flags must either be reserved for that purpose
; or they have to be reset to their initial
; value at the end of the service routine! Otherwise
; nearly unpredictable results will occur.
.DEF        ri = R1
; Register for counting the seconds as packed BCD
.DEF        sec = R2
.DEF    min = R3
.DEF    hour=R4
.DEF   count=R5
.DEF count1= R6
.CSEG
.ORG $0000
; Reset- and Interrupt-vectors
      rjmp Start ; Reset-vector
.org ovf0Addr
      rjmp tc0i
; Reset-vector to address 0000
.org $30
start:  rjmp                main
; Be sure that the jump
; to the interrupt service routine tc0i is exactly at the adress "ovf0",otherwise the interrupt fails.
; The following  sequence takes place : If the timer overflows
; (transition from 255 to 0) the program run is interrupted, the current address in the program counter
; is pushed to the stack, the instruction at address ovf0
; is executed (the  jump instruction). After finishing execution of the interrupt service routine
; the program counter value is restored from the
; stack and program execution proceeds from that point.
tc0i:
       in  ri,SREG ; save the content of the flag register
   inc z1 ; increment the software counter
      out    SREG,ri ; restore the initial value of the flag register
                reti ; Return from interrupt
.org $50
; The main program starts here
main:
  ldi mp,LOW(RAMEND) ;Initiate Stackpointer
  out SPL,mp ; for the use by interrupts and subroutines
  ldi mp,HIGH(RAMEND)
  out SPH,mp       ; Port b (pin 1-8) output-port, port c all output except bit 0,1
  ldi mp,0xFF ; all bits are output
  out DDRb,mp ; to data direction register
  ldi mp,0xFc  ;bits 0,1 input;  pin22 for minutes set; pin23 hour set;
  out  DDRC,mp
  LDI MP,03
  OUT PORTC,MP   ;pull up port C bits 0- 1
                internally itself
  rcall init_lcd          ;initialise LCD module
  ldi R16,$80
  rcall cmd
; Software-Counter-Register reset to zero
  ldi mp,0 ; z1 cannot be set to a constant value, so we set mp
  mov z1,mp ; to zero and copy that to R0=z1
  mov sec,mp ; and set the seconds to zero
  mov min,mp    ;and minutes also
  ldi  mp,$12
  mov hour,mp
; Prescaler of the counter/timer = 64, that is 1 MHz/64 = 15625 Hz = $3D09
  ldi mp,0x03 ;Initiate Timer/Counter 0 Prescaler as /64
  out TCCR0,mp ; to Timer 0 Control Register
; enable interrupts for timer 0
  ldi mp,$01 ; set Bit 0 but for 8515 this was bit  1!
  out TIMSK,mp ; in the Timer Interupt Mask Register
                ; enable all interrupts generally
   sei ; enable all interrupts by setting the flag in the status-register
; The 8-bit counter overflows from time to time and the interrupt service
; routine increments a counter in a register. The main program loop reads this
; counter register and waits until it reaches 3D hex. Then the timer is read until
; it reaches 09 (one second = 15625 (dec)= 3D09(hex) timer pulses). The timer
; and the register are then set to zero and one second is incremented. The seconds
; are handled as packed BCD-digits (one digit = four bits, one byte represents
; two digits). The seconds are refreshed. The seconds
; are displayed on the LCD module, as well.
 ldi mp,0x31     ;just show a "1" to begin with
        rcall lcdwr
 loop:
  ldi mp,$3D ; compare value for register counter
 loop1: rcall lookupdate ; check if user adjusts time-minutes
  rcall lookupdatehr ; check if user adjusts time-hours
  cp z1,mp ; compare with the register
 brlt loop1 ; z1 < mp, wait
 loop2:
    in mp,TCNT0 ; read LSB in the hardware counter
  cpi mp,$09 ; compare with the target value
  brlt loop2 ; TCNT0 < 09, wait
  ldi mp,0 ; set register zero and ...
  out TCNT0,mp ; reset hardware-counter LSB
  mov z1,mp ; and software-counter MSB
  rcall IncSec ; call the subroutine to increment the seconds
  rcall Display ; call subroutine to display the seconds
  rjmp loop ; once again the same
; subroutine increment second counter
; in BCD-arithmetic! Lower nibble = Bit 0..3, upper nibble = 4..7
 IncSec:
  sec ; Set Carry-Flag for adding an additional one to the seconds
  ldi mp,6 ; povoke overflow of the lower nibble by adding 6
  adc sec,mp ; add 6 + 1 (Carry)
  brhs Chk60 ; if overflow of the lower nibble occurred go to 60 check
  sub sec,mp ; subtract the additional 6 as no overflow occurred
 Chk60:
  ldi mp,$60 ; 60 seconds already reached?
  cp sec,mp
  brlt SecRet ; jump if less than 60
  ldi mp,256-$60 ; Load mp to add sec to zero
  add sec,mp ; Add mp to reset sec to zero
  rcall incmin
 SecRet:
  ret ; return to the main program loop
 incmin:  ;subroutine for minutes incrementing
  sec ; Setze Carry-Flag for adding an additional one to the seconds
  ldi mp,6 ; provoke overflow of the lower nibble by adding 6
  adc min,mp ; add 6 + 1 (Carry)
  brhs Chk60_m ; if overflow of the lower nibble occurred go to 60 check
  sub min,mp ; subtract the additional 6 as no overflow occurred
 Chk60_m:
  ldi   mp,$60 ; 60 minutes already reached?
  cp min,mp
  brlt minRet ; jump if less than 60
  ldi mp,256-$60 ; Load mp to add min to zero
  add min,mp ; Add mp to reset min to zero
      RCALL INCHOUR
 minRet:
  ret ; return to the main program loop
 INCHOUR:
  sec
  ldi mp,6
  adc hour,mp
  brhs chk12hour
  sub   hour,mp
  chk12hour:
  ldi mp,$13
  cp hour,mp
  brlt houret
  ldi mp,256-$12
  add hour,mp
  houret:      ret
  lookupdate:
  k2:    sbic pinc,0
  ret  ;if key is not closed, return
        ;if closed, wait for key-debounce and check again
  rcall delay1
  Inc count1
  ldi mp,80
  cp count1,mp
  brlt dd
 RCALL incmin
 Ldi mp,0
 Mov count1,mp
 dd:  rcall display
       ret
   lookupdatehr:
 k3:    sbic pinc,1
 ret  ;if key is not closed, return
      ;if closed, wait for key-debounce and check again
  rcall delay1
  inc count
  ldi mp,80
  cp count, mp
  brlt  dd
      RCALL inchour
  Ldi mp,0
  mov count, mp
  rjmp dd
; subroutine for displaying the time on the LCD
 Display: push r16
  ldi r16,$80
  rcall cmd
  pop r16
  mov r16,hour
  andi r16,0xf0
  ror r16
  ror r16
  ror r16
  ror r16
  ori r16,0x30
```

```
rcall lcdwr
mov r16,hour
andi r16,0b00001111
ori r16,0x30
rcall lcdwr
ldi r16,$3A
rcall lcdwr
mov r16,min
andi r16,0xf0
ror r16
ror r16
ror r16
ror r16
ori r16,0x30
rcall lcdwr
mov r16,min
andi r16,0b00001111
ori r16,0x30
rcall lcdwr
ldi r16,$3A ; For : display
rcall lcdwr
mov r16,sec
andi r16,0xf0
ror r16
ror r16
ror r16
ror r16
ori r16,0x30
rcall lcdwr
```

```
mov r16,sec
andi r16,0b00001111
ori r16,0x30
rcall lcdwr
ldi r16,32
rcall lcdwr
ret
 cmd: cbi portc,2 ; command entry to LCD subroutine
cbi portc,3
cbi portc,4
out portb,R16
sbi portc,4
nop
nop
nop
nop
nop
cbi portc,4
rcall delay1
ret
lcdwr: cbi portc,2 ; write to LCD routine
cbi portc,3
cbi portc,4
sbi portc,2
out portb,R16
sbi portc,4
nop
nop
nop
```

```
nop
nop
cbi portc,4
rcall delay1
ret
init_lcd: ; initialise LCD module
ldi R16,$38
rcall cmd
rcall delay1
rcall delay1
ldi R16,$0e
rcall cmd
rcall delay1
ldi R16,6
rcall cmd
ldi r16,1
rcall cmd
rcall delay1
ret
delay10:
ldi R16,$f0
del_lp: inc R16
brne del_lp
ret
delay1:
clr result
loop22: inc result
brne loop22
ret
```

Let's now examine the use of inbuilt functions of AVR ATmega8535 (such as output compare, ADC and UART) for various applications.

PWM operation of ATmega8535

When the AVR is configured for pulse-width modulated (PWM) operation, the PWM outputs become available at output-compare pins 18 (OC1A) and 19 (OC1B) of ATmega8535. PWM, in conjunction with an analogue filter, can be used to generate analogue output signals and thus it serves as a digital-to-analogue converter.

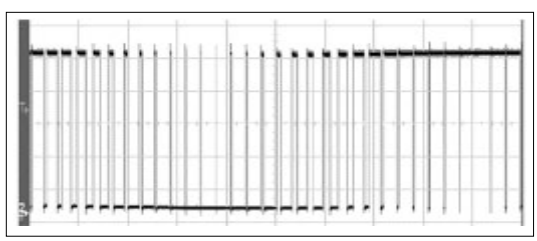


Fig. 15: Variation of pulse width (constant period) with time of a typical PWM wave

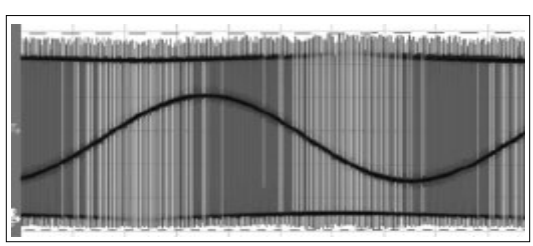


Fig. 16: View of filtered low-frequency sine wave and unfiltered PWM output on an oscilloscope

Principle of pulse-width modulation. To generate different analogue levels, the duty cycle and thereby the pulse-width of the digital signal (base frequency) is changed. If a high analogue level is needed, the pulse width is increased and vice versa (see Figs 15 and 16).

A digital pulse train with a constant period (fixed base frequency) is used as the basis. The base frequency, which can be programmed suitably, should be much higher than the frequency of the output analogue signal obtained after filtering out the base frequency component. For example, to generate a sinewave signal of low frequency (say, 10 Hz, as used for drives or controls), the base frequency of rectangular pulses (with varying duty cycle) may be of the order of 1 kHz or more.

Pulse generation method. The scheme for pulse generation is as follows: Timer/counter 1 is used to count clock ticks. If 8-bit PWM is selected, after the timer counts up to '255,' its count is decremented with each clock tick. Thus, the number increases up to '255' and then decreases, resembling a triangular pattern.

When the number stored in the output-compare register (OCR) matches the loaded count value, pin 19 (output-compare action pin) becomes high or low, as programmed. For example, if the OCR is loaded with a value of '100,' the logic state of OCR pin will be:

Count value	**OCR pin**
0 to 100	Low logic
100 to 255	High logic
255 to 100	High logic
100 to 0	Low logic

Thus, for the total time taken to count 255×2=510 clock ticks, the output pin (pin 19) will be high for (2×155)/(2×256) or 60.5 per cent of the total triangular wave time of one PWM pulse (or the PWM pulse will have a duty cycle of 60.5 per cent). Thus, effectively the voltage transmitted in this period is 60.5 per cent of the maximum, because the pulse is high only for this period of time.

The following program (AVRSINE.ASM**)** will generate a 1Hz sine wave (after filtering) on pin 19 using PWM:

```
AVRSINE.ASM
;———————————————————
; File: avrsine.asm
; Description: Example of how to use the fast PWM
; of the Avr to generate "sine-wave" signal. The PWM
; output requires filtering to shape the sine wave
; form.
;———————————————————
.include "m8535def.inc"
```

```
rjmp init

;Interrupt vector table

.org OVF1Addr ;OC1Aaddr
; Interrupt vector for timer1 output compare match A
rjmp TOF_isr

;Main code
init:
            ldi  R16,low(RAMEND)
; Load low byte address of end
of RAM into register R16
            out SPL,R16
; Initialize stack pointer to end of internal RAM
            ldi  R16,high(RAMEND)
; Load high byte address of end of RAM into
register R16
            out  SPH, R16
; Initialize high byte of stack pointer to end of
internal RAM
                    ldi r16,$ff
                    out ddrb,r16
                    ldi r16,$55
                    out portb,r16

            ldi r16, (1<<PD5) ; Set Pd5 as output
            out DDRd, r16     ;since that is the PWM
                                output pin 19

;SELECT CLOCK SOURCE VIA TCCR1B
       LDI R16,$81


 ;8 BIT PWM NON-INV.
; Set PWM mode: toggle OC1A on compare
            out TCCR1A, r16
; Enable PWM

            ldi r16, 0xFF

; Set PWM top value: OCR1C = 0xFF
            out OCR1AL, r16
             LDI R16,0
             OUT OCR1AH, R16

            ; Enable Timer/Set PWM clock prescaler
            LDI  R16,02
            OUT TCCR1B,R16     ;ck/8 as pwm clock
            (1MHz/8 = 125 kHz)

            ldi r16, (1<<TOIE1)      ; Enable Timer1
Ovrflow interrupt
            out TIMSK, r16
            clr r17
            clr r18
            sei
; Enable global interrupts
idle:
            ldi r16, (1<<SE)         ; Enable sleep
            out MCUCR, r16
            sleep
            rjmp idle

TOF_isr:
            ldi ZH, high(sine_table*2)
; Set up Z to point to the beginning of
sine_table
            ldi ZL, low(sine_table*2)
            add ZL, r17
; Offset Z by r18:r17
            adc ZH, r18
            lpm
            ; Load sine_table[Z] into OCR1A
            out OCR1AL, r0
            inc r17

            reti

sine_table:        ;   256 values
.db 128,131,134,137,140,144,147,150,153,156,159,162,
165,168,171,174
.db 177,179,182,185,188,191,193,196,199,201,204,206,
209,211,213,216
.db 218,220,222,224,226,228,230,232,234,235,237,239,
240,241,243,244
.db 245,246,248,249,250,250,251,252,253,253,254,254,
254,254,254,254
.db 254,254,254,254,254,254,254,253,253,252,251,250,
250,249,248,246
.db 245,244,243,241,240,239,237,235,234,232,230,228,
226,224,222,220
.db 218,216,213,211,209,206,204,201,199,196,193,191,
188,185,182,179
.db 177,174,171,168,165,162,159,156,153,150,147,144,
140,137,134,131
.db 128,125,122,119,116,112,109,106,103,100,97,94,91,
88,85,82
```

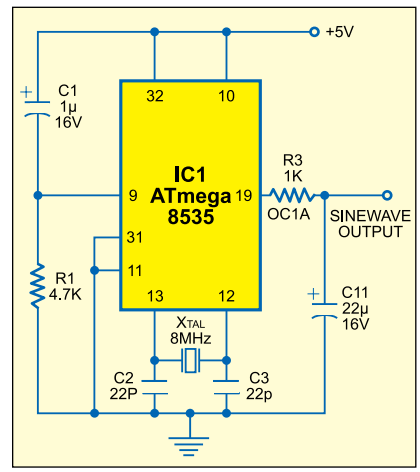


Fig. 17: Circuit for PWM-based sinewave generation

Fig. 18: ADCH and ADCL registers

```
.db 79,77,74,71,68,65,63,60,57,55,52,50,47,45,43,40
.db 38,36,34,32,30,28,26,24,22,21,19,17,16,15,13,12
.db 11,10,8,7,6,6,5,4,3,3,2,2,2,1,1,1
.db 1,1,1,1,2,2,2,3,3,4,5,6,6,7,8,10
.db 11,12,13,15,16,17,19,21,22,24,26,28,30,32,34,36
.db 38,40,43,45,47,50,52,55,57,60,63,65,68,71,74,77
.db 79,82,85,88,91,94,97,100,103,106,109,112,116,119,
122,125
```

For observation of the sine wave on an oscilloscope, use a low-pass filter comprising a 1-kilo-ohm resistor (series element) and a 1µF capacitor (shunt element). However, with an analogue multimeter, the sine wave can be directly observed at pin 19.

In the AVRSINE.ASM program, for each of the triangle wave periods, we read a table of sine values (multiplied by '256') and load these values one by one into the OCR. Since the values vary in a sinusoidal pattern, the pulses that come out are also pulse-width modulated as per these values (see the oscilloscope pattern shown in Fig. 16).

To do the table look-up (as given in the example program 'LCD Table. ASM' of Part 1), the LPM instruction is used. The Z register is used as an indirect indexed register. As stated earlier, the LPM instruction fetches from the table one byte into r_0. The actual loading of the OCR value is done by the instruction:

```
out OCR1AL, r0
```

where OCR1AL refers to pin 19. (OCR1BL refers to pin 18, which is not used here.)

The program contains suitable comments for easy understanding. The table in the program has 256 elements (corresponding to the samples in one complete sinewave period), while each sample period = pulse period (high and low parts) = 510 clock ticks. Thus 256 (samples) × 510 (clock ticks) = 130,560 clock ticks will produce one sinewave cycle. Thus for producing exactly 1Hz frequency, the base frequency should be 130.56 kHz (the nearest value of 125 kHz has been used here).

The circuit for realising the PWM-based sinewave generator is shown in Fig. 17.

The AVRSINE.ASM file and the assembled .HEX file are given in the CD. Using the AT-PROG programmer, load the program into an ATmega8535. Then fix it in a breadboard and make connections as per Fig. 17. Connect the circuit to 5V power supply and observe approximately 1Hz sine wave at pin 19 using an analogue multimeter. The needle on the multimeter will move with the sine wave as a pendulum.

PARTS LIST

Parts list for Figs 17 and 21

Semiconductors:

IC1	-	ATmega8535 microcontroller
IC2	-	LM35 temperature sensor
IC3	-	Max 232, RS-232 level converter

Resistors (all ¼-watt, ±5% carbon):

R1	-	4.7-kilo-ohm
R2	-	100-ohm
R3	-	1-kilo-ohm
VR1, VR3	-	10-kilo-ohm preset
VR2	-	10-kilo-ohm trim potentiometer

Capacitors:

C1, C6	-	1µF, 10V electrolytic
C2, C3	-	22pF ceramic disk
C4, C5, C7-C10	-	10µF, 16V electrolytic
C11	-	22µF, 16V electrolytic

Miscellaneous:

X_{TAL}	-	8MHz crystal
L1	-	100µH inductors
	-	16x1-character Hitachi make LCD or 16×2-character LCD
	-	9-pin female D-connector

Parts list for power supply

Semiconductors:

IC4	-	7805 regulator
D1-D4	-	1N4007 rectifier diode

Capacitors:

C12	-	4700µF, 16V electrolytic
C13	-	0.1µF ceramic disk

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fig. 19: ADMUX register bits

Bit	7	6	5	4	3	2	1	0	
	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fig. 20: ADCSRA register bits

Using the ADC

The inbuilt analogue-to-digital converter (ADC) of ATmega8535 is an 8-channel device with 10-bit resolution and maximum conversion time of 65 µs. The reference voltage for the ADC is connected across pins 32 (positive) and 31 (ground). The 5V Vcc supply (either directly or through a potmeter) can be used as reference voltage, but a capacitor at pin 32 is to be used for decoupling.

To access the ADC, you need to select the ADC channel; while the use of ADC interrupt is left to the discretion of the programmer. The ADC is read after conversion of a sample via the ADCH and ADCL registers (8 bits from the ADCL register and only two bits from the ADCH register) as shown in Fig. 18.

ADMUX and ADCSRA are the other registers used in conjunction with the ADC. Functions of various bits of these registers are explained below.

ADMUX register. The ADMUX register bits are shown in Fig. 19.

Bits 4 through 0 of ADMUX select the ADC channels for single-ended or differential operation including channels with gain. (For full selection details, see Table 85 of the ATmega8535(L) datasheet.)

Bit 5 (ADLAR, or AD left adjust result) affects selection of results in ADCH and ADCL registers. If this bit is made '0,' the ADCL contains the least eight bits and the ADCH contains the remaining two high-order bits in its D1:D0 bit positions. When the ADLAR bit is set to '1,' the ADCH contains the most significant eight bits, while the ADCL contains the least two significant bits in bit positions 7 and 6.

Bits 6 and 7 (REFS0 and REFS1) are reference-selection bits. With bit 7 as '0' and bit 6 as '1,' the external reference voltage is applied to pin A_{ref} (32).

We write E0 (1110 0000b) to ADMUX register in the ADC_LCD.ASM program. That means we choose channel-0 (pin 40) for the signal input, ADCH to give us the most significant eight bits and external 5V reference at pin 32 for analogue-to-digital conversion.

ADCSRA register. This is the control-and-status register for the ADC. Its bit positions are shown in Fig. 20.

The bits of the ADCSRA stand for the following signals: ADC enable (bit 7), ADC start (bit 6), ADC auto-trigger enable for free-running (bit 5), ADC interrupt flag set on completion of conversion (bit 4), ADC interrupt enable when set (bit 3) and ADC prescaler for speed (bits 0, 1 and 2). Bits 0, 1 and 2 determine the division factor between the clock frequency and the input clock to the ADC. The division factor can be selected from '2' to '128' as per Table 86 of the datasheet.

Program for displaying the ADC output on the LCD

The following program (ADC_LCD. ASM) takes the ADC data, converts the 10-bit data into five decimal digits and then shows it continuously on the LCD screen:

```
ADC_LCD.ASM
; ************************************************
; *This program uses channel -0 ADC of ATmega8535
; It reads the ADC and outputs the five-digit
; number on LCD.
; Program authored by  Prof. K. Padmanabhan
; ************************************************
.NOLIST
.INCLUDE "m8535def.inc"
 ;device =ATMega8535
.LIST
.EQU xyz = 12345
.EQU   fq=1000000; clock freq. of
internal oscillator
.EQU   baud=9600; Baudrate of SIO comm.
.EQU   bddiv=(fq/(16*baud))-1; Baudrate
divider
.DEF rmpr = R16
.DEF temp = R14
.DEF result=R12
.DEF mpr =R16
.CSEG
.ORG $0000
; Reset- and Interrupt-vectors
rjmp Start ; Reset-vector
.org OVF0Addr  ; timer-0 overflow
interrupt vector
address
rjmp timer0prg
timer0prg:  ;here take ADC sample
at every 64 µs
ldi r16,$cc
      out portc,r16
      push r16
      in r16,SREG
      PUSH R16
here2:in r16,adcsra
      andi r16,0b01000000
      brne here2  ;value got
in  r16,adcl
      in r17,adch
      rcall lcddisp
      POP R16
      out SREG,R16
      POP R16 ;restart adc
      ldi r16,0b11000101 ;prescale /32
(1x32=32 µs)
;adc enable,adc start,adc
freerun,adcflag,adcno int,
adcprescale/32
      out adcsra,r16
      RETI ;End of ISR
 cmd: cbi portc,2 ;command entry to
LCD routine
      cbi portc,3
      cbi portc,4
      out portb,r16
      sbi portc,4
      nop
      nop
      nop
      nop
      nop
      cbi portc,4
      rcall delay1
      ret
lcdwr:cbi portc,2; wrtite to LCD
routine
      cbi portc,3
      cbi portc,4
      sbi portc,2
      out portb,r16
      sbi portc,4
      nop
      nop
      nop
      nop
      nop
      cbi portc,4
      rcall delay1
      ret
busy: cbi portc,2
      sbi portc,3 ;read/write high?
      cbi portc,4 ;chip select low
      nop
      nop
      sbi portc,4      ;chip select high
busy1:lds R16,pinb
      rol R16
      brcs busy1
      cbi portc,4
      ret
init_lcd: ;initialise LCD
      ldi R16,$38
      rcall cmd
      rcall delay1
      rcall delay1
      ldi R16,$0e
      rcall cmd
      rcall delay1
      ldi R16,6
      rcall cmd
      ldi r16,1
      rcall cmd
      rcall delay1
      ret
delay1:clr result
loop22:ldi R16,$f0
loop2:inc R16
```

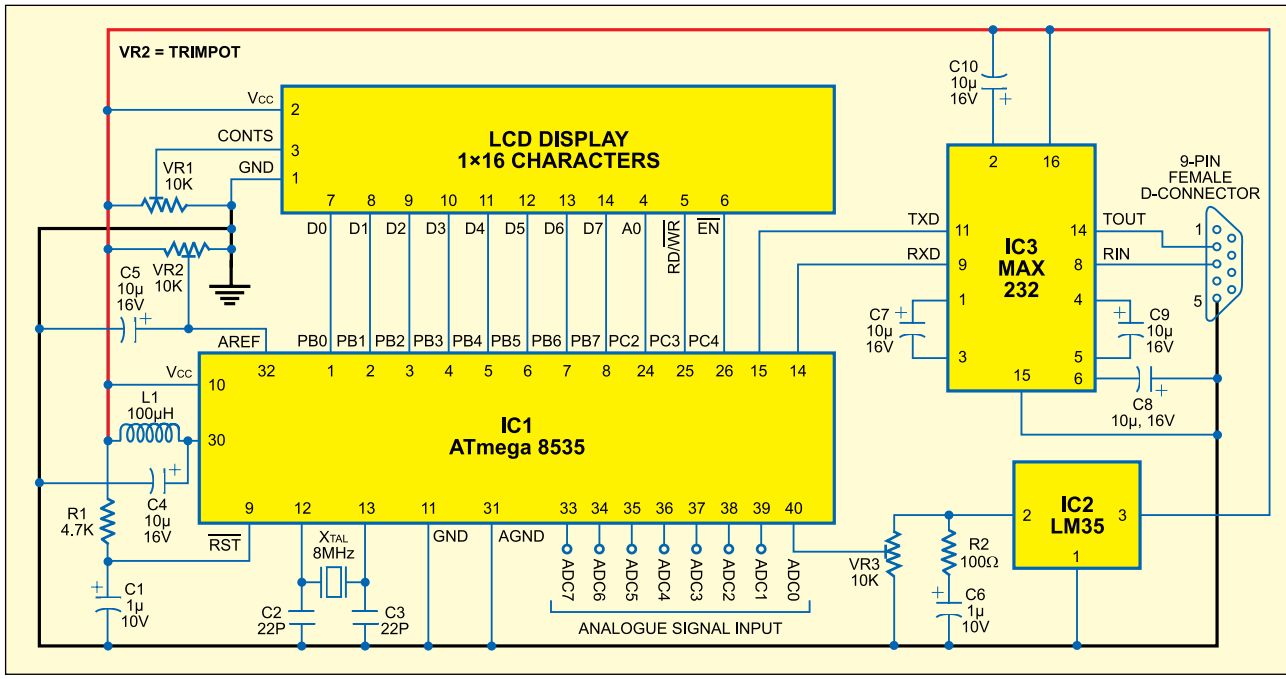


Fig. 21: Circuit for temperature display on either the LCD or the PC

```
      brne loop2
      inc result
      brne loop22
      ret
lcddisp: push r16
      ldi r16,128 ;cursor to left end
      rcall cmd
      pop r16
      rcall binbcd
      mov r16,r15
      andi r16,0x0f
      ori r16,0x30
      rcall lcdwr      ; 1
      mov r16,r14
      andi r16,0b11110000
      ror r16
      ror r16
      ror r16
      ror r16
      ori r16,0x30
      rcall lcdwr      ;2
      mov r16,r14
      andi r16,0x0f
      ori r16,0x30
      rcall lcdwr      ;3
      mov r16,r13
      andi r16,0b11110000
      ror r16
      ror    r16
      ror    r16
      ror      r16
      ori r16,0x30
      rcall lcdwr     ;4
      mov r16,r13
      andi r16,0x0f
      ori r16,0x30
      rcall  lcdwr    ;5
      ret
binbcd:
;* "bin2BCD16" - 16-bit Binary to BCD conversion
;* converts 16-bit number (fbinH:fbinL) to a 5-digit
;* packed BCD number represented by 3 bytes
(tBCD2:tBCD1:tBCD0).
;* MSD of 5-digit number is placed in lowermost
nibble of tBCD2.
;* Number of words       :25
;* Number of cycles      :751/768 (Min/Max)
;* Low registers used    :3 (tBCD0,tBCD1,tBCD2)
;* High registers used  :4(fbinL,fbinH,cnt16a,tmp16a)
;* Pointers used         :Z
Subroutine register variables
.equ        AtBCD0       =13
;address of tBCD0
.equ        AtBCD2       =15
;address of tBCD1
.def        tBCD0        =r13
;BCD value digits 1 and 0
```

```
.def        tBCD1        =r14
;BCD value digits 3 and 2
.def        tBCD2        =r15
;BCD value digit 4
.def        fbinL        =r16
;binary value Low byte
.def        fbinH        =r17
;binary value High byte
.def        cnt16a       =r18
;loop counter
.def        tmp16a       =r19
;temporary value
bin2BCD16:
ldi cnt16a,16            ;Init loop counter
            clr tBCD2
;clear result (3 bytes)
            clr tBCD1
            clr tBCD0
            clr ZH
;clear ZH (not needed for AT90Sxx0x)
bBCDx_1:lsl fbinL ;shift input value
            rol fbinH
;through all bytes
            rol tBCD0                ;
            rol tBCD1
            rol tBCD2
            dec cnt16a
;decrement loop counter
            brne bBCDx_2 ;if counter not zero
            ret                      ;   return

bBCDx_2:ldi  r30,AtBCD2+1
;Z points to result MSB + 1
bBCDx_3: ld tmp16a,-Z    ;get (Z) with
                                      pre-decrement
            subi  tmp16a,-$03        ;add 0x03
            sbrc  tmp16a,3           ;if bit 3 not
                                     clear
            st  Z,tmp16a;store back
            ld  tmp16a,Z;get (Z)
            subi  tmp16a,-$30        ;add 0x30
            sbrc  tmp16a,7           ;if bit 7 not
                                     clear
            st  Z,tmp16a;store back
            cpi  ZL,AtBCD0           ;done all three?
            brne  bBCDx_3            ;loop again if
                                     not
            rjmp  bBCDx_1

; Main program routine starts here
Start:ldi R16,low(RAMEND);Load low byte address
of end of RAM into register R16
            out  SPL,R16; Initialize stack
pointer to end of internal RAM
            ldi  R16,high(RAMEND);Load
high byte address of end of
    RAM into register R16
```

```
out  SPH, R16; Initialize high byte of stack
pointer to end of internal RAM
ldi rmpr,0b00000001;TIMER 0 INTERRUPT ENABLE
out TIMSK,rmpr
    ldi rmpr,05 ; So, we get once 1x10^6/1024=1000 Hz
out TCCR0,rmpr ;prescalar 1024 so that timer
interrupt occurs at 1KHz rate
ldi  r16,$c0      ;c0 for int. ref, e0 with adch
                  alone used.
      out admux,r16  ;channel 0 is selected
      ldi r16,0b11000101 ;prescale /32 (1x32=33 usec)
;adc enable,adc start,adc freerun,adcflag,adcno int,
adcprescale/32
      out adcsra,r16
      ldi r16,0
      out sfior,r16 ;write 0-0-0 to bits d7-d5 for
                  free run
adc
here1:in r16,adcsra
      andi r16,0b01000000
      breq here1  ;value got
      ldi R16,255
      out ddrb,R16   ; port b is all bits output
      out ddrc,R16   ; so is port c
      ldi r16,0
      out ddra,r16         ;port a input
init: sei ;enable global interrupt
LCD:  rcall init_lcd
      ldi R16,$80
      rcall cmd
here3:in r16,adcsra
      andi r16,0b01000000
      brne here3  ;value got
      in  r16,adcl
      in r17,adch
      rcall lcddisp
idle: ldi r16,(1<<SE)
      out mcucr,r16
      sleep
      rjmp idle
restrt:ldi r16,$80    ;point to first cursor
       rcall cmd  ; command to lcd to position cursor
       rcall delay1
       ldi r16,0b11000101 ;prescale /32 (4.43/32=138
usec)=7.2Khz
;adc enable,adc start,adc freerun,adcflag,adcno int,
adcprescale/32
       out adcsra,r16
here4: in r16,adcsra
       andi r16,0b01000000
       brne here4  ;value got
       in  r16,adcl
       in r17,adch
       sbi adcsra,6  ;restart adc
hh:    rcall lcddisp
       RJMP restrt ; Test of the serial interface
```

Note. The ADC_LCD.ASM program together with the .hex file, for directly programming into the chip, is provided in the EFY-CD.

Fig. 21 shows the circuit for viewing the analogue temperature (°C) output of an LM35 temperature sensor IC connected to ADC Ch.0 (pin 40) of the AVR on the LCD screen in 5-digit decimal format after analogue-to-digital conversion using the ATmega8535 chip with the ADC_LCD program. The same circuit with addition of MAX232 chip and ATmega8535 can be used for interfacing to a PC for viewing the temperature data on the PC screen. However, for that you have to program the AVR with the firmware as described in the succeeding paragraphs.

Using the UART in the ATmega8535

Serial communication between the microcontroller and a PC is essential for data transfer to the microcontroller and reading of its ADC output by the PC. The universal asynchronous receiver transmitter (UART) built into the microcontroller can be programmed to operate at certain baud rates.

The ADC_CH.ASM sample program given below is useful for UART applications:

```
ADC_CH.ASM
; ************************************************
; This program read one of ADC channels (0  to 7).
The Channel can be selected by sending Channel
number.
; ATmega8535 receives the Channel no. and outputs
the five digit ADC value
; on RS232 port for reading by a PC's XTALK
program or a VB project.
; Software features:  It is possible to read the ADC
value and also
; transmit to the PC for data logging.
; ************************************************
.NOLIST
.INCLUDE "m8535def.inc"
;device =ATMega8535
.LIST
.EQU xyz = 12345
; Constants for Sio properties
.EQU fq=1000000;  clock frequency of m8535 with
internal oscillator
.EQU baud=4800; Baudrate for SIO communication
.EQU bddiv=(fq/(16*baud))-1; Baudrate divider
.DEF rmpr = R16
.DEF temp = R14
.DEF result=R12
.DEF mpr =R16
.CSEG
.ORG $0000
rjmp Start ; Reset-vector
.org $000b
            rjmp USART_RXC
.org $0100
InitSio:
            LDI rmpr,bddiv ; Init baud generator
            OUT UBRRL,rmpr ; set divider in UART
baud rate register
            ldi rmpr, 0
            out ubrrh,rmpr
            LDI rmpr,(1<<Rxen)|(1<<Txen)|
(1<<RXCIE)
            out UCsRB,rmpr
            LDI ZL,0 ; Wait some time
            LDI ZH,0
InitSio1:
            SBIW ZL,1
            BRNE InitSio1
            ldi r16,(1<<Ursel)|(1<<USBS)|
(3<<UCSZ0)
            out ucsrc,r16
            RET
USART_RXC:
push r16
      in R16,udr
      andi R16,07
      mov r1,r16
      ori r16,$c0
      out admux,r16
      ldi r16,$43
      rcall tout      ; intimate new channel to host
      ldi r16,$48
      rcall tout
      mov r16,r1
      ori r16,$30
      rcall tout
      pop r16
      reti
 cmd: cbi portc,2
      cbi portc,3
      cbi portc,4
      out portb,r16
      sbi portc,4
      nop
      nop
      nop
      nop
      nop
      nop
      nop
      nop
      nop
      cbi portc,4
      rcall delay1
      rcall delay1
      rcall delay1
      ret
lcdwr:cbi portc,2
      cbi portc,3
      cbi portc,4
      sbi portc,2
      out portb,r16
      sbi portc,4
      nop
      nop
      nop
      nop
      nop
      nop
      nop
      nop
      nop
      cbi portc,4
      rcall delay1
      ret
init_lcd:
      ldi R16,$38
      rcall cmd
      rcall delay1
      rcall delay1
      ldi R16,$0e
      rcall cmd
      rcall delay1
      ldi R16,6
      rcall cmd
      ldi r16,1
      rcall cmd
      rcall delay2
ret
tout: sbis UcSRa,UDRE ;TX COMPLETE check
      RJMP tout
      OUT UDR,R16
      Ret
delay1:push r16
       clr result
loop22:ldi R16,$f0
loop2:inc R16
      brne loop2
      inc result
      brne loop22
      pop r16
      ret
delay2:push r16
       clr result
loop221:ldi R16,$f0
loop21:inc R16
       brne loop21
       inc result
       brne loop221
       pop r16
       ret
delay: clr result
ld:    inc result
       brne ld
       ret
lcddisp:push r16
       ldi r16,128 ;cursor to left end
       rcall cmd
       pop r16
       rcall delay1
       rcall delay1
       rcall binbcd
       mov r16,r15
       andi r16,0x0f
       ori r16,0x30
       rcall tout
       rcall lcdwr      ; 1
       mov r16,r14
       andi r16,0b11110000
       ror r16
       ror r16
       ror r16
       ror r16
       ori r16,0x30
       rcall tout
       rcall lcdwr      ;2
       mov r16,r14
       andi r16,0x0f
       ori r16,0x30
       rcall tout
       rcall lcdwr      ;3
       mov r16,r13
       andi r16,0b11110000
       ror r16
       ror    r16
       ror    r16
       ror      r16
      ori r16,0x30
      rcall tout
      rcall lcdwr     ;4
      mov r16,r13
      andi r16,0x0f
      ori r16,0x30
      rcall tout
      rcall  lcdwr    ;5
      ldi r16,$0a
      rcall tout
      ldi r16,$0d
      rcall tout
      ret
binbcd:
;* "bin2BCD16" - 16-bit Binary to BCD conversion
;* convert 16-bit number (fbinH:fbinL) to a 5-digit
;* packed BCD number represented by 3 bytes
(tBCD2:tBCD1:tBCD0).
;* MSD of 5-digit number is placed in the lowermost
nibble of tBCD2.
;* Number of words      :25
;* Number of cycles     :751/768 (Min/Max)
;* Low registers used   :3 (tBCD0,tBCD1,tBCD2)
;* High registers used  :4(fbinL,fbinH,cnt16a,tmp16a)
;* Pointers used        :Z

; Subroutine Register Variables
.equ        AtBCD0 =13          ;address of tBCD0
.equ        AtBCD2 =15          ;address of tBCD1
.def        tBCD0   =r13   ;BCD value digits 1 and 0
.def        tBCD1  =r14    ;BCD value digits 3 and 2
.def        tBCD2  =r15     ;BCD value digit 4
.def        fbinL =r16     ;binary value Low byte
.def        fbinH  =r17    ;binary value High byte
.def        cnt16a =r18    ;loop counter
.def        tmp16a =r19   ;temporary value

bin2BCD16:
            ldi  cnt16a,16 ;Init loop counter
            clr  tBCD2  ;clear result (3 bytes)
            clr  tBCD1
            clr  tBCD0
            clr  ZH     ;clear ZH (not
                        needed for AT90Sxx0x)
bBCDx_1:lsl  fbinL      ;shift input value
            rol  fbinH  ;through all bytes
            rol  tBCD0
            rol  tBCD1
            rol  tBCD2
            dec  cnt16a   ;decrement loop counter
            brne  bBCDx_2 ;if counter not zero
            ret  ;   return
bBCDx_2:ldi  r30,AtBCD2+1 ;Z points to
result MSB + 1
bBCDx_3:
            ld tmp16a,-Z ;get (Z) with pre-decrement
```

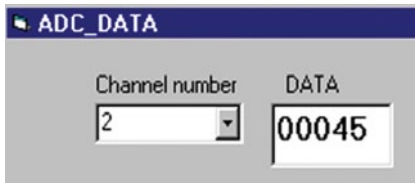


Fig. 22: Screenshot of ADC_CHSEL application

```
        subi  tmp16a,-$03       ;add 0x03
        sbrc  tmp16a,3          ;if bit 3 not
                                clear
        st  Z,tmp16a;store back

        ld  tmp16a,Z;get (Z)
        subi  tmp16a,-$30       ;add 0x30
        sbrc  tmp16a,7 ;if bit 7 not clear
        st  Z,tmp16a ;          store back
        cpi  ZL,AtBCD0 ;done all three?
        brne  bBCDx_3 ;loop again if not
        rjmp  bBCDx_1 ;End of the subroutine
                        section
```

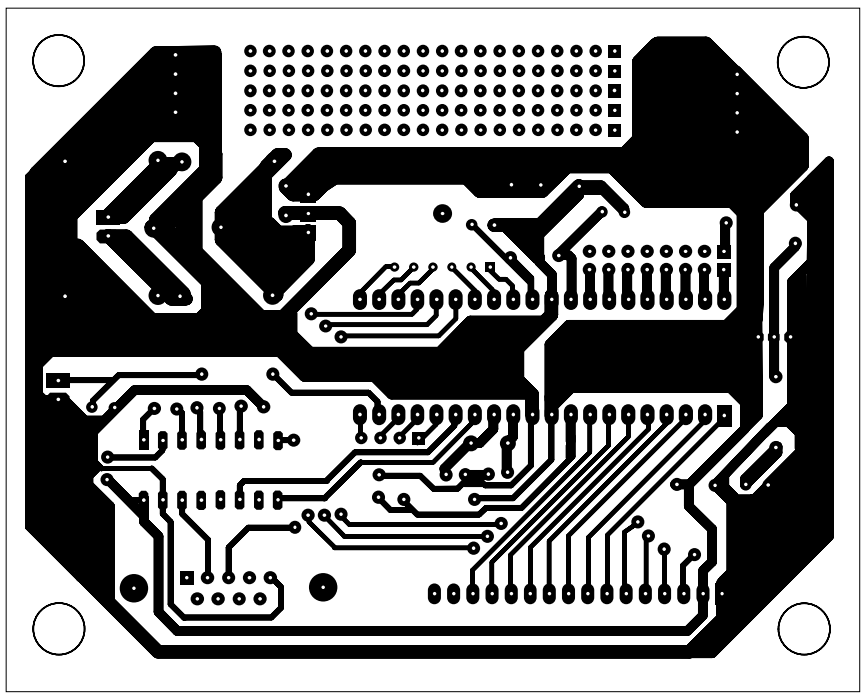


Fig. 23: Integrated actual-size PCB layout (including the 5V power supply circuit given in Part 1) for all the applications described in this 3-part article

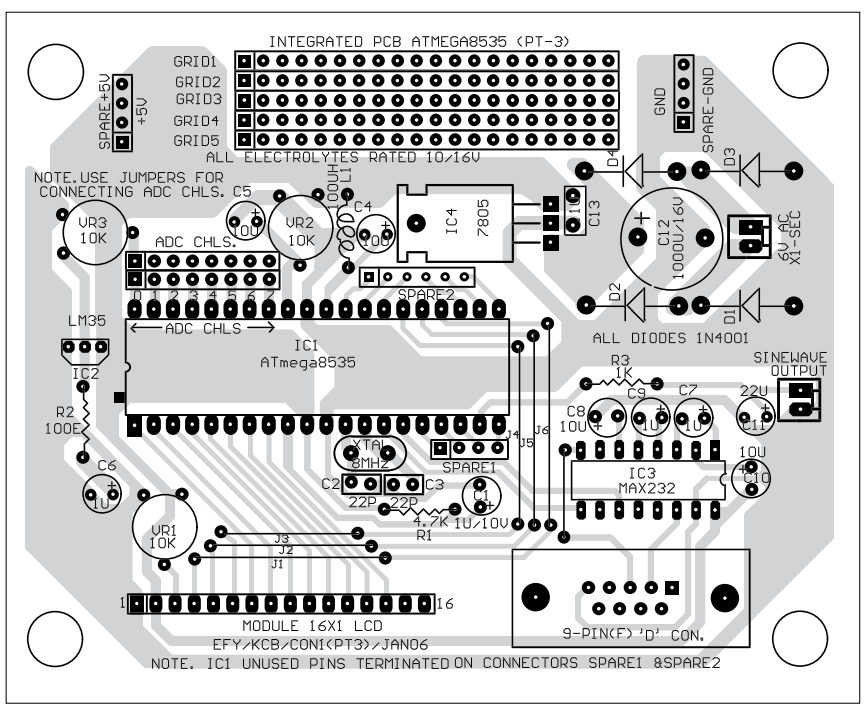


Fig. 24: Component layout for the PCB

```
; Main program routine starts here
Start:ldi R16,low(RAMEND);Load low byte address
                                of end of RAM into
                                 register R16
            out SPL,R16 ; Initialize stack pointer to
                        end of internal RAM
            ldi R16,high(RAMEND);Load high byte
                                    address of end
                                    of RAM into
                                    register R16
            out         SPH, R16 ;Initialize high
                                byte of stack pointer
                        to end of internal RAM
  ldi  r16,$c0 ; c0 for int. ref, e0 with adch alone
                        used.
      out admux,r16  ; channel 0 is selected
      ldi r16,0b11000101 ;prescale /32 (1x32=33 usec)
    ;adc enable,adc start,adc freerun,adcflag,adcno
            int,
adcprescale/32
      out adcsra,r16
      ldi r16,0
      out sfior,r16 ; write 0-0-0 to bits d7-d5 for
                        free
run adc
here1:in r16,adcsra
      andi r16,0b01000000
      breq here1  ;value got
      ldi R16,255
      out ddrb,R16 ; port b is all bits output
      out ddrc,R16 ; so is port c
      ldi r16,0
      out ddra,r16 ;port a input
init: rcall initsio
      sei  ;enable global interrupt
LCD:  rcall init_lcd
lcd1: ldi R16,$80
      rcall cmd
      rcall delay1
      rcall delay1
      rcall delay1
      rcall delay1
here3:in r16,adcsra
      andi r16,0b01000000
      brne here3  ;value got
      in  r16,adcl
      IN R17,adch
      push r16
       ldi r16,0b11000101 ;prescale /32 (1x32=32
usec)
;adc enable,adc start,adc freerun,adcflag,adcno int,
adcprescale/32
      out adcsra,r16
      pop r16
      rcall lcddisp
        rjmp lcd1
      in r16,udr
      andi r16,07
      mov r14,r16
       ori r16,$c0
      out admux,r16
      ldi r16,$43
      rcall tout      ; intimate new channel to host
      ldi r16,$48
      rcall tout
      mov r16,r14
      ori r16,$30
      rcall tout
      rjmp lcd1
```

The above program reads the ADC output data, whose decimal value is output to pin 15 (TX) of ATmega8535 at 4800 bauds in 8-bit ASCII data format. If a MAX232 is wired to pin 15, it can be directly connected to the receive pin of the RS-232 com port of a PC. Then, by using any terminal program (such as XTALK), it can be received by the PC.

The program (ADC_CH.ASM) may be tested as follows:

1. Wire ATmega8535 to the LCD and the serial port through a MAX232 IC as shown in Fig. 21.

2. Connect an analogue signal (e.g.,

a DC voltage in the 0-5V range tapped from a potmeter or the output of LM35 used in the preceding application) to ADC Ch. 0 (pin 40).

3. Program the adc_ch.asm file into the flash memory of ATmega8535 after compilation.

4. Place the IC on the breadboard and press reset.

5. Connect the RS-232 connector of the PC through a 3-wire cable to the MAX232 pins on the board. The TXD output from ATmega8535 should go to RXD pin of the PC's com port and the PC's TXD output should go to RXD pin of ATmega8535.

6. Run the XTALK program on the PC and set the baud rate as '4800,' data as '8 bits,' parity as 'none,' stop as '1,' and com as '1' or '2,' type 'go low,' then press 'Enter' key.

7. Observe the ADC data continuously on the screen.

The PC terminal program can be used to select one of the eight desired channels. For this, type any number from '0' to '7.' For example, to select channel-3 ADC, type '3.' Remember you need not press Enter key thereafter.

The data from the PC terminal is received by the USART_RX subroutine in the interrupt mode. The main program configures the received data to interrupt the processor. In the interrupt routine, the number sent is used to change (by altering the value of the bits in the ADMUX register of the chip) the ADC channel currently chosen. Thus all the following data will pertain to this channel only and the same will be informed to the PC terminal also by sending CH3 followed by data stream.

The XTALK terminal program is given only for testing purposes. The Visual Basic program (ADC_CHSEL) provided in the EFY-CD of this month does the same. It has two windows, one of which is a Combo box for selecting the channel and the other shows the 5-digit data continuously. Selection of the channel is possible via the Combo box (Fig. 22).

Application notes with programs

You may visit Atmel's Website 'www.atmel.com/dyn/products/app_notes.asp?family_id=607' for the following application notes.

1. AVR100: Accessing the EEPROM. This application note contains assembly routines for accessing the EEPROM for all AVR devices. It includes the code for reading and writing EEPROM addresses sequentially and at random addresses.

2. AVR223: Digital Filters with AVR. This document focuses on the use of the AVR hardware multiplier and general-purpose registers for accumulator functionality, scaling of coefficients when implementing algorithms on fixed-point architectures, actual implementation examples and possible ways to optimise/modify the implementations suggested.

3. AVR240: 4x4 Keypad-Wake Up on Keypress. This application note describes a simple interface to a 4×4 keypad designed for low-power battery operation.

Also there are application notes for interfacing the AVR to an IR detector much like the TV remote. Other topics of interest relating to the AVR are use of watchdog, power idle modes, SPI interfacing for communication, etc. Many have tried out the SPI interface for data communication, but it is found to be more complex compared to the RS-232 protocol. The RS-232 link for ADC data, which is described above, makes a really useful serial data-acquisition system.

An integrated actual-size PCB layout (including the 5V power supply circuit given in Part 1) for all the applications described in this 3-part article is shown in Fig. 23. The component layout for the same is shown in Fig. 24. Suitable pads (not shown in the component layout) have been provided for wiring the components. ●

Download source code: http://www.efymag.com/admin/issuepdf/Application%20AVR%20Part%20II.zip

SPEED CHECKER FOR HIGHWAYS

■ DIPANJAN BHATTACHARJEE

While driving on highways, motorists should not exceed the maximum speed limit permitted for their vehicle. However, accidents keep occurring due to speed violations since the drivers tend to ignore their speedometers.

This speed checker will come handy for the highway traffic police as it will not only provide a digital display in accordance with a vehicle's speed but also sound an alarm if the vehicle exceeds the permissible speed for the highway.

The system basically comprises two laser transmitter-LDR sensor pairs, which are installed on the highway 100 metres apart, with the transmitter and the LDR sensor of each pair on the opposite sides of the road. The installation of lasers and LDRs is shown in Fig. 1. The system displays the time taken by the vehicle in crossing this 100m distance from one pair to the other with a resolution of 0.01 second, from which the speed of the vehicle can be calculated as follows:

$$\text{Speed (kmph)} = \frac{\text{Distance}}{\text{Time}}$$

$$= \frac{0.1 \text{ km}}{(\text{Reading} \times 0.01)/3600}$$

or,

$$\text{Reading (on display)} = \frac{36000}{\text{Speed}}$$

As per the above equation, for a speed of 40 kmph the display will read 900 (or 9 seconds), and for a speed of 60 kmph the display will read 600 (or 6 seconds). Note that the LSB of the display equals 0.01 second and each succeeding digit is ten times the preceding digit. You can similarly calculate the other readings (or time).

Circuit description

Fig. 2 shows the circuit of the speed checker. It has been designed assuming that the maximum permissible speed for highways is either 40 kmph or 60 kmph as per the traffic rule.

The circuit is built around five NE555 timer ICs (IC1 through IC5), four CD4026 counter ICs (IC6 through IC9) and four 7-segment displays (DIS1 through DIS4). IC1 through IC3 function as monostables, with IC1 serving as count-start mono, IC2 as count-stop mono and IC3 as speed-limit detector mono, controlled by IC1 and IC2 outputs. Bistable set-reset IC4 is also controlled by the outputs of IC1 and IC2 and it (IC4), in turn, controls switching on/off of the 100Hz (period = 0.01 second) astable timer IC5.

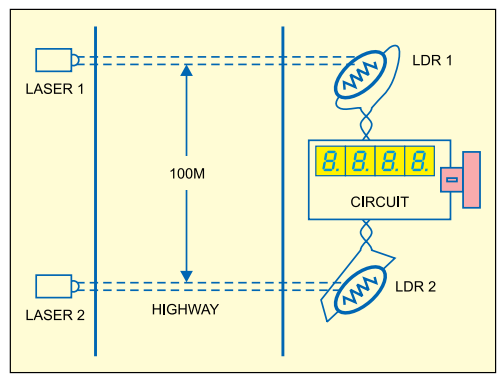


Fig. 1: Installation of lasers and LDRs on highway

The time period of timer NE555 (IC1) count-start monostable multivibrator is adjusted using preset VR1 or VR2 and capacitor C1. For 40kmph limit the time period is set for 9 seconds using preset VR1, while for 60kmph limit the time period is set for 6 seconds using preset VR2. Slide switch S1 is used to select the time period as per the speed limit (40 kmph and 60 kmph, respectively). The junction of LDR1 and resistor R1 is coupled to pin 2 of IC1.

Normally, light from the laser keeps falling on the LDR sensor continuously and thus the LDR offers a low resistance and pin 2 of IC1 is high. Whenever light falling on the LDR is interrupted by any vehicle, the LDR resistance goes high and hence pin 2 of IC1 goes low to trigger the monostable. As a result, output pin 3 goes high for the preset period (9 or 6 seconds) and LED1 glows to indicate it. Reset pin 4 is controlled by the output of NAND gate N3 at power-on or whenever reset switch S2 is pushed.

For IC2, the monostable is triggered in the same way as IC1 when the vehicle

PARTS LIST

Semiconductors:

IC1-IC5	- NE555 timer
IC6- IC9	- CD4026 decade counter/7-segment decoder
IC10	- CD4011 NAND gate
IC11	- 7812 12V regulator
D1, D2	- 1N4148 switching diode
D3-D6	- 1N4007 rectifier diode
LED1	- Green LED
LED2, LED3	- Red LED
DIS1-DIS4	- LTS543 common-cathode, 7-segment display

Resistors (all ¼-watt, ±5% carbon):

R1, R4	- 100-kilo-ohm
R2, R5, R6, R8, R10, R11, R14	- 10-kilo-ohm
R3, R7, R13, R16-R19	- 470-ohm
R9	- 470-kilo-ohm
R12, R15	- 1-kilo-ohm
VR1, VR2	- 100-kilo-ohm preset
VR3	- 20-kilo-ohm preset

Capacitors:

C1	- 100µF, 25V electrolytic
C2, C4, C6, C8, C11	- 0.01µF ceramic disk
C3, C13, C15	- 0.1µF ceramic disk
C5	- 10µF, 25V electrolytic
C7	- 0.47µF, 25V electrolytic
C9	- 0.2µF ceramic disk
C10	- 1µF, 25V electrolytic
C12	- 47µF, 25V electrolytic
C14	- 1000µF, 35V electrolytic

Miscellaneous:

X1	- 230V AC primary to 0-15V, 500mA secondary transformer
PZ1	- Piezobuzzer
LDR1, LDR2	- LDR
S1, S2	- Push-to-on switch
S3	- On/Off switch
	- Pointed laser light

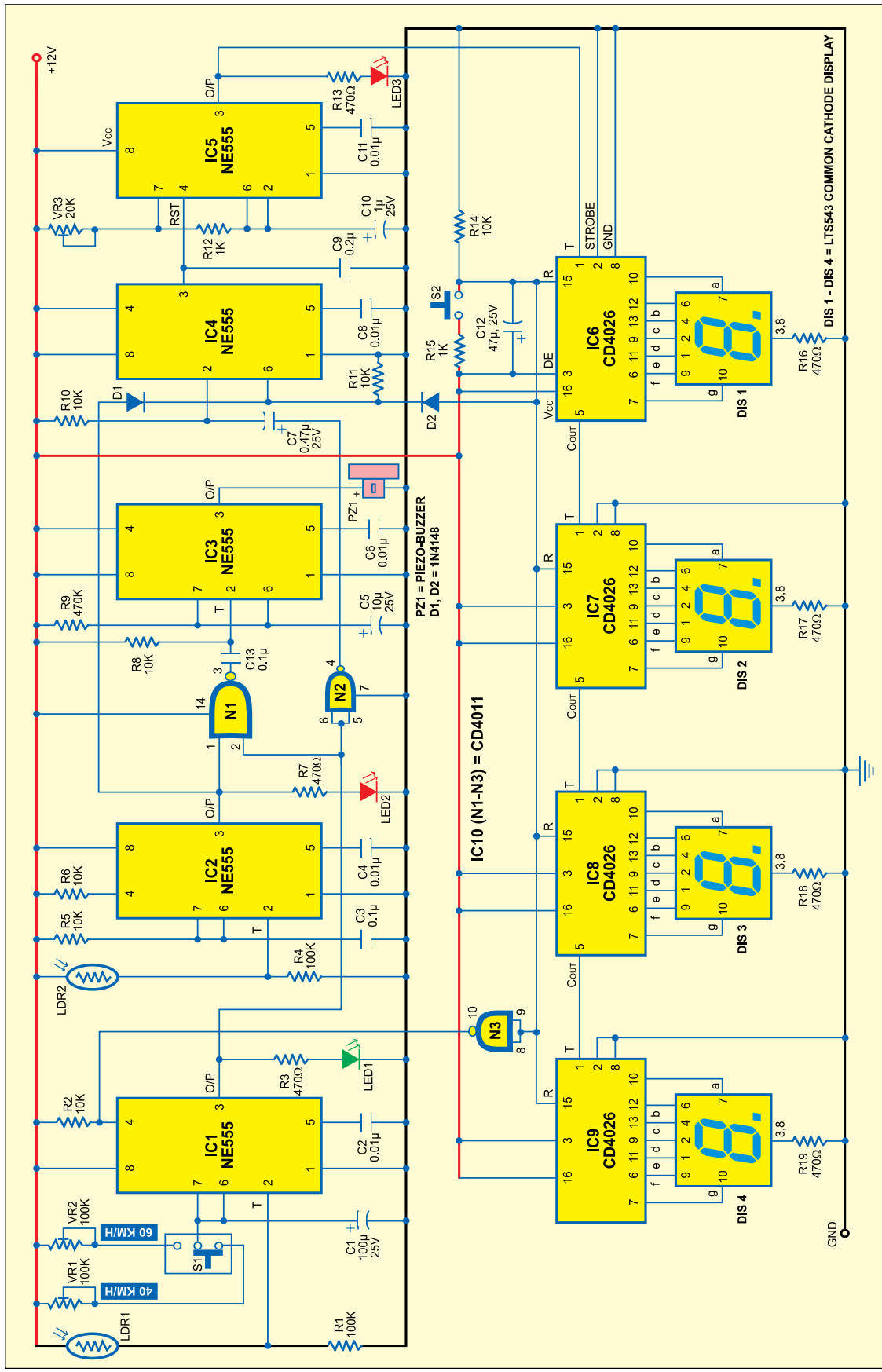


Fig. 2: Circuit of speed checker for highway

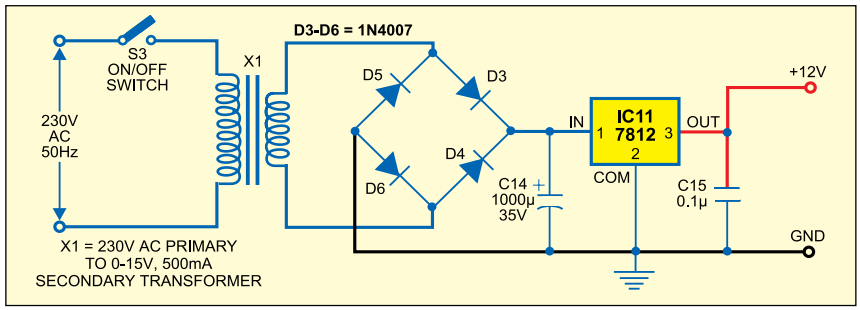


Fig. 3: Power supply

multivibrator whose time period is decided by preset VR3, resistor R12 and capacitor C10. Using preset VR1, the frequency of the astable multivibrator is set as 100 Hz. The output of IC5 is fed to clock pin 1 of decade counter/7-segment decoder IC6 CD4026.

IC CD4026 is a 5-stage Johnson decade counter and an output decoder that converts the Johnson code into a 7-segment decoded output for driving

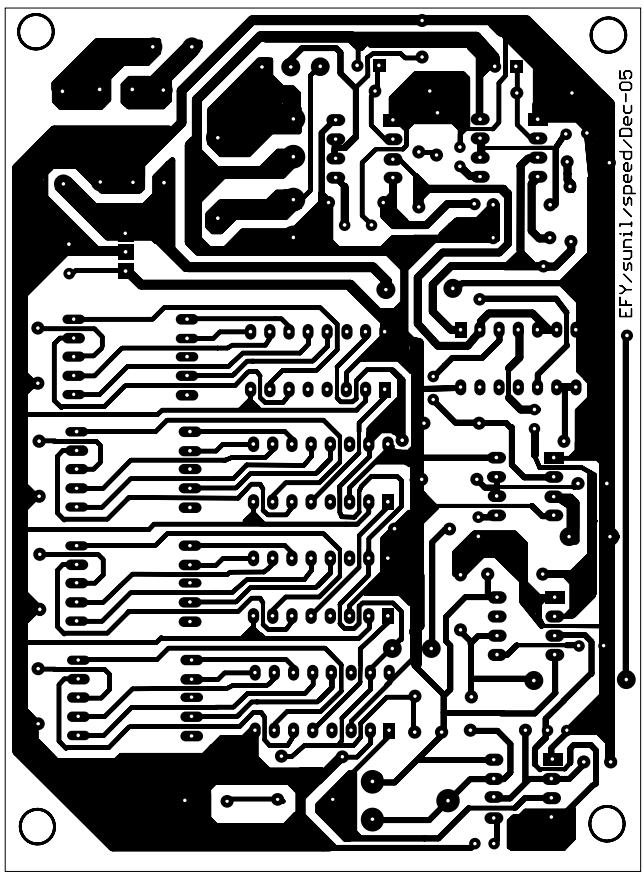


Fig. 4: Actual-size, single-side PCB layout for the speed checker

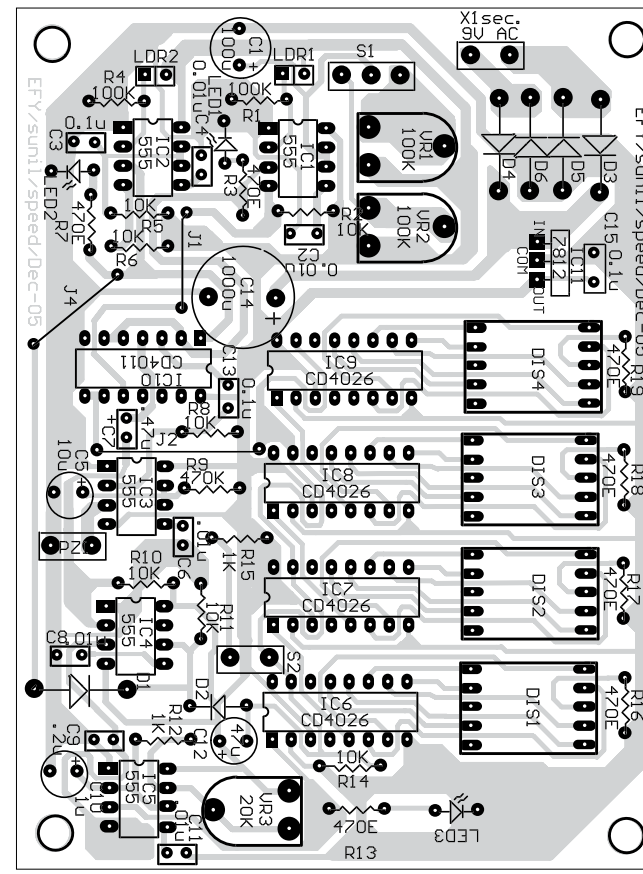


Fig. 5: Component layout for the PCB

intersects the laser beam incident on LDR2 to generate a small pulse for stopping the count and for use in the speed detection. LED2 glows for the duration for which pin 3 of IC2 is high.

The outputs of IC1 and IC2 are fed to input pins 2 and 1 of NAND gate N1, respectively. When the outputs of IC1 and IC2 go high simultaneously (meaning that the vehicle has crossed the preset speed limit), output pin 3 of gate N1 goes low to trigger monostable timer IC3. The output of IC3 is used for driving piezobuzzer PZ1, which alerts the operator of speed-limit violation. Resistor R9 and capacitor C5 decide the time period for which the piezobuzzer sounds.

The output of IC1 triggers the bistable (IC4) through gate N2 at the leading edge of the count-start pulse. When pin 2 of IC4 goes low, the high output at its pin 3 enables astable clock generator IC5. Since the count-stop pulse output of IC2 is connected to pin 6 of IC4 via diode D1, it resets clock generator IC5. IC5 can also be reset via diode D2 at power-on as well as when reset switch S2 is pressed.

IC5 is configured as an astable

IC CD4026 is a 5-stage Johnson decade counter and an output decoder that converts the Johnson code into a 7-segment decoded output for driving DIS1 display. The counter advances by one count at the positive clock signal transition.

The carry-out (Cout) signal from CD4026 provides one clock after every ten clock inputs to clock the succeeding decade counter in a multidecade counting chain. This is achieved by connecting pin 5 of each CD4026 to pin 1 of the next CD4026.

A high reset signal clears the decade counter to its zero count. Pressing switch S2 provides a reset signal to pin 15 of all CD4026 ICs and also IC1 and IC4. Capacitor C12 and resistor R14

generate the power-on-reset signal.

The seven decoded outputs 'a' through 'g' of CD4026s illuminate the proper segment of the 7-segment displays (DIS1 through DIS4) used for representing the decimal digits '0' through '9.' Resistors R16 through R19 limit the current across DIS1 through DIS4, respectively.

Fig. 3 shows the circuit of the power supply. The AC mains is stepped down by transformer X1 to deliver the secondary output of 15 volts, 500 mA. The transformer output is rectified by a bridge rectifier comprising diodes D3 through D6, filtered by capacitor C14 and regulated by IC11 to provide regulated 12V supply. Capacitor C15 bypasses any ripple in the regulated output. Switch S3 is used as the 'on'/'off' switch. In mobile application of the circuit, where mains 230V AC is not available, it is advisable to use an external 12V battery. For activating the lasers used in conjunction with LDR1 and LDR2, separate batteries may be used.

Construction and working

Assemble the circuit on a PCB. An actual-size, single-side PCB layout for the speed checker is shown in Fig. 4 and its component layout in Fig. 5.

Before operation, using a multimeter check whether the power supply output is correct. If yes, apply power supply to the circuit by flipping switch S3 to 'on.' In the circuit, use long wires for connecting the two LDRs, so that you can take them out of the PCB and install on one side of the highway, 100 metres apart. Install the two laser transmitters (such as laser torches) on the other side of the highway exactly opposite to the LDRs such that laser light falls directly on the LDRs. Reset the circuit by pressing switch S2, so the display shows '0000.' Using switch S1, select the speed limit (say, 60 kmph) for the highway. When any vehicle crosses the first laser light, LDR1 will trigger IC1. The output of IC1 goes high for the time set to cross 100 metres with the selected speed (60 kmph) and LED1 glows during for period. When the vehicle crosses the second laser light, the output of IC2 goes high and LED2 glows for this period.

Piezobuzzer PZ1 sounds an alarm if the vehicle crosses the distance between the laser set-ups at more than the selected speed (lesser period than preset period). The counter starts counting when the first laser beam is intercepted and stops when the second laser beam is intercepted. The time taken by the vehicle to cross both the laser beams is displayed on the 7-segment display. For 60kmph speed setting, with timer frequency set at 100 Hz, if the display count is less than '600,' it means that the vehicle has crossed the speed limit (and simultaneously the buzzer sounds). Reset the circuit for monitoring the speed of the next vehicle.

Note. This speed checker can check the speed of only one vehicle at a time. ●

SECTION B:
CIRCUIT IDEAS

AUDIO AMPLIFIER FOR PERSONAL STEREO

■ **M. VENKATESWARAN**

In the output stages of most broadcast receivers and some amplifiers, there is a limit up to which maximum power can be developed without distortion. In the widely accepted output circuit, two output transistors are connected in series between the positive and ground and biasing is adjusted so that each transistor gets half the supply voltage.

The circuit presented here is a simple audio amplifier for a personal stereo system. In this, supply voltage to each transistor can be enhanced to produce a larger output. The audio driver transformer drives the transistors adequately.

A 9V-0-9V, 300mA transformer has been used in the set-up. Out of the four diodes (D1 through D4), two are used for developing the positive voltage rail (+9V) and the other two are used for developing the negative voltage rail (–9V). In the pushpull amplifier, each transistor (T2 or T3) gets double the voltage when activated.

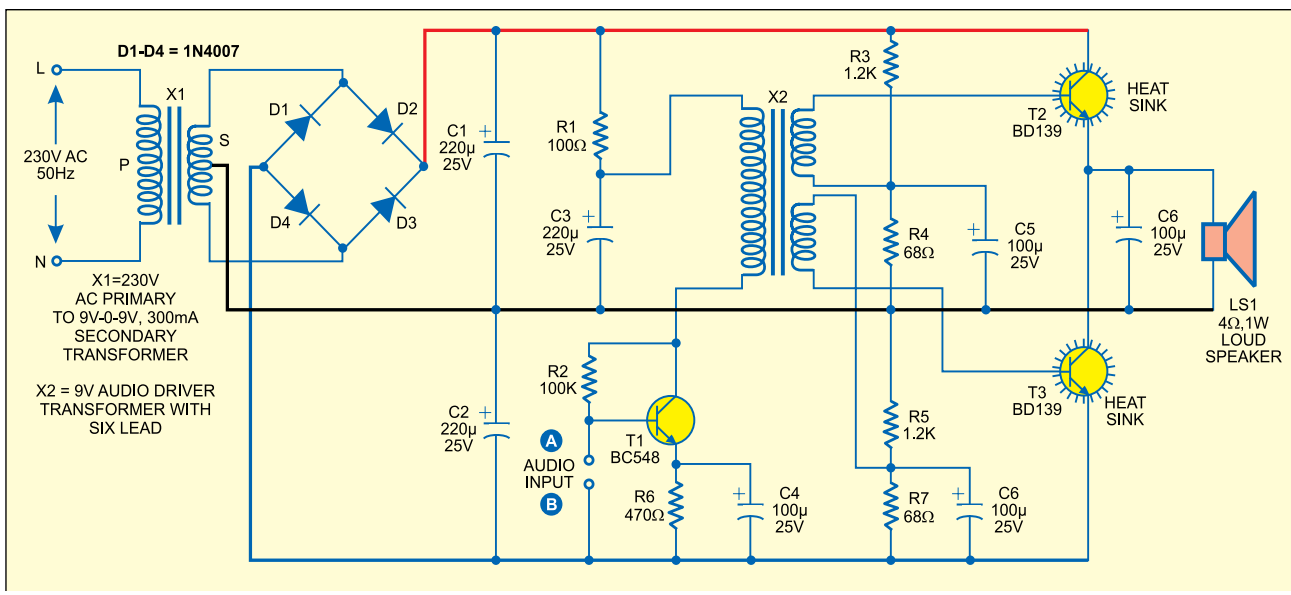
Connect the low audio signal from the stereo system at input terminals A and B of the audio amplifier and provide mains AC to activate the circuit. During the first half cycle of an AF cycle, transistor T2 conducts and the current flows from positive rail to ground rail (centre tap of transformer X1) via the loudspeaker coil (connected between the emitter of transistor T2 and ground) in one direction. While in the second half cycle, transistor T3 conducts and the current flows from ground rail to negative rail via the loudspeaker coil (connected between ground and the collector of transistor T3) in a direction opposite to the previous flow.

Transistors T2 and T3 of the pushpull audio amplifier should be matched correctly. If these transistors get heated, change the bleeding resistor pairs (R3 and R4 for transistor T2 and R5 and R7 for transistor T3) so that the acceptable output without overheating is obtained. You can also replace these transistors with another pair of suitable high-power transistors.

For driving transistors T2 and T3, a 9V audio driver transformer having six leads is used. It is readily available in the market and reasonably matches the output and input impedances of the preceeding and succeeding stages.

To test the quality of the audio output, connect the stereo's outputs to the respective terminals A and B. Now increase the volume level of the stereo slowly. If you get a high-level, high-quality sound across loudspeaker L1, the amplifier is working well. If the sound quality is not good, decrease the volume level until the audio amplifier gives good results.

Note that this audio amplifier works well for low-level audio signals. ●



INFRARED OBJECT COUNTER

■ **RAMBIR SINGH**

This infrared object counter can be installed at the entry gate to count the total number of people entering any venue. For example, it can be used at the railway stations or bus stands to count the people arriving per day or week.

The counter uses an infrared transmitter-receiver pair and a simple, low-cost calculator. It works even in the presence of normal light. The maximum detection range is about 10 metres. That means the transmitter and the receiver are to be installed (at the opposite pillars of the gate) not more than 10 metres apart. No focusing lens is required. If an 8-digit calculator is used the counter can count up to 99,999,999 easily, and if a 10-digit calculator is used the counter can count up to 9,999,999,999.

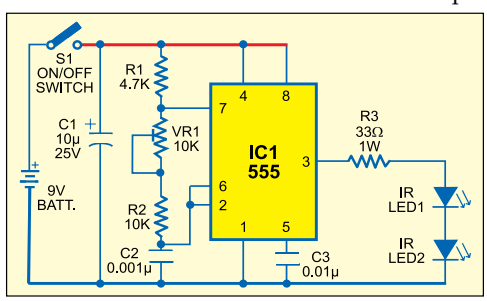


Fig. 1: Transmitter circuit

Powered by a 9V battery, the transmitter circuit (see Fig. 1) comprises IC 555 (IC1), which is wired as an astable multivibrator with a centre frequency of about 38 kHz, and two infrared light-emitting diodes (LEDs). The receiver circuit (see Fig. 2) is powered by a 5V regulated power supply built around transformer X1, bridge rectifier comprising diodes D1 through D4 and regulator IC2. It uses an infrared receiver (IR) module (RX1), optocoupler (IC3) and a simple calculator.

When switch S1 is in 'on' position, the transmitter circuit activates to produce a square wave at its output pin 3. The two infrared LEDs (IR LED1 and IR LED2) connected at its output transmit modulated IR beams at the same frequency (38 kHz). The oscillator frequency can be adjusted using preset VR1.

In the receiver circuit, IR receiver module TSOP1738, which is commonly used in colour televisions for sensing the IR signals transmitted from the TV remote, is used as the sensor.

The IR beams transmitted by IR LED1 and LED2 fall on infrared receiver module IR RX1 of the receiver circuit to produce a low output at its pin 2. This keeps transistor T1 in non-conduction mode.

Now when anyone enters through the gate to interrupt the IR beam, the IR receiver module produces a high output pulse at its pin 3. As a result, transistor T1 conducts to activate IC3 and its internal transistor shorts key '=' of the calculator to advance the count by one.

Both the transmitter and the receiver can be assembled on any general-purpose PCB. Place the transmitter and the receiver around one metre apart.

For calibration, press switches S1 and S2 followed by 'on' key of the calculator. Now press '1' and '+' keys sequentially to get '1' on the screen of the calculator. Then, place a piece of cardboard between the transmitter and the receiver to interrupt the IR rays two times. If the calculator counts '2,' the counter is working properly for that range. Repeat this procedure for higher ranges as well. If there is any problem, adjust VR1.

For installation, switch off the transmitter, receiver and calculator, and mount the transmitter and the receiver on the opposite pillars of the main entry gate such that they are properly orientated towards each other. Mount the calculator where you can read it easily. Connect pins 4 and 5 of IC3 across '=' key connections on the PCB of the calculator.

Now switch on the transmitter and the receiver by pressing switches S1 and S2, respectively. Thereafter, switch on the calculator and press '1' followed by '+' key of the calculator to initialise it. Now your counter is ready to count.

The calculator reads '1' after one interruption, '2' after second interruption and so on. ●

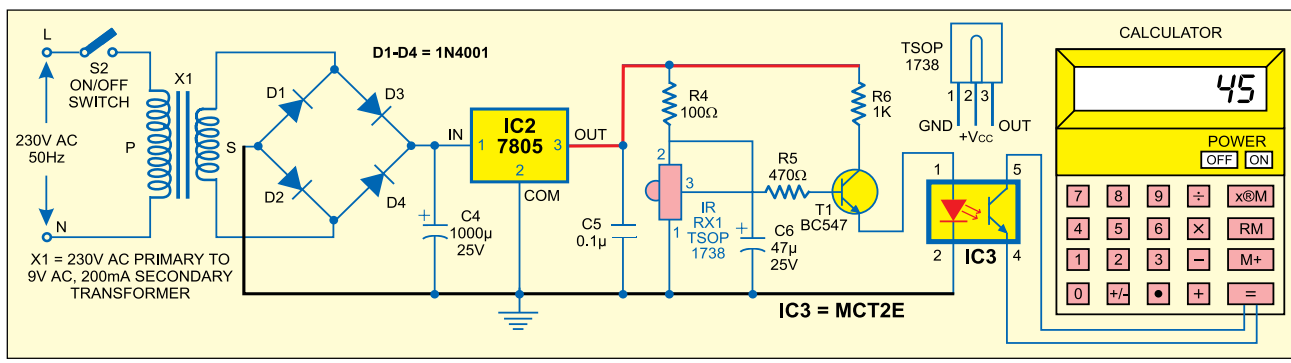


Fig. 2: Receiver-cum-counter circuit

LONG-RANGE BURGLAR ALARM USING LASER TORCH

■ **PRADEEP G.**

Laser torch-based burglar alarms normally work in darkness only. But this long-range photoelectric alarm can work reliably in daytime also to warn you against intruders in your big compounds, etc. The alarm comprises laser transmitter and receiver units, which are to be mounted on the opposite pillars of the entry gate. Whenever anyone enters to interrupt the transmitted laser beam falling on the receiver, the buzzer in the receiver circuit sounds an alarm.

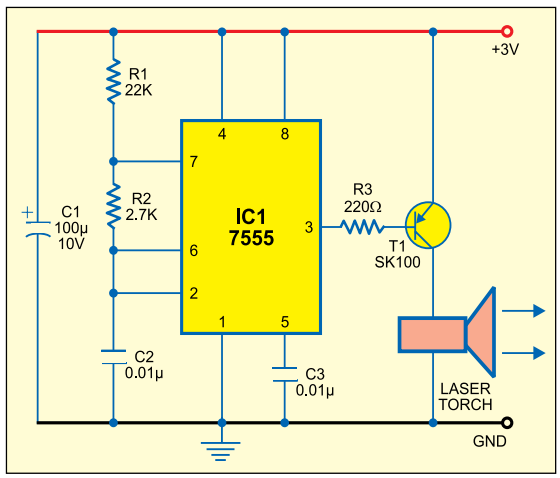


Fig. 1: Circuit of laser torch based transmitter

The range of this burglar alarm is around 30 metres, which means you can place the transmitter and the receiver up to 30 metres apart. Since the laser torch can transmit light up to a distance of 500 metres, this range can be increased by orienting the phototransistor sensor properly. To avoid false triggering by sunlight, mount the phototransistor sensor such that it doesn't directly face sunlight.

The transmitter circuit is powered by 3V DC. The astable multivibrator built around timer 7555 (IC1) produces 5.25kHz frequency. CMOS version of timer 7555 is used for low-voltage operation. The body of the laser torch is connected to the emitter of npn transistor T1 and the spring-loaded lead protruding from inside the torch is connected to the ground.

The receiver circuit is powered by 12V DC. It uses photoDarlington 2N5777 (T2) to sense the laser beam transmitted from the laser torch. The output beam signals from photoDarlington are given to the two-stage amplifier followed by switching circuit, etc. As long as the laser beam falls on photoDarlington T2, relay RL1 remains un-energised and the buzzer does not sound. Also, LED1 doesn't glow.

When anyone interrupts the laser beam falling on photoDarlington T2, npn transistor T6 stops conducting and npn transistor T7 is driven into conduction. As a result, LED1 glows and relay RL1 energises to sound the buzzer for a few seconds (determined by the values of resistor R15 and capacitor C10). At the same time, the large indication load (230V AC alarm for louder sounds or any other device for momentary indication) also gets activated as it is connected to 230V AC mains via normally opened (N/O) contact of relay RL1. ●

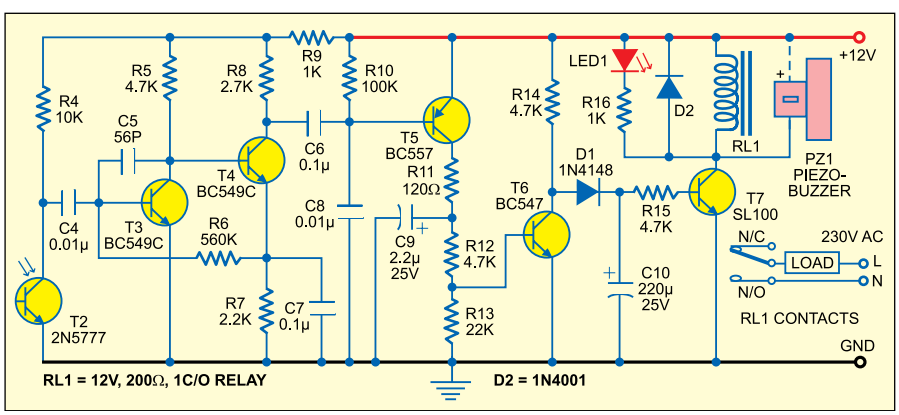


Fig. 2: Receiver circuit

MUSICAL LIGHT CHASER

■ **DEBARAJ KEOT**

This music-operated lighting effect generator comprises five sets of 60W bulbs that are arranged in zig-zag fashion. The bulb sets glow one after another depending on the intensity of the audio signal. No electrical connection is to be made between the music system and the lighting effect generator circuit. You just need to place the gadget near the speakers of the music system.

Fig. 1 shows the complete circuit of the musical light chaser, while Fig. 2 shows pin configurations of 9V regulator 7809, triac BT136 and level meter IC LB1403.

The circuit is powered by regulated 9V DC. The AC mains is stepped down by transformer X1 to deliver a secondary output of 12V AC at 250 mA. The transformer output is rectified by a full-wave rectifier compris-

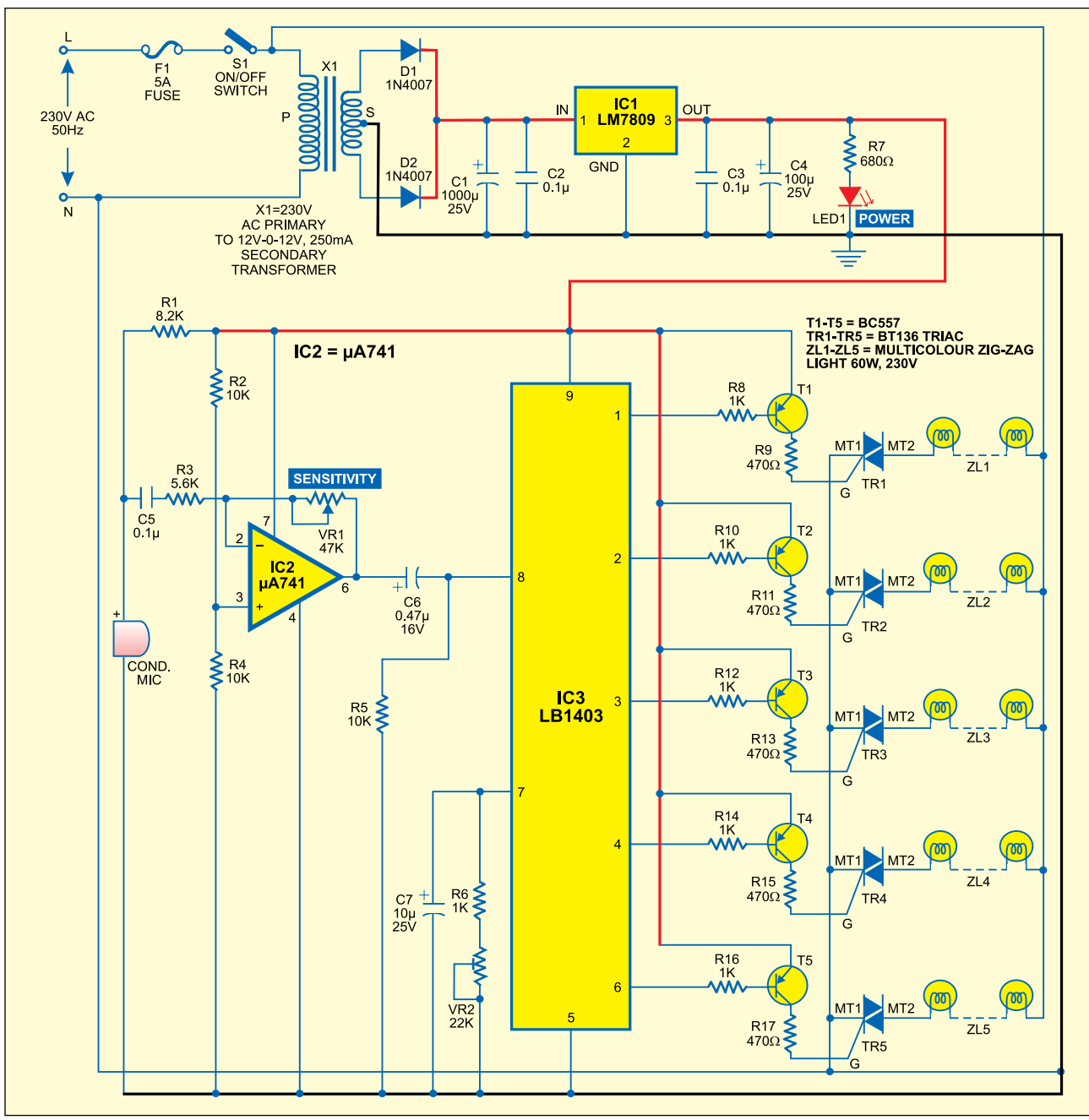


Fig. 1: Circuit diagram of musical light chaser

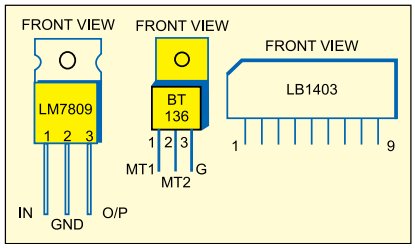


Fig. 2: Pin configuration

ing diodes D1 and D2 and filtered by capacitors C1 and C2. Regulator IC 7809 (IC1) provides regulated 9V power supply to the circuit. Closing switch S1 provides power to the circuit and LED1 glows to indicate that the circuit is ready to work.

When you put your music system in front of the condenser microphone of the light chaser, the sound pressure variation is converted into electrical signals by the condenser microphone. These weak electrical signals are amplified by op-amp µA741 (IC2), which is configured as an inverting amplifier. Using preset VR1 you can set the sensitivity of the circuit.

The amplified output is fed to IC LB1403 (IC3) at its input pin 8. IC3 is a five-dot LED level meter commonly used in stereo systems for LED bargraph displays. It has a built-in amplifier, comparators and constant current source at its output pins.

Depending on the intensity of the input audio signals, all or some outputs of IC3 go low to drive transistors T1 through T5, which, in turn, fire the corresponding triacs TR1 through TR5 via their gates and multicoloured zig-zag bulb sets comprising ZL1 through ZL5 glow.

When the audio level is low, only triac T1 is fired and the zig-zag bulb set ZL1 turns on and off sequentially. When the audio level is high, triacs TR1 through TR5 get fired and all the bulb sets (ZL1 through ZL5) turn on and off sequentially.

Pin 7 of IC3 is used for selecting the response speed of the lighting. The larger the time constant, the slower the response, and vice versa. The time constant can be changed by changing the values of resistor R6, variable resistor VR2 and capacitor C7. Here, variable resistor VR2 is used for varying the response speed of the chaser light as desired. When VR2 is set in the minimum resistance position, the response is very fast, and when it is set at the maximum resistance, the response is slow.

The complete circuit including the power supply can be constructed on any general-purpose PCB or a small Vero board. Triacs TR1 through TR5 should be kept away from the op-amp and its related components. The metallic parts of the triacs should not touch each other and the other parts of the circuit. After assembling the circuit, house it in a suitable shockproof plastic cabinet. Make some holes in the cabinet for heat dissipation.

Note. 1. Some zig-zag lights have a special bulb called 'master bulb' for automatic flickering. It should be removed and replaced with a simple non-flickering colour bulb.

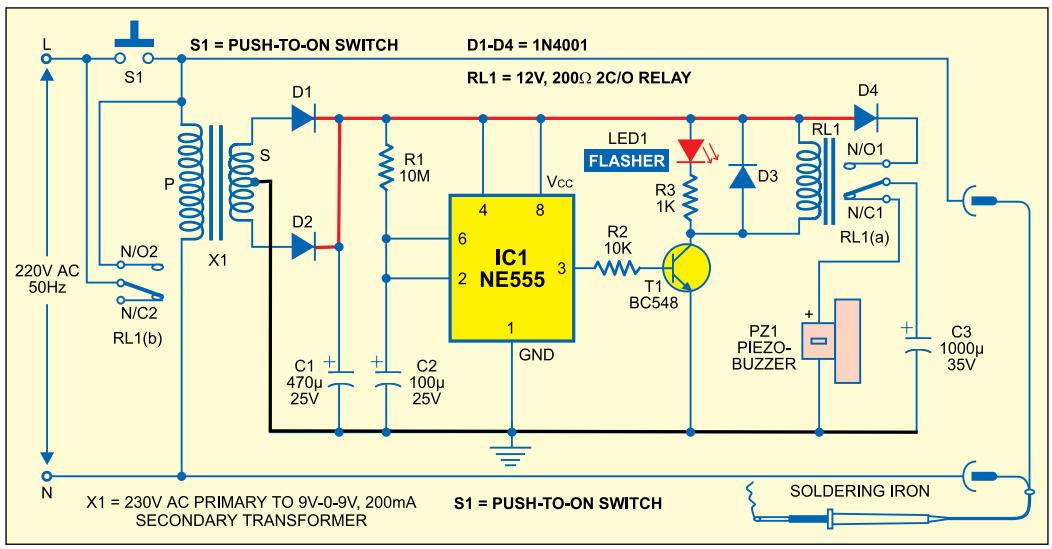
2. Never touch any naked part of the circuit when it is connected to the mains. ●

AUTOMATIC SOLDERING IRON SWITCH

■ T.A. BABU

Quite often, we forget to turn off the soldering iron. This results in not only a smoking oxidised iron but also waste of electricity. To solve this problem, here's a circuit that automatically switches off the soldering iron after a predetermined time. The circuit draws no power when it is inactive. The circuit can also be used for controlling the electric iron, kitchen timer or other appliances.

At the heart of the circuit is a monostable multivibrator built around timer IC 555. When the circuit is in sleep mode, to switch on the soldering iron, you should push switch S1 momentarily. The multivibrator gets triggered and its output pin 3 goes high for around 18 minutes to keep relay RL1 energised via transistor T1. At the same time, capacitor C3 charges and AC supply is provided to switch on the soldering iron via normally opened (N/O) contacts of relay RL1.



The soldering iron remains 'on' for the time period predetermined by resistor R1 and capacitor C2. Here, this time is set for 18 minutes. Flashing of LED1 indicates the heating progress of the soldering iron. When the predetermined time is over, relay RL1 de-energises to turn off the soldering iron and the buzzer sounds until capacitor C3 gets discharged.

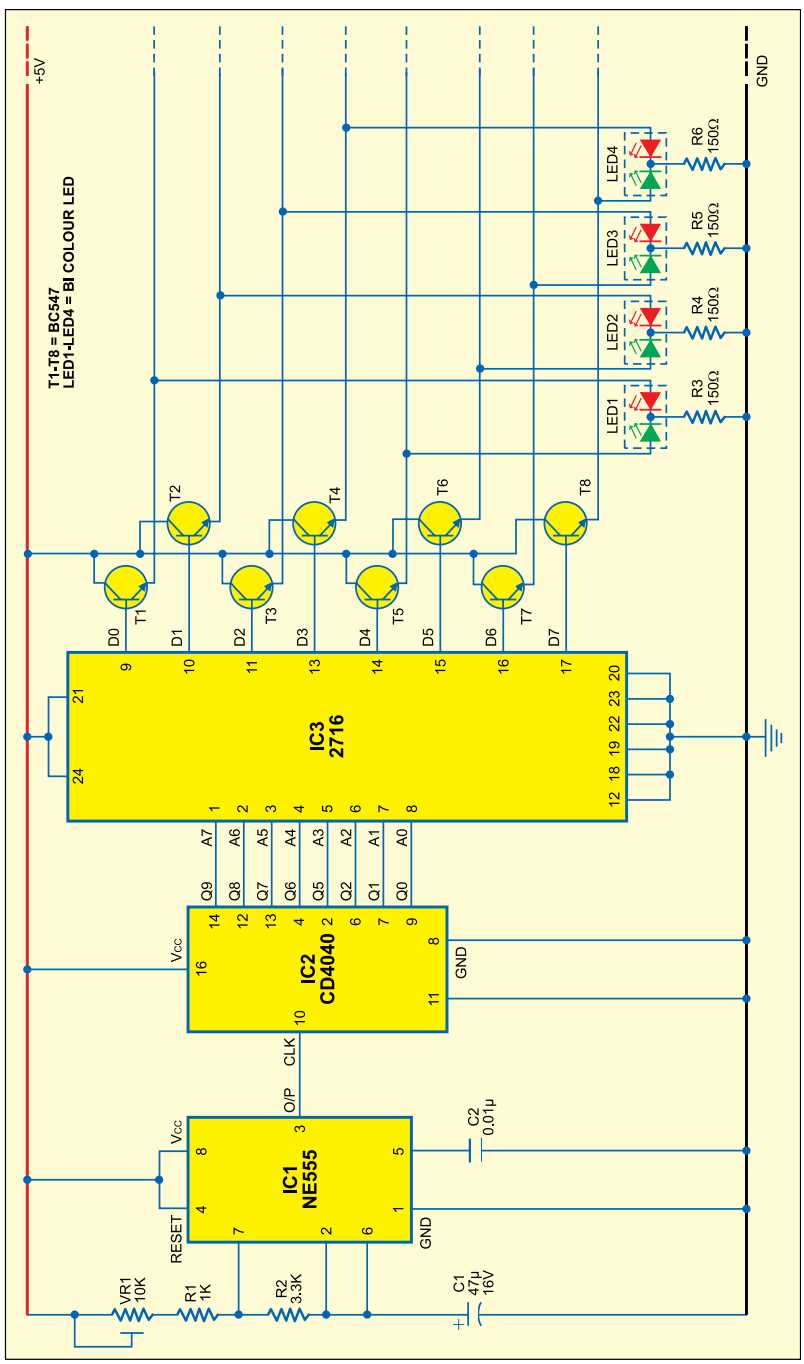
For switching on the circuit, use either a bell push switch or a similar switch with appropriate current carrying capacity. ●

VERSATILE LED DISPLAY

■ **PRIYANK MUDGAL**

This circuit uses an erasable programmable read-only memory (EPROM) to display various light patterns on LEDs. Since bicolour LEDs (comprising green and red LEDs) have been used, display is possible in three colours (green, red and amber).

The circuit is powered by 5V DC. IC 555 (IC1) is wired as an astable multivibrator, whose oscillation frequency can be varied using preset VR1. The output of IC1 clocks 12-stage binary counter IC CD4040 (IC2), which, in turn, provides address data to EPROM IC 2716 (IC3). IC3 contains the code (see Table I) for the display.



The high logic at any data pin causes the corresponding LED to glow. When the data at address location 00H is addressed, the red LED of LED1 glows. The data byte 44H at address location 09H causes both the green and red LEDs of LED2 to glow (refer the table).

The binary outputs of IC2 comprising Q0, Q1, Q2, Q5, Q6, Q7, Q8 and Q9 have been connected to address pins A0 through A7 of EPROM IC3 (2716). Q3 and Q4 outputs of IC2 have not been used. This causes each display pattern to be repeated eight times before the next pattern is displayed. You can adjust the number of times a display pattern repeats by changing the output lines of IC2 connected to the EPROM's address pins A0 through A7.

Binary Code for EPROM

Address	Data	Address	Data
00	01	20	F0
01	02	21	F0
02	04	22	F0
03	08	23	F0
04	10	24	0F
05	20	25	0F
06	40	26	0F
07	80	27	0F
08	88	28	FF
09	44	29	FF
0A	22	2A	FF
0B	11	2B	FF
0C	F8	2C	F0
0D	F4	2D	0F
0E	F2	2E	FF
0F	F1	2F	F0
10	8F	30	0F
11	4F	31	87
12	2F	32	C3
13	1F	33	E1
14	87	34	F0
15	4B	35	E1
16	2D	36	C3
17	1E	37	87
18	78	38	F0
19	B4	39	78
1A	D2	3A	3C
1B	E1	3B	1E
1C	84	3C	0F
1D	42	3D	1E
1E	21	3E	3C
1F	18	3F	78

The circuit uses a total of four bicolour LEDs. However, more LEDs in pairs of four can be added in the dotted lines (see the figure). Suppose you want to connect four more bicolour LEDs (LED5 through LED8, not shown in the figure). For this, you'll have to connect them in parallel to LED1 through LED4, respectively. The speed of the display can be changed by varying preset VR1, which changes the clock frequency.

You can also create other display patterns by coding the EPROM accordingly. Note that the code should be burnt into the EPROM (by using a programmer kit) before it is inserted into the circuit. ●

AUTO TURN-OFF BATTERY CHARGER

■ **Y.M. ANANDAVARDHANA**

This charger for series-connected 4-cell AA batteries automatically disconnects from mains to stop charging when the batteries are fully charged. It can be used to charge partially discharged cells as well.

The circuit is simple and can be divided into AC-to-DC converter, relay driver and charging sections.

In the AC-to-DC converter section, transformer X1 steps down mains 230V AC to 9V AC at 750 mA, which is rectified by a full-wave rectifier comprising diodes D1 through D4 and filtered by capacitor C1. Regulator IC LM317 (IC1) provides the required 12V DC charging voltage. When you press switch S1 momentarily, the charger starts operating and the power-on LED1 glows to indicate that the charger is 'on.'

The relay driver section uses pnp transistors T1, T2 and T3 (each BC558) to energise electromagnetic relay RL1. Relay RL1 is connected to the collector of transistor T1. Transistor T1 is driven by pnp transistor T2, which, in turn, is driven by pnp transistor T3. Resistor R4 (10-ohm, 0.5W) is connected between the emitter and base of transistor T3.

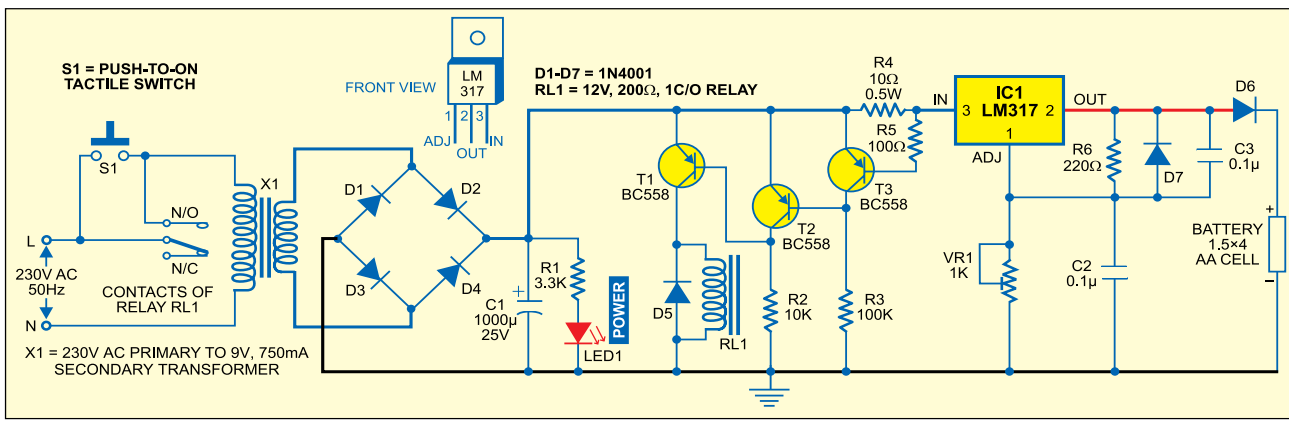
When a current of over 65 mA flows through the 12V line, it causes a voltage drop of about 650 mV across resistor R4 to drive transistor T3 and cut off transistor T2. This, in turn, turns transistor T1 'on' to energise relay RL1. Now even if the pushbutton is released, mains is still available to the primary of the transformer through its normally open (N/O) contacts.

In the charging section, regulator IC1 is biased to give about 7.35V. Preset VR1 is used for adjusting the bias voltage. Diode D6 connected between the output of IC1 and battery limits the output voltage to about 6.7V, which is used for charging the battery.

Pushing switch S1 latches relay RL1 and the battery cells start charging. As the voltage per cell increases beyond 1.3V, the voltage drop across resistor R4 starts decreasing. When it falls below 650 mV, transistor T3 cuts off to drive transistor T2 and, in turn, cuts off transistor T3. As a result, relay RL1 de-energises to cut off the charger and red LED1 turns off.

You may determine the charging voltage depending on the NiCd cell specifications by the manufacturer. Here, we've set the charging voltage at 7.35V for four 1.5V cells. Nowadays, 700mAH cells are available in the market, which can be charged at 70 mA for 10 hours. The open-circuit voltage is about 1.3V.

The shut-off voltage point is determined by charging the four cells fully (at 70 mA for 14 hours). After measuring the output voltage, add the diode drop (about 0.65V) and bias LM317 accordingly. ●



PENCELL CHARGE INDICATOR

■ **D. MOHAN KUMAR**

Small-size AA cells and button cells used in electronic devices providing a terminal voltage of 1.5V are normally rated at 500 mAh. As the cells discharge, their internal impedance increases to form a potential divider along with the load and the battery terminal voltage reduces. This, in turn, reduces the performance of the gadget and we are forced to replace the battery with a new one. But the same battery can be used again in some other application that requires less current.

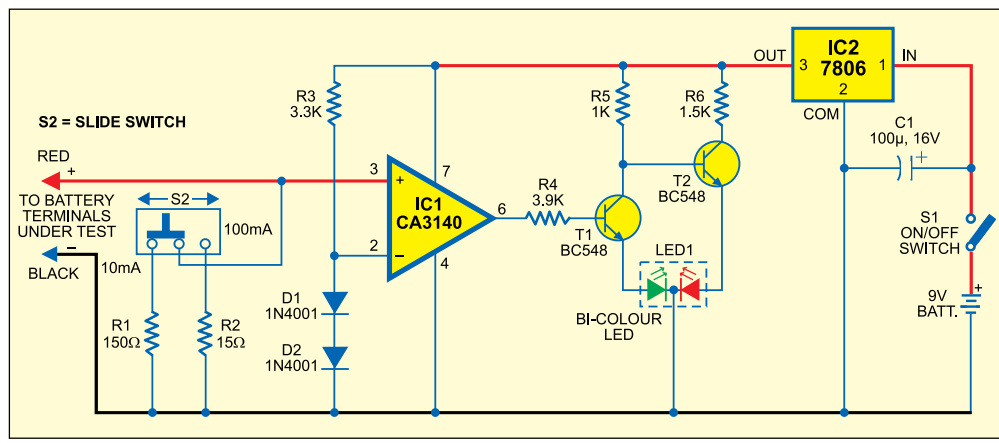
Here's a simple tester for quick checking of discharged pencells and button cells before throwing them away. The tester detects the holding charge of the battery and the terminal voltage to indicate whether the battery is suitable for a particular gadget or not.

A 9V battery can power the circuit with sufficient voltage and current. When you close switch S1, it provides stable 6V DC to the circuit.

The circuit uses op-amp CA3140 (IC1) as a voltage comparator. It can sense even a slight voltage variation between its inverting and non-inverting inputs. The non-inverting input (pin 3) of IC1 is supplied with a voltage obtained from the battery under test, while its inverting input pin 2 is provided with a reference voltage of 1.4V derived by resistor R4 and series combination of diodes D1 and D2. Resistors R1 and R2 provide a loading of 10 mA and 100 mA, respectively, for checking the charge capacity.

When a new battery is connected to the test terminals, the non-inverting input of IC1 gets 1.5V, which exceeds the voltage of the inverting input and the output of IC1 goes high. This high output provides forward bias to transistor T1 through resistor R4 and it conducts to light up the green half of the bicolour LED (LED1). Simultaneously, the base of transistor T2 is pulled down and it turns off and the red half of bicolour LED1 remains off.

When a partially discharged battery (with a terminal voltage of less than 1.4 V) is connected to the test terminals, the output of IC1 goes low to switch off transistor T1. This allows transistor T2 to forward bias by taking bias voltage through resistor R5 and the red LED within bicolour LED1 glows.



Slide switch S2 is used to check whether the battery is holding sufficient current to drive a load of 10 mA or 100 mA. If the discharged battery holds more than 100mA current, the green LED within bicolour LED1 glows, indicating that the battery can be used again in a low-drain circuit.

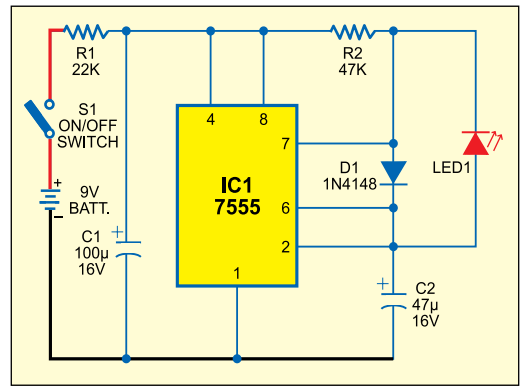
The circuit can be easily constructed on a perforated board using readily available components. Enclose it in a small case with probes or battery holder for testing. ●

MISER FLASH

■ T.A. BABU

A flashing LED at the doorstep of your garage or home will trick the thieves into believing that a sophisticated security gadget is installed. The circuit is nothing but a low-current drain flasher. It uses a single CMOS timer that is configured as a free running oscillator using a few additional components. As the LED flashes very briefly, the average current through the LED is around 150 µA with a high peak value, which is sufficient for normal viewing. This makes it a real miser.

The 9V battery source is connected via 'on'/'off' switch S1 to the circuit. When switch S1 is closed, the IC receives power from capacitor C1, which is constantly charged through resistor R1. As capacitor C1 delivers power to IC1, it saves the battery from drain.



Most LEDs consume a current of 20 mA, which in many instances is higher than the power consumed by the rest of the circuit. This is undesirable if the device is battery-powered. In this circuit, the energy consumed by the LED is a small fraction of the normal value.

Capacitor C2 charges through resistor R2 and diode D1. When the voltage across C2 reaches two-third of the supply voltage, threshold pin 7 of IC1 switches on as a current sink. The capacitor discharges through LED1 into pin 7 rapidly. Diode 1N4148 (D1) provides the one-way charging path for capacitor C2 via resistor R2. LED1 illuminates briefly for a while with the accumulated charges in C2. Again, the charging cycle repeats. This way, LED continues flashing. A 9V PP3 battery can perfectly handle this job. ●

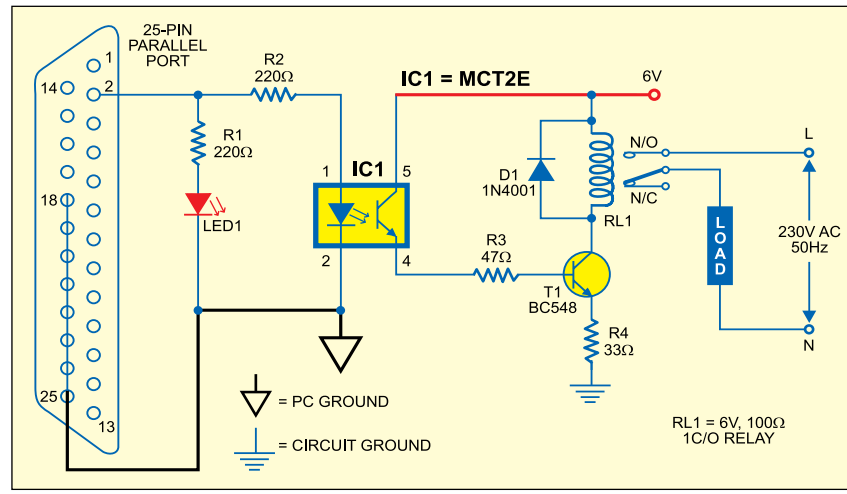
PC-BASED TIMER

■ AKSHAY MATHUR

Timers are very useful both for industrial applications and household appliances. Here is a PC-based timer that can be used for controlling the appliances for up to 18 hours. For control, the timer uses a simple program and interface circuit. It is very cost-effective and efficient for those who have a PC at workplace or home. The tolerance is ±1 second.

The circuit for interfacing the PC's parallel port with the load is very simple. It uses only one IC MCT2E, which isolates the PC and the relay driver circuits. The IC prevents the PC from any short circuit that may occur in the relay driver circuit or appliance. The glowing of LED1 indicates that the appliance is turned on. Transistor BC548 is used as the relay driver.

The program code is written in 'C' language and compiled using 'Turbo C' compiler. When the program is run, it prompts the user to input the time duration in seconds or minutes to control the appliance. After entering the required timing, press any key from the keyboard.



Suppose you input the total duration as 'x' minutes, of which 'on' and 'off' durations are 'y' and 'z' minutes, respectively. The program will repeat the on-off cycle for x/(y+z) number of times. After completion of the total time, to repeat the cycle, you will have to reset the time in the program to activate the circuit.

The program uses two bytes for storing integer type data. So when input is given in terms of seconds or minutes, it can hold 2^{16}–1=65,535 seconds or 18 hours at the maximum. The sleep() function in the program is used to hold the appliance in 'on' or 'off' condition for the 'on' and 'off' periods as entered by the user against prompts. The sound() function is used to give a beep during 'on' condition of the appliance.

EFY note. The source code and executable file of this program can be downloaded from: http://www.efymag.com/admin/issuepdf/pc%20Based%20Timer.zip

TIMER.C

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <stdlib.h>
void main()
{
int PORT=0x0378,a,b,c,d=0;
clrscr();
_setcursortype(_NOCURSOR);
gotoxy(1,15);
textcolor(2);
printf("This is the Program to use PC Parallel port as Timer for external circuits");
gotoxy(1,17);
printf("\nWith this program you can time the circuits for precision upto a second\n");
printf("\nPress any key to continue....\n");
getch();
clrscr();
gotoxy(1,5);
printf("How do you want to enter the time duration ?");
printf("\n\n\n1.Enter time duration in minutes (press m)");
printf("\n\n2.Enter time duration in seconds (press s)\n");
gotoxy(1,15);
switch(getch())
{
case 'm':
{
printf("Enter the total time duration (in minutes): ");
scanf("%d",&a);
printf("Enter the time to keep the circuit on (in minutes): ");
scanf("%d",&b);
printf("Enter the time to switch off the circuit (in minutes): ");
scanf("%d",&c);
a=a*60;
b=b*60;
c=c*60;
printf("Press any key to start the program");
getch();
do
{
outportb(PORT,1);
printf("\nYour circuit is on");
sound(330);
sleep(b);
nosound();
outportb(PORT,0);
printf("\nYour circuit is off");
sleep(c);
d=d+b+c;
}
```

```
while(d<a);
printf("\n Your program completed succes-
fully");
outport(PORT,0);
sleep(5);
exit(0);
}
case 's':
{
printf("Enter the total time duration (in sec): ");
scanf("%d",&a);
printf("Enter the time to keep the circuit on (in
sec): ");
scanf("%d",&b);
printf("Enter the time to switch off the circuit
(in sec): ");
scanf("%d",&c);
printf("Press any key to start the program");
getch();
do
{
outportb(PORT,1);
printf("\nYour circuit is on");
sound(330);
sleep(b);
nosound();
outportb(PORT,0);
printf("\nYour circuit is off");
sleep(c);
d=d+b+c;
}
while(d<a);
printf("\nYour program completed succesfully");
outportb(PORT,0);
sleep(5);
exit(0);
}
}
}
```

ATMEL AVR ISP DONGLE

■ **EFY LAB**

Atmel's AVR microcontroller chips are in-system programmable (ISP), i.e. these can be programmed directly in the target circuit. A special programmer software is used to download the program from the PC into the AVR's flash memory. Atmel offers a software package called the Atmel AVR ISP that allows programming of the AVR microcontrollers in the circuit using a simple dongle. A dongle is nothing but an adaptor cable that connects the PC's parallel port with the ISP pins of the AVR chip for programming.

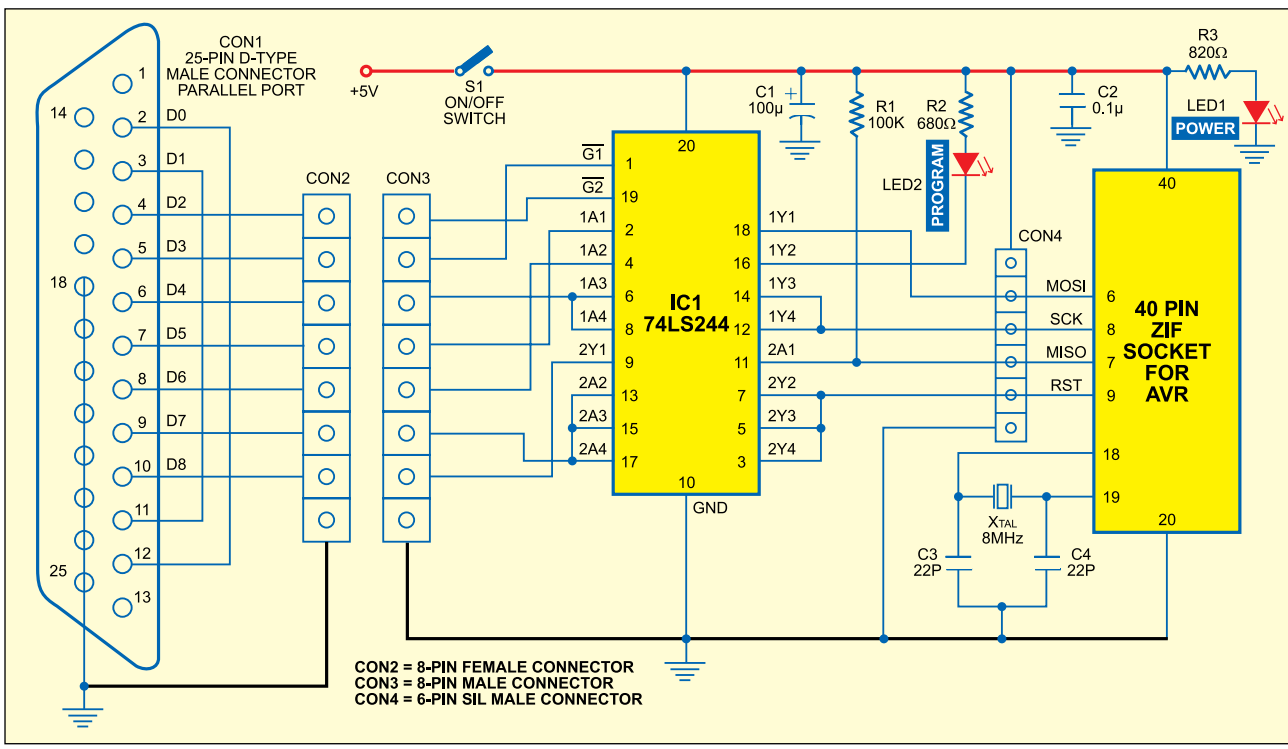
For programming, the four lines required from the AVR chip to the ISP adaptor (dongle) are:

1. MOSI (Master Out, Slave In): Data being transmitted to the AVR being programmed is sent on this pin

2. MISO (Master In, Slave Out): Data received from the AVR being programmed is sent on this pin

3. SCK (Shift Clock): Serial clock generated by the programmer from the PC.

4. RST (Reset): Reset (low pulse) generated by the program. The AVR is programmed while in reset state.



Here's a dongle circuit for in-system programming of Atmel's AVR chip AT90S8515 using such software packages as Atmel ISP 2.65 and PonyProg2000. Though not exactly the same, a similar dongle circuit can be found at the Website 'www.iready.org/projects/uinternet/ispdongle.pdf.'

The PC's parallel-port pins 4 and 5 drive buffer IC 74LS244 by enabling its pins 19 and 1, respectively. A low pulse on these pins will allow the passing of the serial clock and data during programming. MOSI, LED, SCK and RST outputs are buffered from the parallel port's pins 7, 8, 6 and 9, respectively. The MISO input from the AVR is fed into pin 10 of the parallel port.

IC 74LS244 (IC1) acts as a buffer as well as an isolator circuit when the AVR is not in programming mode. In idle mode, all the outputs are tristated so as not to affect the operation of the target system.

When the AVR's ISP mode is selected, the lower half of IC 74LS244 is enabled, pulling the target system's Reset line low. Once the target system is in Reset mode, the SCK, MISO and MOSI lines are no longer loaded by the peripheral circuitry, if any, on the target system. Now, it is safe to enable the upper half of 74LS244, driving the MOSI, LED and SCK lines of the dongle. The RST pin becomes high after the AVR is programmed. Glowing of LED2 indicates that the AVR is in programming mode.

There are two standard connectors for in-system programming of Atmel AVR microcontroller. One is the 10-pin header (dual-in-line (DIL) connector)) used on the Atmel STK kits. The other is a 6-pin header (DIL connector) used in Atmel ISPs. The two loop-back connections, pin 2-to-pin 12 and pin 3-to-pin 11 of the parallel port, are

used to identify the dongle. With only pin 2-to-pin 12 link, the dongle is called STK300 or AVR ISP dongle. With only pin 3-to-pin 11 link, the dongle is called STK200 or old Kanda ISP dongle. With both links in place, the dongle is identified as a value-added pack dongle.

Here, we've used an 8-pin single-in-line (SIL) connector and an additional 6-pin SIL connector for the Atmel programer circuit. With the buffer and the 40-pin ZIF socket in this circuit, it can be used as a standalone programmer. The 6-pin SIL male connector is used for connection between the dongle and the AVR on the target board. Thus, another 6-line cable of about 30cm length is required for connecting this ISP adaptor (dongle) to the target circuit.

If the AVR is not on the target circuit, you can insert the AVR into the ZIF socket and program it. Regulated 5V DC is required for the AVR and the associated dongle circuit, whose terminals are also provided in connector CON4. LED1 is used as the power indicator for the circuit. ●

DIGITAL FREQUENCY COMPARATOR

■ V. GOPALAKRISHNAN

Here's a digital frequency comparator for oscillators that indicates the result through a 7-segment display and a light-emitting diode (LED). When the frequency count of an oscillator is below '8,' the corresponding LED remains turned off. As soon as the count reaches '8,' the LED turns on and the 7-segment display shows '8.'

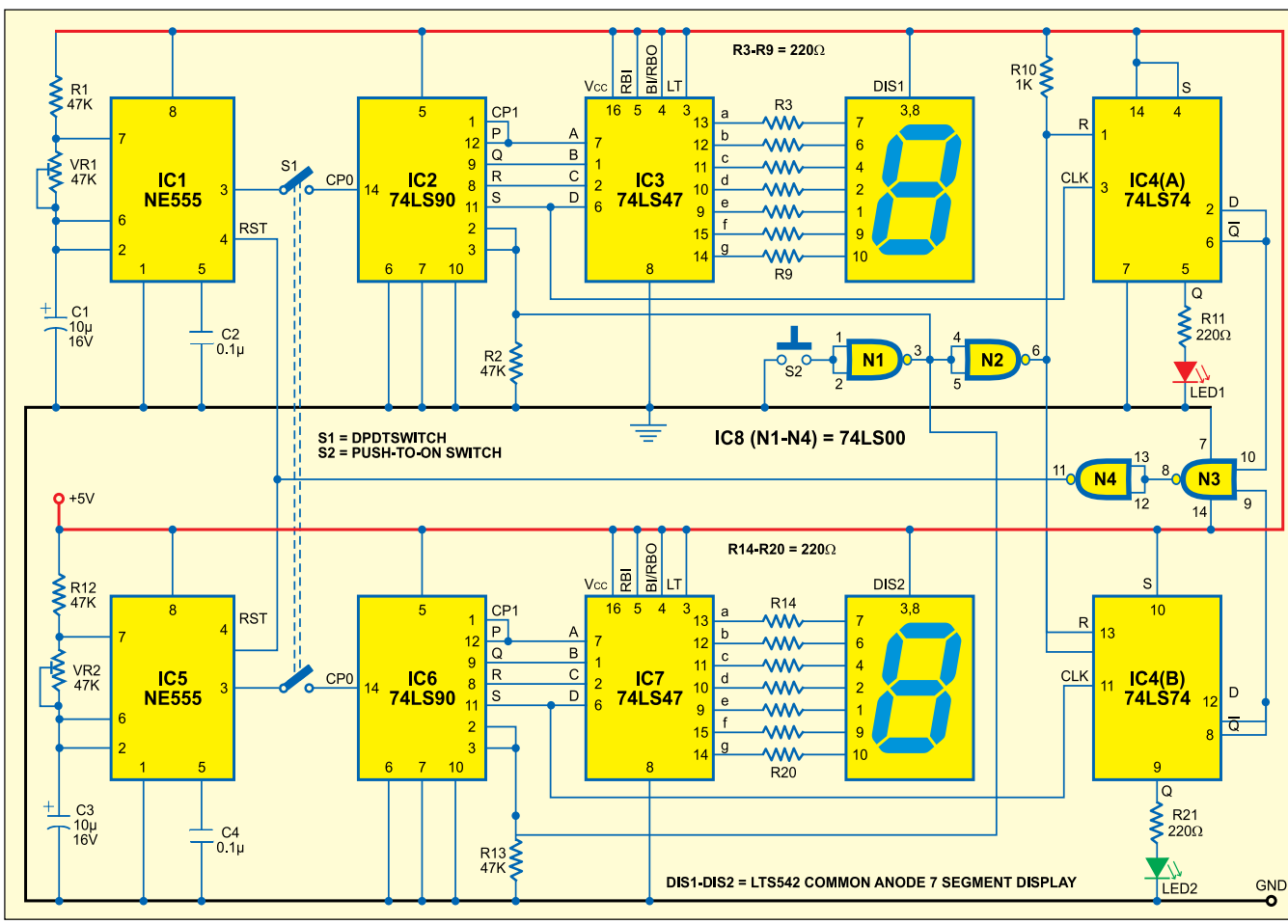
This demo circuit uses two NE555 timers configured as astable free-running oscillators, whose frequencies are to be compared.

The circuit of the digital frequency comparator portion comprises two 74LS90 decade counter ICs (IC2 and IC6), two 74LS47 7-segment display driver ICs (IC3 and IC7), 74LS74 set/reset flip-flop (IC4), 74LS00 NAND gate (IC8) and two 7-segment displays (DIS1 and DIS2). The astable free-running oscillators built around the timers are the frequency sources for the corresponding counters.

When power supply to the circuit is switched on, timing capacitor C1 starts charging through resistor R1 and potmeter VR1. When the capacitor voltage reaches 2/3Vcc, the internal comparator of IC1 triggers the flip-flop and the capacitor starts discharging towards ground though VR1. When the capacitor voltage reaches 1/3Vcc, the lower comparator of IC1 is triggered and the capacitor starts charging again. The charge-discharge cycle repeats. That means, the capacitor charges and discharges periodically between two-third and one-third of the power supply (Vcc). The output of NE555 is high during charging and low during discharging of capacitor C1.

The other oscillator (IC5) works similarly. The oscillator frequency can be varied by the potentiometer (VR1 or VR2). Output pins (pin 3) of the oscillators (IC1 and IC5) are connected to the respective decade counters (IC2 and IC6) through the DPDT switch.

IC2 and IC6 count the initial eight cycles. IC 74LS90 is a 4-bit ripple decade counter. It consists of a divide-by-two section and a divide-by-five section counter. Each section has a separate clock input. The input of the divide-by-five section (CP1) is externally connected to the P output (pin 12) of the divide-by-two section (CP0). When the divide-by-two sec-



tion receives clock pulse, it becomes a divide-by-ten counter.

Decade counter 74LS90 is reset by a high pulse at its pins 2 and 3. Initially, pins 2 and 3 are pulled down by resistor R2. The P through S outputs of IC2 are connected to the A through D inputs of IC3. Pin 11 (S) of IC2 is also connected to pin 3 of IC4(A) for providing the clock pulse. The count is displayed on the 7-segment display.

The 7-segment decoder/driver (74LS47) accepts four binary-coded decimals (8421), generates their complements internally and decodes the data with seven AND/OR gates having the open-collector output to drive the display segments directly. Each segment-driver output is capable of sinking 40mA current in the 'on' state. Pins 3, 4 and 5 of the display driver are connected to Vcc to disable the ripple-blanking input (RBI), blanking input (BI)/ripple-blanking output (RBO) and lamp test (LT).

IC3 provides segment data to the 7-segment display through current-limiting resistors R3 through R9 (each 220 ohms).

IC 74LS74 (IC4) controls the reset pin (RST) of NE555. It is a dual D-type flip-flop with direct clear and set inputs and complementary outputs. The input data is transferred to the outputs on the positive edge of the clock pulse. Since the Q output is connected to the data input D, the flip-flops work in toggle mode.

Initially, reset pins 1 and 13 of the flip-flops are pulled high via resistor R10. When the reset pin of any flip-flop receives a low pulse from NAND gate N2 of IC8, the flip-flop is reset and its Q output goes high. On receiving a clock pulse, the Q and Q_outputs of the flip-flop go high and low, respectively, and the LED turns on. The low output of IC4 resets the oscillators. The reset signal is derived with the help of NAND gates N3 and N4.

When switch S2 is pressed, both the oscillators and the respective counters start working. As soon as any of the counters counts '8,' the corresponding display shows '8' and LED glows. This means that oscillator has a higher frequency. Now both the counters stop counting because the flip-flop output goes low to reset both the astable oscillators.

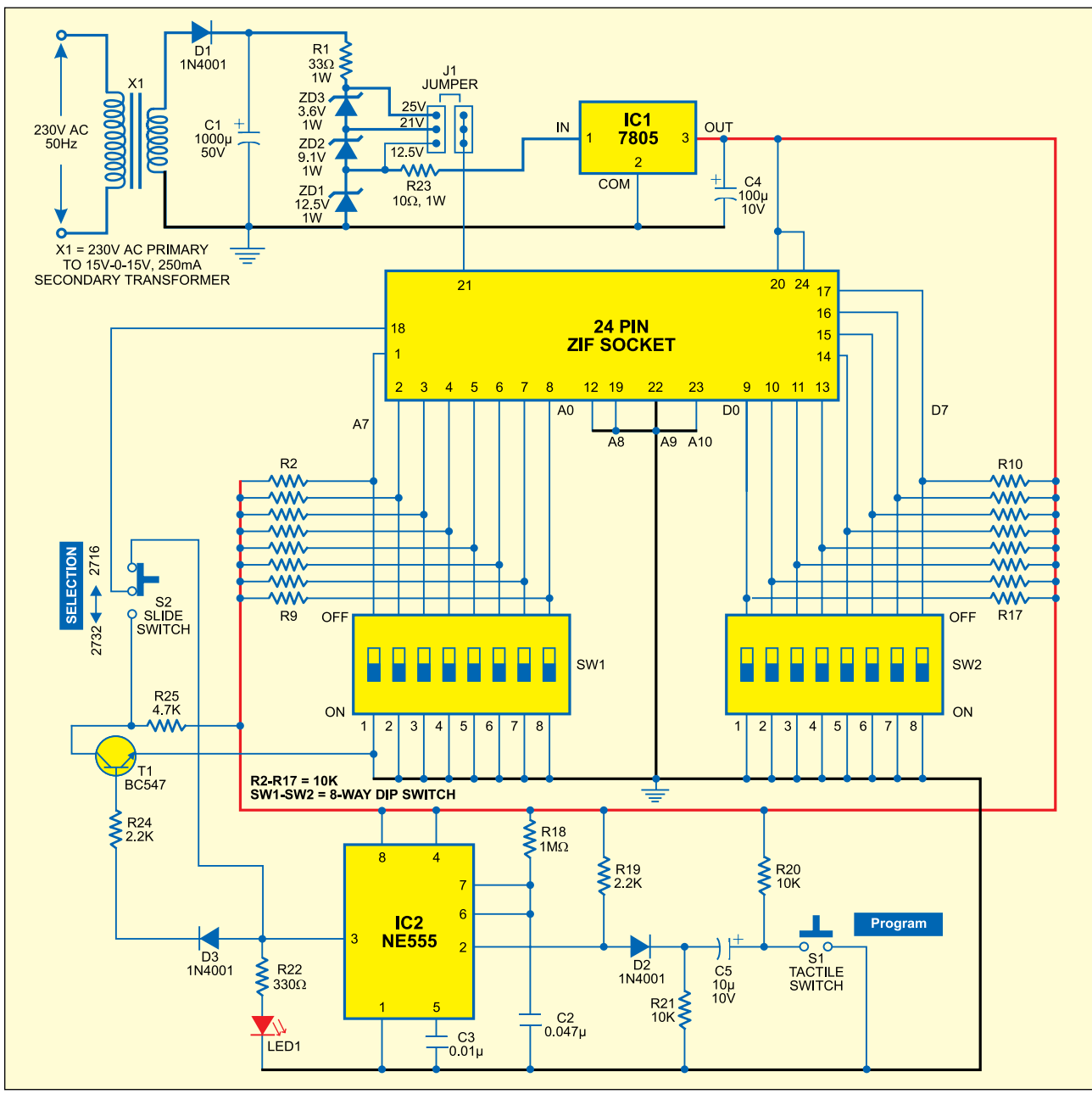
In case the frequencies of both the astable oscillators are same, both the displays show '8' and LED1 and LED2 glow at the same time. ●

MANUAL EPROM PROGRAMMER

■ **PRIYANK MUDGAL**

The programmer devices required for programming the electrically programmable read-only memories (EPROMs) are generally expensive. Here is a low-cost circuit to program binary data into 2716 and 2732 EPROMs.

The circuit uses timer NE555 (IC2) wired as a monostable. When push-to-on switch S1 is pressed, IC2 generates a 50ms pulse, which is given to the program pin 18 of the ZIF SOCKET through switch S2. EPROM is inserted into the ZIF SOCKET for programming. LED1 glows to indicate the application of the programming pulse to the EPROM. Before applying the programming pulse to the EPROM, select the programming voltage (25V, 21V or 12.5V, as specified by the manufacturer) applied to pin 21 of the ZIF SOCKET by using jumper J1. The programming voltage required for an EPROM is sometimes written on its body. The address and data for the EPROM (ZIF SOCKET) are set by using DIP switches SW1 and SW2, respectively, whose pins are initially pulled high via 10-kilo-ohm resistors.



The AC mains is stepped down by transformer X1 to deliver 30V, 250 mA from the secondary. The secondary output is rectified by diode D1 and filtered by capacitor C1. The programming voltages of 25V, 21V and 12.5V are generated with the help of zener diodes ZD1, ZD2 and ZD3. IC1 is used to provide +5V regulated supply to the circuit.

To begin with, first read the programming voltage written on the EPROM. Now insert the EPROM chip into the 24-pin ZIF socket and slide switch S2 as per EPROM. Then connect the power supply to provide regulated 5V DC to the circuit. Select the programming voltage using jumper J1 and set the programming address and data value using switches SW1 and SW2, respectively. After providing the required programming voltage, press switch S1 to program the data at the desired address. Repeat this procedure for the next address and corresponding data. ●

WIRELESS STEPPER MOTOR CONTROLLER

■ JAYDIP APPASAHEB DHOLE

Here is a low-cost and simple wireless stepper motor controller using infrared signals. Using this circuit you can control the stepper motor from a distance of up to four metres.

The circuit comprises transmitter and receiver sections. The communication between the transmitter and receiver sections is achieved through infrared signals.

In the transmitter section, timer NE555 ICs (IC1 and IC2) are configured as astable multivibrators with frequencies of around 1 Hz and 38 kHz, respectively. The output of IC1 is given to reset pin 4 of IC2, so the 38kHz carrier signal is modulated by 1Hz modulating signal. The modulated signal from pin 3 of IC2 is transmitted by the infrared LED. Resistor R5 limits the current through the IR LED.

The transmitted signal is sensed by IR receiver module TSOP1738 (IC6) of the receiver section and its output at pin 3 is used as clocks for dual flip-flop 74LS74 ICs (IC3 and IC4), which are configured as a ring counter.

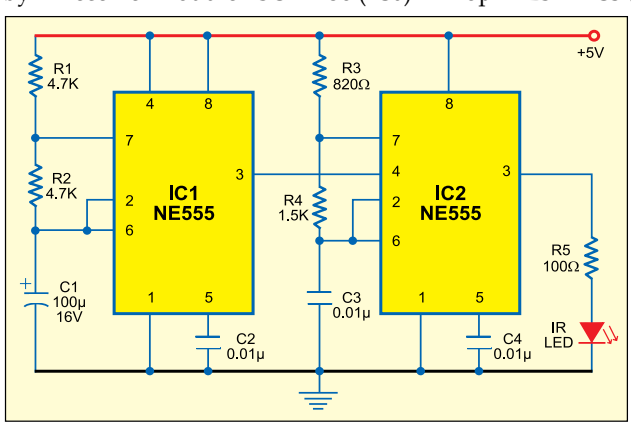


Fig. 1: Infrared transmitter

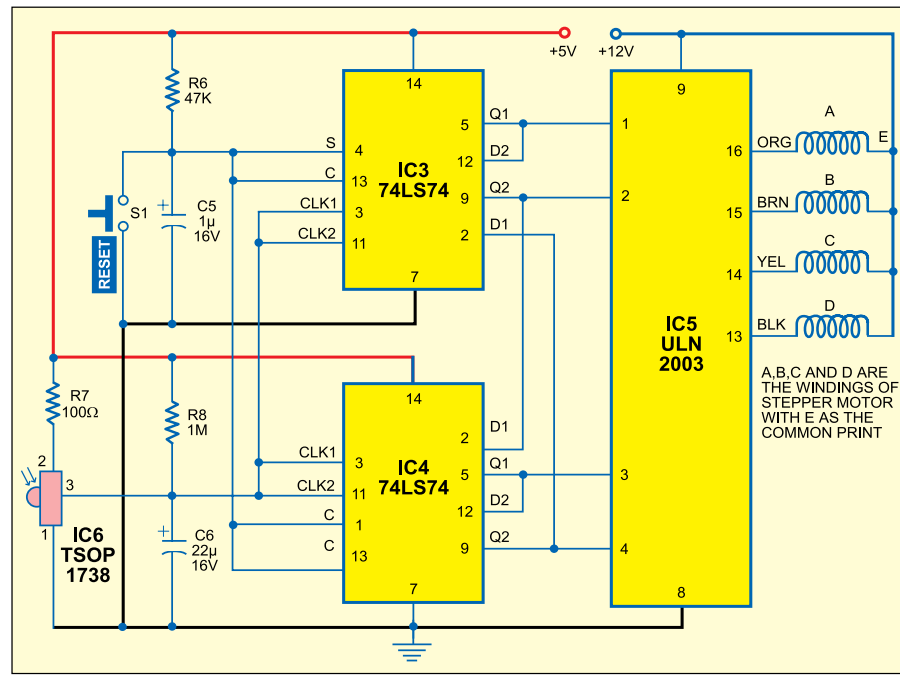


Fig. 2: Infrared receiver and stepper motor driver circuit

When the power is switched on, the first flip-flop is set and its Q1 output goes high, while the other three flip-flops are reset and their outputs go low. On receiving the first clock pulse, the high output of the first flip-flop gets shifted to the second flip-flop. Thus on reception of every clock pulse, the high output keeps shifting in a ring fashion.

The outputs of flip-flops are amplified by the Darlington transistor array inside ULN2003 (IC5) and connected to the stepper motor windings marked 'A' through 'D.' The common point of the windings is connected to +12V DC supply.

To stop the motor, the flip-flops can be reset manually by pressing reset switch S1. On releasing the reset switch, the stepper motor again starts moving. If any interruption occurs between the transmitter and the receiver, the motor stops. ●

SIMPLE DIGITAL SECURITY SYSTEM

■ **PHERDAUS ISLAM**

You can use this simple and reliable security system as a watchdog by installing the sensing loops around your building. You have to stretch the loop wires two feet above the ground to sense the unauthorised entry into your premises.

Wire loops 1, 2 and 4 are connected to the A, B and C inputs of 7-segment decoder 4511 (IC1), respectively, while the D input of IC1 is grounded permanently. The loops are also connected to a dual 3-input NOR gate and inverter CD4000 (IC2) to activate the alarm.

Fig. 1 shows the circuit of the digital security system, while Fig. 2 shows the proposed wiring diagram for the loops around the premises. Before using this security system, make sure that loops shown in Fig. 2 are connected as shown in Fig. 1. If you don't want to use a buzzer, switch it off by opening switch S2.

The circuit works off a 9V regulated power supply. However, battery back-up is recommended. A common-cathode, 7-segment display (LTS543) is used for displaying whether the loops are intact or not.

If loop 1 is broken, the display will show '1'. If two or all the three loops are broken, the display will show the sum of the respective broken loop numbers. For example, if loops 1 and 4 are broken, the display will show 5(1+4).

When all the three loops are intact, the display will show '0.' All the three inputs of gate N1 remain low to give a high output. This high output is further given to gate N2 and, as a result, its output remains low. This keeps transistor T1 in cut-off position and the piezobuzzer does not sound.

When any loop is broken, the output of NOR gate N1 goes low, while the output of gate N2 goes high. Transistor T1 conducts and the buzzer sounds to alert you. You can mute the buzzer by switching off power to the circuit through switch S1. ●

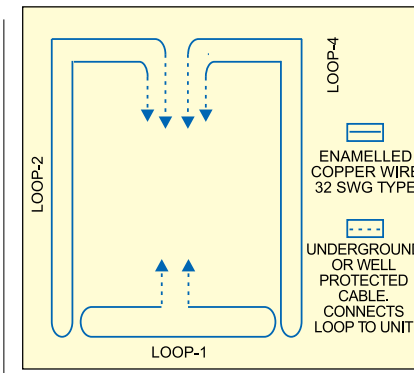


Fig. 2: The proposed wiring diagram of loops

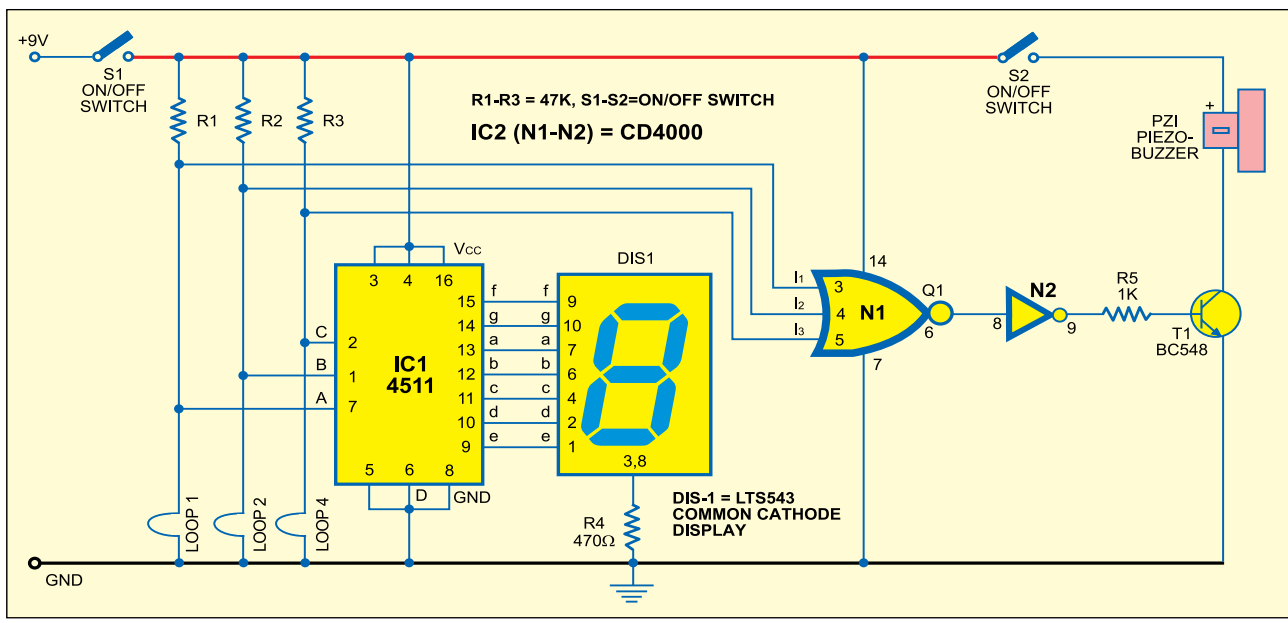


Fig. 1: The digital security system circuit

MULTIPLE APPLICATIONS OF HIGH-POWER LEDS

■ **N.S. HARI SANKAR-VU3NSH**

Nowadays, high-power light-emitting diodes (LEDs) LXHLMW1C are available in the market. These white LEDs contain indium-gallium-nitrogen (InGaN). The LEDs' emitting capacity is 20 candela (Cd). We can use these LEDs for automatic garden lighting and wide voltage operation by applying different voltages.

Fig. 1 shows the circuit for automatic garden lighting. Switch S1 connects 12V to the circuit built around transistors T1 and T2. Light-dependent resistor LDR1 is used to sense the light intensity and preset VR1 is used to adjust the threshold of light. The resistance of LDR remains low in daylight and high at night (in darkness).

In the morning, light falls on LDR1 and transistors T1 and T2 are cut-off. As a result, 12V supply is not available to the LEDs. In the evening, when no light falls on LDR1, transistors T1 and T2 conduct to provide 12V to the LEDs. This turns on all the LEDs (LED1 through LED60). The on/off switching level can be adjusted by 220-kilo-ohm preset according to the intensity of the light.

The emitting capacity of LEDs (UW-510CWH) used here is 8 Cd. Since a total of 60 of these LEDs have been used, this unit will provide luminous intensity equivalent of 480 Cd. The LEDs are arranged in twenty rows, with each row having three LEDs in series. The input voltage is approximately 12V and all the LEDs are spaced 1 to 1.5 cm apart.

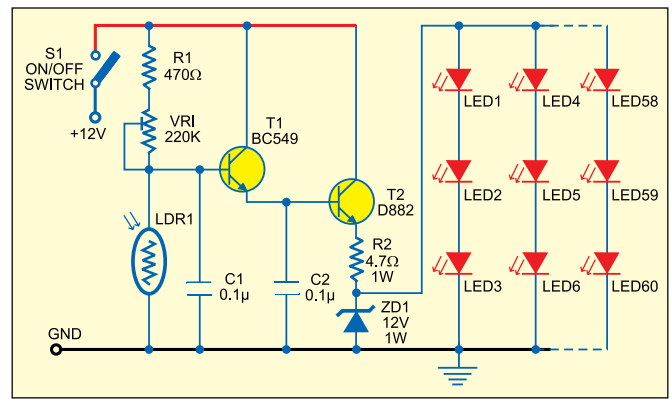


Fig. 1: Circuit for automatic garden lighting

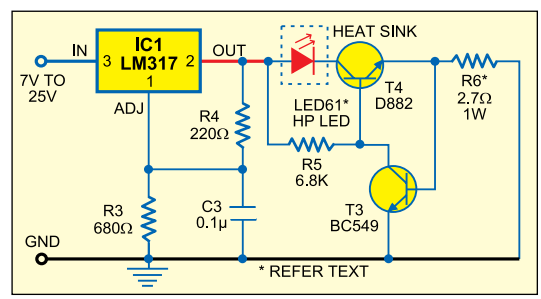


Fig. 2: Circuit for wide-voltage operation

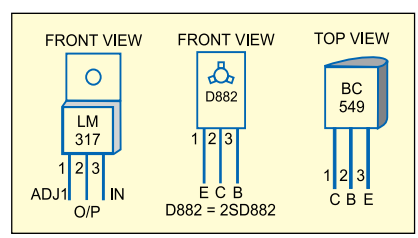


Fig. 3: Pin configuration

The entire circuit, except LDR1, can be assembled on any general-purpose PCB. House the PCB in a box and, using two long wires, mount LDR1 at a place where light falls on it directly. Now place the unit in your garden.

You can use the switching section for other systems as well. You just need to remove the LED section from the circuit and connect the switching section to the desired system. So the system will now automatically switch on in the evening and switch off in the morning.

Fig. 2 shows a wide-voltage operation circuit. Here, the high-power LED61 (LXHLMW1C) gives a power equivalent of 20 Cd. This LED has a metallic back for mounting on a heat-sink. Its rated maximum input DC voltage and current are 3.6V and 350 mA, respectively. Regulator IC LM317 (IC1) provides a constant voltage of 4.7V. Resistors R3 and R4 limit the current through the LED. The LED is very sensitive to voltage inputs. In the 2.5V-3.5V region, each millivolt variation changes the current through the LED logarithmically. Transistors BC549 and D882 (T3 and T4) and resistor R6 provide a constant current to LED61. The unit gives a constant lighting for voltages ranging from 7V to 25V.

Fig. 3 shows pin configuration of regulator LM317 and transistors D882 and BC549. Use heat-sinks in regulator LM317 and transistor D882 before soldering them onto the PCB. ●

AUTOMATIC BATHROOM LIGHT WITH BACK-UP LAMP

■ D. MOHAN KUMAR

Sometimes we forget to switch off the bathroom light and it remains on unnoticed for long periods. This circuit solves the problem of electricity wastage by switching off the lamp automatically after 30 minutes once it is switched on. The back-up LED lamp provided in the circuit turns on for three minutes when mains fails. This is helpful especially when you are taking a shower at night.

The circuit is built around binary counter CD4060 (IC2), which has a built-in oscillator and 14 cascaded bistable multivibrators. The oscillator generates clock pulses based on the values of resistors R3 and R4 and capacitor C3.

For the given values, Q11 output of IC2 goes high after 30 minutes of power-on. Resistor R2 resets the IC for proper operation. The output of IC2 is fed to the gate of the SCR via resistor R6 and LED2, which function as a voltage dropper as well as output status indicator.

When the SCR gets gate drive, it fires to energise relay RL1. The latching function of the SCR keeps the relay energised until the power to the circuit is switched off using switch S1. When the relay energises, its normally closed (N/C) contacts break and light turns off. LED1 indicates that the oscillator is working.

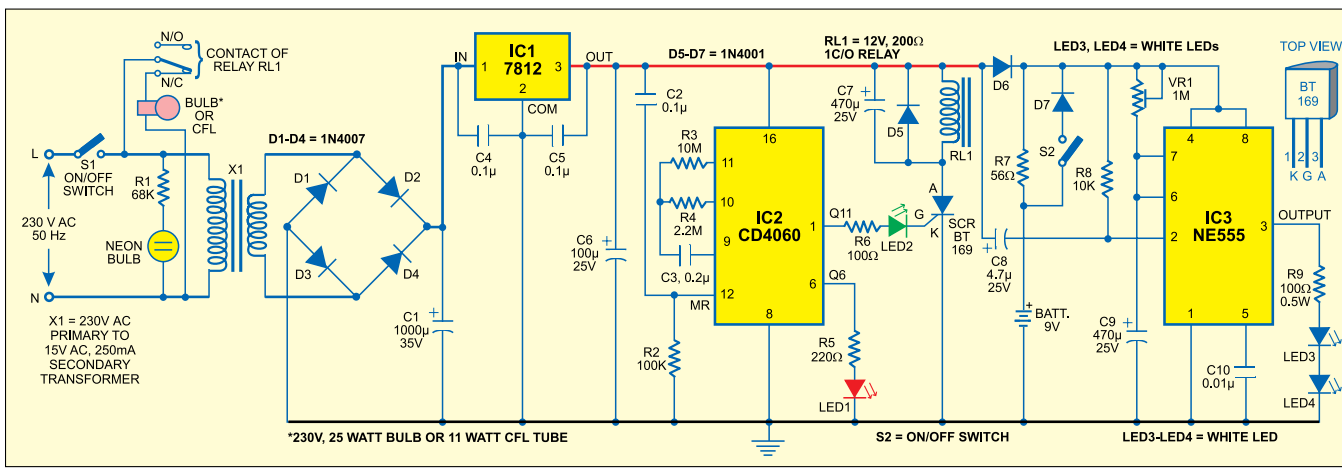
The back-up white-LED lamp comprising LED3 and LED4 gives ample light in the event of mains failure. It is powered by a 9V rechargeable battery, which is charged at around 200mA current via diode D6 and resistor R7 when the circuit is switched on.

The back-up lamp circuit is built around timer NE555 (IC3) designed as a monostable. The output of IC3 goes high for three minutes based on the values of preset VR1 and capacitor C9. When the circuit is switched on, IC3 gets power supply via diode D6 and its trigger pin 2 remains high due to resistor R8. As a result, its output remains low as long as mains is present.

When power fails, pin 2 of IC3 gets triggered via capacitor C8 and the monostable output goes high to switch on the white LEDs (LED3 and LED4). Resistor R9 limits the current through the LEDs to a safe level. Diode D7 is forward biased to give full voltage to the monostable when power fails.

The power supply for the circuit is derived from a 15V AC, 250mA transformer. The secondary output is rectified by a full-wave rectifier comprising diodes D1 through D4. Capacitor C1 smoothes the resulting DC. Regulator IC 7812 (IC1) and capacitors C4 and C5 provide stabilised 12V for the circuit.

Assemble the circuit on a Vero board and enclose it in a watertight plastic case. Connect the bathroom lamp (either 25-watt bulb or 11-watt CFL tube) to the circuit via N/C contacts of the relay, so that it turns on when switch S1 is pressed. For easy access, fix switch S1 along with the neon indicator outside the bathroom. ●



DIGITAL AUDIO/VIDEO INPUT SELECTOR

■ T.K. HAREENDRAN

Need to connect more than one audio-video (AV) source to your colour television? Don't worry, here's an AV input expander for your TV. It is inexpensive and easy to construct.

The working of the circuit is simple and straightforward. Whenever 12V DC is applied to the circuit, power-on LED1 glows. Now reset the decade counter by momentarily pressing switch S2 to make Q0 output of IC1 high. LED2 glows to indicate that the circuit is ready to work.

Switch S1 is used for selecting a particular audio-video (AV) signal. To select the first AV signal, press switch S1 once. To select the second AV signal, press switch S1 twice. In the same way, you can select the other two signals.

Momentarily pressing of switch S1 once results in clocking of the decade counter and relay driver transistor T1 conducts to energise relay RL1. Now normally opened (N/O) contacts of two-changeover relay RL1 connect the television set's inputs to the first AV signal (marked as Video-In 1 and Audio-in 1). LED3 glows to indicate this.

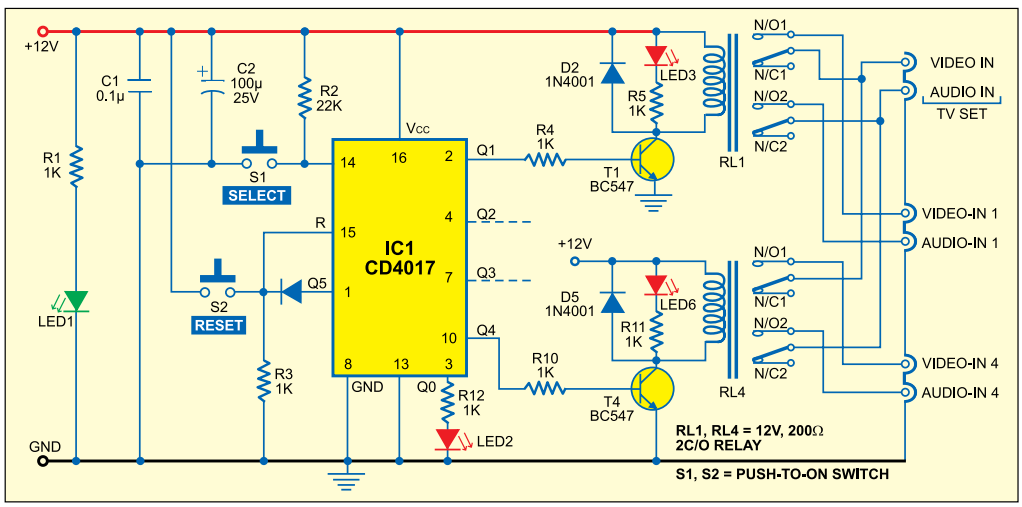
When you press switch S1 twice, the Q2 output of IC1 goes high. Consequently, 2C/O relay RL2 (not shown in the circuit) energises and television inputs are connected to the second AV signal (not shown in the figure). LED4 (not shown in figure) glows to indicate this.

Similarly, pressing switch S1 thrice makes the Q3 output of IC1 high. Consequently, 2C/O relay RL3 (not shown in the figure) energises and the television inputs are connected to the third AV signal source. LED5 (not shown in the figure) glows to indicate this.

Again, pressing switch S1 four times makes the Q4 output of IC1 high. Consequently, 2C/O relay RL4 energises and the TV inputs are connected to the fourth AV signal source (marked as Video-in 4 and Audio-in 4). LED6 glows to indicate this.

Further pressing of switch S1 resets the decade counter and LED2 glows again. Thereafter, the cycle repeats. The circuit is wired for four-input selection, therefore the Q5 output of IC1 is connected to reset pin 15 of IC1.

Enclose the assembled PCB along with the relays in a cabinet with the input/output sockets and indicators mounted on the body of the cabinet. ●



ACCURATE FOOT-SWITCH

■ KAUSHIK HAZARIKA

Certain industrial controls require accurate switching operations. For example, in case of a foot-switch for precise drilling work, even a small error in switching may cause considerable loss. This low-cost but accurate foot-operated switch can prevent that.

IC NE555 is wired in one-shot mode. Its output pin 3 goes high only when both switches S1 and S2 are pressed simultaneously. You can release any one of the switches without changing the output state. When you release both the switches, the output goes low.

The switches are placed under a foot paddle as shown in Fig. 2. LED1 is used as a warning indicator. If either S1 or S2 gets pressed erroneously, LED1 blinks to warn the operator. The operator can then withdraw his foot in case of a mistake or depress the other switch also to trigger the circuit. LED1 is to be mounted on the operator's desk.

The circuit operation is simple. Resistors R2, R3 and R4 form a voltage divider. IC NE555 has two comparators, a flip-flop and power output section built into it. Pressing either S1 or S2 puts the input voltage between the upper comparator (2/3Vcc) and the lower comparator (1/3Vcc). Thus, it has no effect on the state of the internal flip-flop of IC NE555. Pressing the two switches simultaneously sets the flip-flop and the output of NE555 goes high. Transistor T2 energises relay RL1 for driving the load.

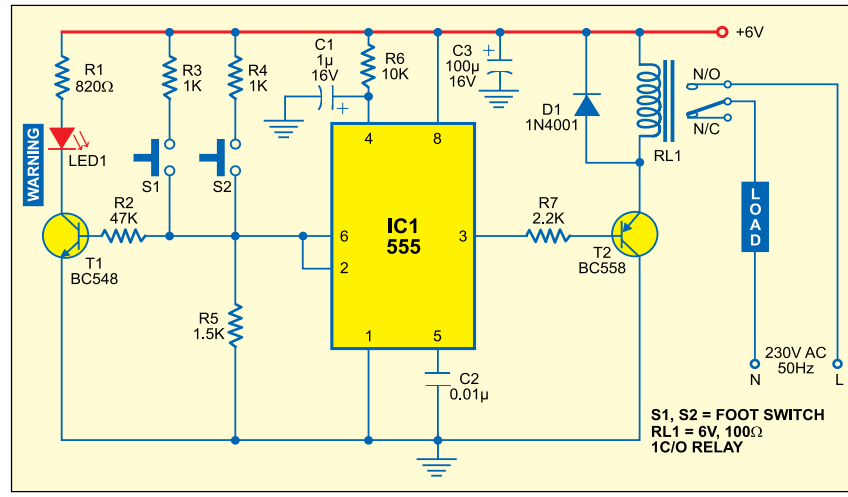


Fig. 1: Circuit of the foot-switch

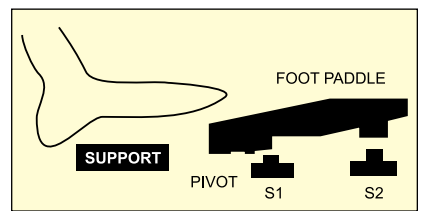


Fig. 2: Foot paddle switch

Releasing any of the switches brings the comparator voltage back to the initial level inside NE555 and it has no effect on the state of the flip-flop. Releasing both the switches brings the input level with respect to ground below the low trigger level, and thus it resets the output.

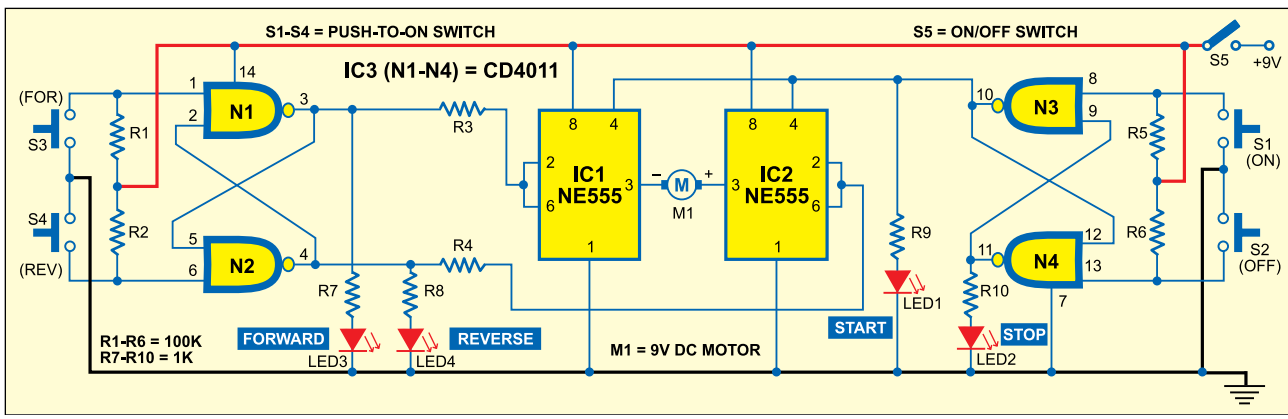
Use of the voltage divider results in stable operation over the entire permissible supply voltage range. The R-C circuit at pin 4 provides power-on reset.

When only S1 is pressed, R3 (1 kilo-ohm) is less than R5 (1.5 kilo-ohms) and IC1 is not triggered. However, transistor T1 (BC548) gets forward biased and LED1 glows. When both S1 and S2 are pressed, the effective resistance between +Vcc and pin 2 of IC1 is about 500 ohms, which is less than R5 (1.5 kilo-ohms), and IC NE555 gets triggered. ●

MICROMOTOR CONTROLLER

■ **V. DAVID**

Using this circuit, you can control the rotation of a DC micromotor simply by pressing two push-to-on switches momentarily.



The circuit is built around two NE555 ICs (IC1 and IC2) and a quad-NAND IC CD4011 (comprising NAND gates N1 through N4). The NE555 ICs (IC1 and IC2) are configured as inverting buffers. IC CD4011 (IC3) NAND gates are configured as bistable flip-flop. The DC motor to be controlled is connected between the outputs (pin 3) of IC1 and IC2.

Closing switch S5 provides power to the circuit. Now, when you press switch S1 momentarily, pin 10 of IC3 goes high, while its pin 11 goes low. Since pin 10 of IC3 is connected to reset pin 4 of IC1 and IC2, the high output at pin 10 of IC3 will enable IC1 and IC2 simultaneously. When switch S2 is pressed, pin 10 of IC3 goes low, while its pin 11 goes high. The low logic at pin 10 disables both IC1 and IC2.

Switches S3 and S4 are used for forward and reverse motion of the motor in conjunction with switch S1. If you press switch S3 after pressing switch S1, pin 3 of IC3 goes high, while its pin 4 goes low. The motor now starts rotating in the forward direction. However, if you press switch S4 after pressing switch S1, the motor will rotate in reverse direction.

Note. The complete kit of this circuit can be obtained from Kits'n'Spares, 303, Website: www.kitsnspares.com; E-mail: info@kitsnspares.com. ●

POWER-ON REMINDER WITH LED LAMP

■ D. MOHAN KUMAR

Many a times equipment at workstations remain switched on unnoticed. In this situation, these may get damaged due to overheating. Here is an add-on device for the workbench power supply that reminds you of the power-on status of the connected devices every hour or so by sounding a buzzer for around 20 seconds. It also has a white LED that provides good enough light to locate objects when mains fails.

Fig. 1 shows the circuit of power-on reminder with LED lamp. Here, IC NE555 (IC1) is wired as an astable multivibrator, whose time period is set to around six minutes using resistors R1 and R2, preset VR1 and capacitor C1 for sounding the buzzer every hour. The output of IC1 is fed to the clock input of IC CD4017 (IC2). Capacitor C3 and resistor R3 provide power-on-reset pulse to IC2.

When power to the circuit is switched on, pin 3 of IC2 goes high. After around one hour, its output pin 11 (Q9) goes high and the buzzer sounds. This cycle repeats until the power to the circuit is switched off.

The automatic lamp is built around a light-dependent resistor (LDR) and two npn transistors. The LDR offers a very high resistance in darkness, i.e., when no light falls on it. Therefore when power fails, transistor T1 gets reverse biased to drive transistor T2 and the white LED (LED2) glows. The lamp circuit is powered by a 9V rechargeable battery, which is charged via resistor R5 when mains is present. Thus in darkness, the LED remains 'on.'

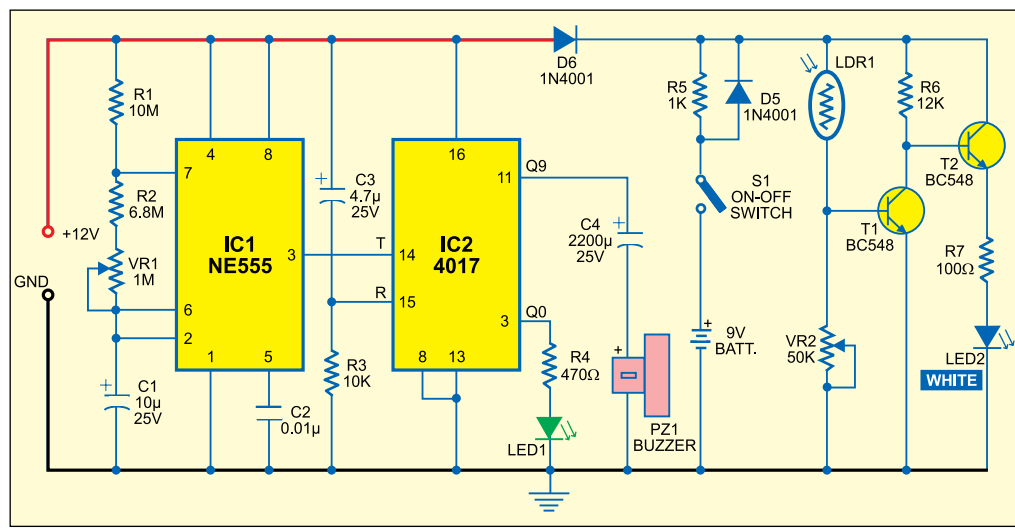


Fig. 1: Circuit of power-on reminder with LED lamp

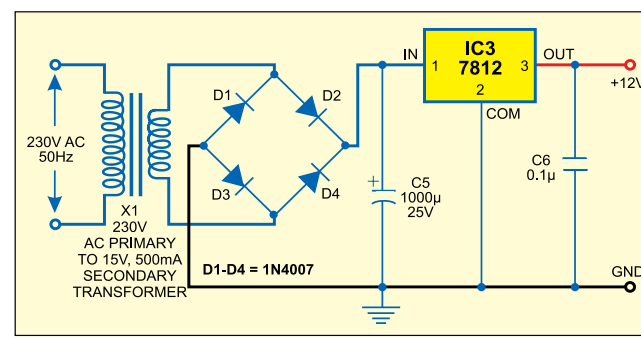


Fig. 2: Power supply circuit

Fig. 2 shows the power supply circuit. The AC mains is stepped down by transformer X1 to deliver a secondary output of 15V AC at 500 mA. The transformer output is rectified by a bridge rectifier comprising diodes D1 through D4, filtered by capacitor C5 and regulated by IC 7812 (IC3) to provide regulated 12V to the circuit. Capacitor C6 bypasses any ripple in the regulated output. ●

MAINS INTERRUPTION COUNTER WITH INDICATOR

■ T.K. HAREENDRAN

This circuit counts mains supply interruptions (up to 9) and shows the number on a 7-segment display. It is highly useful for automobile battery chargers. Based on the number of mains interruptions, the user can extend the charging time for lead-acid batteries.

Fig. 1 shows the circuit of the interruption counter with indicator. A 9V (PP3 or 6F22) battery powers the entire circuit. Fig. 2 shows the block diagram of the mains interruption counter circuit along with the battery charger and lead-acid battery as used in automobile battery charger shops.

When 9V is applied to the circuit, IC2 is reset by the power-on-reset signal provided by capacitor C3 and resistor R5 and the 7-segment display (DIS1) shows '0.' The 230V AC mains is fed to mains-voltage detection optocoupler IC MCT2E (IC1) via capacitor C1 and resistors R1 and R2 followed by bridge rectifier BR1, smoothing capacitor C2 and current-limiting resistor R2. Illumination of the LED inside optocoupler IC1 activates its internal phototransistor and clock input pin 1 of IC2 is pulled down to low level.

IC CD4033 (IC2) is a decade counter/7-segment decoder. Its pin 3 is held high so that the display initially shows '0.' Clock pulses are applied to clock input pin 1 and clock-enable pin 2 is held low to enable the counter.

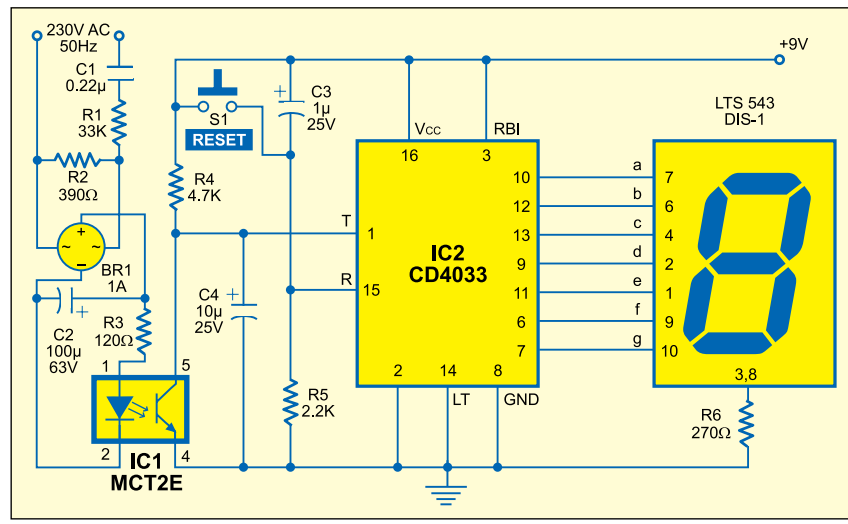


Fig. 1: Circuit of mains interruption counter with indicator

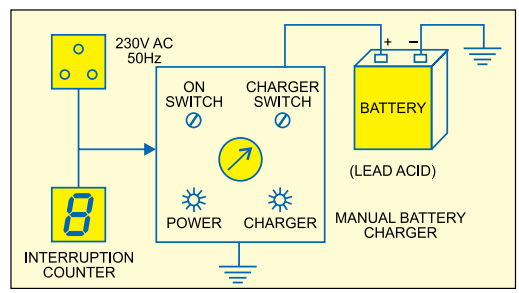


Fig. 2: Block diagram of the arrangement used in automobile battery charger shops

Seven-segment, common-cathode display DIS1 (LTS543) indicates the mains interruption count. Capacitor C2 provides a small turn-on delay for the display.

When mains fails for the first time, clock input pin 1 of IC2 again goes high and display DIS1 shows '1.' When mains resumes, pin 1 of IC2 goes low and DIS1 continues to show '1.' When mains fails for the second time, clock input pin 1 of IC2 goes high and display DIS1 shows '2.' When mains resumes, pin 1 of IC2 again goes low and DIS1 continues to show '2.' This way, the counter keeps incrementing by '1' on every mains interruption. Note that this circuit can count up to nine mains interruptions only. ●

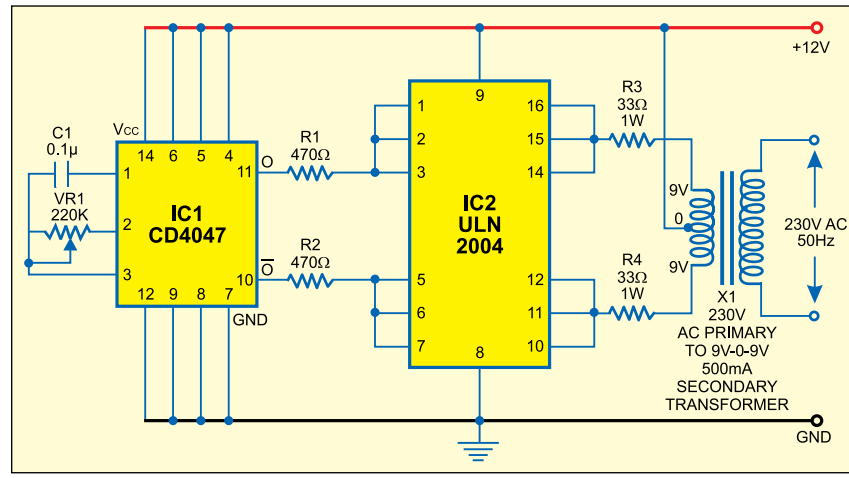
SIMPLE LOW-POWER INVERTER

■ **PRADEEP G.**

Here is a simple low-power inverter that converts 12V DC into 230-250V AC. It can be used to power very light loads like window chargers and night lamps, or simply give shock to keep the intruders away. The circuit is built around just two ICs, namely, IC CD4047 and IC ULN2004.

IC CD4047 (IC1) is a monostable/astable multivibrator. It is wired in astable mode and produces symmetrical pulses of 50 to 400 Hz, which are given to IC2 via resistors R1 and R2.

IC ULN2004 (IC2) is a popular 7-channel Darlington array IC. Here, the three Darlington stages are paralleled to amplify the frequencies received from IC1. The output of IC2 is fed to transformer X1 via resistors R3 and R4.



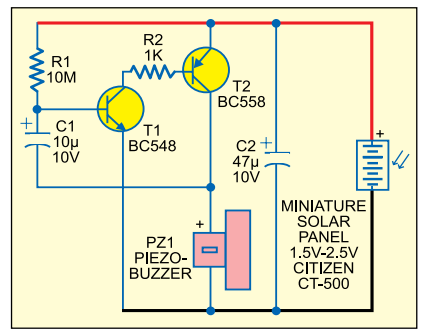
Transformer X1 (9V-0-9V, 500mA secondary) is an ordinary step-down transformer that is used here for the reverse function, i.e., step up. That means it produces a high voltage. Resistors R3 and R4 are used to limit the output current from the ULN to safe values. The 230-250V AC output is available across the high-impedance winding of the transformer's primary windings. ●

SOLAR BUG

■ **D. SOMNATH**

Hide this solar-powered circuit suitably and see the reaction of your friends to the chirpy sound produced by it every few minutes. In all probability, it will coax them to find out where the sound is coming from.

The circuit runs off a miniature solar power panel, which can be taken out from an old calculator such as Citizen CT-500. A panel giving 1.5V to 2.5V is required. Note that the circuit



can work properly from a panel as small as 3 cm^2.

If a digital voltmeter is connected across capacitor C2, a slow build-up of voltage can be observed when the panel is exposed to light. Transistors T1 and T2 form a relaxation oscillator. When C1 charges to 0.6V, transistor T1 conducts and the charge built up in C2 is discharged through the piezobuzzer to produce a short beep.

While testing the circuit, the value of resistor R1 can be reduced to, say, 1 kilo-ohm. Use a good-quality buzzer to ensure that the sound produced is loud enough. ●

REMOTE CONTROL FOR HOME APPLIANCES

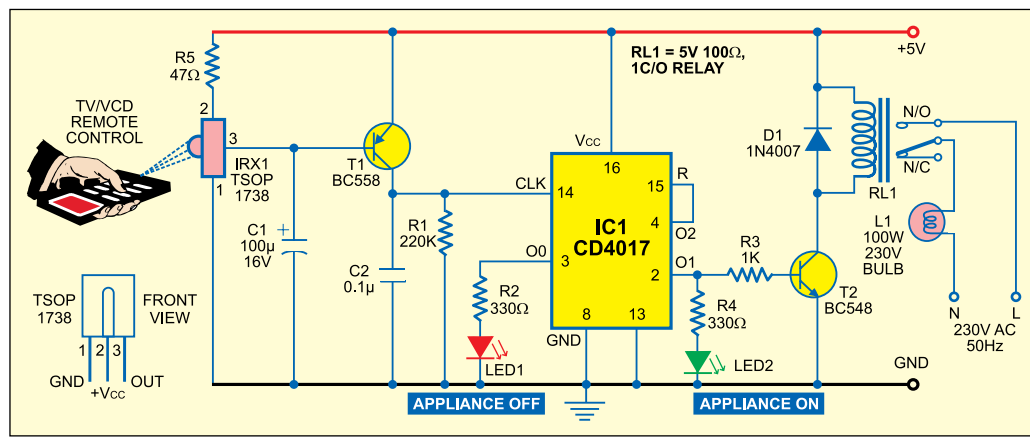
■ **S. MOHAN**

Connect this circuit to any of your home appliances (lamp, fan, radio, etc) to make the appliance turn on/off from a TV, VCD or DVD remote control. The circuit can be activated from up to 10 metres.

The 38kHz infrared (IR) rays generated by the remote control are received by IR receiver module TSOP1738 of the circuit. Pin 1 of TSOP1738 is connected to ground, pin 2 is connected to the power supply through resistor R5 and the output is taken from pin 3. The output signal is amplified by transistor T1 (BC558).

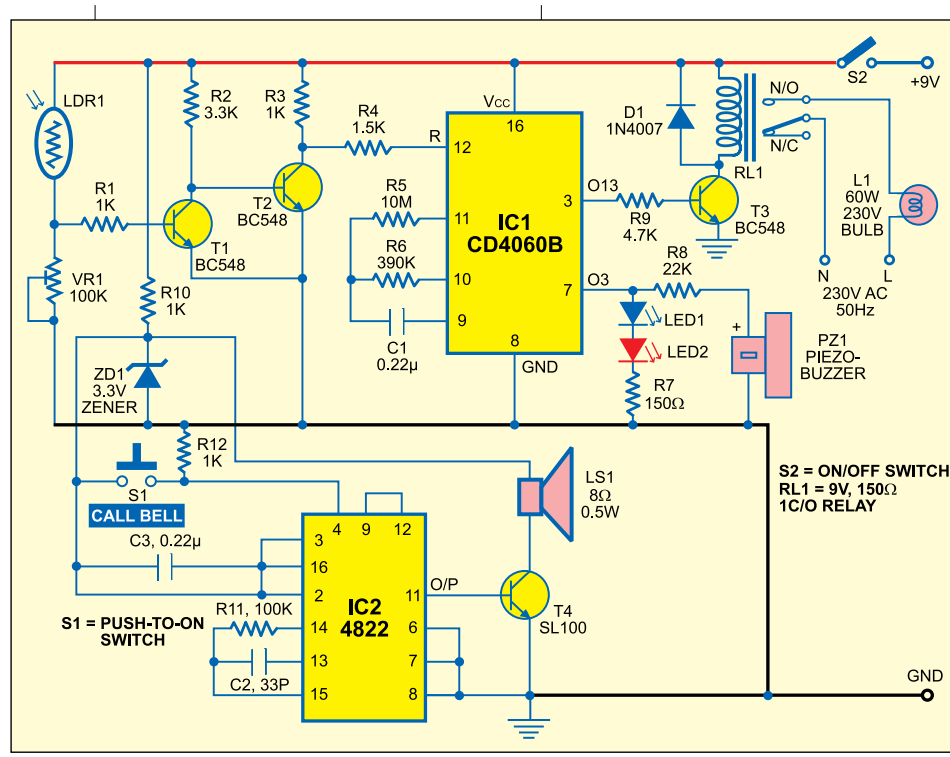
The amplified signal is fed to clock pin 14 of decade counter IC CD4017 (IC1). Pin 8 of IC1 is grounded, pin 16 is connected to Vcc and pin 3 is connected to LED1 (red), which glows to indicate that the appliance is 'off.'

The output of IC1 is taken from its pin 2. LED2 (green) connected to pin 2 is used to indicate the 'on' state of the appliance. Transistor T2 (BC548) connected to pin 2 of IC1 drives relay RL1. Diode 1N4007 (D1) acts as a freewheeling diode. The appliance to be controlled is connected between the pole of the relay and neutral terminal of mains. It gets connected to live terminal of AC mains via normally opened (N/O) contact when the relay energises. ●



MOCK ALARM WITH CALL BELL

■ D. MOHAN KUMAR



Here is a fully automatic mock alarm to ward off any intruder to your house. The alarm becomes active at sunset and remains 'on' till morning. The flashing light-emit ting diodes (LEDs) and beeps from the unit simulate the functioning of a sophisticated alarm system. Besides, the circuit turns on and off a lamp regularly at an interval of 30 minutes throughout the night. It also has a call bell facility.

The circuit is built around CMOS IC CD4060B (IC1), which has an internal oscillator and a 14-stage binary divider to provide a long delay without using a high-value resistor and capacitor.

Press switch S2 to provide 9V power supply to the circuit. During daytime, light-dependent resistor LDR1 offers little resistance and transistor T1 conducts. This drives transistor T2 into the cut-off mode, as its base is pulled to ground via transistor T1. Reset pin 12 of IC1 remains high as long as transistor T2 is cut off. This keeps the oscillator of IC1 (comprising resistors R5 and R6 and capacitor C1) disabled and its outputs remain low. The sensitivity of LDR1 can be adjusted using preset VR1.

When the sunlight decreases in the evening, the resistance of LDR1 increases to cut off transistor T1. This drives transistor T2 into conduction mode and its collector voltage goes low. At the same time, reset pin 12 of IC1 goes low to enable the oscillator of IC1 and the oscillator starts oscillating. The O3 output (pin 7) of IC1 goes high every five seconds to light up the LEDs (LED1 and LED2) and activate the buzzer. Resistor R8 limits the tone produced from the buzzer.

At the same time, O13 output of IC1 (pin 3) goes high every 30 minutes to forward bias transistor T3 to energise relay RL1 and lamp L1 connected to the normally opened (N/O) contacts of relay RL1 glows. This cycle repeats till morning.

The call bell is built around IC 4822 (IC2). Its inbuilt musical tone generator generates different tones at each trigger. The frequency of the tone can be controlled through external components R11 and C2. The output at pin 11 of IC2 is amplified by transistor T4.

When push-to-on switch S1 is pressed once, trigger pin 4 of IC2 gets a positive trigger from the positive rail (reduced by zener diode to 3.3V) via resistor R10 and IC2 starts producing a melody. Resistor R10 limits the current to the trigger pin of IC2 and resistor R12 prevents any false triggering. Zener diode ZD1 provides the 3.3V required for IC 4822.

The circuit works off 9V regulated power supply. Assemble the circuit on any general-purpose PCB and enclose it in a waterproof plastic box with holes for mounting LEDs on the rear and the LDR on the top of the box. Place the LDR such that sunlight falls on it directly. Mount the unit on the pillar of the entrance gate. To avoid unnecessary illumination of the LDR, install lamp L1 away from the unit in the porch of the house. Keep the speaker inside the room. ●

POWER-SAVER LED LAMP

■ K.N. GHOSH

This high-intensity, energy-efficient, long-lasting and durable lamp can withstand input voltage fluctuation of up to ±25 per cent without change in the light output. The lamp consumes only 0.5W power compared to conventional 15W lamps, significantly reducing the energy costs, and can be used as night lamp, path lamp or *mandir* lamp. Also, it has a life of 100,000 hours (11 years of continuous use) against 1000 hours for conventional lamps, thus requiring no replacement for a long time, once fitted.

At the heart of this lamp are seven light-emitting diodes (LEDs). Unlike ordinary incandescent lamps, LEDs don't have a filament that can burn out and are illuminated solely by the movement of electrons in a semiconductor material. So they last much longer. Small size and use of plastic material make them more durable.

But the main advantage of LEDs is efficiency. In traditional incandescent lamps, light is produced by heating the tungsten filament. This results in wastage of energy, as a huge portion of the available electricity isn't used for producing the visible light. LEDs generate little heat. A much higher percentage of the electrical power goes directly to light generation, which cuts down the electricity bill considerably.

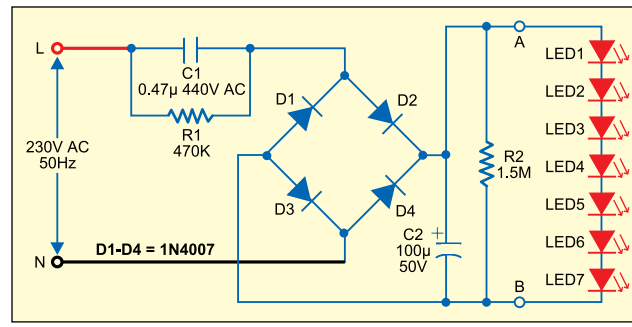


Fig. 1: Circuit diagram of power-saver LED lamps

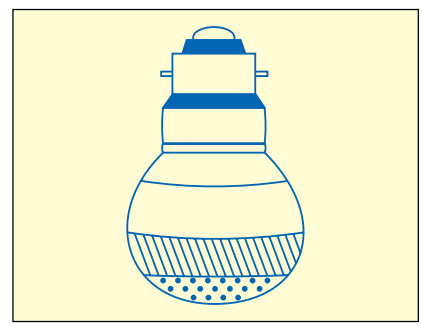


Fig. 2: Proposed enclosure for the lamp

Fig. 1 shows the circuit for red LED-based lamp. The LEDs (LED1 through LED7) are powered from mains without the use of a transformer. Here, capacitor C1 is used as the 'AC voltage dropper'—the well-known transformerless solution. It results in the advantages of a smaller size of the circuit and no heat generation (as the capacitor dissipates negligible power).

The forward-conduction voltage drop required for the LED chain (LED1 through LED7) is provided by C1 alone. C1 discharges through R1 immediately after the circuit is disconnected from mains, which prevents a fatal shock due to any voltage remaining on the input terminals. The AC mains voltage is rectified by diodes D1 through D4 and filtered by capacitor C2. Resistor R2 acts as a bleeder.

You can make lamps in other colours as well by simply disconnecting all the red LEDs (LED1 through LED7) between points A and B and instead connecting yellow LEDs for yellow lamp, blue LEDs for blue lamp, green LEDs for green lamp and white LEDs for white lamp, as desired.

Capacitor C1 should be rated for at least 440V AC, while mains applications also require use of an X2-class capacitor. The circuit may be assembled on a circular PCB and housed in a bulb-shaped enclosure as shown in Fig. 2. Mount capacitors C1 and C2 on the track side of the PCB to save space. ●

MAINS SUPPLY FAILURE ALARM

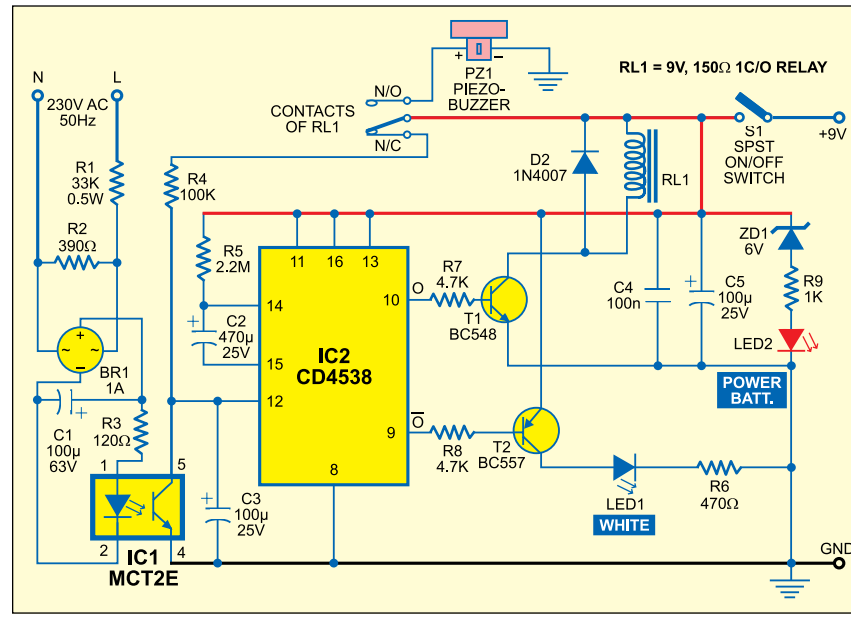
■ T.K. HAREENDRAN

Whenever AC mains supply fails, this circuit alerts you by sounding an alarm. It also provides a backup light to help you find your way to the torch or the generator key in the dark.

The circuit is powered directly by a 9V PP3/6F22 compact battery. Pressing of switch S1 provides the 9V power supply to the circuit. A red LED (LED2), in conjunction with zener diode ZD1 (6V), is used to indicate the battery power level. Resistor R9 limits the operating current (and hence the brightness) of LED2.

When the battery voltage is 9V, LED2 glows with full intensity. As the battery voltage goes below 8V, the intensity of LED2 decreases and it glows very dimly. LED2 goes off when the battery voltage goes below 7.5V.

Initially, in standby state, both the LEDs are off and the buzzer does not sound. The 230V AC mains is directly fed to mains-voltage detection optocoupler IC MCT2E (IC1) via resistors R1, R2 and R3, bridge rectifier BR1 and capacitor C1. Illumination of the LED inside optocoupler IC1 activates its internal phototransistor and clock input pin 12 of IC2 (connected to 9V via N/C contact of relay RL1) is pulled low. Note that only one monostable of dual-monostable multivibrator IC CD4538 (IC2) is used here.



When mains goes off, IC2 is triggered after a short duration determined by components C1, R4 and C3. Output pin 10 of IC2 goes high to forward bias relay driver transistor T1 via resistor R7. Relay RL1 energises to activate the piezobuzzer via its N/O contact for the time-out period of the monostable multivibrator (approximately 17 minutes). At the same time, the N/C contact removes the positive supply to resistor R4. The time-out period of the monostable multivibrator is determined by R5 and C2.

Simultaneously, output pin 9 of IC2 goes low and pnp transistor T2 gets forward biased to light up the white LED (LED1). Light provided by this back-up LED is sufficient to search the torch or generator key.

During the mono time-out period, the circuit can be switched off by opening switch S1. The 'on' period of the monostable multivibrator may be changed by changing the value of resistor R5 or capacitor C2.

If mains doesn't resume when the 'on' period of the monostable lapses, the timer is retriggered after a short delay determined by resistor R4 and C3. ●

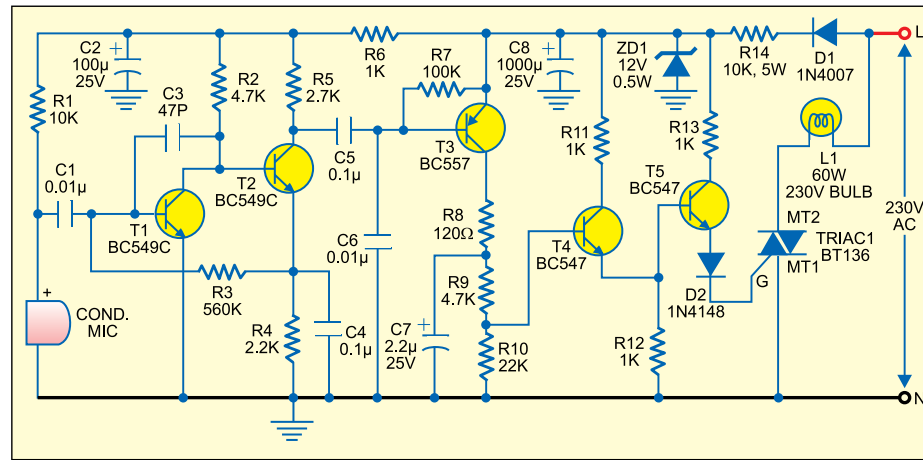
SOUND-OPERATED SWITCH FOR LAMPS

■ **PRADEEP G.**

This inexpensive, fully transistorised switch is very sensitive to sound signals and turns on a lamp when you clap within 1.5 metres of the switch. One of its interesting applications is in discotheques, where lights could be turned on or off in sync with the music beats or clapping.

The condenser microphone senses the sound and converts it into electrical variations. The electrical signals are amplified by the two-stage direct-coupled (DC) amplifier formed by transistors T1 and T2 and fed to the switching circuit. The switching circuit comprises transistors T3, T4 and T5, which conduct only when the circuit senses sound signals. Transistor T5 supplies sufficient gate voltage to the triac to drive the 230V lamp.

The regulated 12V DC power supply for the circuit is derived from AC mains by using resistor R14, diode D1 and zener diode ZD1. The circuit can be assembled on any general-purpose PCB. ●



TV PATTERN GENERATOR

■ **S. ANANTHA NARAYANAN**

This single-IC TV pattern generator is useful for fault finding in TV sets. You can correct the alignment of the timing circuits of the TV set with the help of this circuit. The vertical stripes (bars) produced by the pattern generator on the TV screen help you align the vertical scanning synchronisation circuit of the receiver.

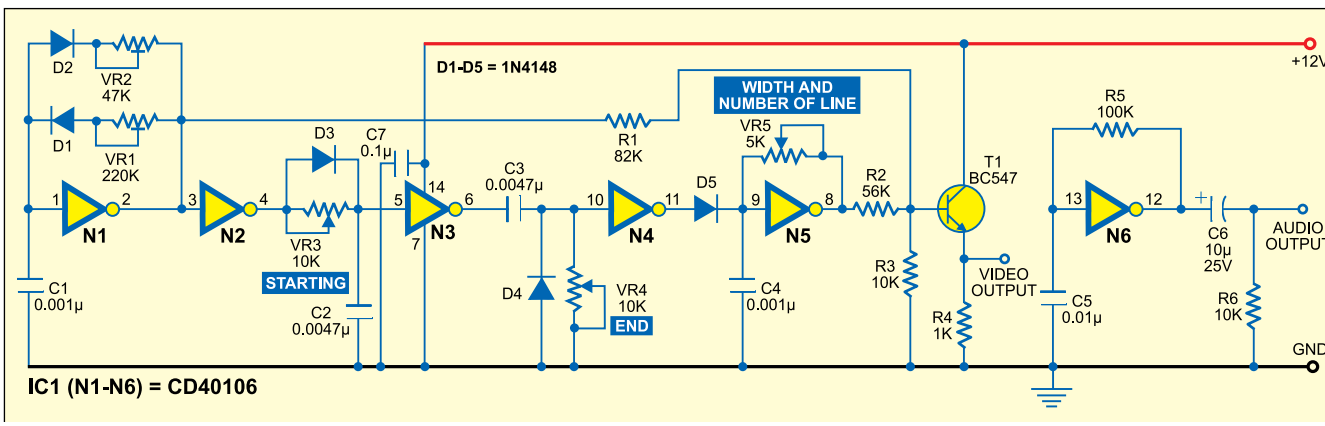
To test the TV set, you need to connect the video and audio outputs of the circuit to the respective inputs of the TV set one by one. If the video section of the TV set is working the circuit generates vertical white lines on the TV screen, and if the audio section is working you hear sound from the TV's speakers. You can also adjust the width of vertical lines.

The circuit uses hex Schmitt inverter IC CD40106 (IC1). NOT gate N1 generates horizontally synchronised (Hsync) pulses for the PAL video signal. Presets VR1 and VR2 are used to control the 'on' and 'off' time durations of the oscillator, respectively. For PAL, you need to adjust VR2 for 'off' duration of 4.7 µs, while VR1 needs to be adjusted for 'on' duration of around 60 µs.

If vertical lines appear on the TV screen on connecting the video output of the circuit to the video input of the TV, the video section of the TV set is working. You can control the starting position of the lines using potmeter VR3, the end position of the lines using potmeter VR4, and the line width and the number of lines using potmeter VR5.

If you don't have an oscilloscope, set presets VR1 and VR2 to 150k and 22k, respectively, to get the required 'on' and 'off' periods for the oscillator and see the vertical line pattern on the TV.

The audio frequency oscillator is built around NOT gate N6. Its oscillation frequency is decided by resistor R6 and capacitor C5. Connect the audio output of the circuit to the audio input of the TV. If you hear sound from the TV's speakers, the audio section of the TV set is working. ●



RECHARGEABLE TORCH BASED ON WHITE LED

■ T.A. BABU

Rechargeable torches don't come without problems. You need to replace the bulbs and charge the batteries frequently. The average incandescent light-emitting diode (LED) based torch, for instance, consumes around 2 watts. Here's a rechargeable white LED-based torch that consumes just 300 mW and has 60 per cent longer service life than an average incandescent torch.

Fig. 1 shows the circuit of the rechargeable white LED-based torch. The reactive impedance of capacitors C1 through C3 (rated for 250V AC) limits the current to the charger circuit. The resistor across the capacitors provides a discharge path for the capacitors after the battery is charged. The red LED1 indicates that the circuit is active for charging.

The torch uses three NiMH rechargeable button cells, each of 1.2V, 225 mAH. A normal recharge will take at least 12 hours. Each full recharge will give a continuous operational time of approximately 2.5 hours. Recharge the battery to full capacity immediately after use to ensure its reliability and durability. The charging current is around 25 mA.

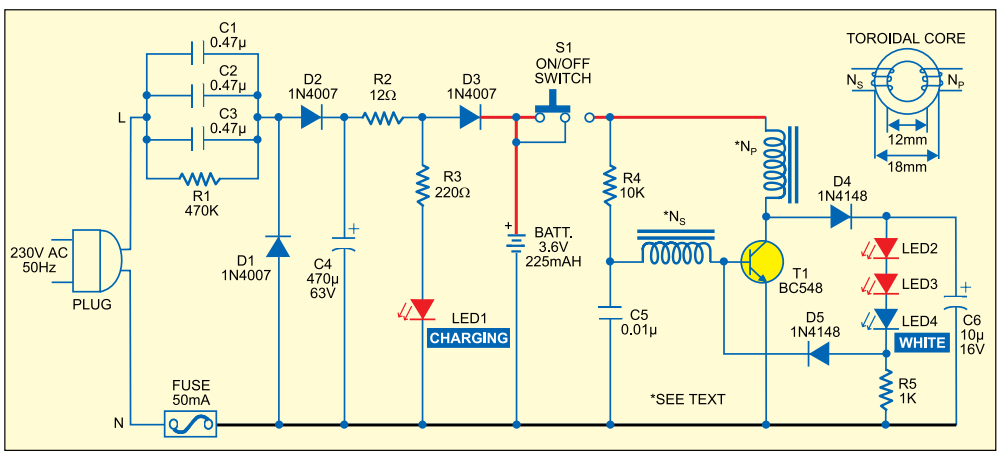


Fig. 1: Circuit diagram of rechargeable torch

A voltage booster circuit is required for powering the white LEDs (LED2 through LED4). An inverter circuit is used to achieve voltage boosting. Winding details of the inverter transformer using an insulated ferrite toroidal core is given in the schematic. The number of 35 SWG wire turns in the primary and secondary coils (N_P and N_S) are 30 and 3, respectively. If the inverter does not oscillate, swap the polarity of either (but not both) the primary or the secondary winding. A reference voltage from resistor R5 provides a reflected biasing to the transistor, and keeps the output constant and regulated.

The suggested enclosure for the torch is shown in Fig. 2. ●

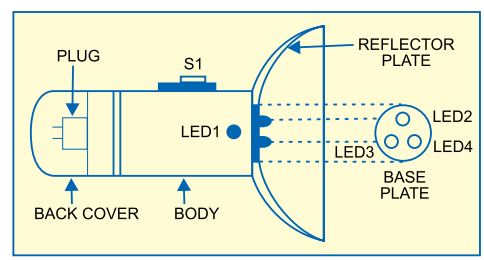


Fig. 2: Suggested enclosure for the torch

16-WAY CLAP-OPERATED SWITCH

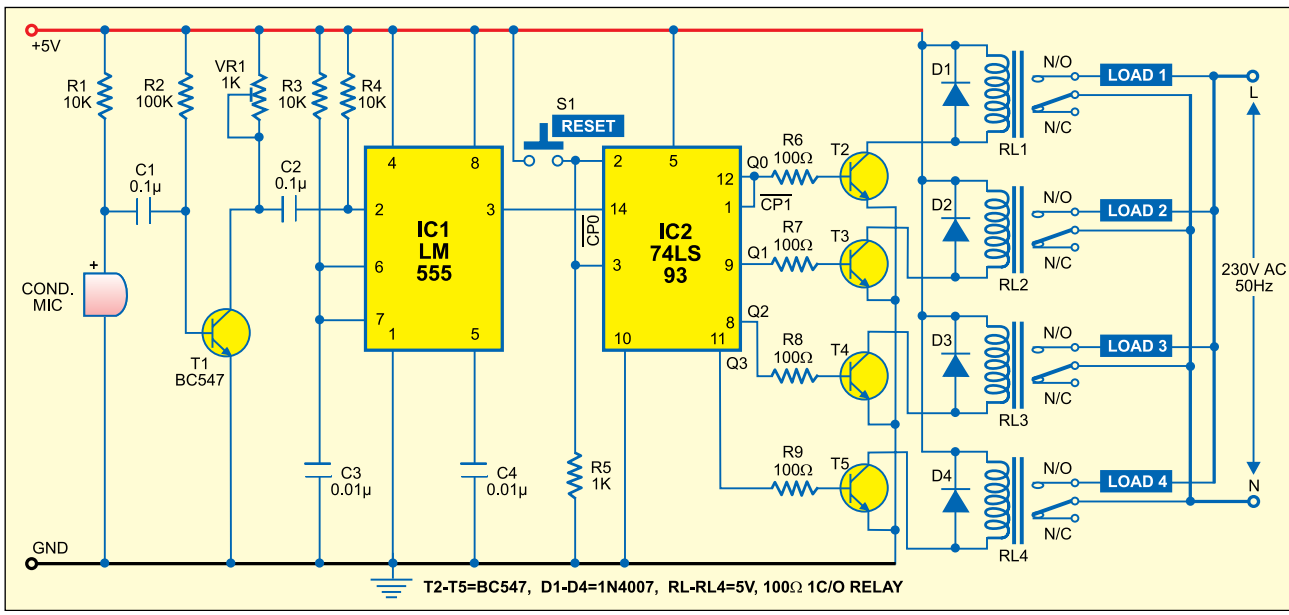
■ **RAJ K. GORKHALI**

Control your home appliances without getting out of your bed. You just have to clap in the vicinity of the microphone used in this circuit, which you can keep by the bedside. You can switch on/off up to four different electrical equipment (TV, fan, light, etc) in 16 different ways.

This circuit is built around timer IC 555 (IC1), CMOS IC 74LS93 (IC2) and five BC547 npn transistors (T1, T2, T3, T4 and T5). Transistor T1 is used as the preamplifier and the rest are used for driving the relays.

A small condenser microphone is connected at the base of transistor T1, which is biased from resistor R1 (10 kilo-ohms). The clapping sound is converted into electrical energy by the microphone and amplified by transistor T1. The transistor output is fed to the monostable circuit wired around IC 555. Output pin 3 of the timer is connected to the clock input of divide-by-16 IC 74LS93.

The outputs of IC2 are fed to npn transistors T2, T3, T4 and T5 via 100-ohm resistors to drive relays RL1, RL2, RL3 and RL4 connected to appliances 1 though 4, respectively. Freewheeling diodes D1 through D4 connected across the relays protect the transistors from the back electromagnetic field (e.m.f.) produced by the relays.



Output of 74LS93

Number of claps	Q0	Q1	Q2	Q3
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	1	1	0	0
4	0	0	1	0
5	1	0	1	0
6	0	1	1	0
7	1	1	1	0
8	1	0	0	1
9	1	0	0	1
10	0	1	0	1
11	1	1	0	1
12	0	0	1	1
13	1	0	1	1
14	0	1	1	1
15	1	1	1	1

Note:1. '1' indicates high logic.'0' indicates low logic.
2. At high logic, the corresponding transistor conducts to energise the corresponding relays and activate the load.

The output states of IC 74LS93 (Q0 through Q3) for different numbers of claps are shown in the table.

The circuit is powered from regulated 5V DC. For testing the circuit, disconnect the resistors from the outputs of IC2 and connect four LEDs in series with 220-ohm resistors between the outputs and ground. Now switch on the power supply and clap near the microphone. You can see the four LEDs glowing in the manner shown in the table. A reset push switch is provided to switch off all the 'on' devices.

Now you can connect the desired appliances to the relays and control them with your claps. ●

BRAKE FAILURE INDICATOR

■ **D. MOHAN KUMAR**

Do you want to get an early warning of brake failure while driving? Here is a brake failure indicator circuit that constantly monitors the condition of the brake and gives an audio-visual indication. When the brake is applied, the green LED blinks and the piezobuzzer beeps for around one second if the brake system is intact. If the brake fails, the red LED glows and the buzzer stops beeping.

The circuit will work only in vehicles with negative grounding. It also gives an indication of brake switch failure.

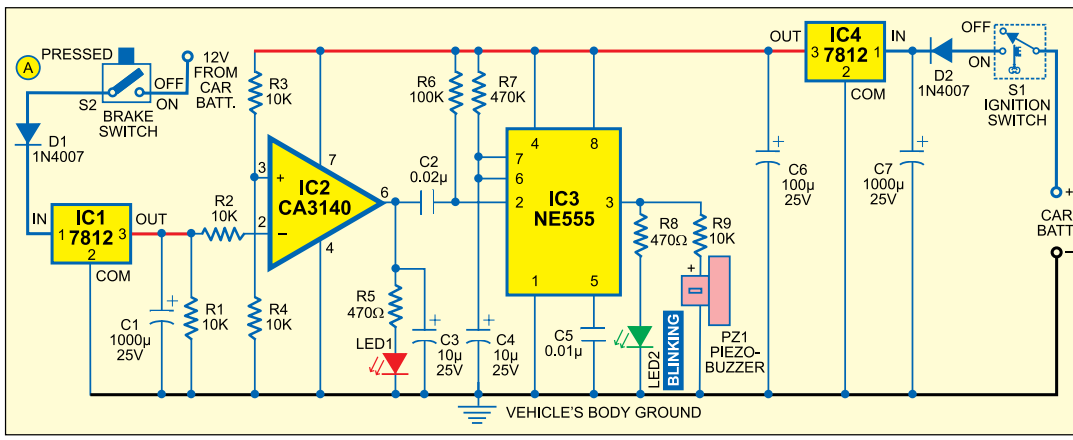
In hydraulic brake systems of vehicles, a brake switch is mounted on the brake cylinder to operate the rear brake lamps. The brake switch is fluid-operated and doesn't function if the fluid pressure drops due to leakage. The fluid leakage cannot be detected easily unless there is a severe pressure drop in the brake pedal. This circuit senses the chance of a brake failure by monitoring the brake switch and reminds you of the condition of the brake every time the brake is applied.

The circuit uses an op-amp IC CA3140 (IC2) as voltage comparator and timer NE555 (IC3) in monostable configuration for alarm. Voltage comparator IC2 senses the voltage level across the brake switch. Its non-inverting input (pin 3) gets half the supply voltage through potential divider resistors R3 and R4 of 10 kilo-ohms each. The inverting input (pin 2) of IC2 is connected to the brake switch through diode D1, IC 7812 (IC1) and resistor R2. It receives a higher voltage when the brake is applied.

Normally, when the brake is not applied, the output of IC2 remains high and the red LED (LED1) glows. The output of IC2 is fed to trigger pin 2 of the monostable through coupling capacitor C2. Resistor R1 is used for the input stability of IC2. IC1 and C1 provide a ripple-free regulated supply to the inverting input of IC2.

IC3 is wired as a monostable to give pulse output of one second. Timing elements R7 and C4 make the output high for one second to activate the buzzer and LED2. Usually, the trigger pin of IC3 is high due to R6 and the buzzer and LED2 remain 'off.'

When the brake pedal is pressed, pin 2 of IC2 gets a higher voltage from the brake switch and its output goes low to switch off the red LED. The low output of IC2 gives a short negative pulse to the monostable through C2 to trigger it. This activates the buzzer and LED2 to indicate that the brake system is working. When there is pressure drop in the brake system due to leakage, LED1 remains 'on' and the buzzer does not sound when the brake is applied.



The circuit can be assembled on any general-purpose PCB or perforated board. Connect point A to that terminal of the brake switch which goes to the brake lamps. The circuit can be powered from the vehicle's battery.

The circuit requires well-regulated power supply to avoid unwanted triggering while the battery is charging from the dynamo. IC4, C6 and C7 provide regulated 12V to the circuit. The power supply should be taken from the ignition switch and the circuit ground should be clamped to the vehicle's body. A bicolour LED can be used in place of LED1 and LED2 if desired. ●

BATTERY CHARGER WITH AUTOMATIC SWITCH-OFF

■ V. GOPALAKRISHNAN

This smart charger automatically switches off when your re-chargeable batteries reach the full charge.

The circuit comprises a bistable multivibrator wired around timer IC 555. The bistable output is fed to an ammeter (via diode D1) and potmeter VR1 before it goes to three Ni-Cd batteries that are to be charged.

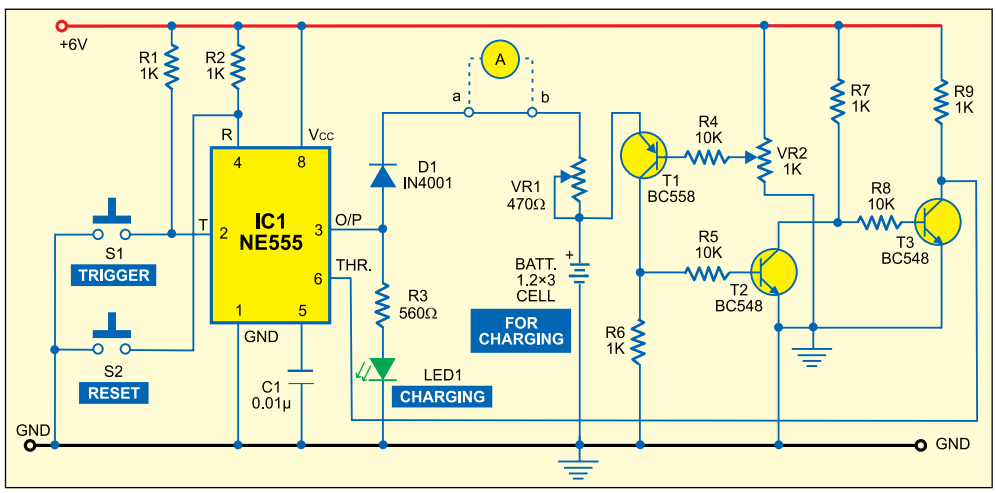
Normally, the full charge potential of an Ni-Cd cell is 1.2V. Trigger the bistable by pressing switch S1 and adjust potmeter VR1 for 60mA current through the ammeter.

Now remove the ammeter and connect a jumper wire between its points 'a' and 'b.' Connect the positive output terminal of the batteries to the emitter of pnp transistor T1. The base of transistor T1 is held at 2.9V by adjusting potmeter VR2. The output of transistor T1 is inverted twice by npn transistors T2 and T3.

Thus when the batteries are fully charged to 3×1.2V=3.6V, a voltage higher than this makes transistor T1 to conduct. Transistor T2 also conducts and transistor T3 goes off. The threshold level of timer 555 reaches 6V, which is more than $2/3 \times V_{CC} = 2/3 \times 6 = 4V$, to turn off the timer.

During charging, the threshold level of the timer is held low. The green LED (LED1) glows during charging of the batteries and goes off at the attainment of full charge.

Note that this circuit can be used only for 1.2V, 600mAH Ni-Cd rechargeable batteries that require 60 mA of current for 15 hours to charge fully. ●



MULTIDOOR OPENING ALARM WITH INDICATOR

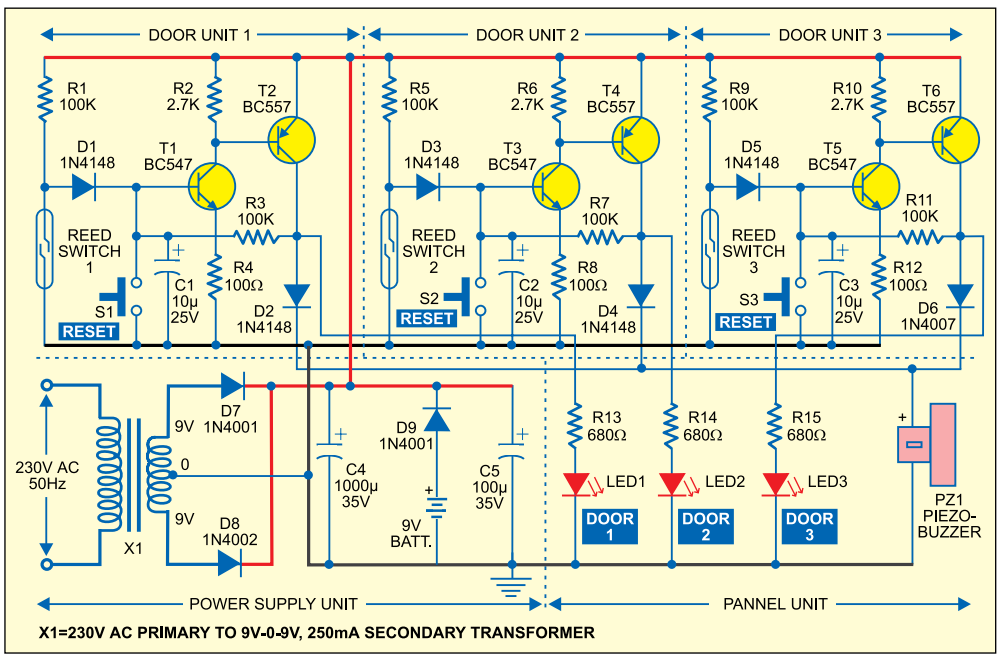
■ **PRADEEP G.**

This door-opening alarm alerts you of intruders. You can use it for up to three doors.

You simply need to fit a small unit including reed switch on each door-frame and fix a magnet on the moving door such that the magnet aligns with the reed switch when the door is closed. A separate unit incorporating the power supply, three LEDs and a buzzer is to be kept by your side inside your room. A three-core ribbon cable from this unit goes to each door unit. One core goes to the positive terminal, second to the negative terminal and third to the output of the unit.

When the door is closed, the reed switch terminals are shorted and the alarm does not sound.

When door 1 is opened by someone, transistors in the corresponding door unit conduct and the buzzer sounds. LED1 glows to indicate opening of door 1. Due to diode latching action, the alarm will sound continuously even after the door is closed. It can be stopped only by pressing the reset switch of the door unit.



X1=230V AC PRIMARY TO 9V-0-9V, 250mA SECONDARY TRANSFORMER

Regulated 9V to 12V DC for operating the circuit is derived from AC mains and fed to the three units mounted on the doors. Battery-backup is also provided. When all the three doors are simultaneously opened, all the three LEDs will glow.

This arrangement can be extended for more doors by increasing the number of door units connected to the audio-visual indication unit. ●

SAFETY GUARD

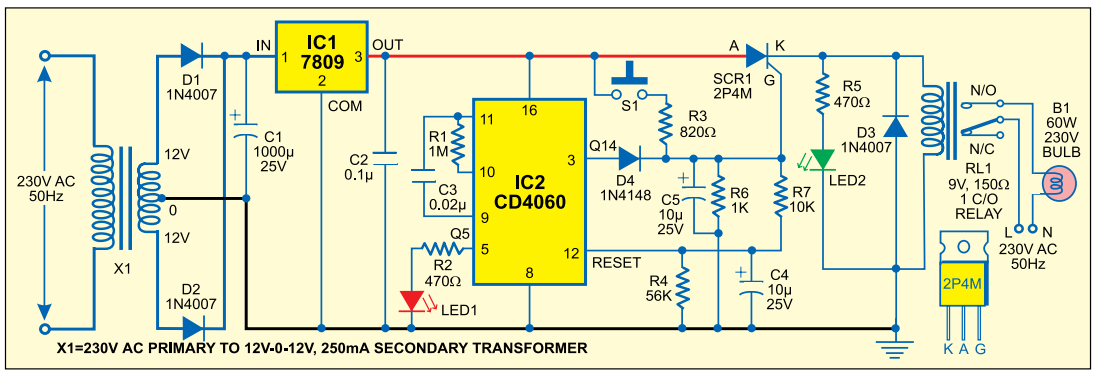
■ A. RAMESH BABU

Protect your home appliances from voltage spikes with this simple time delay circuit.

At the heart of the circuit is IC CD4060, which consists of two inverter gates for clock generation and a 14-bit binary ripple counter. Here the clock oscillations are governed by resistor R1 and capacitor C1. In this circuit, only two outputs of the IC (Q5 and Q14) have been used. Q5 is connected to an LED (LED1) and Q14 is used to trigger the gate of the SCR through D4 as well as reset the counter.

The anode of the SCR is connected to +9V and the cathode is connected to the relay coil. The other pin of the relay coil is connected to the negative supply, while its contacts are used for switching on the appliances.

Whenever power to the appliances is switched on or resumes after mains failure, the oscillator starts oscillating and LED1 blinks. This continues for three minutes. After that, Q14 output of IC CD4060 goes high to trigger the gate of the SCR through D4. At this moment, the voltage is available at the cathode of the SCR, which energises the relay coil to activate the appliance and LED2 glows. Switch S1 is used for quick start without waiting for delay. ●



WHITE LED-BASED EMERGENCY LAMP AND TURNING INDICATOR

■ **ANANTHA KESHAVA I. AND SHIREEN M. BARETTO**

White LEDs are replacing the conventional incandescent and fluorescent bulbs due to their high power efficiency and low operating voltage. These can be utilised optimally for emergency lamp and vehicle turning indication. The circuits for the purpose are given here.

Fig. 1 shows the circuit of a white-LED based emergency lamp. You can also use arrays of white LEDs as day-time running lamps in automobiles.

In the emergency lamp, seven 1.2V AA-size Ni-Cd cells giving 8.4V have been used as the power source. The brightness is controlled by duty-cycle variation of an astable multivibrator working at 1 kHz. The astable multivibrator is built around IC1. Its output is connected to LED-driver transistor T1.

Up to six branches of white LEDs can be connected in parallel, with each branch containing two LEDs in series (only three branches are used here). Depending on the application, different combinations of battery voltages and the number of LEDs in series can be made such as to keep the resistive losses low.

The charger circuit for a Ni-Cd battery is shown in Fig. 2. When the battery voltage is less than 9.8V, charging takes place since the voltage at the emitter of transistor T2 (V_E) is 9.8V. The value of resistor R8 is chosen such that the battery charges at a rate of 70 mA per hour. The full charge voltage of the battery is 9.8V. When the battery reaches full voltage, the current reduces to approach the tickle charge value of few milliamperes.

Assemble both the circuits shown in Figs 1 and 2 on a general-purpose PCB. LEDs can also be mounted on the reflector of a lamp. After assembling, connect points A and GND of the emergency lamp circuit to the respective points of the battery charger circuit. Now your emergency lamp is ready to work.

To use the emergency lamp, switch on the circuit using switch S1. All the LEDs (LED1 through LED6) will glow to provide sufficient light.

Turning indicator shown in Fig. 3 is another application of the LEDs. It can be used for two-wheelers and draws limited power from the dynamo/battery. At low revolutions, headlight dims because of the increase in load. The white LED-based turning indicator circuit draws a fraction of the power drawn by conventional bulbs, and may last longer than the vehicle itself.

The circuit comprises two identical sections for left and right turn indications. The right turn indicator circuit is built around transistors T3 through T5 and white/yellow LEDs (LED8 through LED13). Similarly, the left turn indicator circuit is built around transistors T4, T6 and T7 and white/yellow LEDs (LED15 through LED20). Transistor T4 and the piezobuzzer are common for both-side indicators.

When you slide switch S2 towards

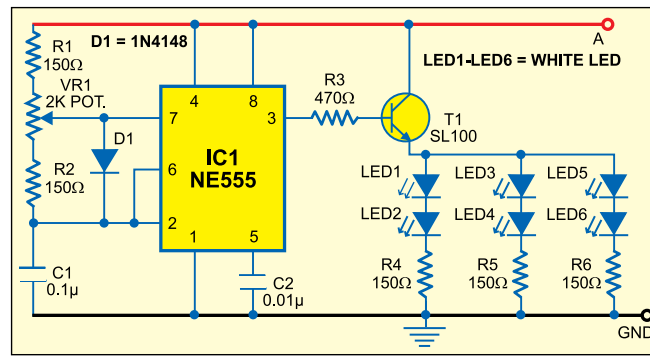


Fig. 1: White LED based emergency lamp

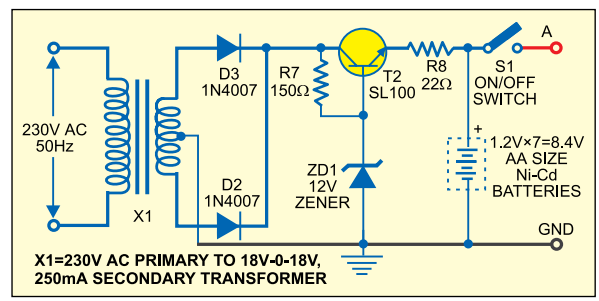


Fig. 2: Battery charger

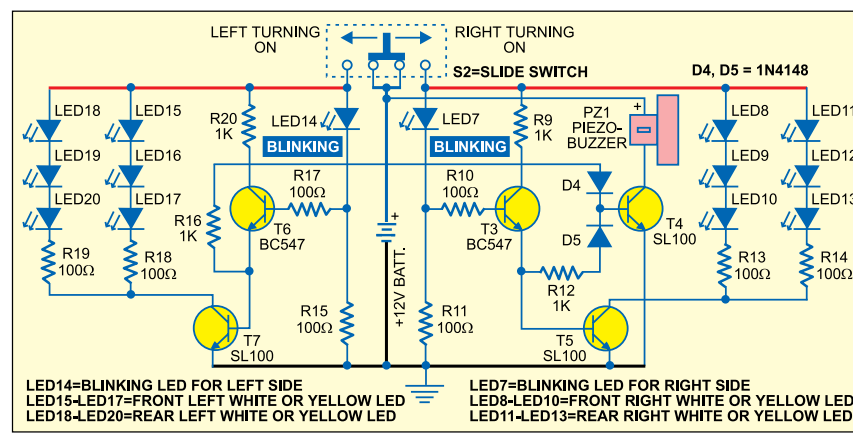


Fig. 3: Turning indicator

right, blinking LED7, right-front LEDs (LED8 through LED10) and rear LEDs (LED11 through LED13) start blinking. Similarly, when you slide switch S2 towards left, blinking LED14, left-front LEDs (LED15 through LED17) and rear LEDs (LED18 through LED20) start blinking.

Transistor T3 acts as the buffer, while transistor T4 drives the buzzer. Transistors T5 and T7 drive the LEDs.

The LED array can be built using white LEDs or yellow LEDs depending on the colour of the indicator's cover. In case you use yellow LEDs, keep in mind that the forward drop voltage is around 1.8V for a single yellow LED and therefore the value of the resistance should be changed in accordance with the increase in the number of LEDs in series.

Three white LEDs produce the light intensity of six yellow LEDs. ●

INEXPENSIVE CAR PROTECTION UNIT

■ **M. VENKATESWARAN AND T.E. PARTHASARATHY**

For car protection, custom-made units are available but they are costly.

Here's a circuit to protect car stereo, etc from pilferage that costs less and requires no adjustments in the car but a good car cover.

Place the circuit at your bedside and bring the two wires from the unit to the car (parked outside your home) and connect one wire-end to the cover and the other to the ground, with both wire-ends shorted by some weight such as a brick. So outwardly the mechanism is not visible.

Normally, the protector operates off AC mains and the battery takes over only when mains fail. As the battery current is not high, the battery will last long.

As long as the two wires remain shorted, transistor T1 remains cut off. When shorting is removed, transistor T1 gets forward biased and its collector voltage drops to trigger IC2 and the piezobuzzer starts sounding.

If mains fails, the battery-takeover indicator (shown in Fig. 2 and connected to points A, B and C in Fig. 1) immediately gets triggered at pin 2 of IC3. Its high output activates the battery-operation alarm for a couple of seconds. IC1 draws power from the battery to activate the protection unit.

After setting up the unit properly and shorting both the wires, press test switch S2. If there is no fault in the circuit, the alarm will sound. Now release test switch S2 and momentarily press reset switch S1 to switch off the alarm. ●

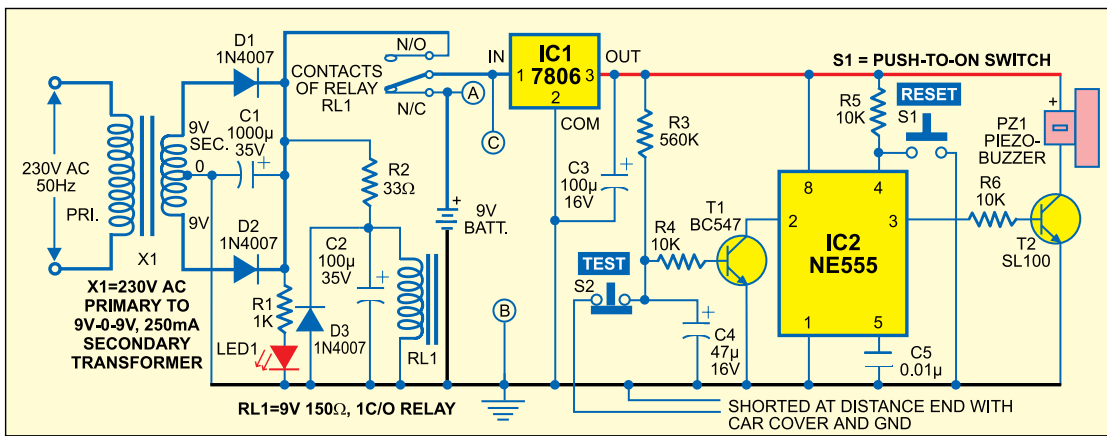


Fig. 1: Circuit of car protection unit with alarm

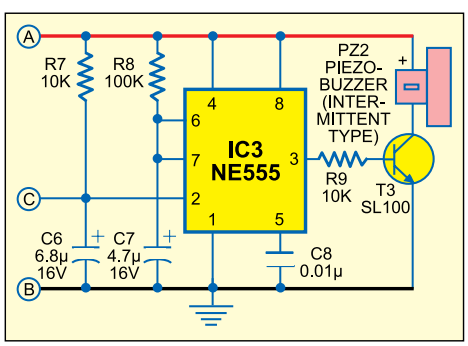


Fig. 2: Battery takeover indicator

If someone tries to remove the cover, the alarm of the circuit starts sounding to alert you. The alarm can be switched off by resetting it using switch S1.

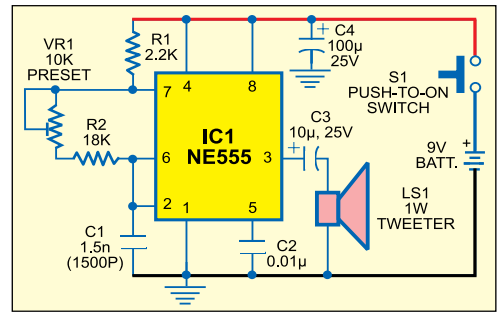
The car protection circuit comprises two timer ICs: one for the alarm circuit (see IC2 in Fig.1) and the other to indicate that the battery has taken over as the power source (see IC3 in Fig. 2).

DOG CALLER

■ **PRADEEP G.**

Dog trainers use a whistle to call dogs. But why blow that irritating, loud whistle when the dog can hear a sound inaudible to the humans? We the humans can hear up to 20 kHz, but dogs can hear ultrasound (sound ranging between 20 and 30 kHz) also.

Here's a circuit that generates 21 to 22 kHz (frequencies just above the audible range), so it can be used to call your pets by generating ultrasonic sound.



IC 555 is used as an oscillator. By adjusting the preset, ultrasonic sound of 21-22kHz frequency can be generated. Whistle effectiveness depends on the speaker used. Use of a low-wattage tweeter is recommended. (Don't use an ultrasonic transducer, because it is designed for 40 kHz only.)

The circuit works off 9V. For portability, use a 9V PP3 battery and house the unit inside a pocket radio cabinet. ●

SMART CELLPHONE HOLDER

■ T.K. HAREENDRAN

This smart cellphone holder makes sure that you don't forget to carry your mobile phone. Fitted in the car, it keeps searching for the mobile phone within the holder using infrared (IR) rays and alerts you through a flashing LED when it doesn't find one. You can attach the circuit to your existing cellphone holder or, with a little skill, construct one as per your requirement.

The circuit, wired around IC LM555 (IC1), derives power from the 12V DC automobile battery. Diode D1 is an accidental wrong-polarity input guard. Resistor R7 limits the inrush current to IC1.

When power is applied to the circuit, the low-frequency astable multivibrator built around IC1 is activated and LED2 at its output pin 3 flashes briefly.

When ignition switch S2 is flipped to 'on' position, the +12V DC from the car's battery disables the astable multivibrator via diode D2 and LED2 turns off.

When the ignition is turned off and the mobile phone is in its holder, LED2 again starts blinking. In case the cellphone holder is empty, IR rays from IR LED1 fall on phototransistor T1 and it conducts to pull the base of LED driver transistor T2 towards ground to disable the visual indicator (LED2).

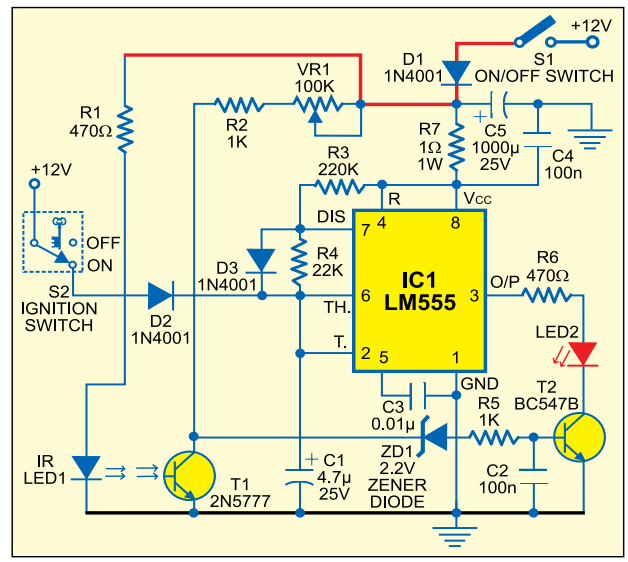


Fig. 1: Circuit of smart cellphone holder

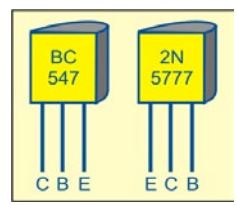


Fig. 2: Pin configurations of BC547 and 2N5777

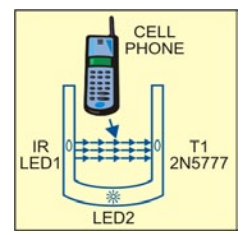


Fig. 3: Proposed cellphone holder

If you've forgotten to carry your cellphone, LED2 fitted in the cellphone holder will stop flashing to indicate that the mobile phone is not in the cell holder of the case. Resistor R1 limits the current flowing through IR LED1 and resistor R6 limits the operating current and hence luminance of LED2. Variable resistor VR1 determines the detection sensitivity of phototransistor T1. The blinking rate of LED2 can be changed by changing the value of capacitor C1 (or R3-R4 resistor combination).

Pin configurations of BC547 and phototransistor 2N5777, and the proposed cellphone holder are shown in Figs 2 and 3, respectively. ●

IC 555 TIMER TESTER

■ **RAJ K. GORKHALI**

This simple and easy-to-use gadget not only tests the IC 555 timer in all its basic configurations but also tests the functionality of each pin of the timer. Once a timer is declared fit by this gadget, it will function satisfactorily in whatever mode or configuration you may try it.

The two basic configurations in which a timer IC 555 can be used are the astable and the monostable modes of operation.

When the DPDT switch (S2) is in position 1-1, the timer under test automatically gets wired as a monostable multivibrator. In this case, the monoshot can be triggered by the microswitch (S1). The debouncing circuit constituted by the two NAND gates of IC1 (N1 and N2) produces a clean rectangular pulse when the microswitch is pressed. Resistor R3, capacitor C1 and diode D1 ensure that the trigger terminal of timer IC 555 (pin 2 is the trigger terminal) gets the desired positive-to-ground trigger pulse. This differentiator circuit also ensures that the width of the trigger pulse is less than the expected monoshot output pulse.

The monoshot output pulse width is a function of the series combination of resistor R8 and potentiometer VR2, and capacitor C4. When DPDT switch S2 is in position 2-2, the timer gets configured for the astable mode of operation. The output is a pulse train with the high time period determined by the series combination of resistors R8, potentiometer VR2, resistor R9 and capacitor C4, whereas the low time period is determined by resistor R9 and capacitor C4.

The reset terminal of timer IC (pin 4) should be tied to Vcc normally. More precisely, the voltage at pin 4 should be greater than 0.8V. A voltage less than that resets the output. Whether you have connected the timer in the monoshot or astable mode of operation, the output goes low the moment you bring the reset terminal below 0.8V.

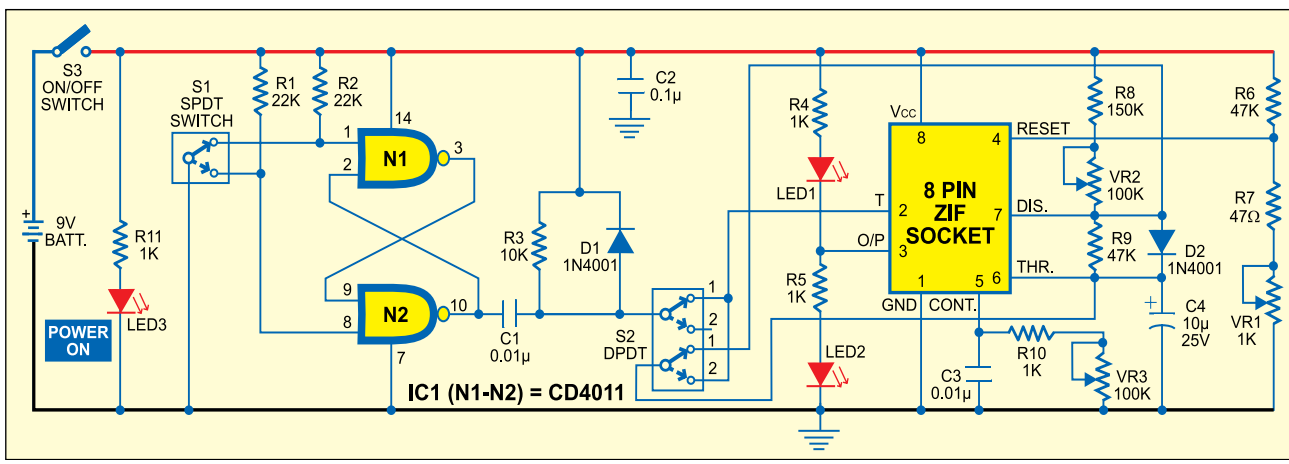
The control terminal (pin 5) can be used to change the high time ('on' time) of the output pulse train in the astable mode and the output pulse width in the monoshot mode by applying an external voltage. This external voltage basically changes the reference voltage levels of the comparators inside the IC. The levels are set by three identical resistors of usually 5 kilo-ohms inside the IC connected from Vcc to ground, at 2/3Vcc for pin 5 and 1/3Vcc for pin 2. These levels can be changed by connecting an external resistor between pin 5 and ground. Resistor R10 and potentiometer VR3 have been connected for this purpose.

The pulse width in the monoshot mode is given by:

1.1×total charging resistance×charging capacitance

This expression is valid when there is no external resistor connected at pin 5. The pulse width can be reduced by connecting an external resistor.

The high and low time periods in the astable mode are:



High time period = 0.69×charging resistance×charging capacitance

Low time period = 0.69×discharge resistance×capacitance

Again the expressions are true with no external resistor at pin 5. The high time period can be made to decrease by connecting an external resistor between pin 5 and ground.

The circuit can thus be used to check:

1. The timer IC in astable configuration.

2. The timer IC in monostable configuration.

3. The capability of the reset terminal to override all functions and rest the output to low.

4. The function of the control terminal to change the 'on' or the 'high' time of the output waveform in astable mode of operation and the output pulse width in monostable

mode of operation.

The circuit operates off a 9V battery, which makes the gadget portable. You can construct it easily on any general-purpose PCB along with the 8-pin socket.

To test an IC 555:

1. Insert it into the socket.

2. Set switch S2 in position 1-1.

3. Switch on the power supply by flipping switch S3 to 'on' position. Power-indicator LED (LED3) glows to indicate that the circuit is ready to test the IC timer.

4. If the IC is okay, LED1 glows because the IC is wired as a monoshot and in the absence of any trigger, its output is low.

5. Apply the trigger pulse by momentarily pressing switch S1. LED1 stops glowing and, in turn, LED2 glows. This confirms that the output of the monoshot has gone high. After the predetermined time period, LED2 goes off and LED1 again glows. Vary preset VR2 and trigger the monoshot again through switch S1. You will find that LED2 glows this time for a longer or a smaller time period depending upon whether you increased or decreased VR2 resistance.

6. For checking the reset function of the timer, trigger the monoshot again, and before the expected time is over, quickly decrease the potmeter VR1 resistance so as to bring the voltage at pin 4 below 0.8V. You will observe the output going low (indicated by glowing LED1 and extinguished LED2).

7. For checking the control function of the timer IC, set potmeter VR1 again in the maximum resistance position. Also set preset VR3 in the minimum resistance position. Trigger the monoshot using switch S1. You'll observe its output going high for a time period that is much less than that determined from the series combination of R8 and VR2, and capacitor C4. In fact, for any fixed setting of this series combination, the output pulse width can be observed to vary for different values of potmeter VR3 resistance—by triggering the monoshot several times, once for each setting of VR3.

8. Now set the DPDT switch in position 2-2. LED1 and LED2 glow alternatively with the timing determined by the resistances in the charge and discharge paths. This means the timer IC is okay and wired in astable mode.

9. The functions of reset and control pins can be checked in astable configuration too in the same way as discussed above for the monoshot configuration. ●

FUEL RESERVE INDICATOR FOR VEHICLES

■ D. MOHAN KUMAR

Here is a simple circuit for monitoring the fuel level in vehicles. It gives an audiovisual indication when the fuel level drops alarmingly below the reserve level, helping you to avoid running out of petrol on the way.

Nowadays vehicles come with a dash-mounted fuel gauge meter that indicates the fuel levels on an analogue display. The 'reserve' level is indicated by a red marking in some vehicles, but the needle movement through the red marking may be confusing and not precise. This circuit monitors the fuel tank below the reserve level and warns through LED indicators and audible beeps when the danger level is approaching.

The fuel sensor system consists of a tank-mounted float sensor and a current meter (fuel meter), which are connected in series. The float-driven sensor attached to an internal rheostat offers high resistance when the tank is empty. When the tank is full, the resistance decreases, allowing more current to pass through the meter to give a higher reading.

The fuel monitoring circuit works by sensing the voltage variation developed across the meter and activates the beeper when the fuel tank is almost empty. Its point A is connected to the input terminal of the fuel meter and point B is connected to the body of the vehicle.

The circuit consists of an op-amp IC CA3140 (IC1), two 555 timer ICs (IC2 and IC3) and decade counter CD4017 (IC4).

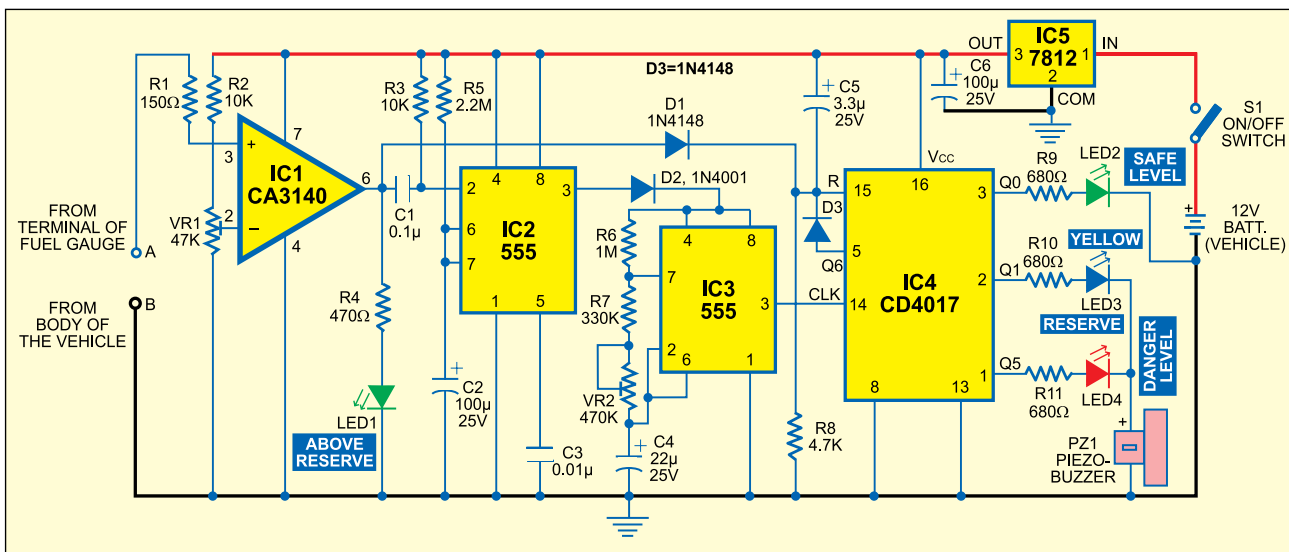
Op-amp IC CA3140 is wired as a voltage comparator. Its inverting input (pin 2) receives a reference voltage controlled through VR1. The non-inverting input (pin 3) receives a variable voltage tapped from the input terminal of the fuel meter through resistor R1.

When the voltage at pin 3 is higher than at pin 2, the output of IC1 goes high and the green LED (LED1) glows. This condition is maintained until the voltage at pin 3 drops below that at pin 2. When this happens, the output of IC1 swings from high to low, sending a low pulse to the trigger pin of the monostable (usually held high by R3) via C1. The monostable triggers and its output goes high for a predetermined time based on the values of R5 and C2. With the given values, the 'on' time will be around four minutes.

The output of IC2 is used to power the astable circuit consisting of timer 555 (IC3) via diode D2. Oscillations of IC3 are controlled by R6, R7, VR2 and C4. With the given values, the 'on' and 'off' time periods are 27 and 18 seconds, respectively. The pulses from IC3 are given to the clock input (pin 14) of decade counter CD4017 (IC4) and its outputs go high one by one.

When the circuit is switched on, LED1 and LED2 glow if your vehicle has sufficient petrol in the tank. When the fuel goes below the reserve level, the output of IC1 goes low, LED1 turns off and a negative triggering pulse is received at pin 2 of IC2. The output of IC2 goes high for around four minutes and during this time period, clock pin 14 of IC4 receives the clock pulse (low to high) from the output of IC3.

For the first clock pulse, Q0 output of IC4 goes high and the green LED (LED2) glows for around 50 seconds. On receiving the second clock pulse, Q1 goes high to light up the yellow LED (LED3) and sound the buzzer for



around 45 seconds. This audio-visual signal warns you that the vehicle is running out of fuel. On receiving the third clock pulse, LED3 and the buzzer go off. There is a gap of around two-and-a-half minutes before Q5 output goes high.

By the time Q5 goes high and the red LED (LED4) glows, four minutes elapse and the power supply to IC3 is cut off. The output state at Q5 will not change unless a low-to-high clock input is received at its pin 14. Thus LED4 will glow continuously along with the beep. The continuous glowing of the red LED (LED4) and the beep from the buzzer indicate that the vehicle will run out of fuel very shortly.

Q6 output of IC4 is connected to its reset pin 15 via diode D3. This means that after 'on' state of Q5, the count will always start from Q0. Capacitor C5 provides power-on reset to IC4 when switch S1 is closed. The output of IC1 is also connected to reset pin of IC4 via diode D1 (1N4148). So when your vehicle is refueled above the reserve level, LED2 glows to indicate that the tank has sufficient fuel.

IC5 provides regulated 12V DC for proper functioning of the circuit even when the battery is charged to more than 12V.

The circuit can be assembled on a perforated board**.** Adjust VR1 until the voltage at pin 2 of IC1 drops to 1.5V. When point A is connected to the fuel meter (fuel gauge) terminal that goes to the fuel sensor, green LEDs (LED1 and LED2) glow to indicate the normal fuel level. VR2 can be varied to set the 'on' time period of IC3 at around 20 seconds.

Enclose the circuit in a small case and mount on the dashboard using adhesive tape. The circuit works only in vehicles with negative grounding of the body. ●

MEDIUM-POWER FM TRANSMITTER

■ **PRADEEP G.**

The range of this FM transmitter is around 100 metres at 9V DC supply.

The circuit comprises three stages. The first stage is a microphone preamplifier built around BC548 transistor. The next stage is a VHF oscillator wired around another BC548. (BC series transistors are generally used in low-frequency stages. But these also work fine in RF stages as oscillator.) The third stage is a class-A tuned amplifier that boosts signals from the oscillator. Use of the additional RF amplifier increases the range of the transmitter.

Coil L1 comprises four turns of 20SWG enamelled copper wire wound to 1.5cm length of a 4mm dia. air core. Coil L2 comprises six turns of 20SWG enamelled copper wire wound on a 4mm dia. air core.

Use a 75cm long wire as the antenna.

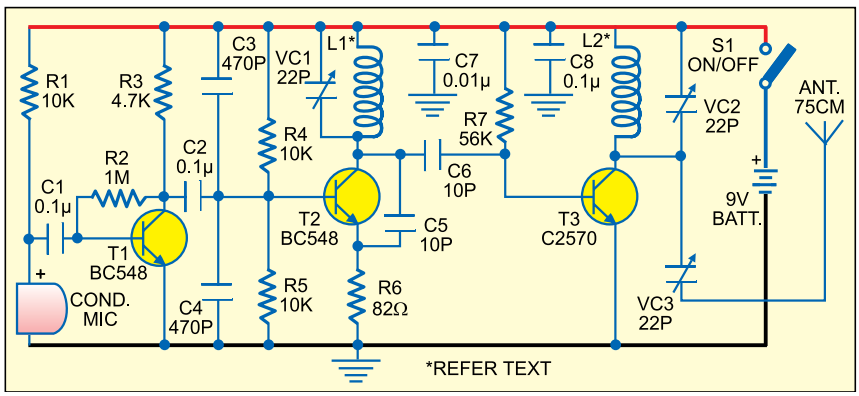


Fig. 1: FM transmitter

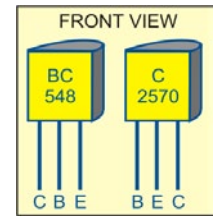


Fig. 2: Pin configurations of transistors BC548 and C2570

For the maximum range, use a sensitive receiver. VC1 is a frequency-adjusting trimpot. VC2 should be adjusted for the maximum range. The transmitter unit is powered by a 9V PP3 battery.

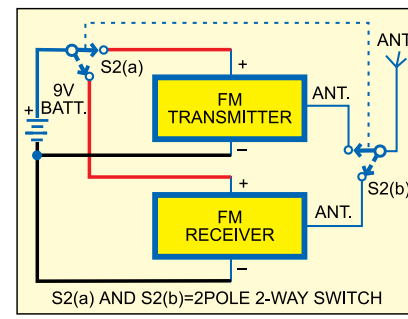


Fig. 3: Walkie-talkie arrangement

It can be combined with a readily available FM receiver kit to make a walkie-talkie set as shown in Fig. 3. ●

TELECONFERENCING SYSTEM

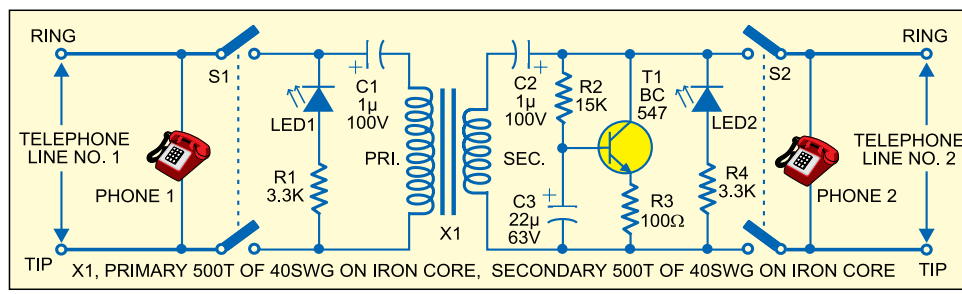
■ **PRINCE PHILLIPS**

Here is a low-cost teleconferencing system that lets you talk to two persons at a time in any part of the world over two telephone lines. The circuit makes use of a coupling transformer and some passive components.

The circuit is connected between the two telephone lines. It works like this: When 'X' calls 'A' on the first telephone line, 'A' puts this call on hold, dials 'Y' on the other telephone line (which is free) and keeps this call too on hold, and slides switches S1 and S2 to 'on' position. Now 'X,' 'A' and 'Y' can talk to one another simultaneously over the two telephone lines.

Both the primary and secondary coils of the coupling transformer consist of 500 turns of 40SWG insulated copper wire. At the secondary side, a small circuit is used for DC holding. This circuit is built around transistor T1 (BC547), resistors R2 and R3 (15 kilo-ohms and 100 ohms, respectively), condenser C3 (22µF, 63V) and two LEDs as indicators for both the primary and secondary sides. It provides proper DC characteristic to hold second telephone line in operation even though no telephone on that line is present.

Here, transistor T1 acts like a resistor to DC and as high impedance for audio signals. The high impedance of the circuit is provided by condenser C3, which prevents any audio signal from appearing at the base of T1. Thus any audio voltage appearing across telephone line No. 2 will not cause a corresponding current in the transistor. ●



LIGHT DIMMER THAT DOUBLES AS VOLTMETER

■ G.D. SEKHRI

Measure AC mains voltage without using a multimeter. All you need to do is to slightly modify the light dimmer fitted at the base of a table lamp for use as a voltmeter. When the dimmer is turned anticlockwise to a point where the filament glow is just visible, that point can be used as the reference point for measuring the voltage.

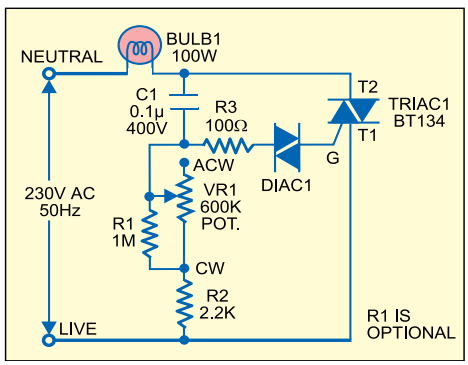


Fig. 1: Light dimmer

First, remove the old knob and fix a circular white paper around the shaft. Now put back a skirted knob with a cursor as close to the paper as possible and mark two extremities of the pot on the paper as CW and ACW (see Fig. 2).

Switch on the lamp via a variac and feed 50 volts. Rotate the potmeter knob anticlockwise until the filament glow is just visible and mark that point against the cursor as 50V. Keep on increasing the voltage to 100, 150, 180, 200 and 220 using the variac and calibrating the scale for all the voltages. Now a voltage scale is created. The only snag is that the voltage is increasing in anticlockwise direction, which should not be a problem. The scale will not however be linear unlike the one shown in the sketch. Accuracy will depend on the calibration standard used and the tolerance is of the order of 1 per cent ±5 volts. The diameter of the knob of potmeter and fineness of cursor can be of help in getting better accuracy and tolerance.

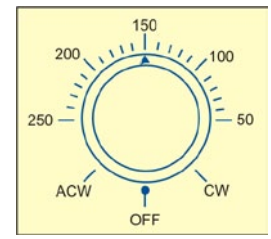


Fig. 2: AC volts scale marking

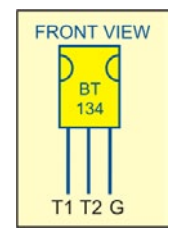


Fig. 3: Pin configuration of BT134

An ordinary fan regulator can be used with a lamp of 40, 60 or 100 watts and calibrated accordingly. The minimum measurable voltage is naturally limited to the one required for 'just visible' condition. With R1 open circuited the maximum scale voltage will be around 220 volts. ●

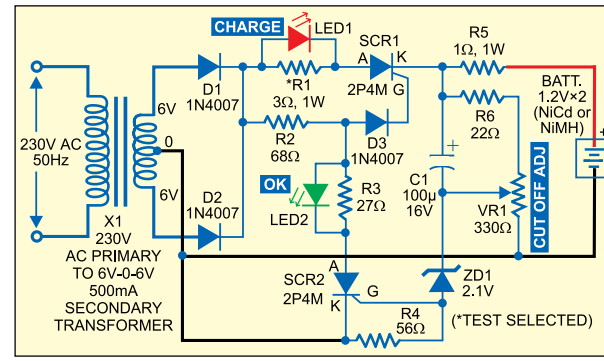
MULTICELL CHARGER

■ T.K. HAREENDRAN

Using this charger, you can safely charge up to two pieces of Ni-Cd cells or Ni-MH cells. The circuit is compact, inexpensive and easy-to-use.

The 230V AC mains is down-converted to 12V AC (at 500 mA) by step-down transformer X1, converted into pulsating DC voltage by diodes D1 and D2, and fed to the battery charger terminals via current-limiting resistor R1 and silicon-controlled rectifier SCR1.

SCR1 is at the heart of the charger. Normally, it conducts due to the gate biasing voltage available through resistor R2 and diode D3, and the battery is in charging mode, which is indicated by LED1. Resistor R2 limits the charging current to a safe value. Charging current of this circuit is about 250 mA.



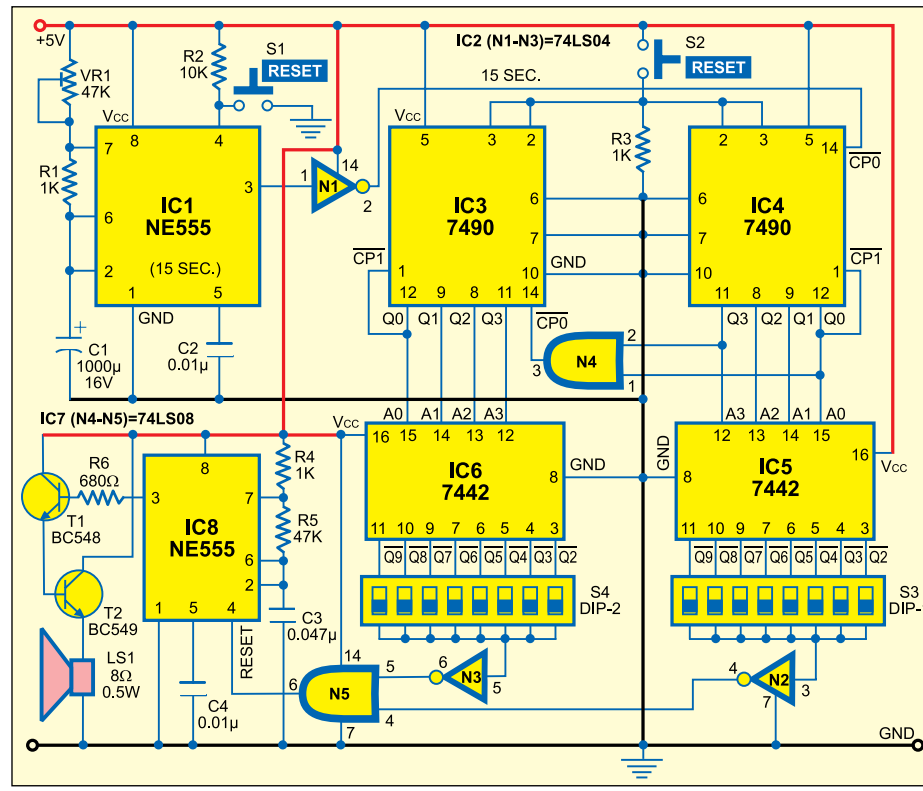
When the battery reaches full charge, SCR2 conducts to pull down the gate of SCR1. This state is indicated by LED2. Now remove the cells from the charger. Normally, Ni-Cd cell with a rating of 500 mAH will take around 2.5 hours to reach full charge, while the charging time for Ni-MH cell with a rating of 1500 mAH will be around 7 hours. Charging time may vary depending on the settings of the charger and input supply line conditions.

After construction, a minor adjustment is required for ensuring proper performance: Power on the circuit without cells and adjust VR1 such that LED2 lights up. Now measure voltage across the charger output terminals, which should be around 5V DC. Now insert the two cells into the holder and connect it to the charger output terminals for charging. LED1 instantly lights up to indicate the charging process. If LED1 glows dimly, readjust VR1 for proper glowing of LED1. Now the circuit is ready for use.

Use of a small heat-sink is recommended for SCR1. ●

TIMER FOR GEYSER

■ V. GOPALAKRISHNAN



This timer circuit for gey-ser sounds an alarm after the set timing of 22 minutes when the water is heated up.

The circuit comprises a timer IC 555 wired as an astable multivibrator with adjustable time period of 15 seconds. The astable output after inversion by an inverter drives decade counters IC3 and IC4 (each IC 7490) connected in cascade. The decade counters output is connected to decoders IC5 and IC6 (each IC 7442), respectively. The decoder outputs ($\overline{Q8}$ outputs of IC5 and IC6) are fed to inverters and the inverter outputs, in turn, are fed to an AND gate. The AND output is connected to the reset pin of the astable multivibrator built around another timer IC 555 to sound the alarm. Now you can turn off the geyser.

A green LED (LED1) has been used as the power supply indicator.

Switch on the timer and the geyser at the same time. When the alarm sounds, it means that the water in the geyser has heated up and can be used.

You can assemble the timer circuit on a general-purpose PCB and install it near your bathroom so that both the timer circuit and the geyser can be switched on simultaneously.

After the siren sounds, if required, we can increase the time by another 22 minutes for geyser by resetting the circuit by pushing reset switches S1 and S2 momentarily.

If you want to change the preset time of the geyser, the same can be easily done by combining appropriate outputs of IC5 and IC6 using DIP switches (S3 and S4) while keeping in mind IC5 outputs ($\overline{Q0}$ through $\overline{Q9}$) are spaced 15 seconds apart and IC6 outputs are spaced 150 seconds (2.5 minutes) apart.

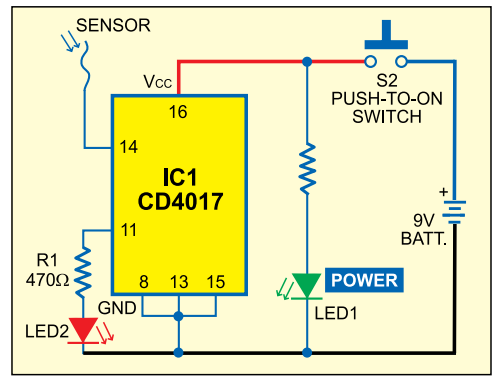
Caution. Please note that the timer circuit has no connection with the geyser circuit. The geyser works off 220V AC, while the timer works off 5V DC. ●

220V LIVE WIRE SCANNER

■ T.A. BABU

This simple circuit lets you scan a 220V live wire. The clock input of the IC is connected to a wire, which acts as the sensor. Here, we have used 10cm length of 22SWG wire as the sensor.

When you hold the sensor (metallic conductor or copper wire) close to the live wire, electric field from mains activates the circuit. As the input impedance of the CMOS IC is high, the electric field induced in the sensor is sufficient to clock it. The output obtained at pin 11 of CD4017 drives the LED. Flashing of the LED (LED2) indicates the presence of mains, while LED1 indicates that the scanner is active.



The circuit can be used to find stray leakage from electrical appliances like fans, mixers, refrigerators, etc. It can be easily assembled on any general-purpose board or the discrete components can be directly soldered on the IC.

A 9V PP3 battery powers the circuit. If you use a mains adaptor, make sure that it is well regulated and isolated; otherwise, even the stray electric field from mains transformer will clock the circuit.

Caution. Use insulated wire as sensor to avoid risk of exposure to live AC mains. ●

DOORBELL-CUM-VISITOR INDICATOR

■ V. GOPALAKRISHNAN

This doorbell circuit can also give identification of the visitor to your home in your absence. When you're home, you can use it simply as a normal doorbell.

The circuit (see Fig. 1) comprises a monostable built around timer IC 555 (IC1), relay driver transistor BC548 (T1), inverter section built around IC 7404 (IC2), latching section built around IC 555 (IC3) and LED display driver transistor BC548 (T2).

The monostable output drives the relay through transistor amplifier T1. The normally-opened (N/O) contact of the relay connects an electric bell with mains supply as shown in Fig. 1. The output of the monostable also goes to inverter N1, which, in turn, enables the latching circuit built around IC3 in conjunction with DPDT slide switch S2. The output of IC3 at its pin 3 is fed to LED1 (visitor-out indicator) and LED2 through LED16 via LED driver transistor T2.

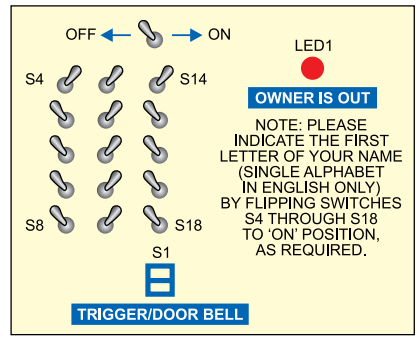


Fig. 2: Switch panel to be mounted on the gate

The switch panel shown in Fig. 2 is to be mounted on the entry gate or door, while the LED display panel shown in Fig. 3 is to be kept inside the house near the owner's desk.

Before you leave your house, slide switch S2 (Out) to 'on' position and press reset switch S3 once. When somebody visits your home and presses doorbell switch S1, it will trigger the monostable (IC1) and also energise the relay to ring the bell. The monostable output through inverter N1 will enable the latching circuit and LED1 will glow continuously to indicate that you're out of the house.

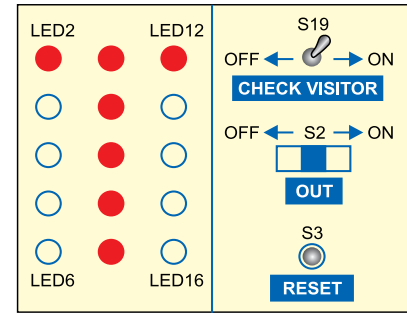


Fig. 3: Suggested LED display to be kept near the owner's desk

The message "Please indicate the first letter of your name (single alphabet in English only) by flipping switches S4 through S18 to 'on' position," as required, is written just below LED1 indicator as shown in Fig. 2.

Suppose the visitor's name is Tina Chopra. As the initial alphabet of her first name is 'T,' she has to flip the topmost-row and middle-column switches towards 'on' position. The position of the switches for this example is shown in Fig. 2.

When you return home, just flip switch S19 to 'on' position to check

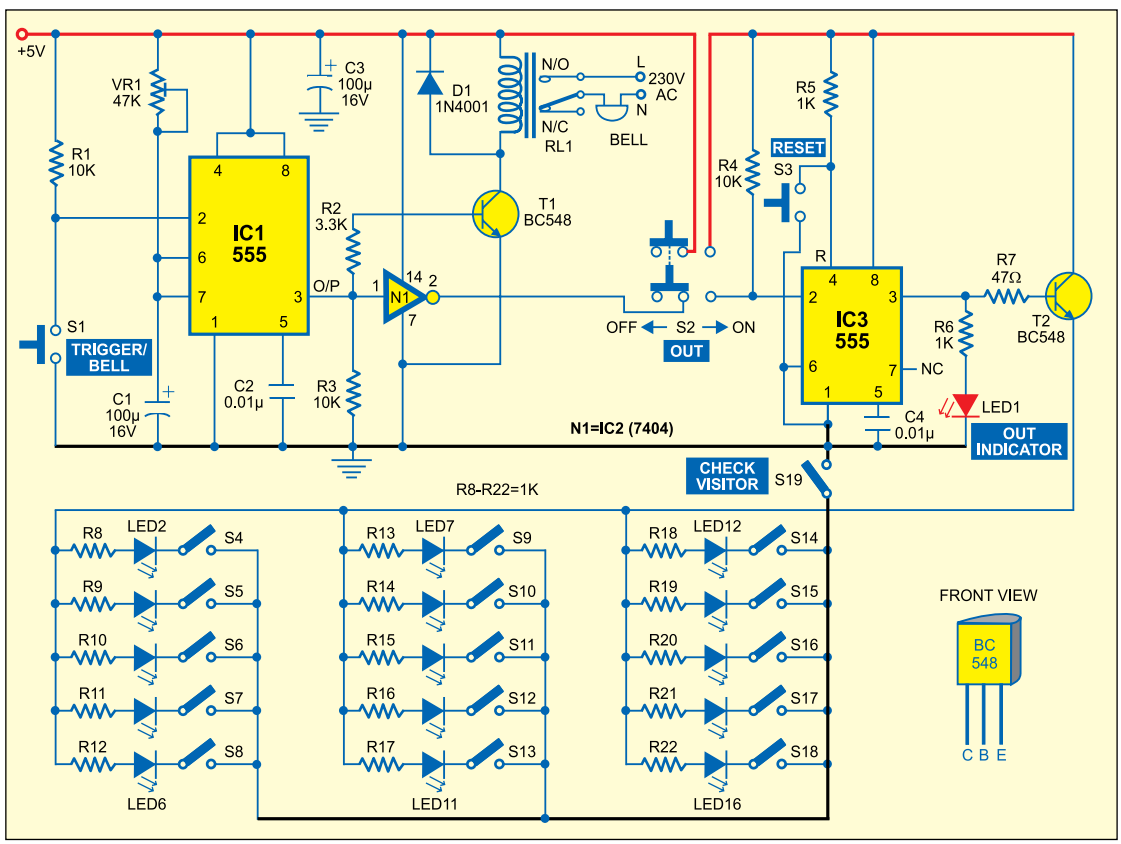


Fig. 1: Circuit diagram of doorbell-cum-visitor indicator

the visitor's initial. The topmost-row and middle-column LEDs will glow to indicate the alphabet 'T,' so you know that one of your friends having name starting with 'T' had visited your place in your absence. The status of the LEDs for this example is shown in Fig. 3 by the red LEDs.

Now you can slide switch S2 to 'off' position or press reset button S3 once. When the indicator is not in use, slide DPDT switch S2 to 'off' position to bypass the latching and LED display sections by cutting off the power supply and thus prevent unnecessary continuous drain of power. Now the rest of the circuit will work as a doorbell only.

The rows and columns may be increased to accommodate a 5×7 display matrix. The circuit can be used for a single visitor only. ●

SMART SWITCH

■ T.A. BABU

To switch on the mains voltage, either a mechanical switch or a relay offers a simple solution. However, the relay and its associated components occupy a lot of space and cannot be accommodated in a standard switch box. The 'smart switch' circuit, shown here, offers a better alternative. It is nothing but an 'on'/'off' controller and uses an electronic circuit that behaves like a normal switch. A flat pushbutton control provides an aesthetic look to your switch panel.

The switching circuit comprises an optocoupler circuit that receives input from a bistable switch formed by a couple of Schmidt trigger gates that control a triac. The load can be switched on/off by simply pushing the pushbutton switch for a brief period. Every time the switch receives a push, the optocoupler toggles the triac. A special zero-crossing detector in the optocoupler supresses radio interference, unlike the arbitrary phase switching.

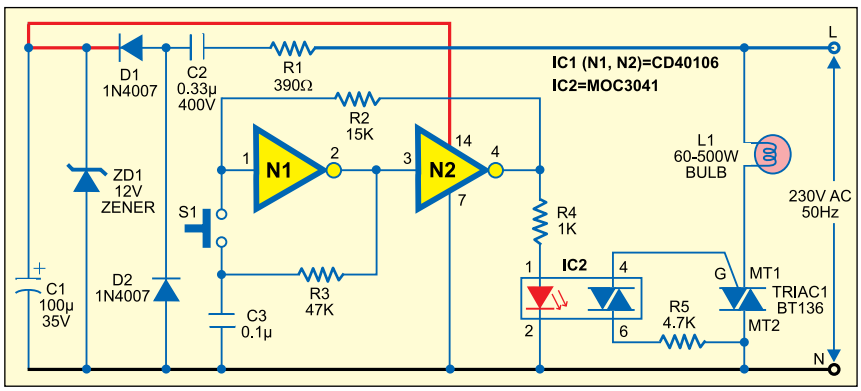


Fig. 1: Circuit of the smart switch

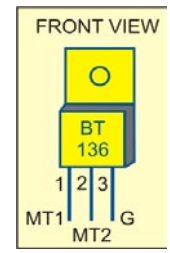


Fig. 2: Pin configuration of triac BT136

Since mains is not isolated, use a good-quality pushbutton switch with proper insulation to avoid lethal shock. Make sure that the triac can handle the current you are going to draw through it. If required, several pushbuttons can be wired in parallel to allow toggling of the triac from different locations. ●

STRESS METER

■ **D. MOHAN KUMAR**

This stress monitor lets you assess your emotional pain. If the stress is very high, it gives visual indication through a light-emitting diode (LED) display along with a warning beep. The gadget is small enough to be worn around the wrist.

The gadget is based on the principle that the resistance of the skin varies in accordance with your emotional states. If the stress level is high the skin offers less resistance, and if the body is relaxed the skin resistance is high. The low resistance of the skin during high stress is due to an increase in the blood supply to the skin. This increases the permeability of the skin and hence the conductivity for electric current.

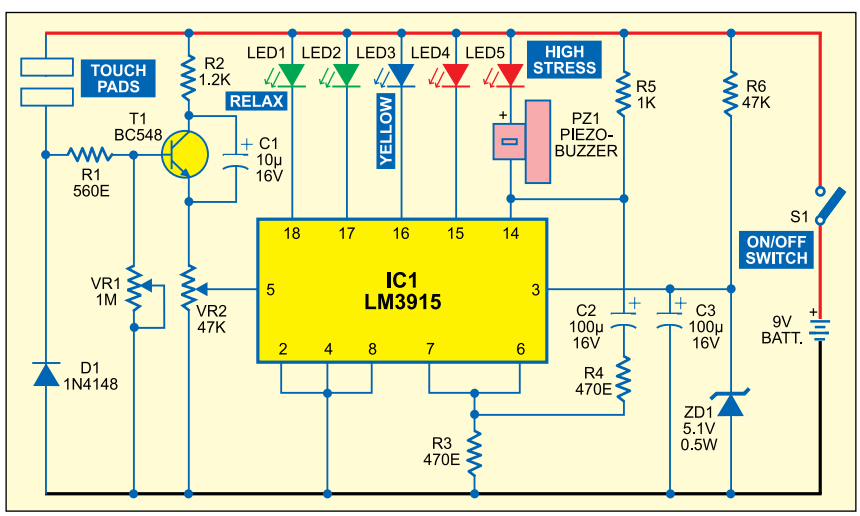


Fig. 1: Circuit of the stress meter

This property of the skin is used here to measure the stress level. The touch pads of the stress meter sense the voltage variations across the touch pads and convey the same to the circuit. The circuit is very sensitive and detects even a minute voltage variation across the touch pads.

The circuit comprises signal amplifier and analogue display sections. Voltage variations from the sensing pads are amplified by transistor BC548 (T1), which is configured as a common-emitter amplifier. The base of T1 is connected to one of the touch pads through resistor R1 and to the ground rail through potmeter VR1. By varying VR1, the sensitivity of T1 can be adjusted to the desired level. Diode D1 maintains proper biasing of T1 and capacitor C1 keeps the voltage from the emitter of T1 steady.

The amplified signal from transistor T1 is given to the input of IC LM3915 (IC1) through VR2. IC LM3915 is a monolithic integrated circuit that senses analogue voltage levels at its pin 5 and displays them through LEDs providing a logarithmic analogue display. It can drive up to ten LEDs one by one in the dot/bar mode for each increment of 125 mV in the input.

Here, we've used only five LEDs connected at pins 14 through 18 of IC1. LED1 glows when input pin 5 of IC1 receives 150 mV. LED5 glows when the voltage rises to 650 mV and LED5 flashes and piezobuzzer PZ1 beeps when the stress level is high.

Resistors R4 and R5 and capacitor C2 form the flashing elements. Resistor R3 maintains the LED current at around 20 mA. Capacitor C3 should be placed close to pin 3 for proper functioning of the IC. Zener diode ZD1 in series with resistor R6 provides regulated 5V to the circuit.

The circuit can be assembled on a small piece of perforated board. Use transparent 3mm LEDs and a small piezobuzzer for audio-visual indications. Enclose the circuit in a small plastic case with touch pads on the back side. Two self-locking straps can be used to tie the unit around your wrist.

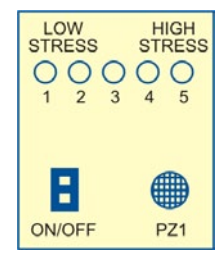


Fig. 2: Display panel

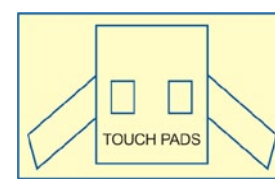


Fig. 3: Self-locking straps

After tying the unit around your wrist (with touch pads in contact with the skin), slowly vary VR1 until LED1 glows (assuming that you are in relaxed state). Adjust VR2 if the sensitivity of IC1 is very high. The gadget is now ready for use. ●

POWER FAILURE AND RESUMPTION ALARM

■ SEEMANT SINGH

This circuit gives audio-visual indication of the failure and resumption of mains power. The circuit is built around dual timer IC LM556. When mains is present the bicolour LED glows in green colour, and when mains fails it turns red.

The AC mains is stepped down by transformer X1 to deliver the secondary output of 12V at 250 mA. The transformer output is rectified by a full-wave bridge rectifier comprising diodes D1 through D4, filtered by capacitor C1 and regulated by IC 7809 (IC1) to give regulated 9V DC to operate the circuit.

9V battery and pnp transistor T1 have been used here as the power source for red light indication of the absence of power. Transistor T1 can be made to conduct or cut-off easily by varying preset VR1.

Initially, when mains is present, pnp transistor T1 is in cut-off state and therefore bicolour LED1 glows in green colour.

When power fails, pnp transistor T1 starts conducting and bicolour LED1 glows in red colour. Due to non-availability of Vcc voltage at pin 14 of IC2, its output pin 9 remains low and transistor T3 does not conduct. However, capacitor C7 (4700µF) holds adequate charge and hence transistor T4 conducts and piezobuzzer PZ1 sounds continuously for around eleven seconds until capacitor C7 discharges completely.

When power resumes, bicolour LED1 glows in green colour and the buzzer beeps for around 14 seconds.

Dual timer IC LM556 (IC2) sections have been used here in monostable and astable modes, respectively.

In the monostable section, location of the external timing capacitor determines whether a positive or negative output pulse is generated. Diode D7 ensures that even a momentary power loss will cause a pulse to be generated when the power resumes. With capacitor C3 connected to ground, a positive output pulse is generated according to the following relationship:

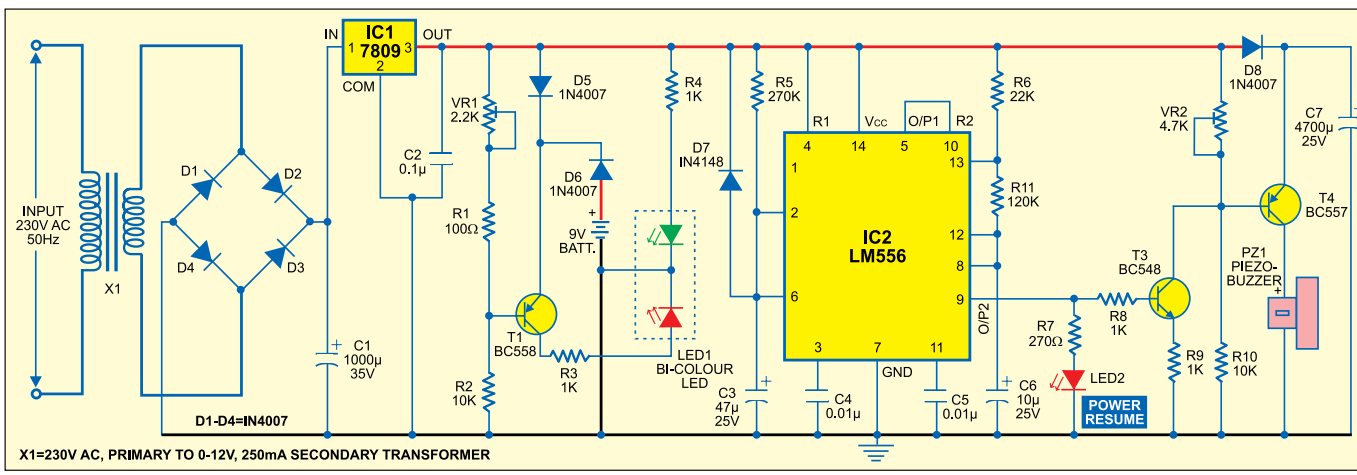
T = 1.1×R5×C3.

This positive output is present at pin 5 of IC2. Since IC2 is a dual-timer IC, its first output is directly fed to reset pin 10 of second section. Therefore the second timer of IC2 starts oscillating. Its frequency of oscillations (F0) is determined by resistors R6 and R11 and capacitor C6 as follows:

F0=1.4/(R6+2R11)×C6.

IC LM556 outputs frequencies in the form of pulses at its pin 9. These pulses are coupled to npn transistor T3, which conducts and cuts off depending on the output at pin 9 of IC2. Red LED2 is connected to pin 9 via current-limiting resistor R7 (270-ohm) to indicate power resumption.

The collector output of transistor T3 is directly fed to the base of pnp transistor T4, due to which base biasing of T4 varies and the buzzer beeps for around 14 seconds. ●



LITTLE DOOR GUARD

■ **T.K. HAREENDRAN**

If some intruder tries to open the door of your house, this circuit sounds an alarm to alert you against the attempted intrusion.

The circuit (Fig. 1) uses readily available, low-cost components. For compactness, an alkaline 12V battery is used for powering the unit. Input DC supply is further regulated to a steady DC voltage of 5V by 3-pin regulator IC 7805 (IC2).

Assemble the unit on a general-purpose PCB as shown in Fig. 4 and mount the same on the door as shown in Fig. 3. Now mount a piece of mirror on the doorframe such that it is exactly aligned with the unit. Pin configurations of IC UM3561 and transistors 2N5777 and BC547 are shown in Fig. 2.

Initially, when the door is closed, the infrared (IR) beam transmitted by IR LED1 is reflected (by the mirror) back to phototransistor 2N5777 (T1). The IR beam falling on phototransistor T1 reverse biases npn transistor T2 and IC1 does not get positive supply at its pin 5. As a result, no tone is produced at its output pin 3 and the loudspeaker remains silent. Resistor R1 limits the operating current for the IR LED.

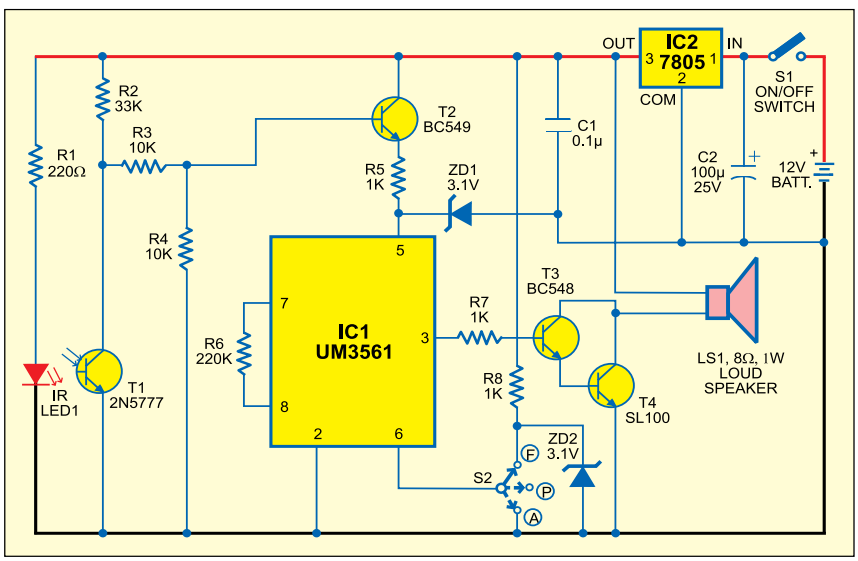


Fig. 1: Circuit of the door guard

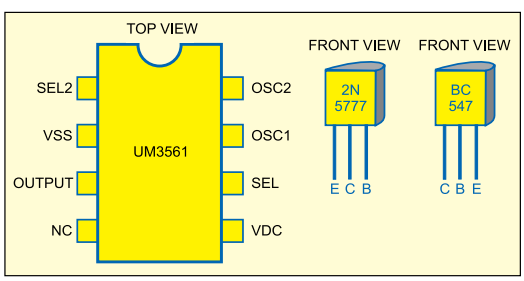


Fig. 2: Pin configurations of UM3561 and transistors 2N5777 and BC547

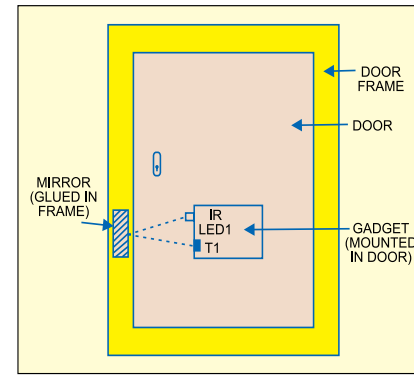


Fig. 3: Back view of the door assembly

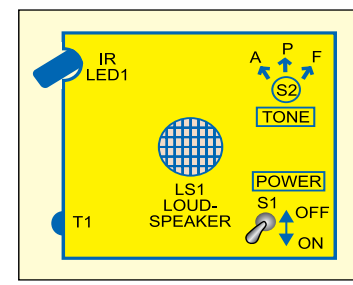


Fig. 4: Suggested enclosure with major components layout

When the door is opened, the absence of IR rays at phototransistor T1 forward biases npn transistor T2, which provides positive supply to IC1. Now 3-siren sound generator IC UM3561 (IC1) gets power via resistor R5. The output of IC1 at pin 3 is amplified by Darlington-pair transistors T3 and T4 to produce the alert tone via the loudspeaker.

Rotary switch S2 is used to select the three preprogrammed tones of IC1. IC1 produces fire engine, police and ambulance siren sounds when its pin 6 is connected to point F, P or A, respectively. ●

ELECTRONIC FUSE

■ T.A. BABU

An absolute necessity of every electronics lab is a workbench power supply. The power supply should be regulated and protected against short circuit.

Most power-supply protection circuits use a low-value, high-wattage resistor connected in series with the load for current sensing. The voltage drop across the sensor resistor is weighed to activate the protection circuit. The given circuit is based on a polyfuse application, which is a resettable fuse by itself.

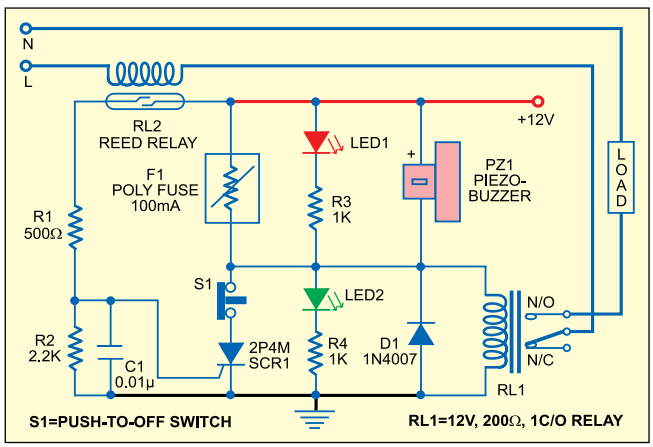


Fig. 1: Electronic fuse

Initially, when the circuit is powered, silicon-controlled rectifier SCR1 is 'off.' Relay RL1 energises through the polyfuse and the load is connected through the normally opened (N/O) contact of the relay. When the current drawn by the load increases above a certain level (which depends on the number of turns in the winding on reed relay, see Fig. 2 and the accompanying table), the contacts of reed relay RL2 close to trigger SCR1. As a result, relay RL1 de-energises and the load gets disconnected. The polyfuse remains in high-resistance state until SCR1 is turned off.

The circuit can be reset either by switching off the power supply or by pushing reset switch S1. LED2 indicates that the power supply is working normally. LED1 indicates that the power supply unit is under the protection mode and the buzzer sounds to warn the user.

The turns of reed relay winding are based on the current drawn through the load, so refer to the table for winding details for your load current requirements. At EFY, testing was done for approximately 1.85A AC load current at 230V AC mains and accordingly 16 turns of 22SWG copper-enamelled wire were wound on the reed relay. ●

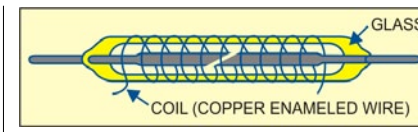


Fig. 2: Reed relay with coil winding

Reed Relay Winding Details for Different Load Currents

Current (amperes)	Turns	SWG
10	5	13
5	10	16
2.5	20	18
1.25	4	21

DIGITAL DICE

■ **SAGAR G. YADAV**

The digital dice presented here acts just like a normal dice. It has six faces (refer Fig. 2) like the normal dice and uses four different logic gate combinations to bring out the six faces of the dice.

At the heart of the circuit is a 14-stage ripple-carry binary counter IC CD4060BC (IC1) with built-in oscillator. The logic section is designed around CMOS quad 2-input NOR gate IC CD4001BC (IC2) and quad 2-input NAND gate IC CD4011BC (IC3). The display section is formed by a group of seven LEDs.

The circuit is divided into three sections: counter, logic and display.

The counter section is built around binary counter IC CD4060BC (IC1). The counter frequency (f) is decided by the in-built oscillator formed by resistor R1 and capacitor C1 as follows:

f=1/2.2R1C1.

Here, the frequency is fixed at around 2056 Hz.

Only the first three outputs of the counter (designated as A, B and C, respectively) have been used in the circuit. The counter is designed to reset at the sixth count (110) as only six counts are required for operation. This is done with the help of diodes D1 and D2 and resistor R3, which are connected such that they generate an AND logic.

From the table it can be noticed that at the sixth count, the counter outputs A and B hold logic 1 simultaneously for the first time, so by ANDing A and B outputs you can give logic 1 to the reset terminal of the counter at the sixth count, thereby resetting the counter.

LED2 and LED5 always glow at the same count, as do LED1 and LED6, and LED3 and LED4. Using these three pairs of LEDs and LED7, four logical combinations have been made in the circuit. LED1 and LED6 glow at all counts, except '0' and '1.' Further, it can be noticed that they glow when 'A' or 'B' is high, hence a NOR gate whose output is $\overline{A+B}$ according to Boolean algebra will perform the job of operating these LEDs.

LED3 and LED4 glow at all counts, except for the first three counts, i.e., they glow when either A, or B and C are high. This logic function can be obtained by using an OR gate and an AND gate, but since we are using only NAND and NOR gates in the circuit we make use of two NAND gates and a NOR gate (with $\overline{A+BC}$ output) to perform this function.

LED2 and LED5 glow only at the first and fifth counts. In other words, they glow only when the complement of B and C outputs goes high. This function can be obtained by using two NAND gates such that their output corresponds to the Boolean expression $\overline{BC}$ or $B+\overline{C}$ according to De Morgan's theorem.

LED7 glows at even counts like 0, 2 and 4. In other words, it glows when the C output is low. This function can be achieved easily by inverting the C output twice using the remaining two NAND gates. The output will also be buffered by these two inverter gates.

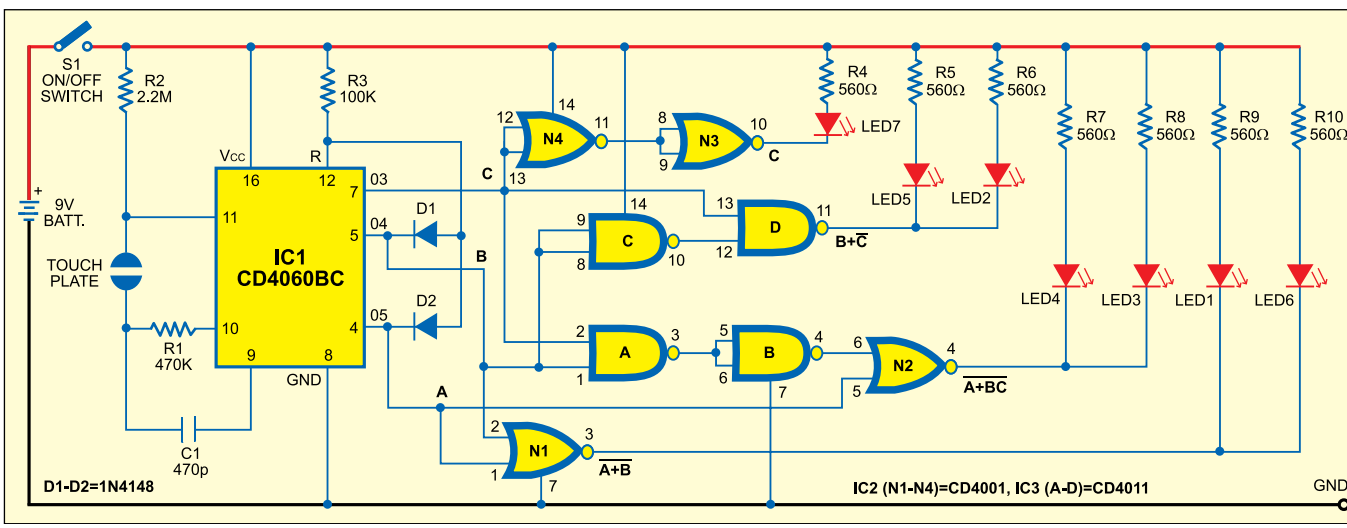


Fig. 1: Digital dice

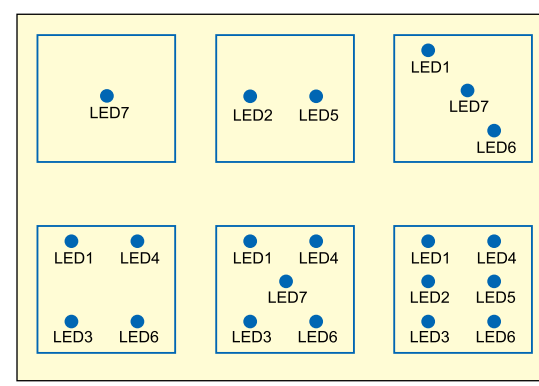


Fig. 2: Different faces of dice

Dice Score and LEDs Lit at Different Counts

Count	Dice score	A	B	C	LEDs lit						
0	1	0	0	0	—	—	—	LE D7			
1	2	0	0	1	—	—	LED2	—	LED5		
2	3	0	1	0	LED1	—	—	LED7	—	—	LED6
3	4	0	1	1	LED1	LED3	—	—	—	LED4	LED6
4	5	1	0	0	LED1	LED3	—	LED7	—	LED4	LED6
5	6	1	0	1	LED1	LED3	LED2	—	LED5	LED4	LED6
6	—	1	1	0	—	—	—	—	—	—	—

The display section comprises seven LEDs. LED1 and LED6 have common cathodes, as do LED2 and LED5, and LED3 and LED4. The anodes of all the LEDs are tied together to the positive terminal of the battery via resistors R4 through R10, respectively.

When you place your finger on the touch pad, the oscillator starts oscillating. The counter will start counting at the rate of 2056 Hz and all the LEDs of the display section will appear to glow simultaneously due to the high counter frequency. This high-frequency counting will make the dice foolproof. When you remove your finger from the touch pad, the counter will stop counting and the display section will show any one of the six possible faces with a probability of 1/6.

The entire circuit can be powered by a 9V battery as the inbuilt oscillator of the counter IC will not work properly below 7V. Use of CMOS ICs means less power consumption.

The circuit can be constructed on a general-purpose PCB and housed inside a plastic case with the LEDs array mounted on the top as shown in Fig. 2. The touch pad can be mounted beside the array. ●

BICYCLE GUARD

■ T.K. HAREENDRAN

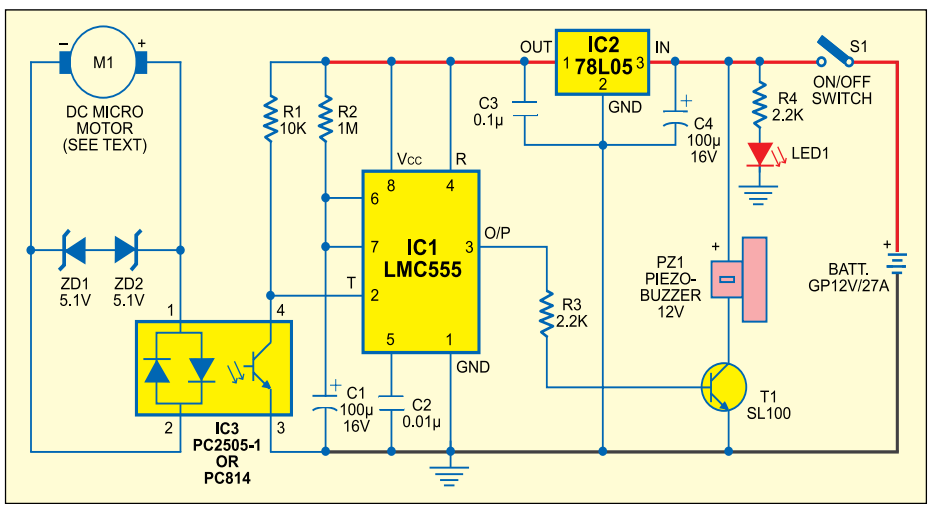
This antitheft device for bicycles is inexpensive and can be constructed easily using a few components.

At the heart of the circuit is a wheel rotation detector, realised using a DC micro motor. For the purpose, you can use the micromotor (spindle motor) of a discarded local CD deck mechanism. With a little skill and patience, you can easily attach a small metallic pulley covered with a rubber washer to the motor spindle. Thereafter, fix the unit in the back wheel of the cycle, like the existing dynamo assembly.

Power supply switch S1 should be kept 'on' when you are using this bicycle guard. When it is flipped towards 'on' position, the circuit gets power from the miniature 12V battery. Now LED1 lights up and resistor R4 limits the LED current. Next, the monostable built around IC1, which is CMOS version of timer LMC555, is powered through a low-current, fixed-voltage regulator IC2 (78L05).

Initially, when the bicycle is standing still, the monostable output at pin 3 of IC1 is low and the circuit is in idle state. In the event of a theft attempt, forward or reverse rotation of the DC motor induces a small voltage at its DC input terminals and the internal LED of 4-pin DIP AC input isolator optocoupler IC3 (PS2505-1 or PC814) glows. As a result, the internal transistor of IC3 conducts and pin 2 of IC1 is pulled low by the optocoupler and the monostable built around IC1 is triggered.

The output at pin 3 of IC1 now drives piezobuzzer-driver transistor T1 via resistor R3 and the buzzer starts sounding to alert you. In this circuit, the buzzer remains 'on' for around two minutes. You can change this time by changing the values of resistor R2 and capacitor C1.



Zener diodes ZD1 and ZD2 (each 5.1V) act as a protector for optocoupler IC3. The costly GP12V/27A battery is used here due to its compact size and reliability. 12V active buzzers with high-pitched tone output may be used with this circuit. These are readily available in the market.

Note. The specific optocoupler is used here deliberately, instead of a bridge rectifier, to increase the circuit's detection sensitivity. Never replace the same with a DC optocoupler. ●

LIQUID-LEVEL ALARM

■ **PRADEEP G.**

In water-level controllers or tanks, a DC current is passed through the metallic probes fitted in the water tank to sense the water level. This causes electrolysis and corrosion of probes, inhibiting the conduction of current and degrading its performance. As a consequence, probes have to be replaced regularly to maintain proper current flow.

The liquid-level alarm given here overcomes this problem. A 1kHz AC signal is passed through the probes, so there will be no electrolysis and therefore the probes last longer.

The block diagram for the liquid-level alarm is shown in Fig. 1. The signal generator sends the generated signal to the first metallic probe. The second metallic probe is connected to the sensing circuit followed by the alarm circuit.

The complete circuit for the liquid-level alarm is shown in Fig. 2. The astable multibrator built around IC 555 (IC1) generates 1kHz square wave signal, which is fed to one of the probes via a DC blocking capacitor.

When the water tank is empty, pnp transistor T1 does not get negative base bias. But as water fills up in the tank, it receives 1kHz signal from IC1 via the probes immersed in water and conducts during the negative half cycle of 1kHz signals. Due to the presence of capacitor C7 (2.2µF), npn transistor T2 continues to get base bias and conducts to provide 3.3V DC to melody generator IC UM66 (IC2). Pin configuration of IC UM66 is shown in Fig. 3. Preset VR1 acts as the output loudness controller. It can be varied to set the alarm sound from the speaker at the desired level.

The circuit works off 12V unregulated power and can be used to detect any conductive liquid. ●

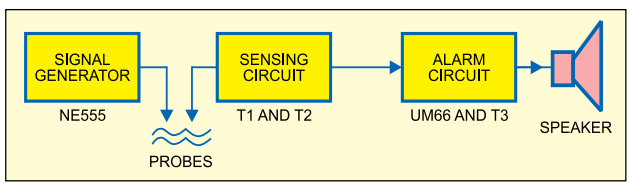


Fig. 1: Block diagram of liquid level alarm

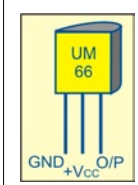


Fig. 3: Pin configuration of UM66

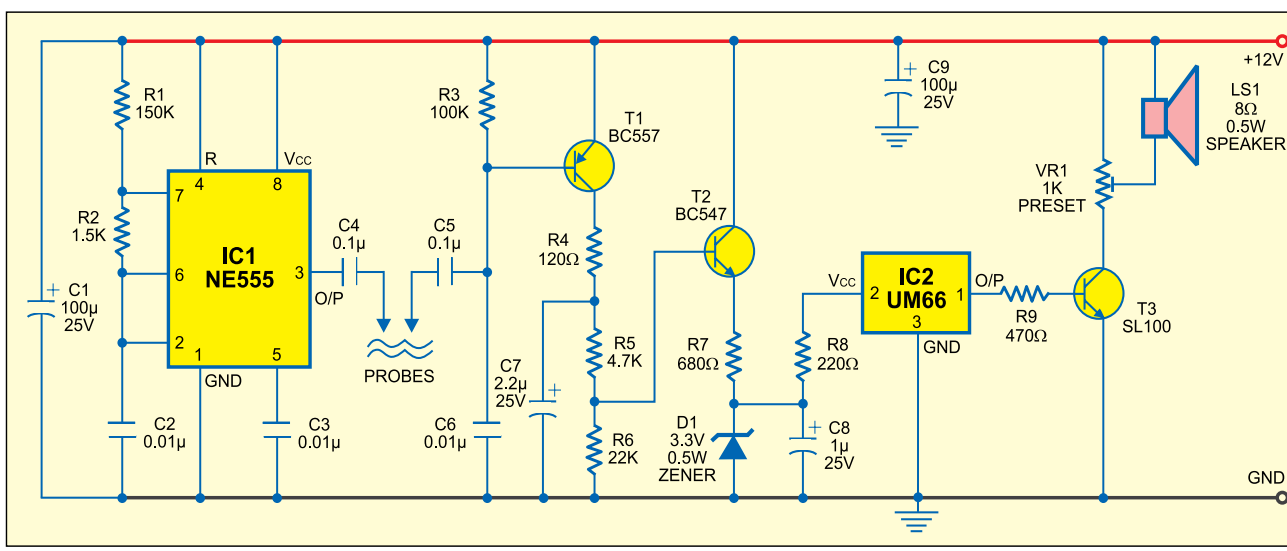


Fig. 2: Liquid level alarm

REMOTE-CONTROLLED POWER-OFF SWITCH

■ **DEBARAJ KEOT**

Remote controllers for various audio/video systems are usually provided with a power 'on'/'off' or standby-mode selector button. But turning the system off from the remote handset actually does not cut off the whole system from mains. Some circuitry inside the system continues to get power from mains even when the power is turned off using the remote handset. One needs to turn off the mechanical switch provided on the system's front panel or wall outlet in order to turn off the entire system.

Also, accessories like TV boosters, stabilisers and additional amplispeaker systems cannot be turned off from the remote handset. And it is very annoying to get out of bed to switch off mains after watching some programme on TV or listening to music.

The circuit given here can disconnect the entire system along with the accessories, including the circuit itself, from mains using the remote for the audio or video systems.

The circuit consists of a timer IC NE555, a decade counter IC HCF4017, three BC548 transistors, an infrared (IR) sensor IC TSOP1738 and a few discrete components. Transformer X1, diodes D1 and D2, and capacitor C1 form power supply for the circuit. Zener diode ZD1 provides regulated voltage to IR sensor TSOP1738 (IC3).

Timer IC NE555 (IC1) is configured as an astable multivibrator that produces a clock pulse every two seconds. The clock pulse is fed to decade counter IC HCF4017 (IC2), whose Q7 output is inverted by transistor T1 and applied to the base of transistor T2 to drive the relay.

The output of sensor IC3 is used to drive transistor T3 and activate the relay via transistor T2.

The outputs of transistors T1 and T3 are ORed and the resultant is applied to transistor T2. Thus if any one or both the inputs connected to the base of transistor T2 are high, relay RL1 energises. The relay de-energises if both the inputs to transistor T2 go low.

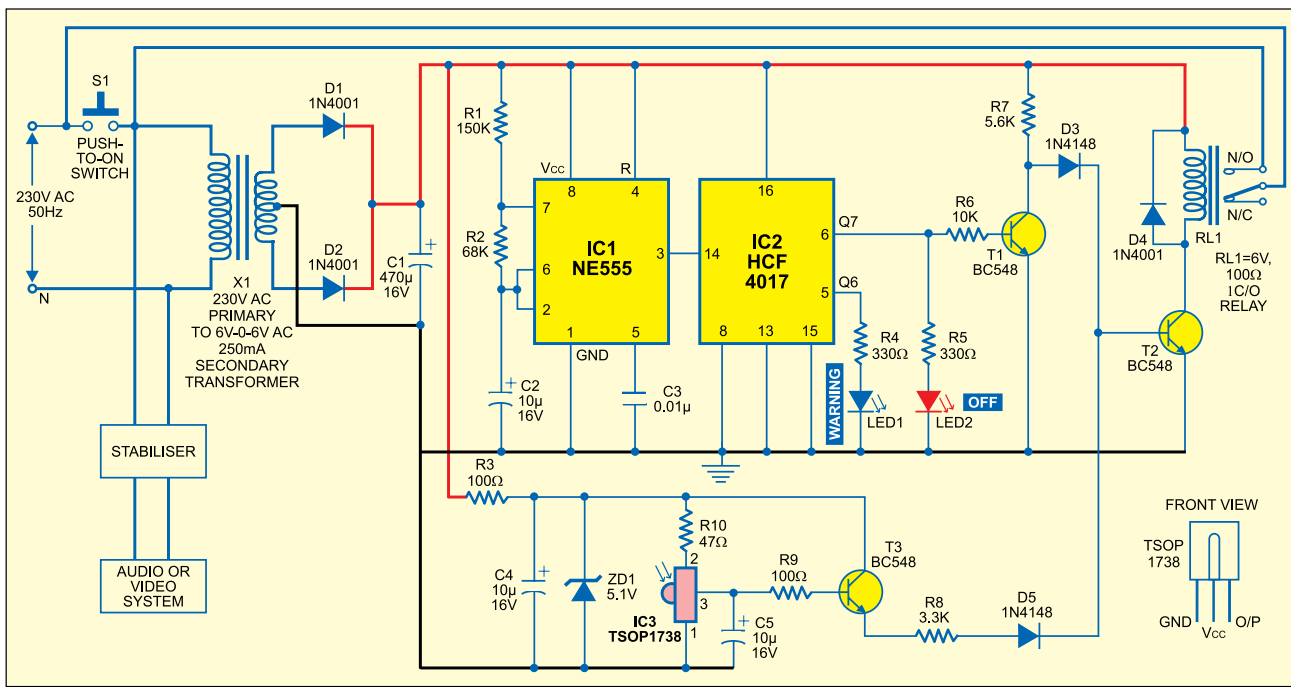
Initially, to switch on mains supply for the audio/video system and the circuit itself, pushbutton switch S1 is pressed momentarily.

Normally, the output of IR receiver module IC3 is high when it is not being activated by a remote, and the relay energises to close the N/O contact and place a short across switch S1. This circuit and the load continue to get power through the N/O contact of relay RL1 even when pushbutton S1 is released.

At the same time, the output of IC2 starts scrolling around its output pins, i.e., pins 5 and 6 go high and low alternately for the clock pulses received. When Q6 output goes high the 'warning' LED (yellow) glows, and when Q7 output goes high the 'off' LED (red) glows.

Yellow LED (LED1) indicates that it's time to switch off the audio or video system.

The entire system can be turned off by pressing any key on the hand-



set once when the red LED (LED2) is glowing. The reason is that red LED2 glows when the Q7 output of IC2 is high. Due to this, the output of transistor T1 is low. Now if any key on the remote handset is pressed, the sensor output goes low for a while. Since both the inputs connected to the base of transistor T2 become low at this time, the transistor stops conducting and the relay de-energises. As a result, the N/O contact of the relay opens to switch off the circuit and the load connected across the transformer's primary winding.

Any normal operation (increasing/decreasing volume, changing channels, etc) can be performed when either both the LEDs are 'off' or the remote is not oriented towards IR receiver module IC3. ●

ZENER VALUE EVALUATOR

■ V. GOPALAKRISHNAN

Using this simple circuit and a known-value zener diode, you can find the breakdown voltage value of any zener diode. The circuit is divided into two sections: zener evaluator and display unit. Regulated 12V and 5V are required to power the zener evaluator section, while the display section works off only 5V. Connect +5V, point A and ground of the zener evaluator section to the respective terminals of the display section.

The zener evaluator circuit (Fig. 1) comprises a linear ramp generator built around timer NE555 and an astable multivibrator built around another NE555. The resistor of the monostable is replaced with a constant-current source formed by transistor T1. Capacitor C2 is charged linearly by the constant-current source formed by transistor T1.

The time period T of the linear ramp generated by IC1 at its pin 6 across capacitor C2 is given by:

$$T=\frac{(2/3)\times V_{CC}\times R5\,(R6+VR2)\times C2}{R6\times V_{CC}-V_{BE}\,(R6+VR2)} \quad \text{.....Eq. (1)}$$

On substituting the values shown in Fig. 1, you get:

T= 0.15 second (approx.)

This value is equal to T_{on} of the monostable without connecting the zener at the control voltage terminal pin 5.

Now connect the zener to the control voltage terminal and trigger the monostable (IC1) by momentarily pressing switch S1. The output pulse width of IC1 is fed to the astable multivibrator (IC2). The time period of the astable multivibrator is around 7 milliseconds (ms) and it oscillates as long as the ramp output of IC1 is high.

The display unit comprising decade counter ICs 74LS90, decoder/driver ICs 74LS47 and 7-segment common-anode displays LTS542 is shown in Fig. 2. Decade counters IC3 and IC4 count the frequency applied on clock pin 14 of IC3 from pin 3 of IC2.

IC 74LS90 is a 4-bit ripple decade counter. When the output of IC3 is '10' (1001), it provides clock at pin 14 of IC4 (via AND gate N1) for further counting.

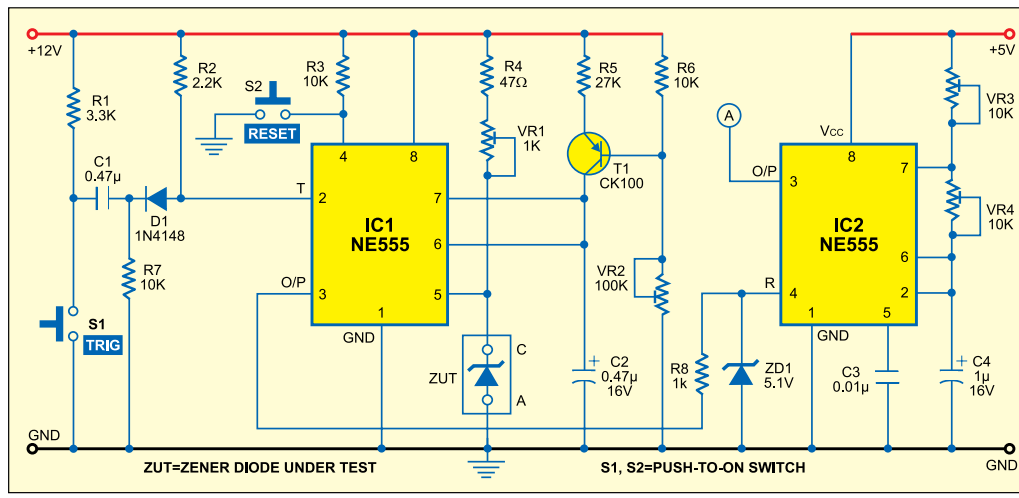


Fig. 1: Circuit diagram of zener evaluator section

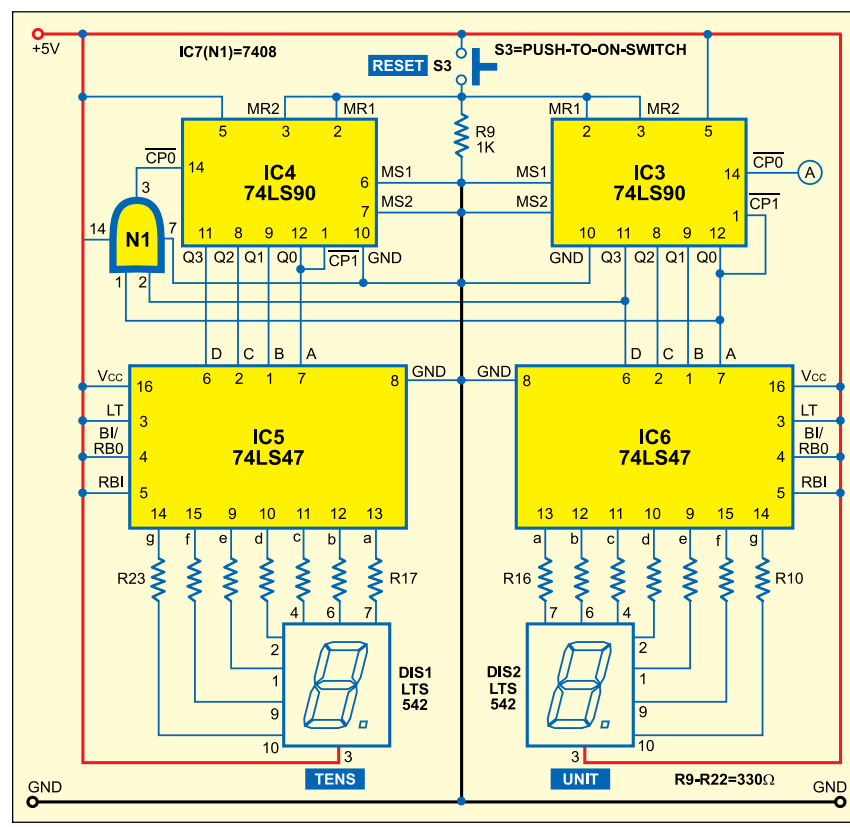


Fig. 2: Display section

For resetting IC3 and IC4, simply press reset switch S3 momentarily.

The outputs of decade counters IC3 and IC4 are connected to 7-segment decoders/drivers IC6 and IC5, respectively, which, in turn, are connected to common-anode displays DIS1 and DIS2 for displaying the frequency of astable multivibrator IC2 used to evaluate the unknown value of the zener diode.

Let's say the counter counts up to N1 for the known zener diode breakdown voltage value X1 and up to N2 for the unknown zener diode value X2. Now, you can calculate the value of the unknown zener diode from the following relationship:

$$X2 = \frac{X1 \times N2}{N1} \quad \text{..... Eq. (2)}$$

Suppose you have a zener diode rated at 6.8V (X1). Insert it at the position marked 'ZUT' of Fig. 1 and press trigger switch S1 momentarily. The counter counts up to, say, '29' (N1), which is shown on the display. Now remove the 6.8V zener and insert the zener of unknown value (X2). The display now shows, say, 15 counts (N2).

From Eq. (2), the value of the unknown zener (X2) can be calculated as 3.5V. ●

SIMPLE MOSFET-BASED CFL

■ **N.S. HARISANKAR, VU3NSH**

This CFL circuit uses only two semiconductor devices and few passive components, which keeps the cost low and simplfies the circuit. Low power consumption is its another advantage.

The circuit works off a 12V, 7AH battery and is built around CMOS hex-inverter IC CD4069 (IC1). Using CMOS IC, the power consumption of the main stage (oscillator) is limited to a few microwatts. The IC is configured as an R-C oscillator with four of its gates connected in parallel to enhance its output drive capability. Its high output can drive TTL loads.

Gates N1 and N2 form an R-C oscillator and the remaining four gates (N3 through N6) are connected in parallel to provide a high output current to the MOSFET switch. The R-C oscillator has only two external components and its output frequency (f) can be roughly calculated using the following equation:

$$f = \frac{0.5}{R1 \times C1}$$

The recommended frequency for a ferrite transformer-based CFL is 18 to 35 kHz. To vary the frequency, you can change the value of resistor R1 in the R-C oscillator (see the table).

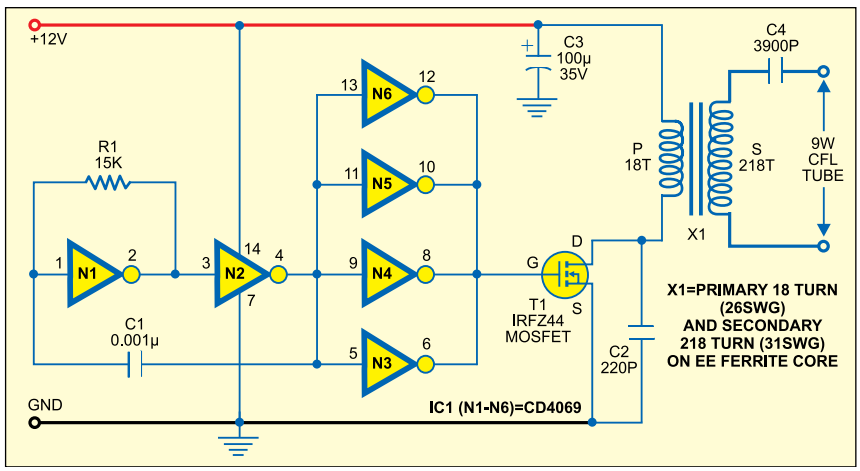


Fig. 1: Circuit of MOSFET-based CFL

Frequencies for Different R-C Combinations

R1	C1	F
20kΩ	1 kpF	25 kHz
15kΩ	1 kpF	35 kHz

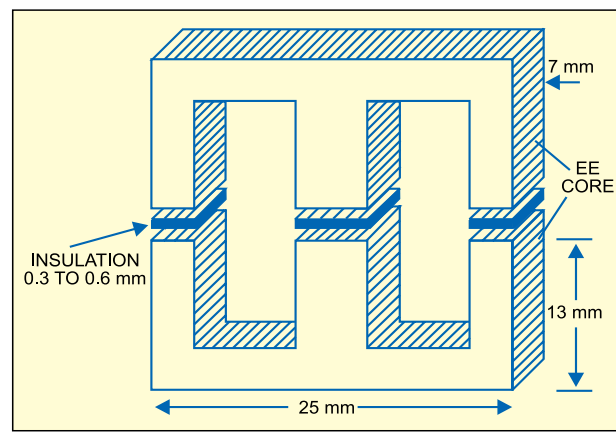


Fig. 2: EE ferrite dimensions

For R1 (15 kilo-ohms) and C1 (0.001 µF) used in this circuit, the selected frequency is 35 kHz.

Resistor R1 connected between pins 1 and 2 of gate N1 provides a negative DC feedback and biases the inverter to a linear region where it works as an amplifier. Capacitor C1 connected between pins 1 and 4 of IC1 provides a positive feedback to enable oscillations.

The n-channel enhancement-mode MOSFET IRFZ44 (T1) is readily available in the market. Transformer X1 is built around an EE-type, 25×13×7mm ferrite core. Use good insulation between the primary and secondary windings. After winding the transformer coils, put some insulating sheet or paper at the edges (tips) of the EE-cores as shown in Fig. 2. This insulation gap between the two cores helps to achieve maximum brightness with minimum current drain.

High-tension (HT) coupling capacitor C4 limits the current to the lamp. Capacitor C2 between drain and ground clips any ripple voltage to give a linear waveform.

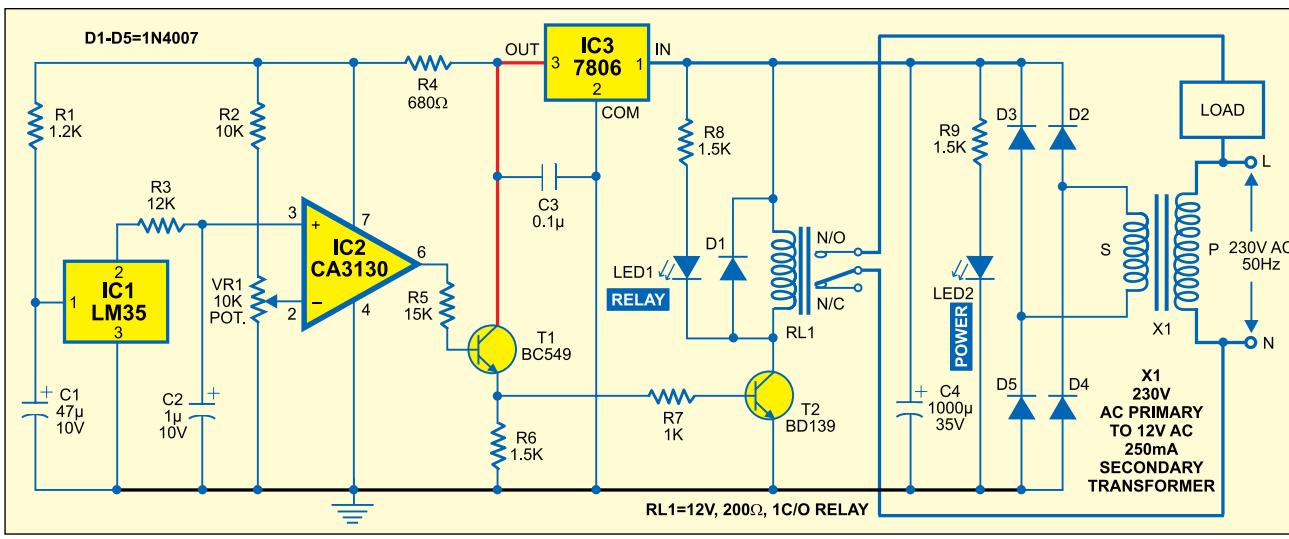
The performance of this CFL depends on the type of the CFL, EE core, oscillation frequency, ferrite core gap, etc. Never use a Schmitt inverter (40106) for this circuit. Use a base for the IC and handle the MOSFET carefully. Before soldering the MOSFET, make sure that the R-C oscillator is oscillating properly. Connect the MOSFET only if the oscillations are proper. In case you don't use an IC base, make sure the soldering iron is earthed properly while soldering the IC.

Lab Note. A Philips 9W CFL was used for testing the circuit. ●

HEAT-SENSITIVE SWITCH

■ **M.K. CHANDRA MOULEESWARAN AND MISS KALAI PRIYA**

At the heart of this heat-sensitive switch is IC LM35 (IC1), which is a linear temperature sensor and linear temperature-to-voltage converter circuit.



The converter provides accurately linear and directly proportional output signal in millivolts over the temperature range of 0°C to 155°C. It develops an output voltage of 10 mV per degree centigrade change in the ambient temperature. Therefore the output voltage varies from 0 mV at 0°C to 1V at 100°C and any voltage measurement circuit connected across the output pins can read the temperature directly.

The input and ground pins of this heat-to-voltage converter IC are connected across the regulated power supply rails and decoupled by R1 and C1. Its temperature-tracking output is applied to the non-inverting input (pin 3) of the comparator built around IC2. The inverting input (pin 2) of IC2 is connected across the positive supply rails via a voltage divider network formed by potmeter VR1.

Since the wiper of potmeter VR1 is connected to the inverting input of IC2, the voltage presented to this pin is linearly variable. This voltage is used as the reference level for the comparator against the output supplied by IC1.

So if the non-inverting input of IC2 receives a voltage lower than the set level, its output goes low (approximately 650 mV). This low level is applied to the input of the load-relay driver comprising npn transistors T1 and T2. The low level presented at the base of transistor T1 keeps it non-conductive. Since T2 receives the forward bias voltage via the emitter of T1, it is also kept non-conductive. Hence, relay RL1 is in de-energised state, keeping mains supply to the load 'off' as long as the temperature at the sensor is low.

Conversely, if the non-inverting input receives a voltage higher than the set level, its output goes high (approximately 2200 mV) and the load is turned 'on.' This happens when IC1 is at a higher temperature and its output voltage is also higher than the set level at the inverting input of IC2. So the load is turned on as soon as the ambient temperature rises above the set level. Capacitor C3 at this pin helps iron out any ripple that passes through the positive supply rail to avoid errors in the circuit operation.

By adjusting potmeter VR1 and thereby varying the reference voltage level at the inverting input pin of IC1, the temperature threshold at which energisation of the relay is required can be set. As this setting is linear, the knob of potmeter VR1 can be provided with a linear dial calibrated in degrees centigrade. Therefore any temperature level can be selected and constantly monitored for external actions like turning on a room heater in winter or a room cooler in summer. The circuit can also be used to activate emergency fire extinguishers, if positioned at the probable fire accident site.

The circuit can be modified to operate any electrical appliance. In that case, relay RL1 must be a heavy-duty type with appropriately rated contacts to match the power demands of the load to be operated. ●

TRANSISTOR TESTER

■ V. GOPALAKRISHNAN

You can test both npn and pnp transistors using this circuit. The circuit indicates whether the transistor is good, open or shorted through two light-emitting diodes (LEDs).

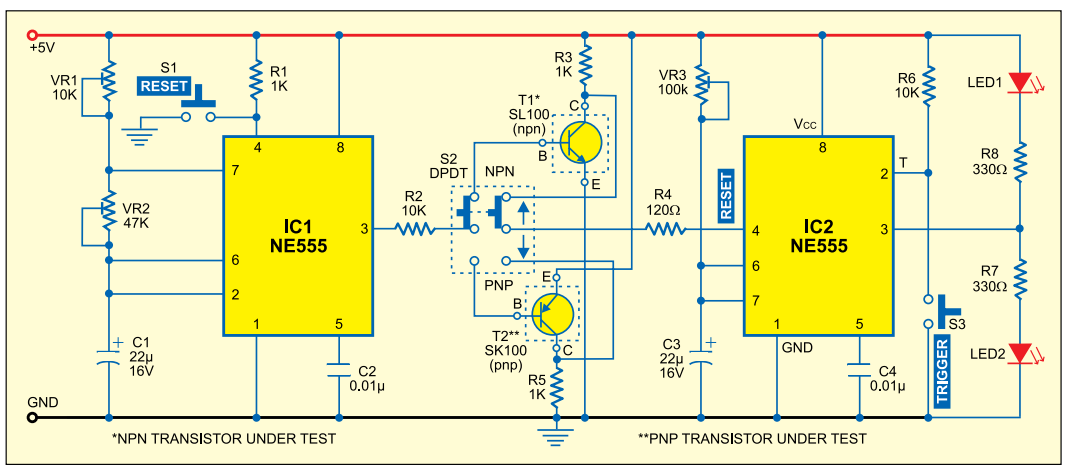
The circuit comprises two NE555 timer ICs: one (IC1) is wired in the astable mode and the other (IC2) in the monostable mode. The time period of the astable multivibrator is around 0.5 second. Its output goes to the base of the npn/pnp transistor under test via DPDT switch S2.

Switch S2 selects the npn/pnp transistor you are going to test, which means that at a time only an npn or a pnp transistor can be tested. The collector of npn or pnp transistor goes to reset pin 4 of the monostable (IC2). Switch S3 is used to trigger the monostable. The time period of the monostable multivibrator is around two seconds.

To test a transistor, insert it at the appropriate place shown within dotted lines and slide switch S2 towards the transistor type (npn or pnp) being tested. From glowing of LED1 and LED2 on triggering of the monostable via switch S3, you can infer whether the transistor is good, short or open-circuited, as shown in the table. ●

Transistor Assessment from Glowing of LEDs

Switch S3	npn transistor	LED1	LED2
Kept pressed	Good	Flickers	Flickers
Pressed momentarily to trigger IC2	Collector-emitter short	Glows	Doesn't glow
Pressed momentarily to trigger IC2	Collector-emitter open	Remains 'off' for two seconds, then glows	Glows for the set time (say, two seconds) and then turns off
Switch S3	**pnp transistor**	**LED1**	**LED2**
Kept pressed	Good	Flickers	Flickers
Pressed momentarily to trigger IC2	Collector-emitter short	Remains 'off' for two seconds, then glows	Glows for the set time (say, two seconds), then turns off
Pressed momentarily to trigger IC2	Collector-emitter open	Glows	Remains 'off'



WATER-TANK OVERFLOW INDICATOR

■ C.H. VITHALANI

Water is a vital but scarce natural resource. To prevent water wastage, this water-tank overflow indicator comes in handy. It gives audio as well visual alarm whenever the water tank overflows.

Fig. 1 shows the water-tank overflow indicator circuit and Fig. 2 shows the power supply circuit.

In the power supply unit, mains AC is stepped down by transformer X1 to deliver secondary output of 9V-0-9V AC at 300 mA. The transformer output is rectified by a full-wave bridge rectifier comprising diodes D1 through D4 and filtered by capacitors C1 and C2 to provide +9V at '+B' point and –9V at '–B' point. Connect '+B,' '–B' and 'GND' terminals of the power supply unit to the respective terminals of the water-tank overflow indicator circuit.

The circuit is built around op-amp LM741 (IC1), which is used as a comparator. The pin configuration of melody generator IC1(UM66) is shown in Fig. 3.

When water in the tank is below the metal plate sensors, inverting pin 2 of IC1 is at a higher potential than non-inverting pin 3. Output pin 6 of the op-amp is low and there is no music from programmable melody generator IC UM66 (IC2). Transistor BC547 (T1) remains cut-off and therefore LED1 doesn't glow and the loudspeaker remains silent.

When water in the tank touches the metal plate sensors, it extends ground to pin 2 of IC1. Now pin 3 of IC1 is at a higher potential than pin 2. The high output of the op-amp generates 3.1V across zener diode ZD1. Melody generator IC2 produces a melody, which drives the transistor to light up LED1 and sound an alarm from the loudspeaker. Rectifier diode D5 is used to prevent negative polarity to the cathode of the zener diode. ●

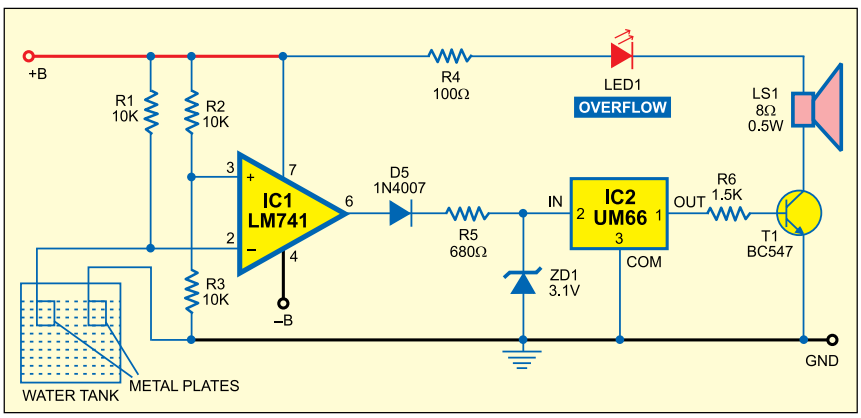


Fig. 1: Circuit of the water-tank overflow audio-visual indicator

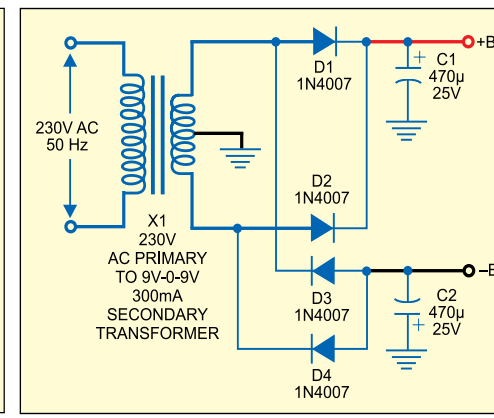


Fig. 2: Power supply circuit

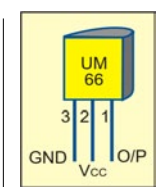


Fig. 3: Pin configuration of UM66

SIMPLE SMOKE DETECTOR

■ **PRADEEP G.**

This simple smoke detector is highly sensitive but inexpensive. It uses a Darlington-pair amplifier employing two npn transistors and an infrared photo-interrupter module as the sensor. The circuit gives audio-visual alarm whenever thick smoke is present in the environment.

The photo-interrupter module (H21A1) consists of a gallium-arsenide infrared LED coupled to a silicon phototransistor in a plastic housing. The slot (gap) between the infrared diode and the transistor (see Fig. 1) allows interruption of the signal with smoke, switching the module output from 'on' to 'off' state.

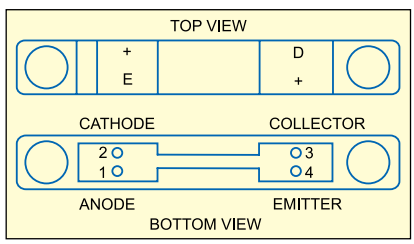


Fig. 1: Top and bottom views of the photo-interrupter module (H21A1)

The circuit of the smoke detector is shown in Fig. 2.

When the smoke enters the gap, the IR rays falling on the photo-transistor are obstructed. As a result, the phototransistor stops conducting and the Darlington-pair transistors conduct to activate the buzzer and light up LED1.

When the smoke in the gap is cleared, light from the IR LED falls on the phototransistor and it starts conducting. As a result, Darlington-pair transistors stop conducting and the buzzer and LED1 turn off.

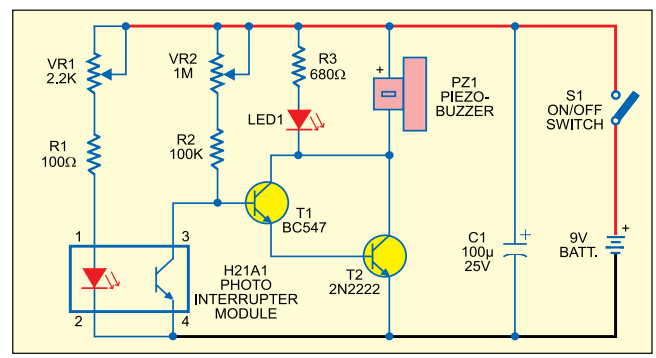


Fig. 2: Schematic of the smoke detector

For maximum sensitivity, adjust presets VR1 and VR2. VR1 is used to control the sensitivity of the photo-interrupter module, while VR2 is used to control the sensitivity of Darlington-pair transistors. ●

SENSITIVE VIBRATION DETECTOR

■ T.K. HAREENDRAN

This vibration detector is realised using readily available, low-cost components. One of its many applications is in a rolling shutter guard for offices and shops. The detector will sense vibration caused by activities like drilling and switch on the connected load (bulb, piezobuzzer, etc) to alert you.

The circuit works off a 6V battery or 6V regulated power supply and uses a piezoceramic element as the vibration detector. The same is easily available from electronics/telephone component vendors or you can take it out from an active buzzer.

Initially, when the power is switched on, decade counter IC1 is reset by power-on-reset components C2 and R1. As a result, Q0 output (pin 3) of IC1 goes high and the entire circuit is in idle state. LED1 indicates the power status.

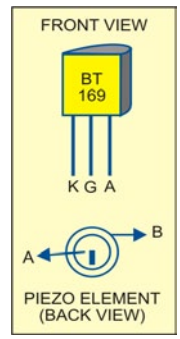


Fig. 2: Pin configuration of SCR1 BT169 and back view of the piezo element

In the event of vibrations, IC2 is clocked by the pulses from the piezoceramic element connected to its clock pin 14. Q1 through Q9 outputs of IC2 are fed to relay-driver switching transistor T1 through diodes D1 through D9 connected in OR mode.

Immediately after clocking, any of the outputs Q1 through Q9 would go high and npn transistor T1 would conduct. As a result, SCR1 is fired through its gate. This, in turn, energises relay RL1. The relay contacts can be used to switch any alarm device to indicate vibration detection. The circuit can be reset by momentarily pressing switch S1.

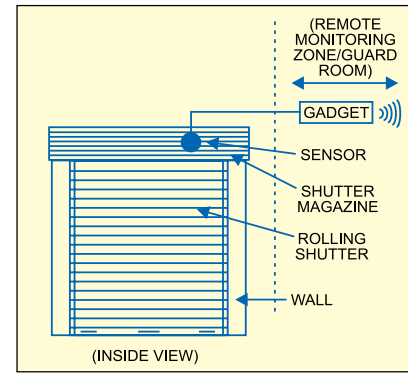


Fig. 3: Arrangement for rolling shutter guard for shops, offices and banks

Zener diodes ZD1 and ZD2 at the clock input of IC1 are used for protection against high voltage input. In the case of repeated false triggering of IC1, add a 100nF capacitor in parallel to the piezoceramic element.

The pin configuration of SCR BT169 and the back view of the piezo element are shown in Fig. 2. Fig. 3 shows suggested location of the vibration detector for rolling shutters of banks, shops, etc. ●

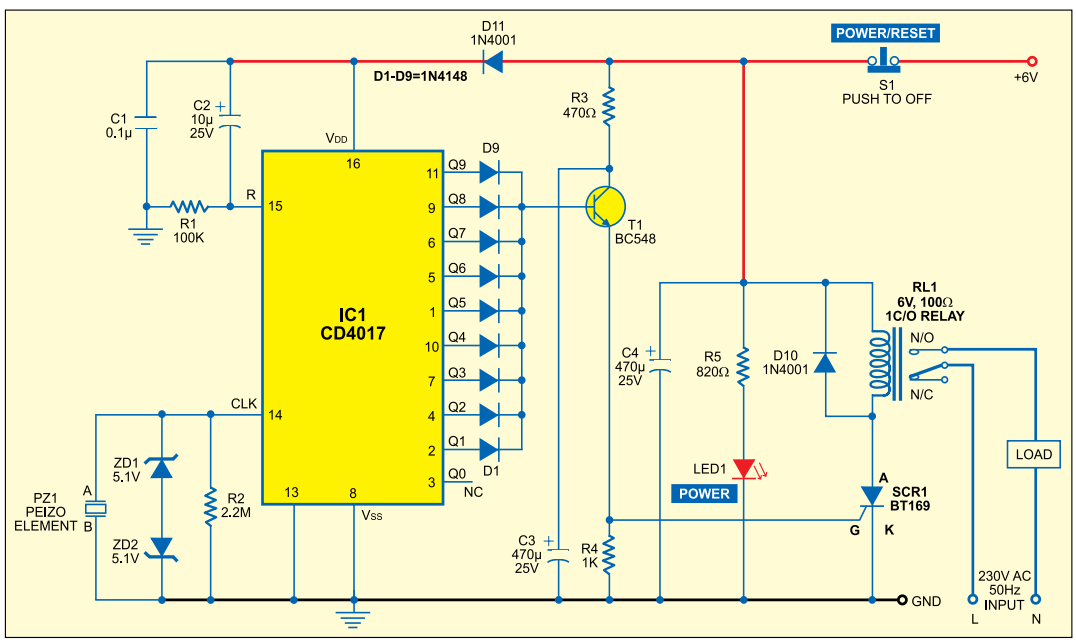


Fig. 1: Circuit of the sensitive vibration detector

SOFT SWITCH

■ **PRADEEP G.**

This circuit lets you switch on/off an AC lamp or any other load by pressing a normally open microswitch. The current passing through the switch is very small.

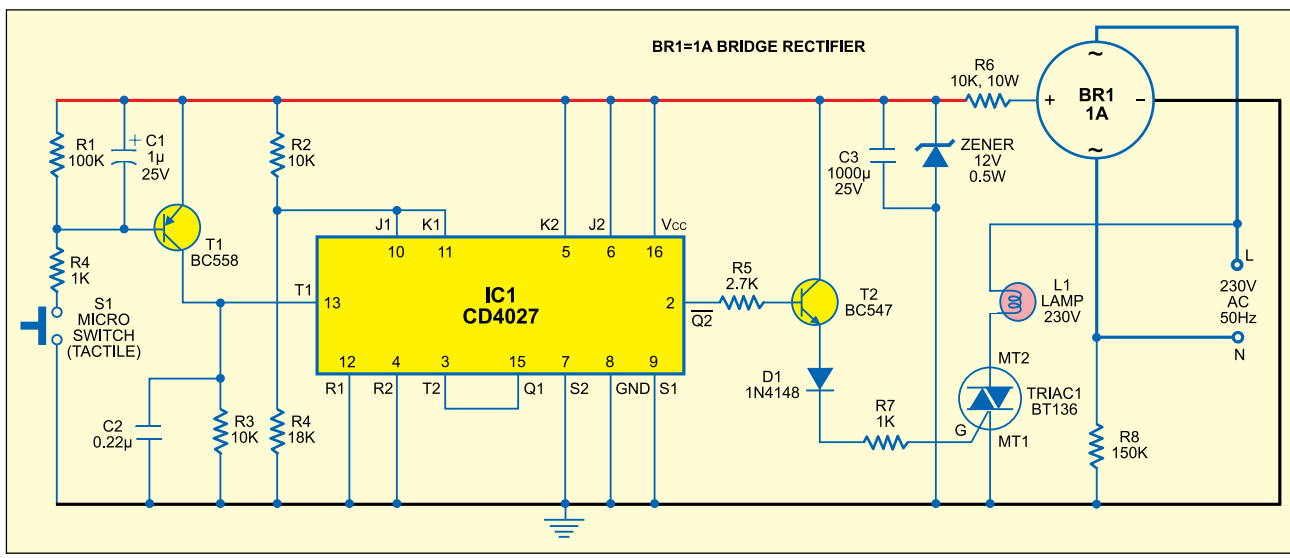
For each press of the microswitch, the output of the IC, which is a CMOS J-K master-slave flip-flop, toggles between high and low states. These digital variations are amplified by an npn transistor to drive a triac. The triac directly activates the AC lamp.

The 12V DC required for the circuit is derived from AC mains. BT136 is a general-purpose triac.

The circuit can be used as a staircase light switch. Connect two microswitches in parallel and fix one switch at the top of the staircase and the other at the bottom. When any of the microswitches is pressed, the lamp will turn on and off alternately.

Since this circuit is not isolated from AC mains, don't touch it after connecting the power supply.

Note. Use of a Texas or ST make IC 4027 is recommended. ●



AUTOMATIC-OFF TIMER FOR CD PLAYERS

■ **SURESH KUMAR K.B.**

Are you in the habit of falling asleep while listening to music? If yes, you'll love this circuit. It will automatically start functioning when you switch off your bedroom light and shall turn your CD player 'off' after a predetermined time. In the presence of ambient light, or when you switch on light of the room in the morning, the CD player will again start playing. Unlike the usual timers, you don't have to set this timer before sleeping.

The circuit derives its power directly from the bridge rectifiers. When 'on'/'off' switch S1 is closed, LED1 glows to indicate that the circuit is powered 'on.'

In the presence of light, the resistance of the light-dependent resistor (LDR1) is low, so transistor T1 conducts to drive transistor T2 into cut-off state and the timer circuit remains inactive.

The collector of transistor T2 is connected to reset pin 12 of IC CD4060 (IC1) via signal diode D5. IC CD4060 is a 14-stage ripple counter with a built-in oscillator. The time period of oscillations (t) is determined by capacitor C3 and resistor R8 connected to pins 9 and 10 of IC1, respectively, as follows:

t=2.3RC

where 'R' is the value of resistor R8 and 'C' is the value of capacitor C3.

When transistor T2 is cut-off, its collector voltage is high. So pin 12 of IC1 is high and IC1 is in reset condition.

When light is switched off, the resistance of LDR1 increases, driving transistor T1 into cut-off state. The collector voltage of transistor T1 goes high to light up LED2 (indicating that the timer circuit is enabled) and transistor T2 starts conducting. As the collector voltage of transistor T2 goes low to around 0.2V, ground potential becomes available at reset pin 12 of IC1. The low state at pin 12 enables the oscillator and it starts counting. LED3 at pin 7 of IC1 starts blinking. Its blinking frequency depends on the R-C components connected between its pins 9 and 10.

LDR1	Timer LED2	Reset pin 12	Count LED3
Light	Off	High	Off
Dark	On	Low	Blink

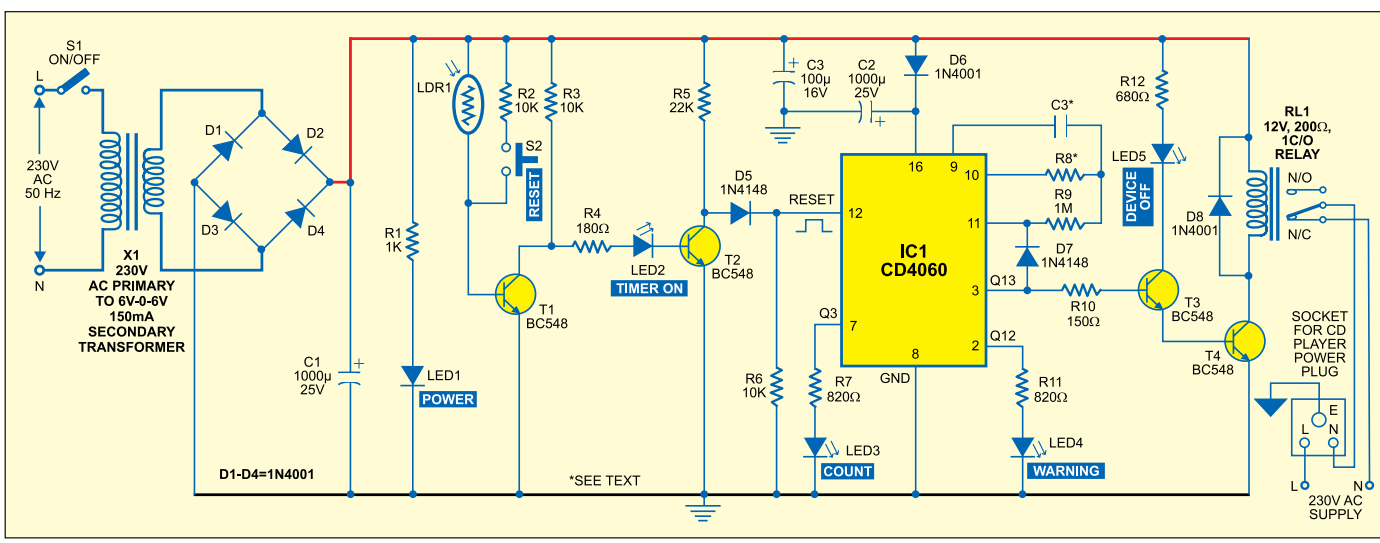
The status of LED2 and LED3 in the circuit with light falling and not falling on LDR1 is given below:

During counting, in case the power fails momentarily, capacitor C2 (1000µF) will provide the necessary power backup for IC1. That is, during the period, pin 3 of IC1 is low. When output pin 3 of IC1 goes high, the relay is energised through transistors T3 and T4 and, at the same time, counting is disabled by the feedback from pins 3 through 11 (clock input) of IC1 via signal diode D7. That is, due to the feedback, output pin 3 remains high unless another high-to-low pulse is received at its reset pin 12.

After the relay is energised, there will be no AC power in the socket. The glowing of LED5 indicates that your CD player has been switched off.

The desired 'off' time period for the timer circuit can be set by choosing proper values of resistor R8 and capacitor C3. If R8 is 680 kilo-ohms and C3 is 0.22 µF, the 'off' time period is around 45 minutes.

The glowing of LED4 gives the warning that your CD player is going to be switched off shortly. In case you want to extend the timer setting for another round, just press reset switch S2 momentarily. LED4 stops glowing and counting starts again from the initial stage. ●



AUTOMATIC WASHBASIN TAP CONTROLLER

■ **AKSHAY MATHUR AND ABHAY MATHUR**

Make your washbasin tap work automatically when you put your hands just below the water tap outlet. This infrared-based system detects any interruption of the IR rays by your hands or utensil and water automatically starts flowing out of the tap.

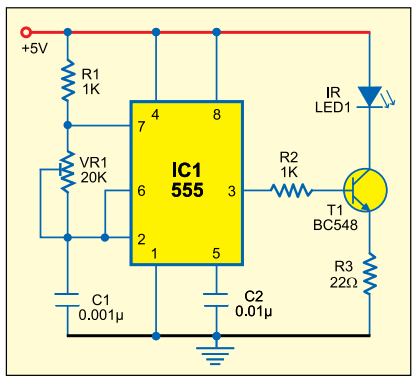


Fig. 1: Transmitter circuit

The circuit is built around 555 timers and comprises transmitter and receiver sections. Both the transmitter and the receiver work off 5V DC. The IR rays continuously emitted by the transmitter fall on the receiver. As soon as an obstacle comes in between the receiver and the transmitter, interrupting the IR rays, the output of the IR sensor goes low momentarily to trigger the timer circuit in the receiver and water comes out for eleven seconds through the tap.

The transmitter is built around timer IC 555, which is used as an astable multivibrator to generate around 38 kHz frequency (see Fig. 1). The timer output is fed to transistor T1, which drives the IR LED (LED1). Note that IR LED1 must be properly oriented towards the IR sensor module of the receiver circuit. Its transmission wavelength of 900 to 1100 nm lies in the peak receptivity range of TSOP1738 receiver module.

The receiver circuit comprises the sensor module, monostable timer and relay driver circuit (see Fig. 2). The sensor module TSOP1738 is sensitive to IR radiation modulated at 38 kHz. Its normally high output goes momentarily low when any IR radiation is detected or interrupted.

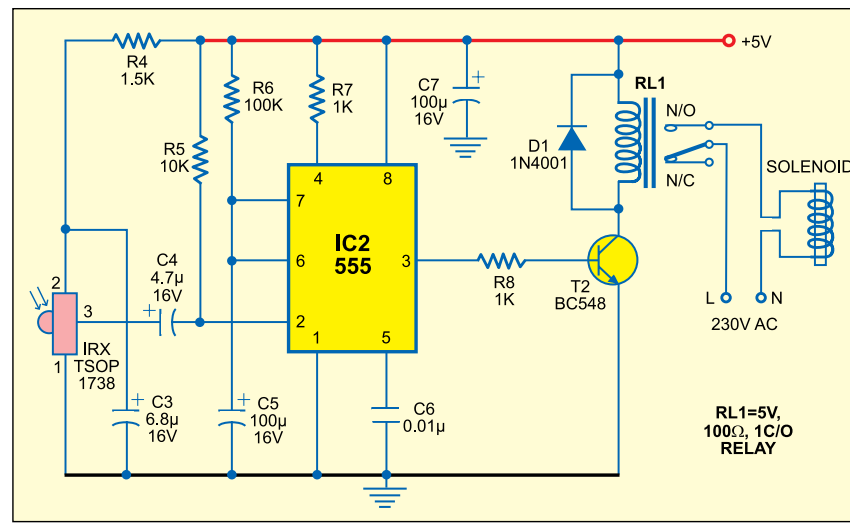


Fig. 2: Receiver circuit

When IR rays falling on the receiver are interrupted, the sensor output goes low momentarily to trigger timer IC2. The output of the timer goes high for eleven seconds and the relay drives the solenoid. During this time period, energisation of the solenoid lifts up the valve fitted in the pipe to let water flow out of the tap. Solenoid valves used specifically for this purpose are shown in Fig. 3.

The relay driver circuit consists of resistor R8, transistor BC548 (T2) and free-wheeling diode D1. Diode D1 protects the relay from damage by high voltages generated by the back emf when the relay is de-energised.

The time period for which the timer goes high can be calculated as follows:

$T_{on}=1.1\ R_6C_5$

$=1.1\times100\times10^{3}\times100\times10^{-6}$

$=11$ seconds

Use shielded wires or leads for installing the IR LED and the IR sensor at opposite sides of the washbasin. Install the IR LED and IR sensor around half a metre apart such that the IR rays transmitted by the IR LED directly fall on the IR sensor. Now switch on the power supply to the circuit.

Fig. 3: Mains 230V AC 2/2-way semi-pilot, diaphragm type, solenoid valves

When you put your hands between the IR LED and IR sensor, the relay energises to make the solenoid open up the valve and water flows out of the tap. ●

REAR-VIEW MONITOR

■ T.K. HAREENDRAN

CMOS colour micro-cameras are readily available from component vendors at reasonable prices. Using such a camera (model FQY888C), you can make a rear-view monitor for your car as described here.

The circuit works off the DC supply directly available from the car's battery. Resistor R1 limits the inrush current and diode D1 protects against wrong polarity. Capacitors C1 and C3 act as the noise suppressor and reservoir filter, respectively.

Before connecting the circuit to the car battery and switching on the car TV, connect points A through E of Fig. 1 to the respective points of the car TV and camera (see Figs 2 and 3) through the phono plug and phono socket.

When the car is moving forward, transistor T1 doesn't conduct and relay RL1 remains de-energised. As a result, external video from the car's AV system connects to the car's TV video input, allowing you to enjoy your favourite programmes. LED2 glows to indicate that the car TV is showing the external AV programme.

When reversing the car, the reverse-lamp supply is turned on as per the mechanical arrangement of the gear lever (not shown in the figure) and positive supply from the lamp terminal is fed to the base of relay-driver transistor T1 via diode D2 and resistor R3. As a result, relay RL1 energises and the video signal output from the camera connects to the car TV via normally-open contact N/O2 of relay RL1 and the TV starts showing rear view of the car.

N/O1 contacts of relay RL1 bypass the master power-'on'/'off' switch (S1) of the car TV to enable the TV even if its indicator LED2 is switched off by the relay contacts. Power supply for the CMOS camera is provided by the car battery through IC1. LED3 raises the output voltage of IC1 to near 11.2V and indicates that the camera is working.

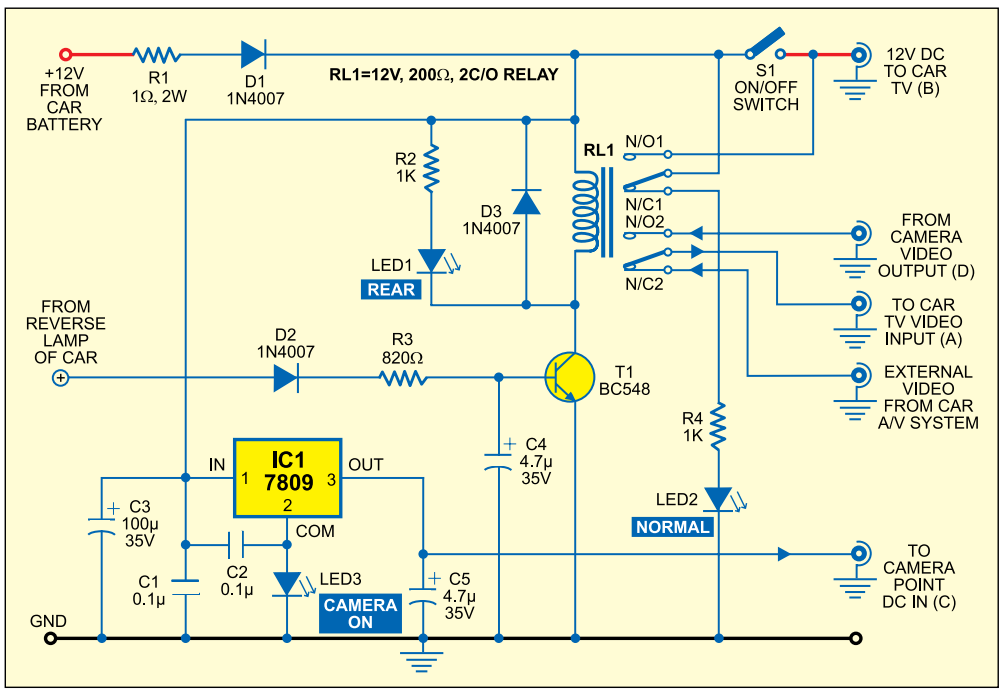


Fig. 1: Rear-view monitor

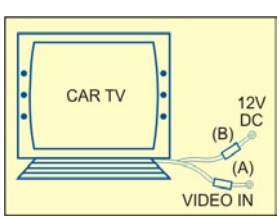


Fig. 2: Car TV

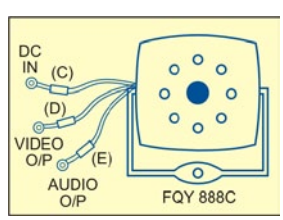


Fig. 3: Camera

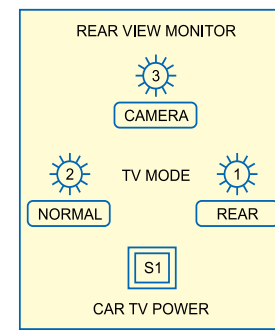


Fig. 4: Front layout of control panel

For safety, you can feed the camera supply through the third contacts of relay RL1 (not shown in the circuit) or use an 'on'/'off' switch between pin 1 of IC1 and the cathode of diode D1.

The circuit can be easily assembled on a medium-size veroboard. You can make it compact by using a PCB-mountable relay. Fixing the camera in the car and focusing it need some patience.

The FQY 888C CMOS camera used here was procured from a component vendor called Eastern Enterprises, Chennai. Since it operates off 6 to 12V DC (120 MW), around 11.2V is applied to it. The camera has three leaded outputs: a yellow RC socket (marked 'D') for video output, a white RC socket (marked 'E') for audio output and a red RC socket (marked 'C') for DC supply. If you are using a different model, carefully study the product catalogue before final wiring. ●

OVER-SPEED INDICATOR

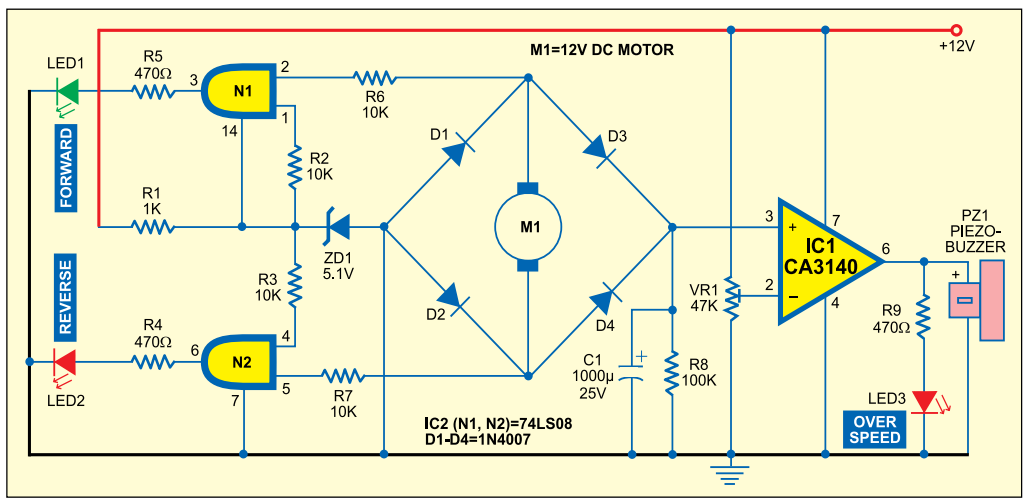
■ V. DAVID

This circuit is designed for indicating over-speed and direction of rotation of the motor used in mini hand tools, water pump motors, toys and other appliances.

A 12V DC motor (M1) is coupled to the rotating part of the appliance with a suitable fixing arrangement. When the motor rotates, it develops a voltage.

This over-speed indicator is built around operational amplifier CA3140 (IC1). Set the reference voltage (depending on the desired speed) by adjusting preset VR1 at pin 2 of IC1. When the voltage developed at pin 3 of IC1 is higher than the reference voltage at pin 2, output pin 6 of comparator IC1 goes high to sound piezobuzzer PZ1 and light up LED3.

The rotation indicator circuit is built around AND gate 74LS08 (IC2). Pin 2 of gate N1 goes high when the motor rotates in forward direction, while pin 1 of gate N1 is pulled high via resistor R2. When both pins 1 and 2 are high, output pin 3 of gate N1 goes high to light up LED1. Similarly, pin 5 of gate N2 goes high when the motor rotates in reverse direction. When both pins 4 and 5 are high, output pin 6 of gate N2 goes high to light up LED2. ●



VERSATILE WATER-LEVEL CONTROLLER

■ **A. SHAFEEK AHAMED**

This simple, economical and versatile circuit switches on the motor pump when water in the overhead tank falls below the lowest level and turns it 'off' when the tank is full. Moreover, if the pump is running dry due to low voltage, it sounds an alarm to alert you to switch off the motor through the inverter and driver circuits. The transistor switch circuitry monitors the flow of water and raises an alarm if the pump runs dry.

Power supply is obtained through step-down transformer X1, diodes D1 through D4, capacitor C1, series current-limiting resistor R1, regulator IC1, and noise-filtering capacitors C2 and C3.

The set-up for the water-level energises. The motor pump now starts running to fill the tank with water. Freewheeling diode D5 prevents chattering of the relay due to the back emf produced by the relay coil.

When the water level rises to bridge the electrodes, because of the conductivity of water, pin 6 (E1) is pulled down to ground (E2). This does not alter the output state of IC2, which

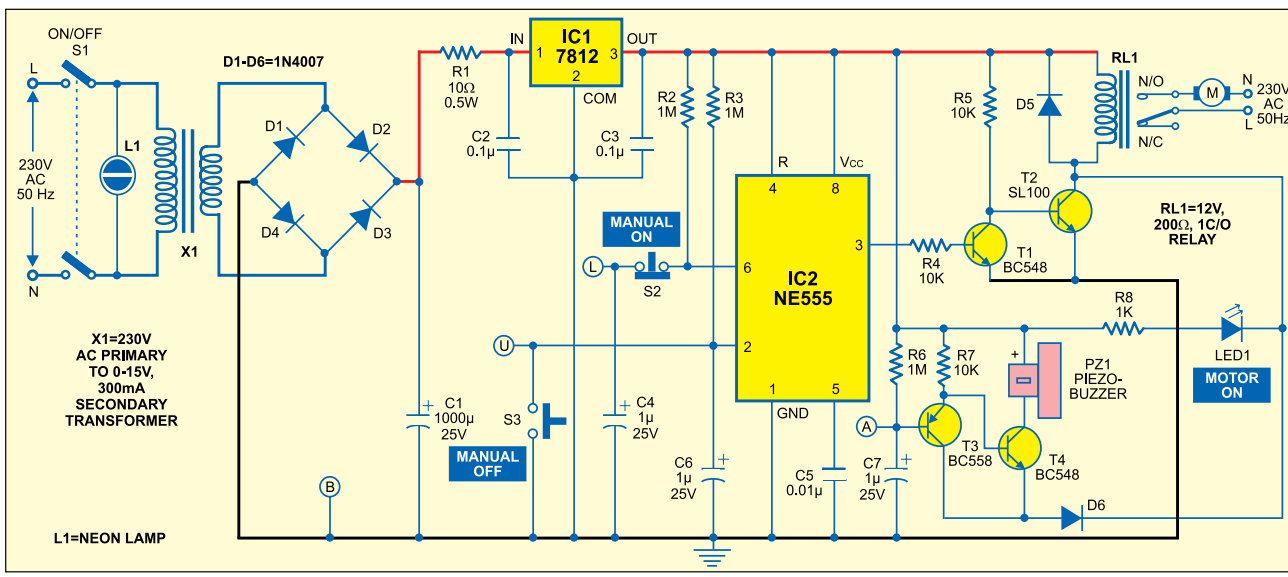


Fig. 1: Circuit of water-level controller

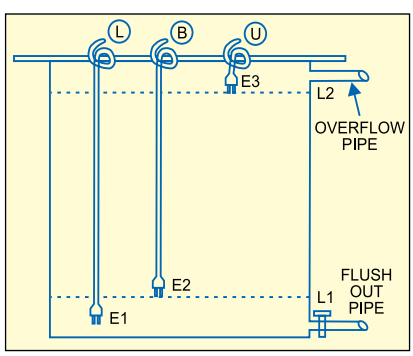


Fig. 2: Water-level electrodes set-up for overhead tank

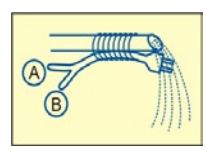


Fig. 3: Dry pump sensor set-up

controller circuit (and hence the motor pump) to avoid coil burn and power wastage.

The water-level controller circuit (see Fig. 1) is built around IC 555 (IC2) to monitor the water level in the overhead tank and 'on'/'off' status of the sensing electrodes is shown in Fig. 2. Electrodes are suspended into the tank such that they don't touch each other. Points B, L and U of the water-level controller circuit are connected to the respective points of the sensor electrodes assembly.

When water in the tank is below the lowest level L1, all the electrodes are electrically separated and hence points L and U (pins 6 and 2 of IC2, respectively) are pulled up to the supply voltage through resistors R2 and R3, respectively. Therefore, to reset IC2 the output of IC2 at pin 3 goes low. As a result, transistor T1 stops conducting to drive transistor T2 and relay RL1 maintains its previous state, and the motor keeps running.

When water rises to the overflow level L2 and touches electrode E3, point U (pin 2 of IC2) is connected to already sunken ground electrode E2, thereby triggering it. IC2 resets to give a high output at pin 3. This is inverted by transistor T1 to cut off transistor T2 and de-energise relay RL1. The motor pump now stops to prevent water overflow.

As water is consumed, the water level comes down leaving electrode E3 isolated from ground electrode E2. Now point U (pin 2 of IC2) is pulled up to the supply voltage. This does not change the output state of IC2 and the motor remains switched off.

When water level again falls below electrode E2, IC2 resets to cut off

transistor T1. Transistor T2 conducts to energise relay RL1 and the motor is powered to run. This is how the process continues. LED1 glows whenever the relay energises, indicating that the motor pump is running.

As the values of resistors R2 and R3 are very high, corrosion of electrodes is very little. Capacitors C2 through C7 filter out unwanted noise. Switches S2 and S3 can be used to manually switch on and off the motor pump, respectively, when water is in between the upper and lower levels. Switch S1 is used to disable the unit during dry pump run or while flushing the tank.

For the sensor electrodes, use a moulded-type AC chord (used for tape recorders) with its pair of wires sleeved at the end and connected together to form the electrode. Other electrodes can be made similarly. These three AC chords are suspended inside the tank from a longitudinally cut PVC pipe (used for electrical wiring).

The arrangement for the dry pump sensor is shown in Fig. 3. A moulded-type AC chord with its pair of wires sleeved at the end can be attached firmly to the delivery pipe such that water falls onto the plug leads. The sleeved ends are connected to points A and B of the water-level controller circuit.

The circuit for dry-run alarm comprises transistors T3 and T4, piezobuzzer PZ1, resistors R6 and R7, and capacitor C7. When points A and B of the dry-running sensor (see Fig. 3) are bridged by water being delivered by the pipe, transistor T3 conducts to drive transistor T4 into cut-off state and therefore the DC buzzer remains silent.

When the pump runs dry, points A and B are electrically apart causing transistor T3 to cut off because of pull-up resistor R6. Transistor T4 conducts due to the emitter drop of transistor T3, which activates the DC buzzer to sound an alarm indicating dry running of the pump.

The alarm circuit is enabled only when transistor T2 conducts, i.e., only when the motor pump runs. Diode D6 isolates the relay driver circuitry to prevent transistor T3 from extending ground to the relay through transistor T3 and water being delivered. As soon as the pump is switched on, the alarm sounds until water reaches the delivery port.

House the controller circuit (including the power supply) in a cabinet. Use a four-core shielded cable for wiring the tank electrodes to the controller unit fixed near the motor switch.

To test the circuit, proceed as follows:

1. Switch on power to the circuit.

2. LED1 glows and relay RL1 energises to produce an alarm from piezobuzzer PZ1, indicating that none of the circuit points A, B, U and L is shorted through water (i.e., water in the tank is below the lowest limit). The energised relay indicates 'on' status of the motor.

3. Immerse points A and B in water. The buzzer stops sounding to indicate that water is flowing out of the pipe to short points A and B. This confirms no dry run.

4. Immerse points B and L in water, as would be the case when the water level rises. Momentarily touch point U to water. LED1 goes off and the relay de-energises to turn the pump 'off.' This would be the case when water touches the overflow limit.

5. Remove points A and B from water assuming that the flowing water that was shorting points A and B has stopped. Now, although water is not flowing, the buzzer does not sound as the relay is already de-energised.

6. Remove points U and B from water, assuming that water has fallen below the lowest limit because of consumption. Two seconds later, LED1 glows and the relay energises.

Precautions. 1. Make sure that water being delivered from the water pipe doesn't touch any of the suspended water-level sensors.

2. Mount the alarm sensor firmly onto the water pipe such that electrodes A and B are shorted by water flowing out of the pipe.

3. Use a properly shielded cable to carry signals from the tank to the water-level controller unit. ●

TOP 20 Projects (Out of 92)

- Standalone Scrolling Display Using AT90S8515 AVR
- Remote Controlled Digital Audio Processor
- Using AVR Microcontroller for Projects
- Simple Digital Security System
- Remote Control for Home Appliances
- Long Range Burglar Alarm Using Laser Torch
- Medium Power FM Transmitter
- Simple Smoke Detector
- Speed Checker for Highways
- Multiple Applications of High-power LEDs
- Digital Dice
- Programmable Timer Based on AT90S4433 AVR
- Auto Turn-off Battery Charger
- Wireless Stepper Motor Controller
- Audio Mixer with Multiple Controls
- Automatic Bathroom Light with Backup Lamp
- Automatic Washbasin Tap Controller
- 16-way Clap-operated Switch
- Inexpensive Car Protection Unit
- Smart Cell Phone Holder

About *Electronics For You* Magazine

Started in 1969, the magazine (print edition) is read by over half-a-million electronics professionals and enthusiasts in India. Another half-a-million professionals, from all across the globe, access its Web portal, www.electronicsforu.com, every month. And now the e-zine version of the magazine, which is available on tablets, mobile devices, smartphones, desktops and laptops, is gaining popularity. The magazine is adored for its focus on technology trends, coupled with a lot of Do-It-Yourself content.

EFYGROUP
Technology Drives Us

EFY Enterprises Pvt Ltd
D-87/1, Okhla Industrial Area, Phase 1, New Delhi 110020
Ph: 011-26810601/2/3; E-mail: info@efyindia.com
Website: www.efyindia.com

ISBN: 978-81-88152-26-1
PUBLISHED BY EFY