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.

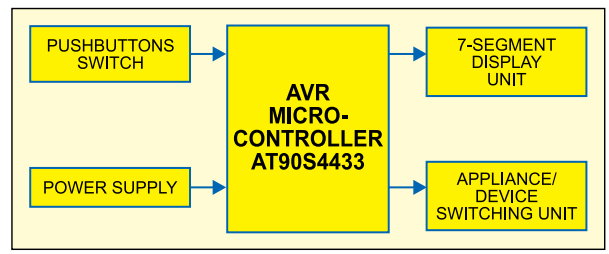

Fig. 1: Block diagram of programmable timer

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

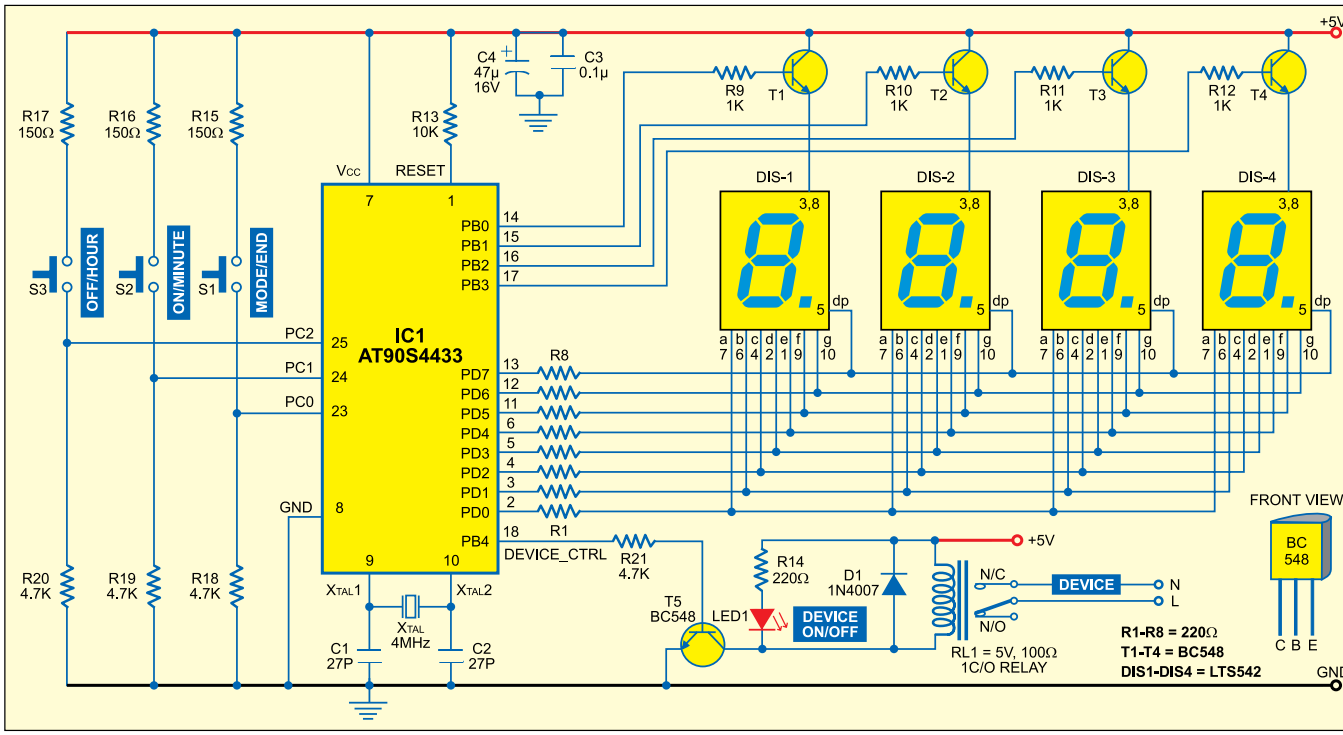

Fig. 2: Circuit of programmable timer using AT90S4433

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.

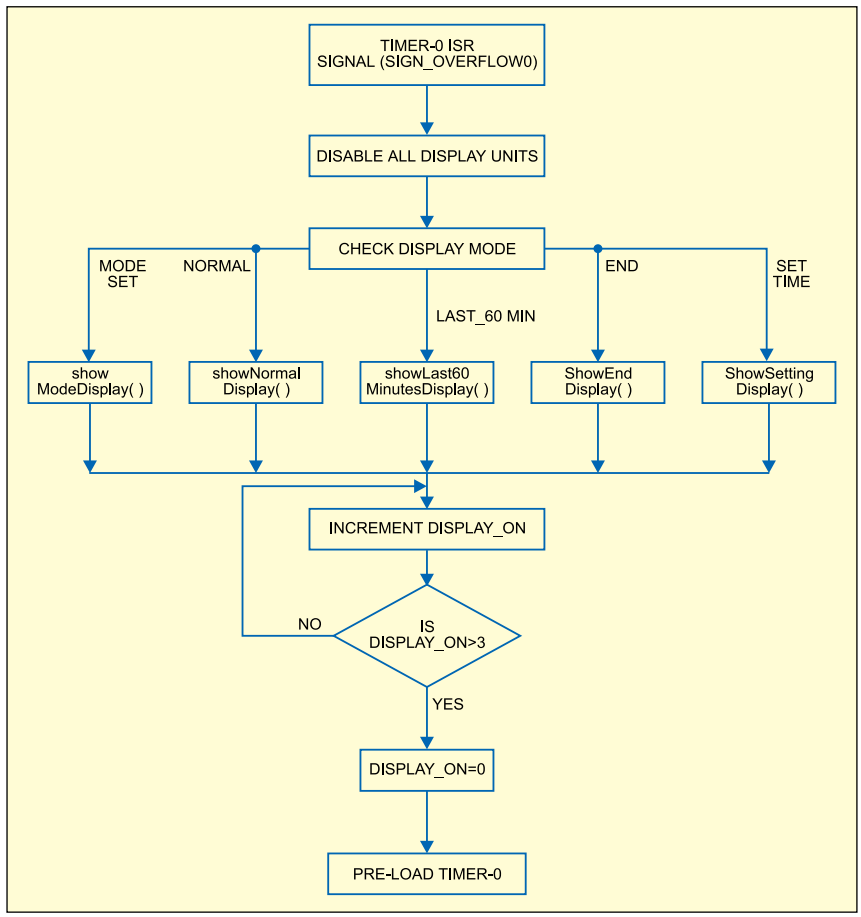

Fig. 3: Flow-chart of timer-0 ISR

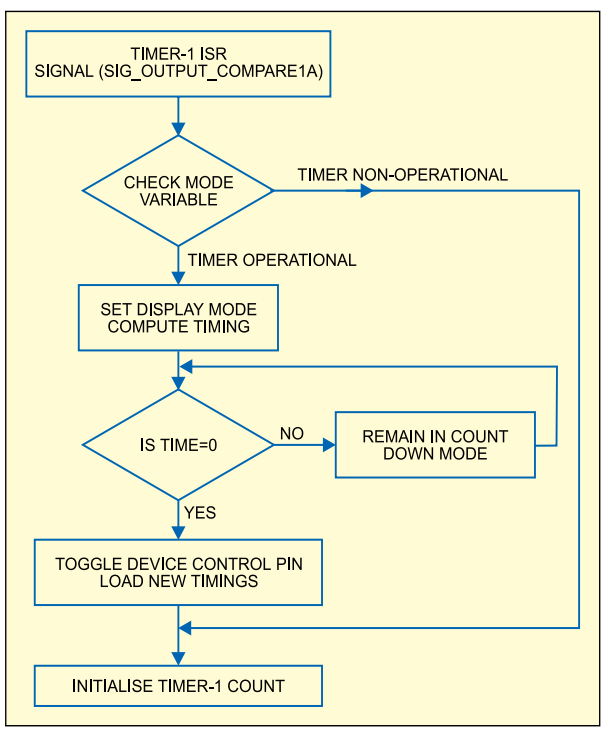

Fig. 4: Flow-chart of timer-1 ISR

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-

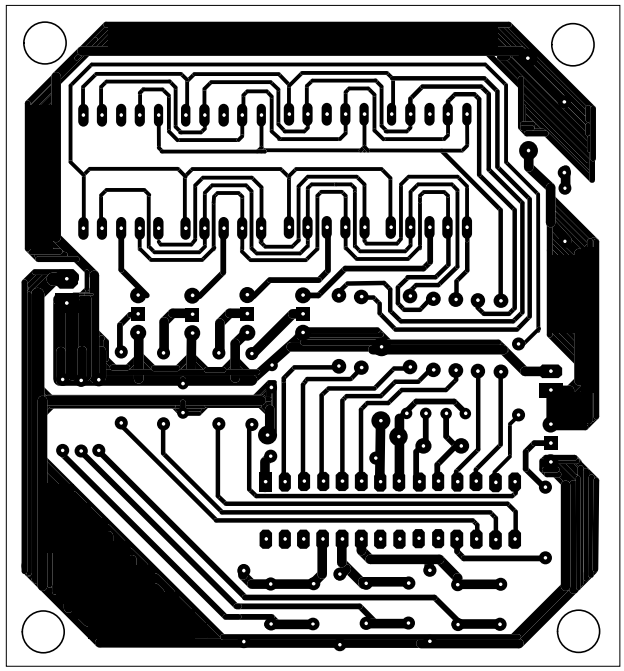

Fig. 5: Actual-size, single-side PCB for programmable timer

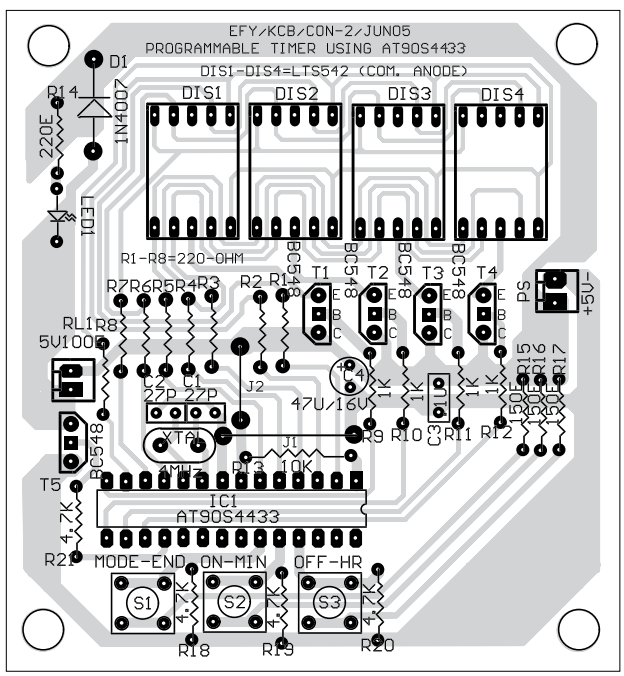

Fig. 6: Component layout for the PCB

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