MICROCONTROLLER-BASED AUTOMATIC FLUSH SYSTEM

■ K.S. SANKAR

In manual flush systems, the user presses a button, which opens a flush valve allowing mains-pressure water to flow into the bowl, or sometimes the user presses directly a flush lever (a handle connected to a flushometer). The valve contains a pneumatic mechanism that closes it after a preset time.

Today, manual flush system has been replaced with a sensor-operated system that automatically flushes the fixture when the user departs.

The microcontroller-based automatic flush system presented here uses an infrared sensor to detect a user approaching the fixture, then it waits until the user departs. A solenoid is used to actuate the flush from a 6V power supply with battery backup inside the unit that also powers the sensor circuit. This flush system is fully controlled by a microcontroller. It also flushes before the person departs if the person is present for more than the preset time (5 minutes).

Fig. 1: Installation of the automatic flush system

Installation of this microcontroller-based automatic flush system is shown in Fig. 1. The IR transmitter LED and the IR receiver modules are mounted side by side such that when the user approaches the mechanism, the IR receiver module receives the IR signal reflected off the person. A solenoid-operated water valve is used in the system.

Circuit description

Fig. 2 shows the circuit of the microcontroller-based flush control system. It is built around Atmel 89C2051 microcontroller that controls the process of automatically flushing the toilet.

The AT89C2051 is an 8-bit microcontroller with 2 kB of flash-based program memory, 128 bytes of RAM, 15 input/output lines, two 16-bit timers/counters, on-chip oscillator and clock circuitry. A 6MHz crystal is used for providing clock. Port pins P1.0 through P1.4 of the microcontroller are connected to buffers N1 through N5 of CD4050 via 10-kilo-ohm pull-up resistors, respectively.

All the input/output (I/O) pins are reset to '1' as soon as RST (pin 9) goes high on pressing switch S3. Holding the RST pin high for two machine cycles while the oscillator is running resets the device. Power-on-reset is achieved by capacitor C2 and resistor R9.

Pin 12 (P1.0) of microcontroller IC2

PARTS LIST

Semiconductor:
IC1 - 7805, 5V regulator
IC2 - AT89C2051 microcontroller
IC3 - CD4050 hex non-inverting buffer
T1 - BC548 npn transistor
IRX1 - TSOP1738 IR receiver module
D1-D5 - 1N4007 rectifier diode
LED1-LED5 – 5mm LED
IR LED1,
IR LED2 – IR LED

Resistors (all ¼-watt, ±5% carbon):
R1, R11-R14 - 330-ohm
R2 - 220-ohm
R3-R8 - 10-kilo-ohm
R9 - 4.7-kilo-ohm
R10 - 150-ohm
R15 - 1.2-kilo-ohm

Capacitors:
C1 - 1000µF, 25V electrolytic
C2, C3 - 10µF, 16V electrolytic
C4, C5 - 22pF ceramic disk

Miscellaneous:
X1 - 230V AC primary to 7.5V, 300mA secondary transformer
RL1 - 6V, 1C/O relay
S1, S2 - On/off switch
S3 - Push-to-on switch
BATT. - 6V battery
- Solenoid (operated with 6V)

provides the 38kHz clock frequency, which is buffered by N1 to drive the two parallel IR-LEDs. These IR-LEDs act as the infrared signal transmitter. Resistor R10 limits the current through the LEDs. Port pins P1.1, P1.2, P1.3 and P1.4 are used for indication of standby, alert, active and flush, respectively. Port pin P1.4 also drives relay RL1 through transistor T1. Diode D5 acts as a free-wheeling diode. The solenoid coil operated off 6V is connected to the contacts of relay RL1.

External interrupt 0 (INT0) is used to receive the reflected

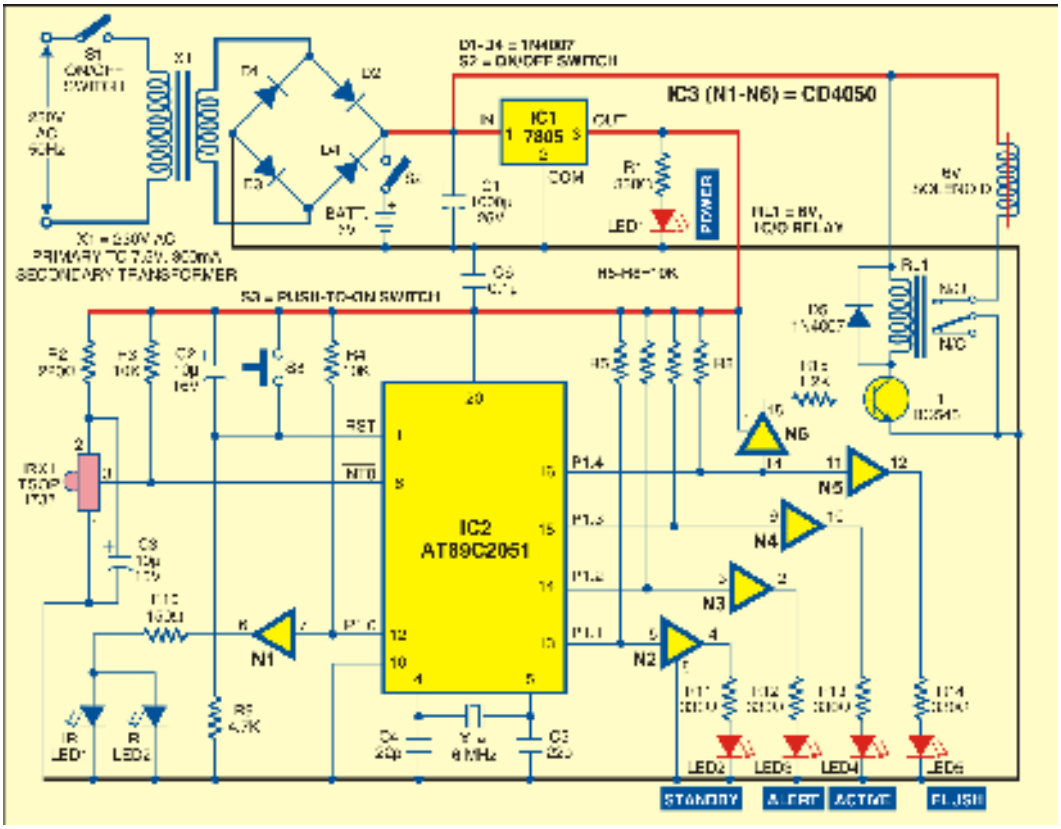

Fig. 2: Circuit of microcontroller-based flush control system

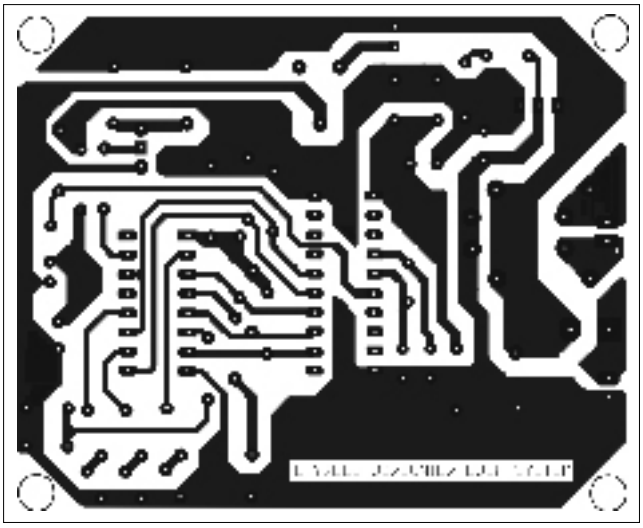

Fig. 3: Actual-size, single-side PCB of microcontroller-based flush control system

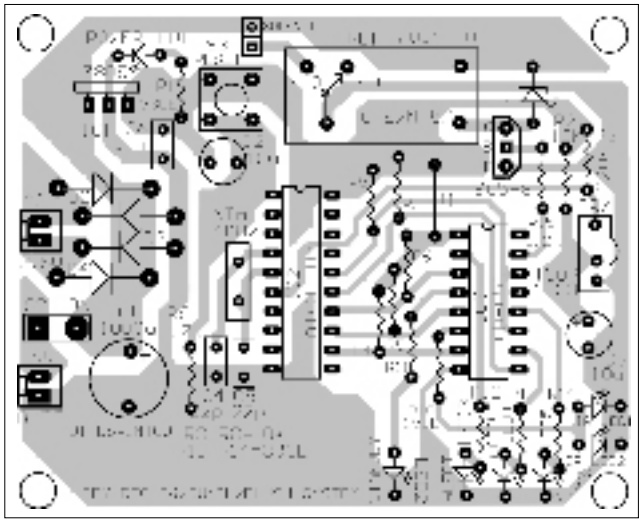

Fig. 4: Component layout for the PCB

IR signal. INT0 (pin 6) of the microcontroller is pulled up with resistor R3 and connected to pin 3 of TSOP1738 IR receiver module.

Pin 2 of TSOP1738 is pulled high with resistor R2, while pin 1 is grounded. In the IR receiver module TSOP1738, the PIN diode and the preamplifier are assembled on the lead frame, and the epoxy package is designed as an IR filter. The demodulated output from the receiver module can be directly decoded by the microcontroller.

The IR-LEDs continuously transmit the IR signal and standby LED2 is always 'on.' When any person comes near the IR-LEDs, the IR receiver module receives the reflected IR signal and alert LED3 lights up. If the alert LED glows for 5 seconds, the active LED (LED4) lights up, indicating that the circuit is now ready to flush. This 5-second time allows for validation of the

pot use by the person. When the person goes away, the flush is activated for 10 seconds, which is indicated by LED5. If the person is there for more than 5 minutes, the system flushes once and the software goes back to waiting for the object to move away.

The 5V regulated power supply for the circuit is provided by a conventional circuit. The AC mains is stepped down by transformer X1 to deliver a secondary output of 7.5V, 300mA, which is rectified by a full-wave rectifier comprising diodes D1 through D4, filtered by capacitor C1 to eliminate ripples and regulated by IC 7805 (IC1) to provide regulated 5V power supply for the circuit. LED1 acts as the power indicator. Relay coil and solenoid coil are powered by 6V unregulated power supply. A 6V rechargeable battery is used for power backup.

An actual-size, single-side PCB for the microcontroller-based automatic flush system (Fig. 2) is shown in Fig. 3 and its component layout in Fig. 4.

The software

The software for flush system is written in 'Basic' language and compiled using Bascom-8051 version. The demo version of Bascom-8051 is available on website 'www.mcselec.com/index.php?option=com_docman&task=doc_download&gid=166&Itemid=54.'

First, instruct the compiler to use 89C2051.dat for microcontroller AT89C2051 by statement '$regfile.' After this, instruct the compiler to override the crystal frequency options setting by statement '$crystal.' Then declare the variables as bits, bytes and words. Initialise port-1 to '0' and port-3 to '1.' (Port-3 acts as the input port.) Enable the interrupt after initialisation. Now write the subroutine 'Fn38K6' to generate 38kHz frequency for transmission of the IR signal.

Standby LED glows when external interrupt 'INT0' is high, i.e., there is no interruption of IR transmission. When 'INT0' goes low, i.e., the transmission is interrupted, alert LED glows. After 5 seconds, active LED lights up. When the person moves away (no interrupt) within 5 minutes, the system flushes for 10 seconds. Otherwise, it flushes every 5 minutes if the person is there. 'Wait' and 'waitms' statements provide the delay in seconds and milliseconds, respectively. Delay time basically depends on the crystal frequency.

EFY note. The source code and other relevant files of this article have been included in this month's EFY-CD.

FLUSH.BAS

```
$regfile = "89c2051.dat"
' the micro controller's include file
$crystal = 6000000
'6 mhz crystal  used
' define variables below
Dim J As Byte
Dim Irrecd As Bit , I As Byte
Dim K As Byte
Dim L As Bit
Declare Sub Fn38k6(period As Byte)
Dim Period As Word
Irrecd = 0
' another name for port p1.0
Irport Alias P1.0
' make all ports 0
P1 = 0
P3 = 255
' make port-3 high for interrupt to work
' on interrupt - call int0_int fuction
On Int0 Int0_int
Set Tcon.0
' int enabled
Enable Interrupts
Beg1:
' show standby mode on port-1 (yellow led on)
P1.1 = 1
' enable the int0 to work now
Enable Int0
Call Fn38k6 100
' call the subroutine to send out a beam
'  of IR at 38khz freq
' int would have taken place if ir recd
Disable Int0
' disable the int now
' check if int occured
If Irrecd = 0 Then
' no int occured
' so go back to standby mode
P1.1 = 0
' flash standby led
Waitms 100
' wait for ( 1/10th of a second)
Goto Beg1
End If
' here int recd
' wait for about 5 secs to get into alert mode
' ir beam should be reflected for this period of
5 secs
Irrecd = 0
P1.2 = 1
' alert led on now
'below for loop will work for approx 5 seconds
For J = 1 To 30
Irrecd = 0
Enable Int0
Call Fn38k6 100
' call the subroutine to send out a beam of IR at
38khz freq
' int would have taken place if ir recd
Disable Int0
' check if int occured
If Irrecd = 0 Then
' no int occured
' so out of loop - reflective object gone
Goto Nothing
End If
Waitms 100
Next J
' here ir has been recd for 5 secs
' so turn on flush for 10 seconds
'After Object Has Moved Away
' wait for obect to move away
'below for loop will work for approx 5 minutes
P1.3 = 1
For I = 1 To 35
' active led on
For J = 1 To 60
Irrecd = 0
Enable Int0
Call Fn38k6 100
' call the subroutine to send out a beam of IR at
38khz freq
' int would have taken place if ir recd
Disable Int0

' check if int occured
If Irrecd = 0 Then
' no int occured
' so out of loop - reflective object gone
Exit For
' get out of the FOR loop
End If
Waitms 100
Next J
Next I
' time period over so flush
' or object has moved away within 5 minutes
P1.4 = 1
' flush led and buzzer on for 10 seconds
Wait 5
Wait 5
P1 = 0
' all leds off
' get back to start
Waitms 100
Goto Beg1
Nothing:
' no ir recd during the 5 min alert period
' so object has moved away
' go back to start
P1 = 0
Waitms 100
Goto Beg1
' =-=-=-=-subroutines below =-=-=-=-=-=-=-=
Sub Fn38k6(period As Word)
' parameter 1000 = 1 second approx
' function to oscillate a port pin at 38,000 times
a sec
Dim Ii As Byte , Jj As Byte , Kk As Byte
Dim Periods As Word
Periods = Period / 100
Ii = 0
While Ii < Periods
Incr Ii
Jj = 0
While Jj < 5
Incr Jj
Kk = 0
While Kk < 255
Incr Kk
Irport = 1
NOP
Irport = 0
NOP
NOP
Wend
Wend
Wend
End Sub
Rem The Interrupt Handler For The Int0 Inter-
rupt
Int0_int:
' program comes here if int0 occurs
Irrecd = 1
' just set a flag and get back
' let the main program handle the flag condition
Return
```