MICROCONTROLLER-BASED BIDIRECTIONAL VISITOR COUNTER

EFY
SUNIL KUMAR
TESTED

■ AKSHAY MATHUR, KULDEEP SINGH NAGLA

Visitor counting is simply a measurement of the visitor traffic entering and exiting offices, malls, sports venues, etc. Counting the visitors helps to maximise the efficiency and effectiveness of employees, floor area and sales potential of an organisation.

Visitor counting is not limited to the entry/exit point of a company but has a wide range of applications that provide information to management on the volume and flow of people throughout a location. A primary method for counting the visitors involves hiring human auditors to stand and manually tally the number of visitors who pass by a certain location. But human-based data collection comes at great expense.

Here is a low-cost microcontroller-based visitor counter that can be used to know the number of persons at a place. All the components required are readily available in the market and the circuit is easy to build.

Two IR transmitter-receiver pairs are used at the passage: one pair comprising IR transmitter IR TX1 and receiver phototransistor T1 is installed at the entry point of the passage, while the other pair comprising IR transmitter IR TX2 and phototransistor T2 is installed at the exit of the passage. The IR signals from the IR LEDs should continuously fall on the respective phototransistors, so proper orientation of the transmitters and phototransistors is necessary.

PARTS LIST

Semiconductors:	
IC1	- LM324 quad op-amp
IC2	- 74LS76 J-K flip-flop
IC3	- AT89C52 microcontroller
IC4	- 7805 5V regulator
T1, T2	- L14F1 npn phototransistor
T3, T4	- 2N3904 npn transistor
IR TX1, IR TX2	- IR transmitting LED
BR1	- 1A bridge rectifier
DIS1-DIS3	- LTS543 CC 7-segment display
Resistors (all ¼-watt, ±5% carbon):	
R1, R2	- 68-ohm
R3, R4	- 6.8-kilo-ohm
R5, R6	- 100-ohm
R7, R8, R10, R11	- 10-kilo-ohm
R9	- 4.7-kilo-ohm
R12-R32	- 220-ohm
VR1, VR2	- 20-kilo-ohm preset
RNW1	- 10-kilo-ohm resistor network
Capacitors:	
C1, C2	- 0.2µF ceramic disk
C3, C4	- 33pF ceramic disk
C5	- 10µF, 16V electrolytic
C6	- 470µF, 25V electrolytic
C7	- 0.1µF ceramic disk
Miscellaneous:	
XTAL	- 12MHz crystal
X1	- 230V primary to 7.5V, 250mA secondary transformer
S1	- Push-to-on switch
S2	- On/off switch

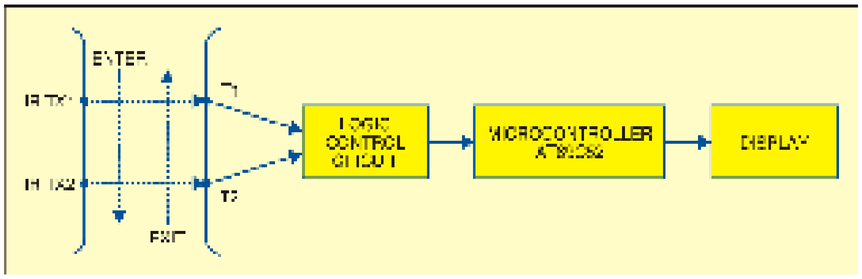

Fig. 1: Transmitter-receiver set-up at the entrance-cum-exit of the passage

Circuit description

Fig. 1 shows the transmitter-receiver set-up at the entrance-cum-exit of the passage along with block diagram. Two similar sections detect interruption of the IR beam and generate clock pulse for the microcontroller. The microcontroller controls counting and displays the number of persons present inside the hall.

Fig. 2 shows the circuit of the microcontroller-based visitor counter, wherein the transmitter and the receiver form the IR detection circuit. Control logic is built around transistors, operational amplifier LM324 (IC1) and flip-flop (IC2).

When nobody is passing through the entry/exit point, the IR beam continuously falls on phototransistor T1. Phototransistor T1 conducts and the high voltage at its emitter drives transistor T3 into saturation, which makes pin 3 of comparator N1 low and finally output pin 1 of comparator N1 is high.

Now if someone enters the place, first the IR beam from IR TX1 is interrupted and then the IR beam from IR TX2. When the beam from IR TX1 is interrupted, phototransistor T1 and transistor T3 cut-off and pin 3 of comparator N1 goes high.

The low output (pin 1) of comparator N1 provides negative trigger pulse to pin 1 of J-K flip-flop IC2(A). At this moment, the high input at 'J' and 'K'

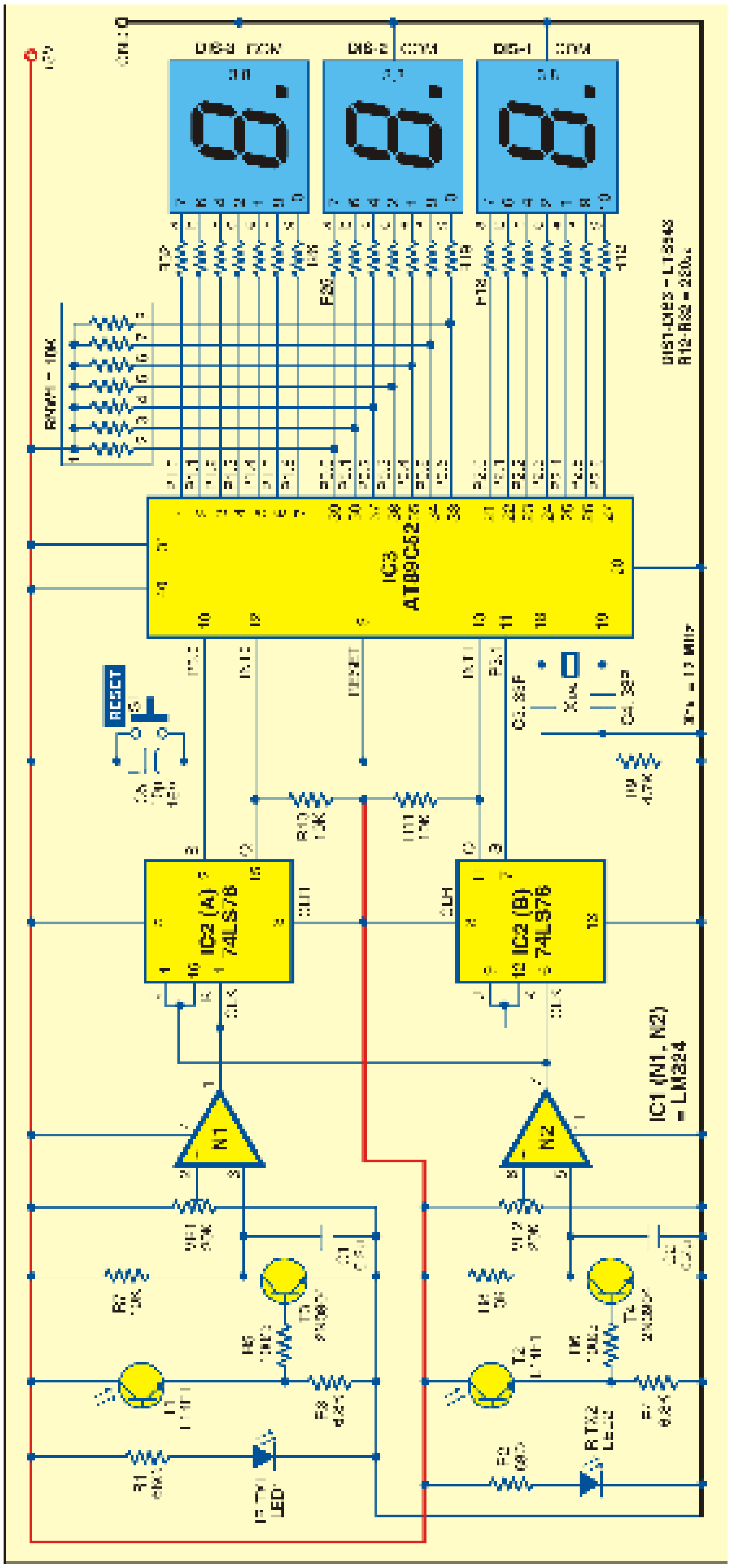

Fig. 2: Circuit of the microcontroller-based visitor counter

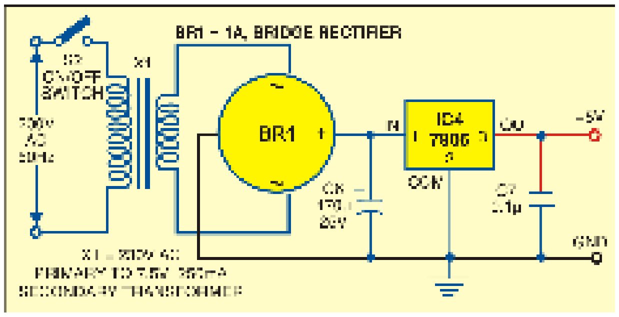

Fig. 3: Power supply circuit

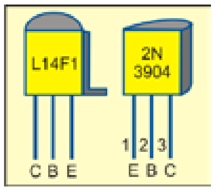

Fig. 4: Pin configuration of L14F1 and transistor 2N3904

pins of flip-flop IC2(A) toggles its output to low. On the other hand, the low input at 'J' and 'K' pins of IC2(B) due to clock pin 1 of IC2(A) and 'J' input (pin 9) and 'K' input (pin 12) of IC2(B) are connected to pin 1 of comparator N1. The negative-going pulse is applied to clock pin 6 of IC2(B) when the person interrupts the IR beam from IR TX2. There is no change in the output of IC2(B) flip-flop. This triggers the external interrupt INT0 (pin 12) of microcontroller AT89C52.

The AT89C52 is an 8-bit microcontroller with 8 kB of flash-based program memory, 256 bytes of RAM, 32 input/output lines, three 16-bit timers/counters, on-chip oscillator and clock circuitry. A 12MHz crystal is used for providing clock.

Ports 0, 1 and 2 are configured for 7-segment displays. Port-0 pin is externally pulled up with 10-kilo-ohm resistor network RNW1 because port-0 is an 8-bit, open-drain, bidirectional, input/output (I/O) port. Port-1 and port-2 are 8-bit bidirectional I/O ports with internal pull-ups (no need of external pull-ups).

Port pins 3.0 and 3.1 are configured to provide the set pulse to J-K flip-flops IC2(A) and IC2(B), respectively. External interrupts INT0 and INT1 receive the interrupt pulse when the person interrupts the IR beams. Resistor R9 and capacitor C5 provide power-on-reset pulse to the microcontroller. Switch S1 is used for manual reset.

When the microcontroller is reset, the flip-flops are brought in 'set' state through the microcontroller at software run time by making their 'set' pin high for a moment.

The value of the counter increments by '1' when the interrupt service routine for INT0 is executed. The output of the corresponding J-K flip-flop is set to 'high' again by making its 'set' input pin low through the microcontroller. The microcontroller is configured as a negative-edge-triggered interrupt sensor.

Similarly, if somebody exits the place, first the IR beam from IR TX2 is interrupted and then the IR beam from IR TX1. When the beam from IR TX2 is interrupted, output pin 7 of comparator N2 goes low. This provides clock pulse to pin 6 of J-K flip-flop IC2(B).

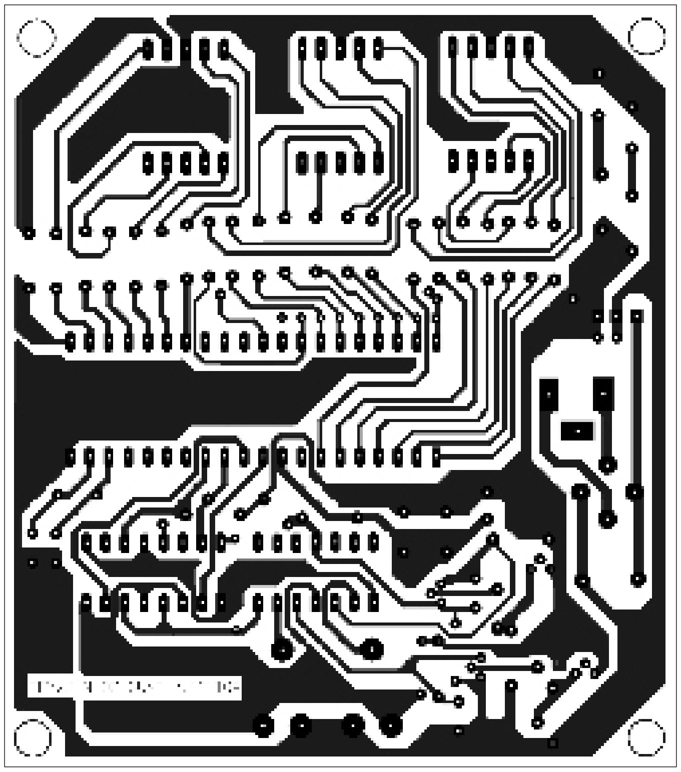

Fig. 5: An actual-size, single-side PCB for the microcontroller-based visitor counter (Fig. 2) including its power supply (Fig. 3)

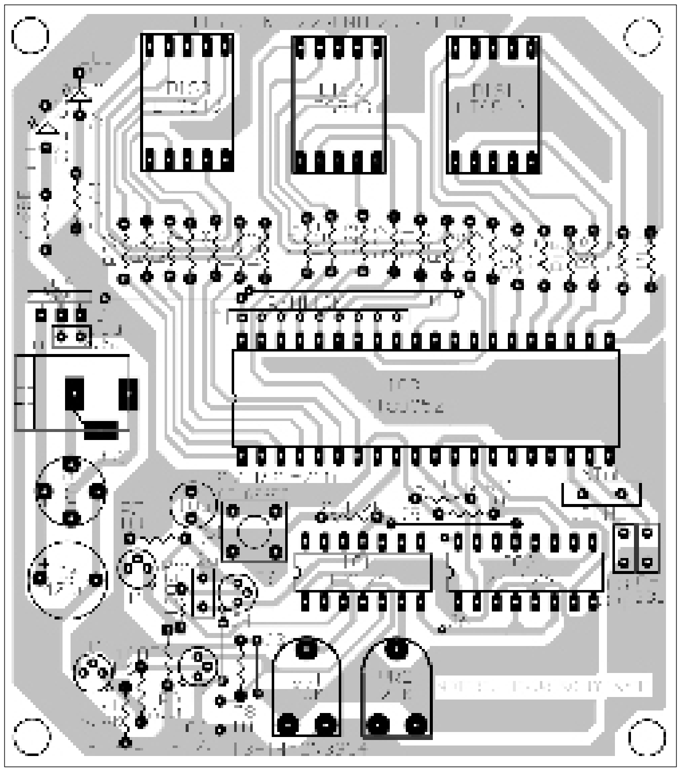

Fig. 6: Component layout for the PCB

At this moment, the high input at 'J' and 'K' pins of flip-flop IC2(B) toggles its output to low. On the other hand, the low input at 'J' and 'K' pins of IC2(A) due to clock pin 6 of IC2(B) and 'J' input (pin 4) and 'K' input (pin 16) of IC2(A) are connected to pin 7 of comparator N2.

The negative-going pulse is applied to clock pin 1 of IC2(A) when the person interrupts the IR beam from IR TX1. There is no change in the output of IC2(A) flip-flop. This triggers the external interrupt INT1 (pin 13) of microcontroller AT89C52. The value of the counter decrements by '1' when interrupt service routine for INT1 is executed. The output of the corresponding J-K flip-flop is set to 'high' again by making its 'set' input pin low through the microcontroller.

The circuit is powered by regulated 5V. Fig. 3 shows the circuit of the power supply. The AC mains is stepped down by transformer X1 to deliver secondary output of 7.5V, 250mA, which is rectified by bridge rectifier BR1, filtered by capacitor C6 and regulated by IC 7805 (IC4). Capacitor C7 bypasses any ripple in the regulated output.

Construction

An actual-size, single-side PCB for the microcontroller-based visitor counter (Fig. 2) including its power supply (Fig. 3) is shown in Fig. 5 and its component layout in Fig. 6.

Software

The software for the visitor counter is written in 'C' language and compiled using C51 Keil compiler. The demo version of this compiler is available for free on the website 'www.keil.com.' It can compile programs up to 2 kB only, which is sufficient for writing most programs.

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

VISITOR.C

```
#include <AT89x52.h>
int i=0,j,k,l,m,a[]={63,6,91,79,102,109,125,7,127,111};
void enter (void) interrupt 0
{
i++;
if(i>999) i=999;
P3_1=0;
for(m=0;m<=1000;m++);
P3_1=1;
}
void exit (void) interrupt 2
{
i—;
if(i<0) i=0;
P3_0=0;
for(m=0;m<=1000;m++);
P3_0=1;
}
void main()
{
IE = 133;
TCON = 5;
P3_0=1;
P3_1=1;
i=0;
while(1)
{
j=i%10;
k=i/10;
l=i/100;
k=k-l*10;
P2=a[j];
P0=a[k];
P1=a[l];
}
}
```