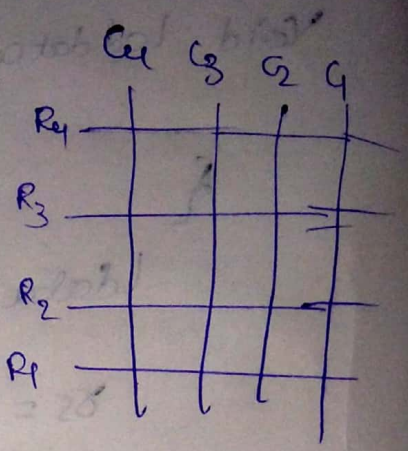
4x4 Keypad




```
#include<reg51.h>

#define display-port P2

sbit rs = P3^2;
sbit rw = P3^3;
sbit en = P3^4;

sbit C4 = P1^0;
 "   C3 = P1^1;
 "   C2 = P1^2;
 "   C1 = P1^3;
 "   R4 = P1^4;
 "   R3 = P1^5;
 "   R2 = P1^6;
 "   R1 = P1^7;

void msdelay(unsigned char time)
{
  unsigned char i, j;
  for(i=0; i<time; i++);
    for(j=0; j<1275; j++)
}
```

```
Void lcdcmd (unsigned char value)
{
    display_port = value;
    rs = 0;
    rw = 0;
    en = 1;
    msdelay (10);
    en = 0;
}

Void lcddata (unsigned char x)
{
    display_port = x;
    rs = 1;
    rw = 0;
    en = 1;
    msdelay (10);
    en = 0;
}
```

```
Void lcd_init()
{
    lcdcmd(0x38);
    msdelay(10);
    lcdcmd(0x0F);
    msdelay(10);
    lcdcmd(0x01);
    msdelay(10);
    lcdcmd(0x81);
    msdelay(10);
}
```

	C_1	C_2	C_3	C_4
R_1	1	2	3	A
R_2	4	5	6	B
R_3	7	8	9	C
R_4	*	0	#	D

```
Void keypad_find()
{
    R1=R2=R3=R4=1;
    C1=C2=C3=C4=0;
    if(R1==0)
        lcddata('1');
    if(R2==0)
        lcddata('4');
```

```
if (R3==0)
  lcddata('7');
  if (R4==0)
    lcddata('*');
  }
```

Similary write for rowfinder2, 3, 4.

```
Void main()
{   lcd_init();
    R1=R2=R3=R4=0;
    C1=C2=C3=C4=1;
    while(1)
    {
      msdelay(30);
      C1=C2=C3=C4=0;
      R1=R2=R3=R4=1;
      if(C1==0)
        rowfinder1();
      if(C2==0)
        row_finder2();
```

```
if (c3==0)
    row_finder3();
    if (c4==0)
        row_finder4();
        }
}
```

Great program

```
#include <reg52.h>

sbit switch=P1^0;

#define Count_port P3

#define switch_not_pressed (bit) 0

#define switch_pressed (bit) 1

void switch_init (void);

bit switch_get_input (const unsigned cha

void Count_init ();

void Count_get_update (const unsigned
    char);
```

```
Void delay_loop_wait (const unsigned int);

Void main (void)
{
    unsigned char Switchpressed = 0;
    Switch_init ();
    Count_init ();
    while (1)
    {
        if (switch_get_input (30)
            == Switchpressed)
        {
        Switchpressed++;
        }
    Count_get_update (Switchpressed);
    }
}
```

```
Void Switch_init ( )
{
    Switch = 1;
}

Void Count_init ( )
{
    Count_post = 0;
}

~~Void~~ bit Switch_get_input (unsigned char Debounce_
                               Period)
{
    bit ReturnValue = Switch_not_pressed;
    if (Switch == 0)
    {
        Delay_loopcount (DebouncePeriod);
        If (Switch == 0)
        {
            while (Switch == 0);
            Return_value = Switch_pressed;
        }
```

```
}
return return_value;
}

void count_get_update (const unsigned char Lcount)
{
    Lcount_port = Lcount;
}

void delay_loop_wait (const unsigned char time)
{
    unsigned char i, j;
    for(i=0; i< time; i++)
        for(j=0; j<120; j++);
}
```

Example 9-20

Write a 8051 C program to toggle all the bits of port P1 continuously with some delay in between. Use Timer 16-bit mode to generate the delay.

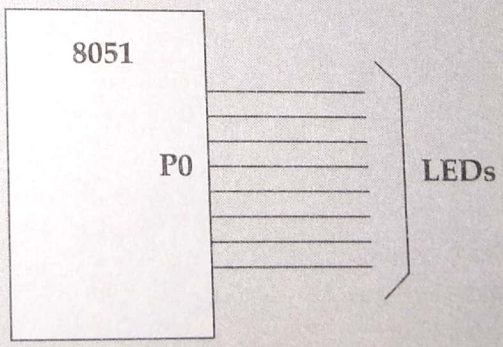
Solution:

```
#include <reg51.h>
void T0Delay(void);
void main(void)
  {
   while(1)                 //repeat forever
   {
      P1=0x55;              //toggle all bits of P1
      T0Delay();            //delay size unknown
      P1=0xAA;              //toggle all bits of P1
      T0Delay();
   }
  }
```

```
void T0Delay()
  {
    TMOD=0x01;              //Timer 0, Mode 1
    TL0=0x00;               //load TL0
    TH0=0x35;               //load TH0
    TR0=1;                  //turn on T0
    while(TF0==0);          //wait for TF0 to roll over
    TR0=0;                  //turn off T0
    TF0=0;                  //clear TF0
  }
```

FFFFH – 3500H = CAFFH = 51967 + 1 = 51968

51968 × 1.085 µs = 56.384 ms is the approximate delay.



Example 9-21

Write an 8051 C program to toggle only bit P1.5 continuously every 50 ms. Use Timer 0, mode 1 (16-bit) to create the delay. Test the program (a) on the AT89C51 and (b) on the DS89C420.

Solution:

```
#include <reg51.h>
void T0M1Delay(void);
sbit mybit=P1^5;
void main(void)
  {
   while(1)
   {
      mybit=~mybit;      //toggle P1.5
      T0M1Delay();       //Timer 0, mode 1(16-bit)
   }
  }
```

(a) Tested for AT89C51, XTAL=11.0592 MHz, using the Proview32 compiler

```
void T0M1Delay(void)
  {
    TMOD=0x01;           //Timer 0, mode 1(16-bit)
    TL0=0xFD;            //load TL0
    TH0=0x4B;            //load TH0
    TR0=1;               //turn on T0
    while(TF0==0);       //wait for TF0 to roll over
    TR0=0;               //turn off T0
    TF0=0;               //clear TF0
  }
```

(b) Tested for DS89C420, XTAL=11.0592 MHz, using the Proview32 compiler

```
void T0M1Delay(void)
  {
    TMOD=0x01;           //Timer 0, mode 1(16-bit)
    TL0=0xFD;            //load TL0
    TH0=0x4B;            //load TH0
    TR0=1;               //turn on T0
    while(TF0==0);       //wait for TF0 to roll over
    TR0=0;               //turn off T0
    TF0=0;               //clear TF0
  }
```

FFFFH – 4BFDH = B402H = 46082 + 1 = 46083

Timer delay = 46083 × 1.085 μs = 50 ms

Example 9-22

Write an 8051 C program to toggle all bits of P2 continuously every 500 ms. Use Timer 1, mode 1 to create the delay.

Solution:

```
//tested for DS89C420, XTAL = 11.0592 MHz, using the Proview32 compiler
#include <reg51.h>
void T1M1Delay(void);
void main(void)
  {
    unsigned char x;
    P2=0x55;
    while(1)
      {
        P2=~P2;                    //toggle all bits of P2
        for(x=0;x<20;x++)
          T1M1Delay();
      }
  }
void T1M1Delay(void)
  {
    TMOD=0x10;                     //Timer 1, mode 1(16-bit)
    TL1=0xFE;                      //load TL1
    TH1=0xA5;                      //load TH1
    TR1=1;                         //turn on T1
    while(TF1==0);                 //wait for TF1 to roll over
    TR1=0;                         //turn off T1
    TF1=0;                         //clear TF1
  }
```

A5FEH = 42494 in decimal

65536 – 42494 = 23042

23042 × 1.085 µs = 25 ms and 20 × 25 ms = 500 ms

NOTE THAT 8051 TIMERS USE 1/12 OF XTAL FREQUENCY, REGARDLESS OF MACHINE CYCLE TIME.

Example 9-23

Write an 8051 C program to toggle only pin P1.5 continuously every 250 ms. Use Timer 0, mode 2 (8-bit auto-reload) to create the delay.

Solution:

```
//tested for DS89C420, XTAL = 11.0592 MHz, using the Proview32 compiler
#include <reg51.h>
void T0M2Delay(void);
sbit mybit=P1^5;
```

```
void main(void)
  {
    unsigned char x, y;
    while(1)
      {
        mybit=~mybit;          //toggle P1.5
        for(x=0;x=250;x++)     //due to for loop overhead
          for(y=0;y=36;y++)    //we put 36 and not 40
            T0M2Delay();
      }
  }

void T0M2Delay(void)
  {
    TMOD=0x02;                 //Timer 0, mode 2(8-bit auto-reload)
    TH0=-23;                   //load TH0(auto-reload value)
    TR0=1;                     //turn on T0
    while(TF0==0);             //wait for TF0 to roll over
    TR0=0;                     //turn off T0
    TF0=0;                     //clear TF0
  }
```

256 – 23 = 233

23 × 1.085 µs = 25 µs

25 µs × 250 × 40 = 250 ms by calculation.

However, the scope output does not give us this result. This is due to overhead of the for loop in C. To correct problem, we put 36 instead of 40.

Example 9-24

Write an 8051 C program to create a frequency of 2500 Hz on pin P2.7. Use Timer 1, mode 2 to create the del

Solution:

//tested for DS89C420, XTAL = 11.0592 MHz, using the Proview32 compiler

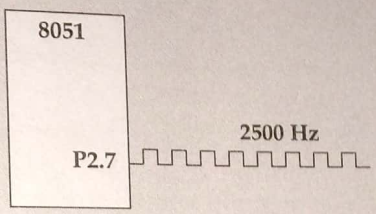
```
#include <reg51.h>
void T1M2Delay(void);
sbit mybit=P2^7;
void main(void)
  {
    unsigned char x;
    while(1)
      {
        mybit=~mybit;          //toggle P2.7
          T1M2Delay();
      }
  }
```

```
void T1M2Delay(void)
  {
    TMOD=0x20;                  //Timer 1, mode 2(8-bit auto-reload)
    TH1=-184;                   //load TH1(auto-reload value)
    TR1=1;                      //turn on T1
    while(TF1==0);              //wait for TF1 to roll over
    TR1=0;                      //turn off T1
    TF1=0;                      //clear TF1
  }
```

1/2500 Hz = 400 µs

400 µs/2 = 200 µs

200 µs/1.085 µs = 184



Example 9-25

A switch is connected to pin P1.2. Write an 8051 C program to monitor SW and create the following frequencies on pin P1.7:

SW=0: 500 Hz

SW=1: 750 Hz

Use Timer 0, mode 1 for both of them.

Solution:

//tested for AT89C51/52, XTAL = 11.0592 MHz, using the Proview32 compiler

```
#include <reg51.h>
sbit mybit=P1^5;
sbit SW=P1^7;
void T0M1Delay(unsiged char);
void main(void)
  {
    SW=1;                       //make P1.7 an input
    while(1)
      {
        mybit=~mybit;           //toggle P1.5
        if(SW==0)               //check switch
          T0M1Delay(0);
        else
          T0M1Delay(1);
      }
  }
```

C Programming of timers 0 and 1 as counters

In Section 9.2 we showed how to use timers 0 and 1 as event counters. A timer can be used as a counter if vide pulses from outside the chip instead of using the frequency of the crystal oscillator as the clock source. By pulses to the T0 (P3.4) and T1 (P3.5) pins, we turn Timer 0 and Timer 1 into counter 0 and counter 1, respectively the next few examples to see how timers 0 and 1 are programmed as counters using the C language.

Example 9-26

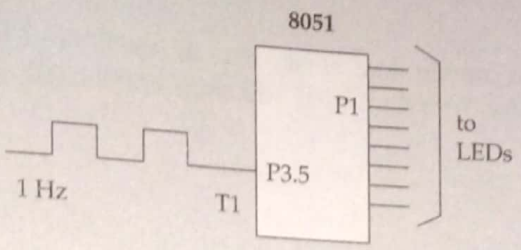
Assume that a 1-Hz external clock is being fed into pin T1 (P3.5). Write a C program for counter 1 in mode 2 (auto reload) to count up and display the state of the TL1 count on P1. Start the count at 0H.

Solution:

```
#include <reg51.h>
sbit T1 = P3^5;
void main(void)
  {
    T1=1;                    //make T1 an input
    TMOD=0x60;               //
    TH1=0;                   //set count to 0
    while(1)                 //repeat forever
      {
        Do
          {
            TR1=1;           //start timer
            P1=TL1;          //place value on pins
          }
```

```
        while(TF1==0);          //wait here
        TR1=0;                  //stop timer
        TF1=0;                  //clear flag
    }
}
```

P1 is connected to 8 LEDs. T1 (P3.5) is connected to a 1-Hz external clock.

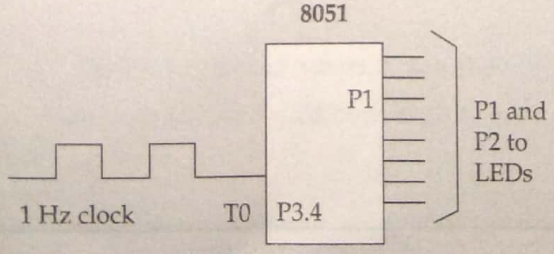


Example 9-27

Assume that a 1-Hz external clock is being fed into pin T0 (P3.4). Write a C program for counter 0 in mode 1 (16-bit) to count the pulses and display the TH0 and TL0 registers on P2 and P1, respectively.

Solution:

```
#include <reg51.h>
void main(void)
  {
    T0=1;                       //make T0 an input
    TMOD=0x05;                  //
    TL0=0;                      //set count to 0
    TH0=0;                      //set count to 0
    while(1)                    //repeat forever
      {
        do
          {
            TR0=1;              //start timer
            P1=TL0;             //place value on pins
            P2=TH0;             //
          }
        while(TF0==0);          //wait here
        TR0=0;                  //stop timer
        TF0=0;
      }
  }
```



Example 9-28

Assume that a 2-Hz external clock is being fed into pin T1 (P3.5). Write a C program for counter 0 in mode (8-bit auto reload) to display the count in ASCII. The 8-bit binary count must be converted to ASCII. Display the ASCII digits (in binary) on P0, P1, and P2 where P0 has the least significant digit. Set the initial value of TL0 to 0.

Solution:

To display the TL1 count we must convert 8-bit binary data to ASCII. See Chapter 7 for data conversion. The ASCII values will be shown in binary. For example, '9' will show as 00111001 on ports.

```
#include <reg51.h>
void BinToASCII(unsigned char);
void main()
  {
    unsigned char value;
    T1=1;
    TMOD=0x06;
    TH0=0;
    while(1)
      {
       do
        {
         TR0=1;
         value=TL0;
         BinToASCII(value);
        }
       while(TF0= =0);
       TR0=0;
       TF0=0;
      }
  }

void BinToASCII(unsigned char value)        //see Chapter 7
  {
    unsigned char x,d1,d2,d3;
    x = value / 10;
    d1 = value % 10
    d2 = x % 10;
    d3 = x / 10
    P0 = 30 | d1;
    P1 = 30 | d2;
    P2 = 30 | d3
  }
```

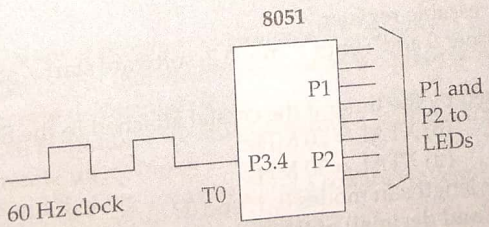
Example 9-29

Assume that a 60-Hz external clock is being fed into pin T0 (P3.4). Write a C program for counter 0 in mode 2 (8-bit auto-reload) to display the seconds and minutes on P1 and P2, respectively.

Solution:

```
#include <reg51.h>
void ToTime(unsigned char);
void main()
  {
   unsigned char val;
   T0=1;
   TMOD=0x06;                    //T0, mode 2, counter
   TH0=-60;                      //sec = 60 pulses
   while(1)
     {
       do
         {
           TR0=1;
           sec=TL0;
           ToTime(val);
         }
       while(TF0==0);
       TR0=0;
       TF0=0;
     }
  }

void ToTime(unsigned char val)
  {
    unsigned char sec, min;
    min = value / 60;
    sec = value % 60;
    P1 = sec;
    P2 = min;
  }
```



By using 60 Hz, we can generate seconds, minutes, hours.

For Examples of Timer 2, see the www.MicroDigitalEd.com Web site.

3. With XTAL = 11.0592 MHz, what value should be loaded into TH1 to have a 28,800 baud rate? Give the both decimal and hex.
4. To transfer a byte of data via the second serial port, it must be placed in register ____.
5. SCON1 refers to ____ and it is a(n) ____-bit register.
6. Which register is used to set the data size and other framing information such as the stop bit for the second serial port?

SECTION 10.5: SERIAL PORT PROGRAMMING IN C

This section shows C programming of the serial ports for the 8051/52 and DS89C4x0 chips.

Transmitting and receiving data in 8051 C

As we stated in the last chapter, the SFR registers of the 8051 are accessible directly in 8051 C compilers by using the reg51.h file. Examples 10-15 through 10-19 show how to program the serial port in 8051 C. Connect your 8051 Trainer to the PC's COM port and use HyperTerminal to test the operation of these examples.

Example 10-15

Write a C program for the 8051 to transfer the letter "A" serially at 4800 baud continuously. Use 8-bit data and 1 stop bit.

Solution:

```
#include <reg51.h>
void main(void)
  {
      TMOD=0x20;                  //use Timer 1,8-BIT auto-reload
      TH1=0xFA;                   //4800 baud rate
      SCON=0x50;
      TR1=1;
      while(1)
       {
         SBUF='A';                //place value in buffer
         while(TI==0);
         TI=0;
       }
  }
```

Example 10-16

Write an 8051 C program to transfer the message "YES" serially at 9600 baud, 8-bit data, 1 stop bit. Do this continuously.

Solution:

```
#include <reg51.h>
void SerTx(unsigned char);
void main(void)
  {
    TMOD=0x20;          //use Timer 1,8-BIT auto-reload
```

```
xample   TH1=0xFD;              //9600 baud rate
         SCON=0x50;
         TR1=1;                 //start timer
         while(1)
           {
              SerTx('Y');
              SerTx('E');
              SerTx('S');
           }
    }
void SerTx(unsigned char x)
    {
       SBUF=x;                  //place value in buffer
       while(TI==0);            //wait until transmitted
       TI=0;
    }
```

Example 10-17

Program the 8051 in C to receive bytes of data serially and put them in P1. Set the baud rate at 4800, 8-bit data, and 1 stop bit.

Solution:

```
#include <reg51.h>
void main (void)
    {
       unsigned char mybyte;
       TMOD=0x20;                 //use Timer 1,8-BIT auto-reload
       TH1=0xFA;                  //4800 baud rate
       SCON=0x50;
       TR1=1;                     //start timer
       while(1)                   //repeat forever
         {
            while(RI==0);         //wait to receive
            mybyte=SBUF;          //save value
            P1=mybyte;            //write value to port
            RI=0;
         }
    }
```

Example 10-18

Write an 8051 C program to send two different strings to the serial port. Assuming that SW is connected to P2.0, monitor its status and make a decision as follows:
SW = 0: send your first name
SW = 1: send your last name
Assume XTAL = 11.0592 MHz, baud rate of 9600, 8-bit data, 1 stop bit.

Solution:

```
#include <reg51.h>
sbit MYSW=P2^0;                          //input switch
void main(void)
{
  unsigned char z;
  unsigned char fname[]="ALI";
  unsigned char lname[]="SMITH";
  TMOD=0x20;                             //use Timer 1,8-BIT auto-reload
  TH1=0xFD;                              //9600 baud rate
  SCON=0x50;
  TR1=1;                                 //start timer
  if(MYSW==0)                            //check switch
    {
      for(z=0;z<3;z++)                   //write name
    {
      SBUF=fname[z];                     //place value in buffer
      while(TI==0);                      //wait for transmit
      TI=0;
    }
  }
else
  {
  for(z=0;z<5;z++)                       //write name
    {
      SBUF=lname[z];                     //place value in buffer
      while(TI==0);                      //wait for transmit
      TI=0;
    }
  }
}
```

Example 10-19

Write an 8051 C program to send the two messages "Normal Speed" and "High Speed" to the serial port. Assuming that SW is connected to pin P2.0, monitor its status and set the baud rate as follows:
SW = 0 28,800 baud rate
SW = 1 56K baud rate
Assume that XTAL = 11.0592 MHz for both cases.

Solution:

```
#include <reg51.h>
sbit MYSW=P2^0;                          //input switch
void main(void)
  {
    unsigned char z;
    unsigned char Mess1[]="Normal Speed";
    unsigned char Mess2[]="High Speed";
```

```
    TMOD=0x20;                    //use Timer 1,8-BIT auto-reload
    TH1=0xFF;                     //28,800 for normal speed
    SCON=0x50;
    TR1=1;                        //start timer
    if(MYSW==0)
      {
       for(z=0;z<12;z++)
        {
          SBUF=Mess1[z];          //place value in buffer
          while(TI==0);           //wait for transmit
          TI=0;
        }
      }
    else
      {
       PCON=PCON|0x80;            //for high speed of 56K
       for(z=0;z<10;z++)
        {
          SBUF=Mess2[z];          //place value in buffer
          while(TI==0);           //wait for transmit
          TI=0;
        }
      }
}
```

8051 C compilers and the second serial port

Since many C compilers do not support the second serial port of the DS89C4x0 chip, we have to declare byte addresses of the new SFR registers using the sfr keyword. Table 10-6 and Figure 10-12 provide the SFR and bit addresses for the DS89C4x0 chip. Examples 10-20 and 10-21 show C versions of Examples 10-11 and in Section 10.4.

Notice in both Examples 10-20 and 10-21 that we are using Timer 1 to set the baud rate for the second serial Upon reset, Timer 1 is the default for the second serial port of the DS89C4x0 chip.

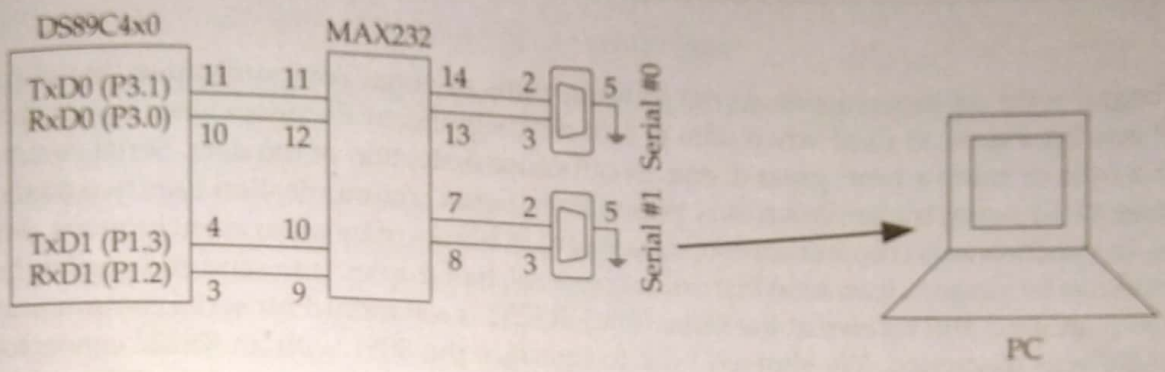
Example 10-20

Write a C program for the DS89C4x0 to transfer letter "A" serially at 4800 baud continuously. Use the second serial port with 8-bit data and 1 stop bit. We can only use Timer 1 to set the baud rate.

Solution:

```
#include <reg51.h>
sfr SBUF1=0xC1;
sfr SCON1=0xC0;
sbit TI1=0xC1;
void main(void)
{
    TMOD=0x20;                    //use Timer 1 for 2nd serial port
    TH1=0xFA;                     //4800 baud rate
    SCON1=0x50;                   //use 2nd serial port SCON1 register
    TR1=1;                        //start timer
```

```
while(1)
  {
   SBUF1='A';          //use 2nd serial port SBUF1 register
   while(TI1==0);      //wait for transmit
   TI1=0;
  }
}
```



Example 10-21

Program the DS89C4x0 in C to receive bytes of data serially via the second serial port and put them in P1. Set the baud rate at 9600, 8-bit data, and 1 stop bit. Use Timer 1 for baud rate generation.

Solution:

```
#include <reg51.h>
sfr SBUF1=0xC1;
sfr SCON1=0xC0;
sbit RI1=0xC0;
void main(void)
 {
  unsigned char mybyte;
  TMOD=0x20;              //use Timer 1, 8-BIT auto-reload
  TH1=0xFD;               //9600
  SCON1=0x50;             //use SCON1 of 2nd serial port
  TR1=1;
  while(1)
    {
     while(RI1==0);       //monitor RI1 of 2nd serial port
     mybyte=SBUF1;        //use SBUF1 of 2nd serial port
     P2=mybyte;           //place value on port
     RI1=0;
    }
}
```

Triggering the interrupt by software

There are times when we need to test an ISR by way of simulation. This can be done with simple [illegible] the interrupts high and thereby cause the 8051 to jump to the interrupt vector table. For example, if the [illegible] 1 is set, an instruction such as "SETB TF1" will interrupt the 8051 in whatever it is doing and force it [illegible] interrupt vector table. In other words, we do not need to wait for Timer 1 to roll over to have an interrupt. We an interrupt with an instruction that raises the interrupt flag.

Review Questions

1. True or false. Upon reset, all interrupts have the same priority.
2. What register keeps track of interrupt priority in the 8051? Is it a bit-addressable register?
3. Which bit of IP belongs to the serial interrupt priority? Show how to assign it the highest priority.
4. Assume that the IP register contains all 0s. Explain what happens if both INT0 and INT1 are activated at [illegible] time.
5. Explain what happens if a higher-priority interrupt is activated while the 8051 is serving a lower-priority [illegible] (that is, executing a lower-priority ISR).

SECTION 11.6: INTERRUPT PROGRAMMING IN C

So far all the programs in this chapter have been written in Assembly. In this section we show how to progr 8051/52's interrupts in 8051 C language. In reading this section, it is assumed that you already know the mate the first two sections of this chapter.

8051 C interrupt numbers

The 8051 C compilers have extensive support for the 8051 interrupts with two major features as follows:

1. They assign a unique number to each of the 8051 interrupts, as shown in Table 11-4.
2. It can also assign a register bank to an ISR. This avoids code overhead due to the pushes and pops of the R registers.

Example 11-14 shows how a simple interrupt is written in 8051 C.

Table 11-4: 8051/52 Interrupt Numbers in C

Interrupt	Name	Numbers used by 8051 C
External Interrupt 0	(INT0)	0
Timer Interrupt 0	(TF0)	1
External Interrupt 1	(INT1)	2
Timer Interrupt 1	(TF1)	3
Serial Communication	(RI + TI)	4
Timer 2 (8052 only)	(TF2)	5

Example 11-14

Write a C program that continuously gets a single bit of data from P1.7 and sends it to P1.0, while simultaneously creating a square wave of 200 µs period on pin P2.5. Use timer 0 to create the square wave. Assume that XTAL = 11.0592 MHz.

Solution:

We will use timer 0 in mode 2 (auto-reload). One half of the period is 100 µs.
100/1. 085 µs = 92, and TH0 = 256 – 92 = 164 or A4H

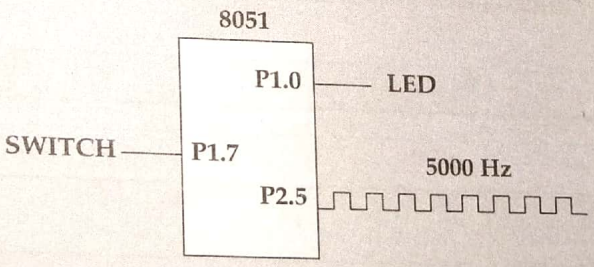
```
#include <reg51.h>

sbit SW   = P1^7;
sbit IND  = P1^0;
sbit WAVE = P2^5;

void timer0(void) interrupt 1
  {
  WAVE = ~WAVE;          //toggle pin
  }

void main()
  {
    SW = 1;              //make switch input
    TMOD = 0x02;
    TH0 = 0xA4;          //TH0 = -92
    IE = 0x82;           //enable interrupts for timer 0
    while(1)
      {
        IND = SW;        //send switch to LED
      }
  }
```

200 µs / 2 = 100 µs
100 µs / 1.085 µs = 92



Example 11-15

Write a C program that continuously gets a single bit of data from P1.7 and sends it to P1.0 in the main, while simultaneously (a) creating a square wave of 200 µs period on pin P2.5, and (b) sending letter 'A' to the serial port. Use Timer 0 to create the square wave. Assume that XTAL = 11.0592 MHz. Use the 9600 baud rate.

Solution:

We will use Timer 0 in mode 2 (auto-reload). TH0 = 100/1.085 µs = –92, which is A4H

```
#include <reg51.h>
```

```
sbit SW = P1^7;
sbit IND = P1^0;
sbit WAVE = P2^5;

void timer0(void) interrupt 1
  {
  WAVE = ~WAVE;          //toggle pin
  }

void serial0() interrupt 4
  {
    if(TI == 1)
      {
        SBUF = 'A';      //send A to serial port
        TI = 0;          //clear interrupt
      }
    else
      {
        RI = 0;          //clear interrupt
      }
  }
void main()
  {
    SW = 1;              //make switch input
    TH1 = -3;            //9600 baud
    TMOD = 0x22;         //mode 2 for both timers
    TH0 = 0xA4;          //-92=A4H for timer 0
    SCON = 0x50;
    TR0 = 1;
    TR1 = 1;             //start timer
    IE = 0x92;           //enable interrupt for T0
    while(1)             //stay here
      {
        IND = SW;        //send switch to LED
      }
  }
```

Example 11-16

Write a C program using interrupts to do the following:
(a) Receive data serially and send it to P0,
(b) Read port P1, transmit data serially, and give a copy to P2,
(c) Make timer 0 generate a square wave of 5 kHz frequency on P0.1.
Assume that XTAL = 11.0592 MHz. Set the baud rate at 4800.

Solution:

```
#include <reg51.h>]
sbit WAVE = P0^1;

void timer0() interrupt 1
  {
    WAVE = ~WAVE;          //toggle pin
  }
```

```
void serial0() interrupt 4
 {
  if(TI == 1)
    {
      TI = 0;                  //clear interrupt
    }
  else
    {
      P0 = SBUF;               //put value on pins
      RI = 0;                  //clear interrupt
    }
 }
void main()
 {
  unsigned char x;
  P1 = 0xFF;                   //make P1 an input
  TMOD = 0x22;
  TH1 = 0xF6;                  //4800 baud rate
  SCON = 0x50;
  TH0 = 0xA4;                  //5 kHz has T = 200 µs
  IE = 0x92;                   //enable interrupts
  TR1 = 1;                     //start timer 1
  TR0 = 1;                     //start timer 0
  while(1)
    {
      x = P1;                  //read value from pins
      SBUF = x;                //put value in buffer
      P2 = x;                  //write value to pins
    }
 }
```

```
    P0 = cnt;                    //display value on pins
  }
void main()
  {
    cnt = 0;                     //set counter to zero
    TMOD = 0x42;
    TH0 = 0x-46;                 //10000 Hz
    IE = 0x86;                   //enable interrupts
    TR0 = 1;                     //start timer 0
    TR1 = 1;                     //start timer 1
    while(1);                    //wait until interrupted
  }
```

1 / 10000 Hz = 100 µs

100 µs / 2 = 50 µs

50 µs / 1.085 µs = 46