Example 4-6

A switch is connected to port pin P0.1. Write a program to check the status of the switch and perform the fo
(a) If switch = 1, send a high-to-low pulse to activate a siren connected to pin P1.7.
(b) Continue monitoring the pin status

Use the carry flag to check the switch status.

Solution:

```
        SETB P0.1          ;make P0.1 an input
AGAIN:  MOV  C,P0.1        ;read the contents of P0.1 into carry flag
        JNC  AGAIN         ;if P0.1 is not high, continue to monitor it
        SETB P1.7          ;if P0.1 is high, send a high to P1.7
        CLR  P1.7          ;send low now, i.e., a H-to-L pulse on P1.7
        SJMP AGAIN         ;continue monitoring the pin status
```

Example 4-7

A switch is connected to pin P1.0 and an LED to pin P2.7. Write a program to get the status of the switch a
it to the LED.

Solution:

```
        SETB P1.7          ;make P1.7 an input
AGAIN:  MOV  C,P1.0        ;read the SW status into CF
        MOV  P2.7,C        ;send the SW status to LED
        SJMP AGAIN         ;keep repeating
```

Note: The instruction "`MOV P2.7, P1.0`" is wrong since such an instruction does not exist. However, "`M`
`P1`" is a valid instruction.

Reading a single bit into the carry flag

We can also use the carry flag to save or examine the status of a single bit of the port. To do that, we use t
tion "`MOV C, Px.y`" as shown in Examples 4-6 and 4-7.

Notice in Examples 4-6 and 4-7 how the carry flag is used to get a bit of data from the port.

Reading input pins vs. port latch

In reading a port, some instructions read the status of port pins while others read
Therefore, when reading ports there are two possibilities:

1. Read the status of the

Example 9-11

With a frequency of 22 MHz, generate a frequency of 100 KHz on pin P2.3. Use Timer 1 in mode 1.

Solution:

```
;Tested for an AT89C51 with a crystal frequency of 22MHz.
```

For a 100-KHz square wave,
(a) T = 1/f = 0.01 ms = 10 µs
(b) 1/2 of it for high and low portions each = 5 µs
(c) 5 µs/0.546 µs = 9 cycles
(d) 65,536 – 9 = 65,527 = FFF7H

The program is as follows.

```
        MOV   TMOD,#10H          ;Timer 1, Mode 1
BACK:   MOV   TL1,#0F7H          ;TL1=F7H
        MOV   TH1,#0FFH          ;TH1=FFH
        SETB  TR1                ;start Timer 1
AGAIN:  JNB   TF1,AGAIN          ;wait for timer rollover
        CLR   TR1                ;stop Timer 1
        CPL   P2.3               ;complement P2.3
        CLR   TF1                ;clear timer flag
        SJMP  BACK               ;reload timer
```

8051. This section is [illegible] choice. The timer's use as an event counter is [illegible]

Example 9-1

Find the values of TMOD to operate as timers in the following modes.
(a) Mode 1 Timer 1
(b) Mode 2 Timer 0, Mode 2 Timer 1
(c) Mode 0 Timer 1

Solution:

From Figure 9-3

(a) TMOD is 00010000 = 10H
The gate control bit and C/T bit are made 0, and the unused timer (Timer 0 bit is also 0)
(b) TMOD is 01010010 = 52H
(c) TMOD is 00000000H = 00H

Example 9-4

In the following program, we are creating a square wave of 50% duty cycle (with equal portions high a the P1.5 bit. Timer 0 is used to generate the time delay. Analyze the program.

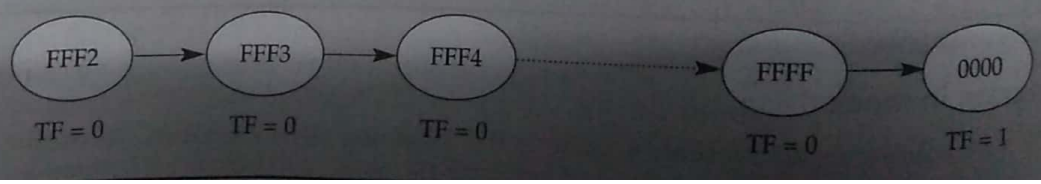
```
        MOV   TMOD,#01          ;Timer 0, mode 1(16-bit mode)
HERE:   MOV   TL0,#0F2H         ;TL0 = F2H, the Low byte
        MOV   TH0,#0FFH         ;TH0 = FFH, the High byte
        CPL   P1.5              ;toggle P1.5
        ACALL DELAY
        SJMP  HERE              ;load TH, TL again
;-------------delay using Timer 0
DELAY:
        SETB TR0                ;start Timer 0
AGAIN:  JNB TF0,AGAIN           ;monitor Timer 0 flag until
                                ;it rolls over
        CLR   TR0               ;stop Timer 0
        CLR   TF0               ;clear Timer 0 flag
        RET
```

Solution:

In the above program notice the following steps.

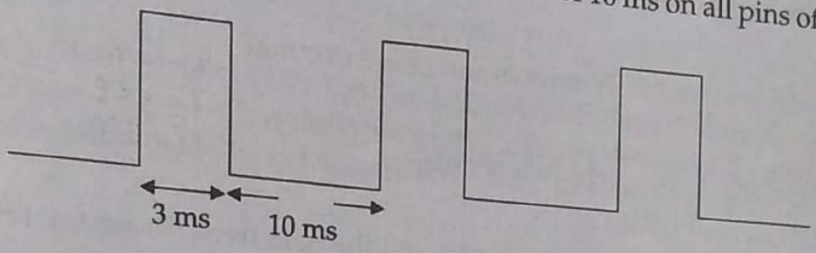
1. TMOD is loaded.
2. FFF2H is loaded into TH0 - TL0.
3. P1.5 is toggled for the high and low portions of the pulse.
4. The DELAY subroutine using the timer is called.
5. In the DELAY subroutine, Timer 0 is started by the "`SETB TR0`" instruction.
6. Timer 0 counts up with the passing of each clock, which is provided by the crystal oscillator. As t counts up, it goes through the states of FFF3, FFF4, FFF5, FFF6, FFF7, FFF8, FFF9, FFFA, FFFB, and so it reaches FFFFH. One more clock rolls it to 0, raising the timer flag (TF0 = 1). At that point, the JNB tion falls through.
7. Timer 0 is stopped by the instruction "`CLR TR0`". The DELAY subroutine ends, and the process is rep

Notice that to repeat the process, we must reload the TL and TH registers and start the timer again.



Example 9-12

Generate a square wave with an ON time of 3 ms and an OFF time of 10 ms on all pins of port 0. Assume an XTAL of 22 MHz.



Solution:

;Tested for an AT89C51 with a crystal frequency of 22MHz.

Let us use Timer 0 in Mode 1.

```
        MOV   TMOD,#01H      ;Timer 0 in mode 1
BACK:   MOV   TL0,#075H      ;to generate the OFF time, load TL0
        MOV   TH0,#0B8H      ;load OFF time value in TH0
        MOV   P0,#00H        ;make port bits low
        ACALL DELAY          ;call delay routine
        MOV   TL0,#8AH       ;to generate the ON time, load TL0
        MOV   TH0,#0EAH      ;load ON time value in TH0
        MOV   P0,#0FFH       ;make port bits high
        ACALL DELAY          ;call delay
        SJMP  BACK           ;repeat for reloading counters to get a
                             ;continuous square wave
        ORG   300H
DELAY:  SETB  TR0            ;start the counter
AGAIN:  JNB   TF0,AGAIN      ;check timer overflow
        CLR   TR0            ;when TF0 is set, stop the timer
        CLR   TF0            ;clear timer flag
        RET
        END                  ;end of file
```

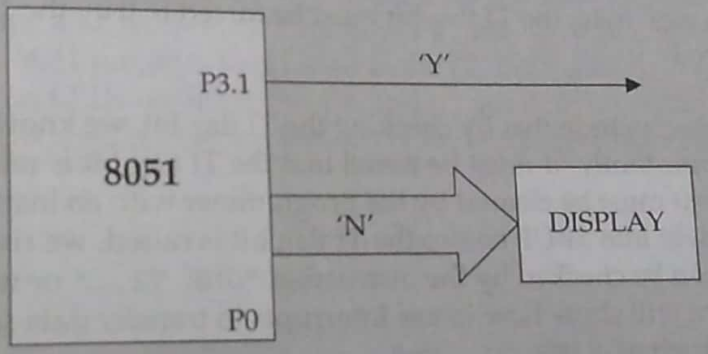
For OFF time calculation:
10 ms/0.546 µs = 18,315 cycle
65,536 – 18,315 = 47,221 = B875H

Example 10-2

Write a program to transfer a letter 'Y' serially at 9600 baud continuously, and also to send a letter 'N' through port 0, which is connected to a display device.

Solution:

Here one byte of data is transmitted serially through pin 11 (P3.1), and another is sent in parallel form through port 0.




```
        MOV   TMOD,#20H    ;Timer 1, mode 2 (auto-reload)
        MOV   TH1,#-3      ;9600 baud rate
        MOV   SCON,#50H    ;8 bit, 1 stop, REN enabled
        SETB  TR1          ;start Timer 1
AGAIN:  MOV   SBUF,#'Y'    ;transfer 'Y' serially
HERE:   JNB   TI,HERE      ;wait for transmission to be over
        CLR   TI           ;clear TI for next transmission
        MOV   P0,#'N'      ;move 'N' to P0 for parallel transfer
        SJMP  AGAIN        ;repeat
```

Example 10-4

Write a program to receive the data which has been sent in serial form and send it out to port 0 in parallel form. Also save the data at RAM location 60H.

Solution:

```
        MOV     TMOD,#20H       ;Timer 1,mode 2,auto-reload
        MOV     TH1,#-3         ;9600 baud
        MOV     SCON,#50H       ;8 bit, 1 stop,REN enabled
        SETB    TR1             ;start Timer 1
        CLR     RI              ;RI is cleared for reception
RPT:    JNB     RI, RPT         ;wait for character to come in
        MOV     A,SBUF          ;move received data into A
        MOV     P0,A            ;move it to P0
        MOV     60H,A           ;move it to RAM location 60H
        END
```

Example 11-2

Write a program that displays a value of 'Y' at port 0 and 'N' at port 2 and also generates a square wave of 10 kHz with Timer 0 in mode 2 at port pin P1.2 .XTAL = 22 MHz.

Solution:

```
;Tested for an AT89C51 with a crystal frequency of 22MHz.
;--- upon wake up, go to main,avoid using memory space allocated to interrupt vector
table
            ORG   0000H
            LJMP  MAIN          ;bypass interrupt vector table
;---ISR for Timer 0 to generate square wave
            ORG   000BH         ;Timer 0 interrupt vector
            CPL   P1.2
            RETI
;---the main program for initialization
            ORG   0030H         ;a location after the interrupt vectors
MAIN:       MOV   TMOD,#02H     ;Timer 0, mode 2 (auto-reload)
            MOV   TH0,#0B6H     ;move count value into TH0
            MOV   IE,#82H       ;enable interrupt timer 0
            SETB  TR0           ;start Timer 0
BACK:       MOV   P0,#'Y'       ;display'Y'at port P0
            MOV   P2,#'N'       ;display'N'at port P2
            SJMP  BACK          ;keep doing this until interrupted
            END
```