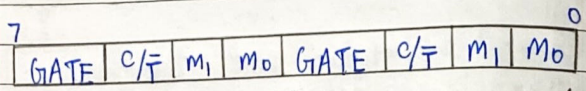
* TMOD :

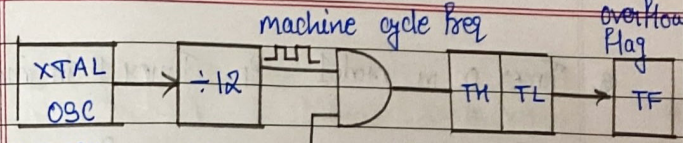


if gate = 1; h/w, INTx → INT0 → Timer 0, INT1 → Timer 1
if gate = 0; s/w, TRx → TR0 → Timer 0, TR1 → Timer 1.

For M1 & M0 :

M1	M0	Timer
0	0	mode 0 (13-bit)
0	1	mode 1 (16-bit)
1	0	mode 2 (8-bit)
1	1	mode 3 (split timer mode).

* Timer MODE1 (16 bit) :



Machine cycle = 12 clock pulses.

TF goes high when FFFF = 0

* Configure

1. Timer 0 in mode 1.

Ans.

Timer1				Timer0			
G	C/T	M1	M0	G	C/T	M1	M0
0	0	0	0	0	0	0	1

TMOD = 0X01

2. Timer 1 in mode 2

Ans

G	C/T	M1	M0	G	C/T	M1	M0
0	0	1	0	0	0	0	0

TMOD = 0X20

3. Timer 0 in mode1 & Timer 1 in mode 2.

Ans.

7							0
G	C/T	M_1	M_0	G	C/T	M_1	M_0
0	0	1	0	0	0	0	1

TMOD = 0X21.

* Note :
Whenever a single timer is configured in a particular mode by default the other unused timer will be configured in mode0.

* Find the timers clock frequency and its time period for various 8051 based systems with the following crystal frequencies:

1. 11.0592 MHz = XTAL freq.
Calculate m/c freq & time period.

Ans. m/c cycle frequency = $\frac{\text{crystal freq}}{12}$

$= \frac{11.0592}{12}$

$= 921.6$ KHz.

time period = $T = \frac{1}{\text{m/c cycle freq}}$

$= \frac{1}{921.6 \text{ KHz}}$

$= 1.085$ μsec

2. XTAL freq = 22 MHz.
Calculate m/c frequency & Time period.

Ans. m/c cycle frequency = $\frac{\text{crystal freq}}{12}$

$= \frac{22}{12}$

$= 1.833$ KHz

Time Period = $\frac{1}{\text{m/c cycle frequency}}$

$= \frac{1}{1.833}$

$= 0.5455$ μsec.

* Questions :

* Generate 25 ms time delay using timer 1 in mode 1. Assume XTAL = 11.0592 MHz.

Ans. Required delay = [(FFFF − YYXX)+1] × time period

(YYXX: count to be loaded; YY → TH, XX → TL; +1 to roll back)

Time period = 1 / (m/c cycle freq.) = 1 / (11.0592 / 12) = 1.085 μs

∴ 25 = [(FFFF − YYXX)+1] × 1.085

$\frac{25}{1.085}$ = [(FFFF − YYXX)+1]

23041 − 1 = FFFF − YYXX

23040 = FFFF − YYXX

∴ YYXX = FFFF − 5A00

= A5FF.

* Obtain the maximum possible delay using timer 0 in mode 1. Assume XTAL = 11.0592 MHz.

Ans. Required delay = [(FFFF − YYXX)+1] × 1.085

∴ To get max delay put YYXX as 0000

∴ Required delay = [(FFFF − 0000)+1] × 1.085

= (65535 + 1) × 1.085 μsec.

= 71106.56 sec

= 71.1 msec

* Obtain the values to be loaded into TH & TL register to generate a delay of 10 ms.

Ans. Required delay = [(FFFF − YYXX)+1] × time period

$\frac{10 \times 10^{-3}}{1.085 \times 10^{-6}}$ 10 = [(FFFF − YYXX)+1] × 1.085 μsec.

(9217 − 1) = FFFF − YYXX (decimal to hexa.)

YYXX = FFFF − 2400

= DBFF (0BFF).

TH TL.

mode1 - 16bit mode.
mode2 - 8bit mode.

* Calculate the delay requised for the given count 5C00 using timers in mode 1.

Ans. Requised delay = [(FFFF - YYXX) + 1] × time period

= [(FFFF - 5C00) + 1] × 1.085 μsec

= (A3FF + 1) × 1.085 μsec

= (41983 + 1) × 1.085

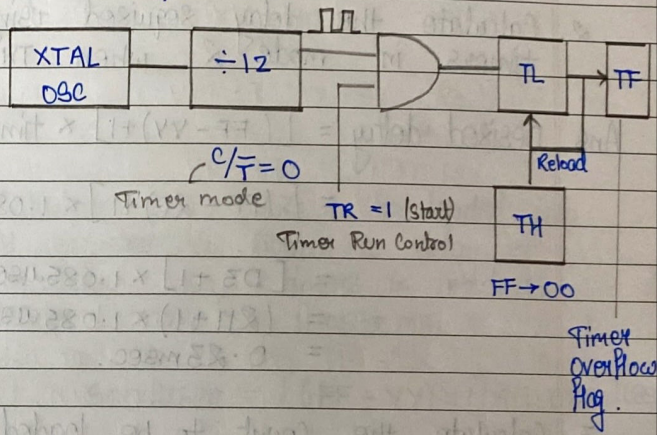
= 45552.64 msec / 1000

= 46.5 msec.

* Note:
Formula to calculate delay in decimal:

delay = [65536 - NNNNN] × time period.

* Mode 2: (8-bit mode):
(Auto Reload).



* Problems:

1. Calculate the max. delay obtained, using timers in mode 2.

Ans. Desired delay = [(FF - YY) + 1] × time period.

To obtain max. delay put YY as 00.

Desired delay = [(FF - 00) + 1] × 1.085 μsec.

= (255 + 1) × 1.085 μsec

= 0.27 msec.

2. Calculate the delay sequised using timers in mode 2 when TH = 2C.

Ans. Desired delay = [(FF - YY) + 1] × time period.

= [(FF - 2C) + 1] × 1.085 μsec.

= [D3 + 1] × 1.085 μsec

= (211 + 1) × 1.085 μsec

= 0.23 msec.

3. Calculate the count to be loaded into TH register to obtain a delay of 50 μsec. Use timers in mode 2. Assume XTAL = 11.0592 MHz

Ans. Desired delay = [(FF - YY) + 1] × time period.

$50\ \mu sec = [(FF - YY) + 1] \times 1.085\ \mu sec$

$\frac{50\ \mu sec}{1.085\ \mu sec} = (FF - YY) + 1.$

$46 - 1 = FF - YY$

$YY = FF - 45.$

$= FF - 2D$

$= D2.$

$\therefore TH = 0XD2.$

4. Calculate the count to be loaded into TH register to obtain a delay of 500 μsec using timers in mode 2. Assume XTAL = 11.0592 MHz

Ans.

$500\ \mu sec = [(FF - YY) + 1] \times 1.085\ \mu sec$

$461 - 1 = FF - YY$

$YY = FF - 460$

$= FF - 1CC$

$= FFFFFF33$

$\frac{500}{1.085} = 460.82$

> 0.27

0.27 msec

255 - 278 μsec? Hence not possible.

[Not possible to generate 500 μsec delay using mode 2].

5. Same abv problem use delay of 3 μsec.

FF - 2 76

=> FF - 2 => FD

Ans. Desired delay = [(FF - YY) + 1] × time period.

$3\mu sec = [(FF - YY) + 1] \times 1.085 \mu sec.$

$\left(\frac{3}{1.085}\right) - 1 = FF - YY.$

$1.7649 = FF - YY$

$\therefore YY = FF - 02$

$= FD.$

* Write an 8051 C program to toggle all the bits of port 1 continuously with some delay in between. Use Timer 0 in mode 1 to generate the delay.

```
#include <reg51.h>
void TODelay (void);
void main (void)
{
  while (1)
  {
    P1 = 0x00;
    TODelay ();
    P1 = 0xFF;
    TODelay ();
  }
}

void TODelay ()
{
  TMOD = 0X01 ; // timer 0 in mode 1.
  TL0  = 0XFE ;
  TH0  = 0XBD ; // delay size unknown.
  TR0  = 1    ; // start the timer
  while (TF0 == 0); // moniter timer overflow
                    flag.
  TR0 = 0 ;
  TF0 = 0 ;
}
```

Timer 0 in mode 1 has TH & TL always. ~~as mode 1 is 16 bit mode.~~ (FFFF - YYXX).

* Write an 8051 C program to toggle only bit $P_{1.5}$ continuously every 50msec, use timer 0 in mode 1.

```
#include <reg51.h>
void TODelay (void);
sbit mybit = P1^5;
```

Write an ... only bit $P_{2.7}$ continuously with 200msec delay use T1 in ...

```
void main (void)
{
  while (1)
  {
   mybit = ~mybit ; // using NOT operator
   T0Delay ();          to toggle P1.5.
  }
}

void T0Delay (void)
{
  TMOD = 0x01 ;   // timer 0 in mode 1.
  TH0  = 0x4B ;   // Count for 50msec delay
  TL0  = 0xFE ;
  TR0  = 1 ;      // start the timer.
  while (TF0 == 0) ;
  TR0  = 0 ;
  TF0  = 0 ;
}
```

* Write an 8051 C program to toggle all bits of P2 continuously every 25msec use timer 1 in mode 1.

T1 · 1 - 0msec
max delay
↓ so divide 500.

* Write an 8051 C program to toggle bits of P2 continuously every 500msec, use timer 1 in mode 1.

```
#include <8egs1.h>
void T1M1Delay (void) ;
void main (void)
{
  unsigned char i ;
  while (1)                    500 msec
  {                            50msec x 10.
    P2 = 0x00 ;
    T1M1Delay () ;
    P2 = 0xFF ;
    T1M1Delay () ;
    for (i = 0 ; i<10 ; i++)
  }
}

void T1M1Delay (void)
{
   TMOD = 0x10 ; // Timer 1 in mode 1.
   TH1  = 0x4B ; // 50msec delay.
   TR1  = 1 ;        → TL1 = 0xFE ;
   while (TF1 == 0) ;
```

0.27 msec in mode 2

```
    TR1 = 0 ;
    TF1 = 0 ;
}
```

* Write an 8051 C program to toggle only P1.5 continuously every 250 msec, use timer 0 in mode 2 to generate the delay.

Ans. Delay required = 250msec

$\therefore$ Delay = [(FF-YY)+1] × 1.085 μsec

250msec = [(FF-YY)+1]
1.085 μsec

250 × 40 — max delay 71.1 msec

choose 25μsec delay × 10000 = 250 msec delay

250 × 10000 = 2500000 μsec

25 μsec = [(FF-YY)+1]
1.085 μsec

28-1 = FF-YY

$\therefore$ YY = FF - 16 (in hexa).
= E9.

$\therefore$ TH0 = 0xE9.

```
#include <reg51.h>
void Tom2Delay (void);
void mybit = P1^5 ;
void main ()
{
   unsigned char x,y;
   while(1)
   {
     mybit = ~ mybit ;
     for (x=0; x<250 ; x++)
        for (y=0 ; y<400 ; y++) ; // for loop to repeat the delay 10000 times.
           Tom2Delay ();
   }
}

void Tom2Delay (void)
{
  TMOD = 0x02 ; // Timer 0 in mode 2.
  TH0 = 0xE9 ;
  TR0 = 1 ; // Start the timer.
  while (TF0 == 0) ;
  TR0 = 0 ;
  TF0 = 0 ;
}
```

* Write an 8051 C program to toggle only bit 2.7 continuously with 200 msec delay use T1 in mode 1.

250×40
20 μsec × (10000)
= 200 msec

```
#include <8051.h>
void delay(void);
sbit mybit = P2^7;
void main()
{
  unsigned char x, y;
  while(1)
  {
    mybit = ~mybit;
    for(x=0; x<250; x++)
      for(y=0; y<40; y++)
        delay();
  }
}
void delay()
{
  TMOD = 0X10;
  TH1 = 0XFF;
  TL1 = 0XEE;
  TR1 = 1;
  while(TF1 == 0);
  TR1 = 0;
  TF1 = 0;
}
```

$\frac{20}{1.085} - 1 = FFFF - YYYY$

$\therefore YYYY = FFEE$.

TH1 = FF

TL1 = EE

* Write an 8051 C program to generate a frequency of 2500 Hz on pin 2.7, use timer 1 mode 2 to create the delay.

Ans. Pin → 2.7

Freq → 2500 Hz.

w.k.t : $T = \frac{1}{f} = \frac{1}{2500} = 400$ μsec

$T = T_{ON} + T_{OFF}$

$T_{ON} = T_{OFF} = \frac{400}{2} = 200$ μsec

Delay $= [(FF - YY) + 1] \times 1.085$ μsec

$\frac{200}{1.085} = (FF - YY) + 1$

184 − 1 = FF − YY

YY = 255 − 183

= 48 (in hexadecimal).

∴ TH = 0X48 for 200 μsec delay.

```
#include <8051.h>
void TIM2Delay(void);
sbit mybit = P2^7;

void main(void)
{
  while(1)
  {
    mybit = ~mybit; // toggle P2.7
    TIM2Delay();
  }
}

void TIM2Delay()
{
  TMOD = 0X20; // Configure T1 in mode 2.
  TH1 = 0X48; // count for 200 micro sec delay
  TR1 = 1;
  while (TF1 == 0); //moniter TF1
  TR1 = 0; //stop the timer.
  TF1 = 0; //clear TF1.
}
```

* A switch is connected to Pin $P_{1.2}$, write an 8051 C program to monitor switch & create the following frequencies on Pin $P_{1.7}$. When sw = 0, 500 Hz needs to be generated & when sw = 1, 750 Hz needs to be generated use Timer 0 in mode 1 for both of them.

Ans. sw → $P_{1.2}$ $P_{1.7}$ → Square wave.

sw = 0, 500 Hz
sw = 1, 750 Hz
in timer 0 in mode 1.

$$T = \frac{1}{f} = \frac{1}{500} = 2ms \quad (0.002 \times 1000).$$

$$\therefore T_{ON} = T_{OFF} = \frac{2ms}{2} = 1ms.$$

$$T = \frac{1}{f} = \frac{1}{750} = 1.33ms$$

$$\therefore T_{ON} = T_{OFF} = \frac{1.33}{2} = 0.6ms.$$

```
#include <reg51.h>
sbit  sw  = P1^2;
sbit  out = P1^7;
void  Tom1Delay (unsigned char);

void main (void)
{
  sw = 1 ; //Configure sw as input.
  while(1)
  {
    out = ~out ; //toggling/generating square
                 wave on P1.7
    if (sw == 0)
        Tom1Delay (0) ;
    else
        Tom1Delay (1) ;
  }
}

void  Tom1Delay (unsigned char c)
{
  TMOD = 0x01 ; // Timer 0 in mode 1.
  if (c == 0)
  {
    TH0 = 0xFC ;   // generate square
    TL0 = 0x66 ;   wave of 500 Hz
  }                freq on P1.7
  else
  {
    TL0 = 0x9A ;   // generate square
    TH0 = 0xFD ;   wave of 750 Hz
  }                freq on P1.7.
  TR0 = 1 ; // start timer
  while( TF0 == 0) ; //Moniter flag.
  TR0 = 0 ; // stop timer
  TF0 = 0 ; // clear flag
}
```

* Programming Timers as counters :

P3 Bit	Function	Pin
P3.0	RxD	10
P3.1	TxD	11
P3.2	$\overline{INT0}$	12
P3.3	$\overline{INT1}$	13
P 3.4	T0	14
P3.5	T1	15
P3.6	$\overline{WR}$	16
P3.7	$\overline{RD}$	17

* Assume that 1Hz external clock is being fed into pin T1 (P3.5). Write a C program for counter 1 in mode 2 to count up and display the state of the TL count on P1. Start the count at 00.

```
#include <reg51.h>
//sbit T1 = P3^5;   T1 is P3^5 no need to declare
void main (void)
{
  T1 = 1; // make T1 as input
  TMOD = 0x60; // Timer 1 as counter 1 in mode2
  TH1 = 0; // start counting from zero.
  while (1)
  {
    do
    {
      TR1 = 1;      //start counter
      P1 = TL1;  // send count on P1
    } while (TF1 == 0);
  }
  TR1 = 0;  // stop the counter 1
  TF1 = 0;  // clear flag
}
```

* Assume that 1Hz external clock is being fed into pin T0 (P3.4). Write a C program for counter 0 in mode 1 to count the pulses and display the state of TH0 and TL0 registers on P2 & P1 reply.

```
#include <reg51.h>
void main (void)
{
  T0 = 1;  //make T0 as input.
  TMOD = 0x05;  // Configure T0 as counter 0
                   in mode1.
  TH0 = 0;
  TL0 = 0;  //clear TH & TL
  while (1)
  {
    do
    {
      TR0 = 1;      //start counter
      P1 = TL0;   //send TL value on P1
      P2 = TH0;   //  "  TH  "  "  P2.
    } while (TF0 == 0);
  }
```

```
    TR0 = 0 ; //stop counter
    TF0 = 0 ; // clear flag.
  }
}
```

* Serial data transmission

Note:

Imp * Timer 1 is used in mode 2 to generate the baud rate for serial data transmission

* With XTAL = 11.0592 MHz find the TH1 value needed to have the following baud rates:

a. 9600.
b. 2400.
c. 1800.

Ans: XTAL = 11.0592 MHz

Machine cycle freq = $\frac{11.0592}{12}$ = 921.6 KHz

Now: = $\frac{921.6}{32}$ by UART

= 28800 Hz

Now to get 1st case [9600]

we have: $\frac{28800}{3}$ = 9600

Now convert 3 into 2's compliment:

```
        0000 0011
1's     1111 1100
+              1
2's     1111 1101   = 0XFD.
```

∴ To generate a baud rate of 9600, FD is loaded into TH1.

ii. To get 2nd case [2400]

we have: $\frac{28800}{2400}$ = 12.

Now convert 12 into 2's compliment

```
        0000 1100
1's     1111 0011
+              1
2's     1111 0100   = 0XF4.
```

∴ F4 must be loaded into TH1 for 2400.

iii To get 3rd case: [1200]:

we have: $\frac{28800}{1200} = 24$.

Now convert 24 into 2's compliment.

```
   0001 1000
is 1110 0111
+             1
-------------------
   1110 1000   = 0XE8.
```

∴ E8 must be loaded into TH1 for 1200.

* Note:
Start bit is always low.
Stop bit/bits are always high.

* Note:
There are two lines on 8051 meant for serial data transmission / reception those are TxD & RxD apply.
Pin 3.0 for TxD. RxD
Pin 3.1 for RxD. TxD

* SCON:
It is 8 bit register used to program the start bit, stop bit and data bits of data framing, among other things.

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0	SCON.7	Serial Port Mode Specifier.
SM1	SCON.6	Serial " " "
SM2	SCON.5	Used for multiprocess comm.
REN	SCON.4	Set/Cleared by software to enabl
TB8	SCON.3	Not used disable see
RB8	SCON.2	Not used
TI	SCON.1	Transmit interrupt flag. Set by HW at the begin of the stop bit mode1 and cleared by SW.
RI	SCON.0	Receive interrupt flag set by HW at the begin of the stop bit mode1 and cleared by SW.

Note: Always make SM2, TB8 & RB8 = 0.

* SM0, SM1 :

SM0	SM1	
0	0	Serial mode 0
0	1	Serial mode 1
1	0	" " 2
1	1	" " 3

Always serial mode 1 is used. It has 8bit data, 1 stop bit & 1 start bit.

* SM2 : This enables the multiprocessing capability of 8051.

* REN (Receive enable) REN always 1 as it is full duplex.
It is bit addressable register :
→ When it is high, it allows 8051 to receive data on RxD pin.
→ If low it is disabled.

* TI (transmit interrupt)
When 8051 finishes the transfer of 8 bit character
→ It raises TI flag to indicate that it is ready to transfer another byte.
→ TI bit is raised at the beginning of the stop bit.

* RI (receive interrupt)
When 8051 receives data serially via RxD, it gets rid of the start & stop bit and places the byte in SBUF register.
→ It raises the RI flag bit to indicate that a byte has been received and should be picked up before it is lost.
→ RI is raised halfway through the stop bit.

* Syntax is : [transfer character serially] :

```
TMOD = 0x20 ; //configure Timer1 in mode2.
TH1  = 0xFD ; //9600 baud rate.
SCON = 0x50 ; //selects mode1 start bit
TR1 = 1 ;  //start timer1.
TI = 0 ;  //clear TI.
SBUF = Byte of data to be transmitted ;
while (TI == 0) ; //Monitor TI flag.
```

Receive
RI = 0

* Write a c program for 8051 to transfer the letter or character 'A' serially at 4800 baud rate continuously. Use 8bit data & one stop bit.

```
#include<reg51.h>
void main (void)
{
  TMOD = 0x20; //configure T1 in M2 to set
                 baud rate.
  TH1  = 0xFA; //baud rate = 4800.
  SCON = 0x50; // serial M1 & REN enabled.
  TR1 = 1 ; //start timer
  while(1)
  {
    SBUF = 'A' ; //place value in SBUF
    while (TI == 0) ; //moniter transmit flag
    TI = 0; // clear TI after the completion
              of transmission.
  }
}
```

* Same prev problem statement. characters 'GIT' and baud rate : 9600.

```
#include<reg51.h>
void  serialTx (unsigned char);
void main (void)
{
  TMOD = 0x20;
  TH1  = 0xFD;
  SCON = 0x50;
  TR1 = 1;
  while(1)
  {
     serialTx ('G');
     serialTx ('I');
     serialTx ('T');
  }
}

SerialTx( unsigned char x)
{
   SBUF = x; //load with char to be transmitted
   while (TI==0) ; //moniter TI.
   TI = 0;
{
```

20 in (hex) → binary space is obtained.

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

```
#include<reg51.h>
void main (void)
{
  TMOD = 0x20;
  TH1  = 0xFA;
  SCON = 0x50;
  TR1 = 1;  //start timer
  unsigned char mybyte;
  while (1)
  {
    while (RI==0); //wait to receive
    mybyte = SBUF; //write SBUF content
                   to mybyte.
    P1 = mybyte; // place the byte on P1.
    RI = 0;      //clear
  }
}
```

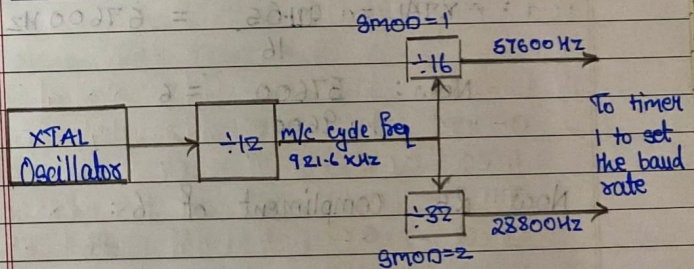
SMOD = 0 (Normal baud)
SMOD = 1 (baud x 2)
PCON = Power Control

* Doubling Baud Rate:
There are two ways to increase the baud rate of data transfer:
1. To use a high frequency crystal.
2. To change a bit in the PCON register.

PCON register is an 8-bit register:
When 8051 is powered up, SMOD is zero. We can set it to high by software and thereby double the baud rate.

(not used by us)

SMOD	-	-	-	GF1	GF0	PD	IDL



TH1 (Decimal)	(Hex)	SMOD=0	SMOD=1
-3	FD	9600	19200
-6	FA	4800	9600
-12	F4	2400	4800
-24	E8	1200	2400

Special function registers:

1. SCON → selects the mode, REN = 1
2. SBUF → holds byte for data
3. PCON → SMOD = 0 ; normal
   SMOD = 1 ; x2

* Find the TH1 value (in both decimal & hex) to set the baud rate to each of the following
a. 9600
b. 4800
if SMOD = 1. Assume XTAL = 11.0592MHz.

Ans.

a. 9600 baud rate, SMOD = 1.

Now: $XTAL = \frac{11.0592}{12}$ (x1000) $= 921.6$ KHz.

$XTAL = \frac{921.6}{16}$ (x1000) $= 57600$ Hz.

Now: $\frac{57600}{9600} = 6$

Now 2's compliment of 6:

```
    0 0 0 0 0 1 1 0
1's 1 1 1 1 1 0 0 1
+               1
2's 1 1 1 1 1 0 1 0
```

1111 1010 ⇒ 0xFA.

b. 4800 baud rate, SMOD = 1.

Now: $XTAL = \frac{11.0592}{12} = 921.6$ KHz.

$= \frac{921.6}{16} = 57600$ Hz

Now: $\frac{57600}{4800} = 12$.

Decimal value = -12

Now 2's compliment of 12:

```
    0 0 0 0 1 1 0 0
1's 1 1 1 1 0 0 1 1
+               1
2's 1 1 1 1 0 1 0 0
```

1111 0100 ⇒ 0xF4.

* Write an 8051 C program to send the two messages "Normal Speed" and "High speed" to the serial port. Assume that Sw is connected to pin P2.0, monitor its status and set the baud as follows:
Sw = 0, 28800 baud rate (0xFF)
Sw = 1, 56K baud rate
Assume XTAL = 11.0592 MHz for both cases.

```
#include<reg51.h>
sbit  sw = P2^0; // input switch
void main (void)
{
   unsigned char mess1[] = " Normal Speed";
   unsigned char mess2[] = " High Speed";
   TMOD = 0x20; // Timer 1 in mode 2
   TH1  = 0xFF; // Count for 28800 baud.
   SCON = 0x50; // Serial mode 1, REN=1
   TR1 = 1; //start timer.
   unsigned char z;
   if (sw==0)
   {
     for (z=0; z<12; z++)
```

PCON is not bit addressable
MSB of PCON is used to double the baud rate.

```
     {
       SBUF = mess1[z]; // place char in SBUF
       while (TI==0); //wait for transmission
       TI = 0; // clear TI
     }
   }
   else
     {
                                  (1000 0000)
       PCON = PCON & 0x80; // SMOD=1
       for(z=0; z=10; z++)
       {
       SBUF = mess2[z]; // place char in SBUF
       while(TI==0);
       TI = 0;
       }
     }
}
```

* LCD interfacing:

LCD Pin Descriptions:

Pin	Symbol	I/O	Description
1		-	
2		-	
3		-	
4	RS	I	
5	R/W	I	
6	E	I/O	Enable
7	DB0	I/O	The 8 bit data bus
8	DB1	I/O	"
9	DB2	I/O	"
10	DB3	I/O	"
11	DB4	I/O	"
12	DB5	I/O	"
13	DB6	I/O	"
14	DB7	I/O	"

* LCD interfacing diagram:

* LCD Command codes:

Code (Hex)	Command
1	clear display screen
2	
3	
4	
5	
6	
7	
8	
A	
C	
E	
F	
80	
C0	
38	

* Write an 8051 C program to send the letters 'M', 'D' and 'E' to the LCD. Use the busy flag method.

```
#include <8051.h>
// P1 is holding LCD data
sbit xs = P2^7;
```

(Lab program can also be written instead of this)

```
sbit rw = P2^5;
sbit en = P2^6;
sbit busy = P1^7;
void main(void)
{
lcdcmd (0x38) ; //use 2 lines 5x7 matrix.
lcdcmd (0x0E) ; //display ON cursor blinking.
lcdcmd (0x01) ; //clear display screen.
lcdcmd (0x06) ; //shift cursor to right.
lcdcmd (0x80) ; //line 1 position 1.

lcddata ('M') ; //send char M.
lcddata ('O') ;
lcddata ('E') ;
}

void lcdcmd (unsigned char value)
{
    lcdready () ; //check the busy flag.
    P1 = value ; // put the value/char on P1 pins
    rs = 0 ; //select cmd register
    rw = 0 ; // write to lcd.
    en = 1 ;
    msDelay(1) ;
    en = 0 ;
    return ;
}

void lcddata (unsigned char value)
{
    lcdready () ; // check busy flag.
    P1 = value ; // place value on P1.
    rs = 1 ;     // select data register.
    rw = 0 ;
    en = 1 ;
    msDelay(1) ;
    en = 0 ;
    return ;
}

void lcdready ()
{
    busy = 1 ; //make P1.7 as input.
    rs = 0 ; // select cmd register.
    rw = 1 ; // read the value.
    while (busy == 1)
    {
        en = 0 ;
```

en=0 to 1, accept & display.

```
        MSDelay(1);
        en = 1;
    }
}

void MSDelay(..)
{
...
}
```