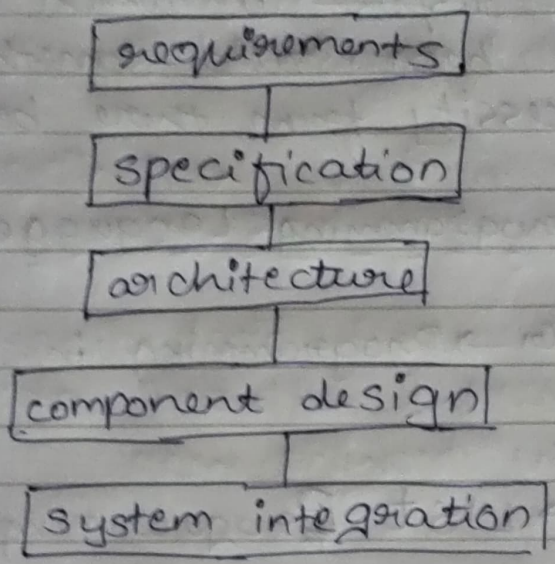
Embedded Computing
Unit I

Embedded System
A special purpose computer which performs a specific task

Define Embedded Computing System
Diff b/w general purpose & embedded computing system
List & explain characteracterics of ~~computing~~ embedded computing system

Levels of abstraction



Sample requirement form
name
purpose
inputs
outputs
functions
performance
manufacturing cost
power
physical size/weight

Prepare requirement form for any embedded product of your choice

Key
V4

Embedded Systems

1) List & explain the challanges in embedded computing system design.

2) Write a note on performance of embedded computing systems.

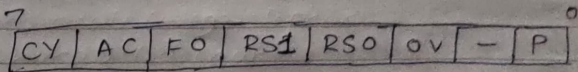
3) With a neat diagram explain the embedded system design process.

4) What is requirement & specification. Develop the sample requirement form for any embedded product of your choice

5) Explain with a neat diagram requirement analysis of a gps moving map

6) With a neat block diagram explain hardware & software architectures for the moving map.

- With a neat diagram explain program status



CY : Carry
AC : Auxiliary carry
F0 : Reserved for general purpose
RS1, RS0 : Register Select bits
OV : Overflow
P : Parity

- Write an 8051C program to send hex values for ASCII characters 0, 1, 2, 3, 4, 5, a, b, c, d, & to port P1

- Write an 8051C program to toggle all the bits of port1 continuosly

Date: / /

- Write a 8051 C program to send the values -4 to +4 to port P1

```
#include <reg51.h>
void main(void)
{
    signed char myarray[ ]={-4,+4,-3,+3,-2,+2,-1,+1,0};
    unsigned char z;
    for(z=0; z<9; z++)
    {
        P1=myarray[z];
    }
}
```

- Write a 8051 C program to toggle the LSB of port P1 50000 times

```
#include <reg51.h>
sbit mybit=P1^0;
void main(void)
{
 unsigned int z;
 for(z=0; z<50000; z++){
       mybit=0;  } Toggle
       mybit=1;  } P1_0
 }
}
```

- Write an 8051c program to toggle bits of P1 continuously with some delay.

```
#include <reg51.h>
void main (void)
{ unsigned int z;
  while (1)
  {
    P1 = 0x00; // clear P1
    for(z=0; z<40000; z++);
    P1 = 0xff; // set P1
  }
  for(z=0; z<40000; z++);
}
```

- Write an 8051c program to toggle port P1 continuously with 250ms delay

- Special function registers

- TMOD } 8-bit wide
- TCON }

TMOD

7	6	5	4	3	2	1	0
Gate	C/T	M_1	M_0	Gate	C/T	M_1	M_0
Timer 1				Timer 0			

Gate = 0; Software means of controlling timer

C/T = 0 ; Timer
1 ; Counter

Mode control

M_1	M_0	Mode selected
0	0	Mode 0 (13-bit)
0	1	Mode 1 (16-bit)
1	0	Mode 2 (8-bit)
1	1	Mode 3 (Split timer)

TCON

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0				

Timer overflow flag (T1)
Timer Run control bit (T1)

TR = 0 ; // stop timer
TR = 1 ; // Start timer

Date: / /

- Generate a delay of 50ms using timer 0 in mode 1

Desired delay = [(FFFF - YYXX) + 1] × 1.085 μs
(YYXX = THO, TLO)

XTAL freq = 11.0592 MHz

Machine cycle freq = XTAL freq / 12

= 11.0592 MHz / 12

= 921.6 kHz

$t = \frac{1}{F} = \frac{1}{921.6\,kHz} = 1.085\,\mu s$

Delay = [(FFFF - YYXX) + 1] × 1.085 μs

$\frac{50ms}{1.085\mu s} = [FFFF - YYXX] + 1$

46082 − 1 = [FFFF − YYXX]

46081 = FFFF − YYXX

B401 = FFFF − YYXX

YYXX = FFFF − B401 = 4BFE

∴ THO = 0x4B

TLO = 0xFE

1) TMOD = 0x01 ; // Timer 0 in mode 1

2) THO = 0x4B, TLO = 0xFE } 50 ms delay

3) TRO = 1; Start timer

4) while(TFO == 0); // wait for TFO to become

5) TRO = 0; // stop the timer 1

6) TFO = 0; // clear TFO for next round

```
#include "at89c51ed2.h"
void TOM1 delay();
void main(void)
{
  while(1)
  {
    P0 = 0x00;
    TOM1delay();
    P0 = 0x10;
    TOM1delay();
    P0 = 0x20;
    TOM1delay();
    P0 = 0x30;
    TOM1delay();
  }
}

void TOM1delay (void)
{
  TMOD = 0x01;
  TH0 = 0x4B;
  TL0 = 0xFE;
  TR0 = 1;
  while(TF0 == 0);
  TR0 = 0;
  TF0 = 0;
```

- Write an 8051c program to toggle all units of port 0, port1 and port2 continuously with a 250ms delay. Use logical operators for toggling

```
#include <reg51.h>
void delay(unsigned int);
void main(void)
{
  while(1)
  {
    P0 =~P0  //Not used to toggle
    P1 =P1^0xff; //Ex-or used to toggle
    P2 =~P2;
    delay(250);
  }
}
void delay(unsigned int itime)
{
   unsigned int i,j;
   for(i=0;i<itime;i++)
   for(j=0;j<1275;j++);
}
```

- Write an 8051c program to read bit P1.0 and send it to P2.7 after inverting it

```
#include <reg51.h>
sbit inbit = P1^0;
sbit outbit = P2^7;
```

```
bit membit;
void main(void)
{
  inbit=1; //make it as input pin
  while(1)
  {
    membit =inbit; //read from P1.0 store
                   it in membit
    outbit=~membit;
  }
}
```

- Write an 8051 c program to read P1.0 and P1.1, & send the ascii character to port 0 according to following table.

P1.1	P1.0	
0	0	8
0	1	0
1	0	5
1	1	1

} P_0

Sol")

```
#include<reg51.h>
void main(void)
{
  unsigned z;
  z=P1;
  z=z&0x03; //mask all the bits except last two
  switch(z)
  {
    case(0):
    {
      P0='8';
      break;
    }
    case(1):
    {
      P0='0';
      break;
    }
    case(2):
    {
      P0='5';
      break;
    }
    case(3);
    {
      P0='1';
      break;
    }
  }
}
```


- Data conversion program
- Write a 8051 c program to convert packed BCD 29 to ascii & display the bytes on P1 & P2

```
#include<reg51.h>
void main(void)
{
  unsigned char mybyte=0x29;
  unsigned char x,y,z;
  x=mybyte&0x0f; //mask higher nibble
```

```
P1 = x | 0x30; //ASCII value of 9 on P1
y = mybyte & 0xf0; //mask lower nibble
z = y >> 4;
P2 = z | 0x30; //ASCII value of 2 on P2
}
```

ASCII characters of packed BCD

- Write an 8051c program to convert ASCII digits of 4 & 9 to packed BCD and display them on port 1

```
#include<reg51.h>
void main(void)
{
    unsigned char bcdbyte;
    unsigned char x='4';
    unsigned char y='9';
    x = x & 0x0f; //mask 3
    x = x<<4; //x = 0x40
    y = y & 0x0f; //mask 3
    bcdbyte = x | y;
    P1 = bcdbyte;
}
```

- Assume that we have 4 bytes of hexadecimal data 25H, 62H, 3FH, 52H.
1) Find the checksum byte
2) Perform the checksum operation to ensure data integrity

3) If the second byte 62H has been changed to 22H show how checksum detects the error?

Sol: 1) checksum byte:

```
  25H
  62H
  3FH
  52H
 ------
 (1)18H
```

18H = 0001 1000

```
  1110 0111   1's complement
        1
 ---------
  1110 1000  = E8H
```

2) checksum operation

```
  25H
  62H
  3FH
  52H
  E8H
 ------
 (2)00  → data is not corrupted
```

3)

```
  25H
  22H
  3FH
  52H
  E8H
 ------
  1C0H
```

- Write an 8051C program to calculate the checksum byte for the data given in the previous example

```
#include<reg51.h>
void main(void)
{
 unsigned char mydata[] ={0x25,0x62,0x3F,0x52};
 unsigned char sum=0;
 unsigned char x, checksumbyte;
 for(x=0; x<4; x++)
   {
     P2=mydata[x];
     Sum=Sum+mydata[x];
     P1=sum;
   }
 chksumbyte=~sum+1;
  P0=chksumbyte;
}
```

- Write an 8051C program to check the integrity of the program to check the data given in the previous example. If the data is send, the ASCII char G to P0 otherwise send ascii character B to port 0

- Write an 8051C program to convert the hexadecimal value 0xfd to decimal and display the digits of port 0, port 1 and port 2