**MC Codes**

1. **Write a program in C language**
2. Write a program in C language to toggle all the LED’s interfaced to port P2 of 8051 / 89C51 continuously with delay of 100 ms. Use looping for delay.

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

while(1)

{

P2 = 0XFF;

msdelay(100);

P2 = 0X00;

msdelay(100);

}

}

void msdelay(unsigned int itime)

{

unsigned int i,j;

for(i=0;i<itime;i++)

{

for(j=0;j<1275;j++)

{ }

}

}

1. Write a program in C language for up/down counting of hex numbers up to two digits. Display the result on LEDs connected to PORT2.

**A] UP hex counter**

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

unsigned int z;

while(1)

{

for(z=0;z<=255;z++)

{

P2=z;

msdelay(500);

}

}

}

void msdelay(unsigned int itime)

{

unsigned int i,j;

for(i=0;i<itime;i++)

{

for(j=0;j<1275;j++)

{ }

}

}

**B] DOWN hex counter**

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

unsigned int z;

while(1)

{

for(z=255;z>=0;z--)

{

P2=z;

msdelay(500);

}

}

}

void msdelay(unsigned int itime)

{

unsigned int i,j;

for(i=0;i<itime;i++)

{

for(j=0;j<1275;j++)

{ }

}

}

1. **Write a program in C language**
2. Write a program in C language for up / down counting of BCD numbers up to two digits. Display the result on LEDs connected to PORT2.

**A] UP BCD counter**

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

unsigned int x,y;

while(1)

{

for(x=0;x<=9;x++)

{

for(y=0;y<=9;y++)

{

P2=(x<<4)|y;

msdelay(500);

}

}

}

}

void msdelay(unsigned int itime)

{

unsigned int i,j;

for(i=0;i<itime;i++)

{

for(j=0;j<1275;j++)

{}

}

}

**B] DOWN BCD counter**

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

unsigned int x,y;

while(1)

{

for(x=9;x>=0;x--)

{

for(y=9;y>=0;y--)

{

P2=(x<<4)|y;

msdelay(500);

}

}

}

}

void msdelay(unsigned int itime)

{

unsigned int i,j;

for(i=0;i<itime;i++)

{

for(j=0;j<1275;j++)

{}

}

}

1. Write a program in C language to display ASCII values from ‘A’ to ‘Z’ on LEDs connected to PORT2.

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

unsigned int z;

while(1)

{

for(z='A';z<='Z';z++)

{

P2=z;

msdelay(500);

}

}

}

void msdelay(unsigned int itime)

{

unsigned int i,j;

for(i=0;i<itime;i++)

{

for(j=0;j<1275;j++)

{}

}

}

1. **Write a program in C language**

Write a program to interface DAC 808 to 8051. Apply the digital input to obtain a square wave & sine wave.

1. **Square wave**

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

while(1)

{

P1 = 0X00;

msdelay(500);

P1 = 0XFF;

msdelay(500);

}

}

void msdelay(unsigned int itime)

{

unsigned int i,j;

for(i=0;i<itime;i++)

{

for(j=0;j<1275;j++)

{}

}

}

1. **Sine wave**

#include<reg51.h>

void main(void)

{

unsigned char sinvalue[12] = {128,192,238,255,238,192,128,64,17,0,17,64};

unsigned char i;

while(1)

{

for(i=0;i<12;i++)

{

P1 = sinvalue[i];

}

}

}

1. **Write a program in C language**

Write a program to interface DAC 808 to 8051. Apply the digital input to obtain a triangular wave & sine wave.

1. **Triangular wave**

#include<reg51.h>

void main(void)

{

unsigned int x,y;

while(1)

{

for(x=0;x<255;x++)

{

P1 = x;

}

for(y=255;y>0;y--)

{

P1 = y;

}

}

}

1. **Sine wave**

#include<reg51.h>

void main(void)

{

unsigned char sinvalue[12] = {128,192,238,255,238,192,128,64,17,0,17,64};

unsigned char i;

while(1)

{

for(i=0;i<12;i++)

{

P1 = sinvalue[i];

}

}

}

1. **Write a program in C language**
2. Write a program to interface stepper motor to 8051 and rotate it clock-wise / anti clockwise direction continuously.

**A] Clockwise direction**

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

while(1)

{

P1 = 0X06;

msdelay(500);

P1 = 0X0C;

msdelay(500);

P1 = 0X09;

msdelay(500);

P1 = 0X03;

msdelay(500);

}

}

void msdelay(unsigned int time)

{

unsigned int i,j;

for(i=0;i<time;i++)

{

for(j=0;j<1275;j++)

{}

}

}

**B] Anticlockwise direction**

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

while(1)

{

P1 = 0X03;

msdelay(500);

P1 = 0X09;

msdelay(500);

P1 = 0X0C;

msdelay(500);

P1 = 0X06;

msdelay(500);

}

}

void msdelay(unsigned int time)

{

unsigned int i,j;

for(i=0;i<time;i++)

{

for(j=0;j<1275;j++)

{}

}

}

1. Draw the complete interfacing diagram & calculate the count to be loaded for rotating the motor through 90 degrees.

**Calculation**

**Step angle = 7.5**

**Count value = 90/(7.5\*4)**

**= 3**

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

unsigned int k;

for(k=0;k<3;k++)

{

P1 = 0X06;

msdelay(500);

P1 = 0X0C;

msdelay(500);

P1 = 0X09;

msdelay(500);

P1 = 0X03;

msdelay(500);

}

while(1);

}

void msdelay(unsigned int time)

{

unsigned int i,j;

for(i=0;i<time;i++)

{

for(j=0;j<1275;j++)

{}

}

}

1. **Write a program in C language**
2. Write a program to interface stepper motor to 8051 and rotate it clock-wise / anti clockwise direction continuously.

**DONE above**

1. Draw the complete interfacing diagram & calculate the count to be loaded for rotating the motor through 180 degrees.

**Calculation**

**Step angle = 7.5**

**Count value = 180/(7.5\*4)**

**Count value = 6**

#include<reg51.h>

void msdelay(unsigned int);

void main(void)

{

unsigned int k;

for(k=0;k<6;k++)

{

P1 = 0X06;

msdelay(500);

P1 = 0X0C;

msdelay(500);

P1 = 0X09;

msdelay(500);

P1 = 0X03;

msdelay(500);

}

while(1);

}

void msdelay(unsigned int time)

{

unsigned int i,j;

for(i=0;i<time;i++)

{

for(j=0;j<1275;j++)

{}

}

}

1. **Write a program in Assembly / C language**
2. Write a program to perform addition, subtraction of two 16-bit numbers. Store the result in RAM locations 50H (LS Byte) & 51H (MS Byte).

First 16 bit number = 1208H

Second 16 bit number = A56DH

Result : 12 08

+ A5 6D

B7 75

RAM address Data

50H 08H+6DH

51H 12H+A5H+carry

**A] Addition**

ORG 00H

CLR C

MOV A, #08H

ADD A, #6DH

MOV 50H, A

MOV A, #12H

ADDC A, #A5H

MOV 51H, A

MOV 52H, C

END

**B] Subtraction**

**55ACH - 1234H = 4378H**

ORG 00H

CLR C

MOV A, #ACH

SUBB A, #34H

MOV 50H, A

MOV A, #55H

SUBB A, #12H

MOV 51H, A

MOV 52H, C

END

1. Write a program for non overlapping (10 Bytes from 20H to 40H onwards) and overlapping (10 Bytes from 20H to 25H onwards) memory block transfer.

**A] Non overlapping**

ORG 00H

MOV R2, #0AH

MOV R0, #20H

MOV R1, #40H

loop : MOV A, @R0

MOV @R1, A

INC R0

INC R1

DJNZ R2, loop

END

**B] Overlapping**

ORG 00H

MOV R2, #0AH

MOV R0, #29H

MOV R1, #2EH

loop : MOV A, @R0

MOV @R1, A

DEC R0

DEC R1

DJNZ R2, loop

END