**Embedded Systems Lab**

**Assignment 5**

**Name** – Neelangshu Roy

**Reg No** – 20214060

**Group** – C1

Q1. Write a C program in Keil to print Fibonacci numbers. Take input from port P0 and display output to port P3 with delay of 100 milliseconds in between each output.

Create manual delay using loop.

Code -

#include<reg51.h>

void createDelay(unsigned int t);

void main(void)

{

unsigned int n,i,a,b,c;

a=0;b=1;

n=P0;

if(n>0)

{

P3=a;

createDelay(100);

n--;

P0=n;

}

if(n>0)

{

P3=b;

createDelay(100);

n--;

P0=n;

}

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

{

c=a+b;

P3=c;

a=b;b=c;

createDelay(100);

P0=n-i-1;

}

}

void createDelay(unsigned int t) // t is in milliseconds

{

unsigned int i,j;

t\*=10;

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

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

}

// 0,1,1,2,3,5,8,13,21,34 -> 00100010

Q2. Write a C program in Keil to find square root of a number and display it as output on port.

Code -

#include<reg51.h>

#include<math.h>

void main(void)

{

unsigned int res;

while(1)

{

res=P0;

res=(int)sqrt(res);

P1=res;

}

}

Q3. Write a C program in Keil to sort a list of n numbers and display them periodically on P0,P1,P2 and P3 with a delay of 150 milliseconds in between each display.

Code -

#include<reg51.h>

#include<stdlib.h>

void createDelay(int t);

void main(void)

{

unsigned int arr[]={9,8,7,6,5,4,3,2,1,10};

unsigned int i,j,n;

n=sizeof(arr)/sizeof(arr[0]);

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

{

int mini=arr[i],pos=i;

for(j=i+1;j<n;j++)

if(arr[j]<mini)

{

mini=arr[j];pos=j;

}

if(pos!=i)

{

arr[i]+=arr[pos];

arr[pos]=arr[i]-arr[pos];

arr[i]-=arr[pos];

}

}

i=0;j=0;

P0=0;P1=0;P2=0;P3=0;

while(1)

{

if(j==0)P0=arr[i];

if(j==1)P1=arr[i];

if(j==2)P2=arr[i];

if(j==3)P3=arr[i];

createDelay(150); // in milliseconds

i=(i+1)%n;

j=(j+1)%4;

}

}

void createDelay(int t)

{

int i,j;

t\*=10;

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

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

}

EdSim programs :

Q1. Sort the string "embedded systems", storing each character as its integer ASCII equivalent.

mov r2, #00

loop1: mov a, r2

anl psw, #7fh

subb a, 30h ; 30h stores the length of the string, or the integer array in this case

jz getout

mov a, r2

inc a

mov r3, a

loop2: mov a, r3

anl psw, #7fh

subb a, 30h

jz exit

mov a, #40h ; 40h is the starting or base address of the integer array

add a, r2

mov r0, a

subb a, r2

add a, r3

mov r1, a

mov a, @r1 ; indirect addressing mode is used here

subb a, @r0

jnc go\_on

mov a, @r1

mov r4, a

mov a, @r0

mov @r1, a

mov a, r4

mov @r0, a

go\_on: inc r3

ljmp loop2

exit: inc r2

ljmp loop1

getout: nop

end

Q2. Count number of times 'd' occurs in the string "embedded systems" and display it at a memory location using indirect memory addressing mode.

mov r3, #50h ; starting point in array, representing the characters as their ASCII values

mov r4, #00 ; index pointer

loop: mov a, r4

subb a, 40h ; 40h stores array length, checks if i==n, then break out of loop

jz getout

mov a, r4

add a, r3

mov r0, a ; r0 stores the address = starting point + offset

mov a, @r0 ; indirect addressing mode is used here

subb a, 30h ; 30h stores 'd' i.e. 100, ASCII value of 'd', if deducting this equals exactly zero,

jnz go\_on ; then we do NOT jump to go\_on

inc r1 ; we increase the value in register r1

go\_on: inc r4 ; increment index pointer

ljmp loop

getout: mov 20h, r1 ; store the frequency of 'd' at address 20h

end