ADC:

#include <avr/io.h>

#include <util/delay.h>

#define LCD\_DATA PORTD //LCD data port

#define LCD\_ctrl PORTB

#define en PB2 // enable

#define rw PB1 // read/write

#define rs PB0 // register select

#define F\_CPU 1000000UL

void LCD\_cmd(unsigned char cmd);

void init\_LCD(void);

void LCD\_write(unsigned char data);

void LCD\_Sting\_Write(unsigned char \*string) ;

void LCD\_set\_cursor(char a, char b);

unsigned int read\_adc();

void init\_adc();

int main(void)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Prog Initialization\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int val,one,ten,b;

float a;

DDRD|=0xFF; // setting the port B as output

DDRB|=0X07; // setting for port D pin 0,1,2 as output

init\_LCD();

// initialization of LCD

init\_adc();

\_delay\_ms(100);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

LCD\_String\_Write("TS ACTIVATED");

\_delay\_ms(500);

init\_LCD(); // CLEAR LCD

LCD\_String\_Write("ROOM TEMP=");

while(1)

{ LCD\_set\_cursor(1,11);

a=read\_adc();

\_delay\_ms(500);

a=(a/2);

b=a;

one=(b%10)+(0x30);

ten=(b/10)+(0x30);

if(ten != 0x30)

{

LCD\_write(ten);

LCD\_set\_cursor(1,12);

}

LCD\_write(one);

}

return 0;

}

void init\_LCD(void)

{

LCD\_cmd(0x38); // initialization of 8bit mode

\_delay\_ms(50);

LCD\_cmd(0x01); // clear LCD

\_delay\_ms(50);

LCD\_cmd(0x0E); // Setting cursor ON

\_delay\_ms(50);

LCD\_cmd(0x80); // —8 go to first line and –0 is for 0th position

\_delay\_ms(50);

}

void init\_adc()

{

// AREF = AVcc

ADMUX = (1<<REFS0);

// ADC Enable and prescaler of 128

// 16000000/128 = 125000

ADCSRA = (1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0);

}

unsigned int read\_adc()

{// select the corresponding channel 0~7

// ANDing with ’7' will always keep the value

// of ‘ch’ between 0 and 7

int ch=2;

ch &= 0b00000111; // AND operation with 7

ADMUX = (ADMUX & 0xF8)|ch; // clears the bottom 3 bits before ORing

// start single convertion

// write ’1' to ADSC

ADCSRA |= (1<<ADSC);

// wait for conversion to complete

// ADSC becomes ’0' again

// till then, run loop continuously

while(ADCSRA & (1<<ADSC));

return (ADC);

}

void LCD\_cmd(unsigned char cmd)

{

LCD\_ctrl =(0<<rs)|(0<<rw)|(1<<en);

//LCD\_ctrl = 0X04;

LCD\_DATA=cmd;

\_delay\_ms(1);

LCD\_ctrl =(0<<rs)|(0<<rw)|(0<<en);

//LCD\_ctrl = 0X00;

\_delay\_ms(1);

}

void LCD\_write(unsigned char data) // Writing a character

{

LCD\_ctrl = (1<<rs)|(0<<rw)|(1<<en);

LCD\_ctrl = 0X05;

LCD\_DATA= data;

\_delay\_ms(1);

LCD\_ctrl = (1<<rs)|(0<<rw)|(0<<en);

//LCD\_ctrl = 0x01;

\_delay\_ms(1);

}

void LCD\_String\_Write(unsigned char \*string) //store address value of the string in //pointer\*string

{

int i=0;

while(string[i]!= '\0') // Loop will execute till it detects null character

{

LCD\_write(string[i]); // sending data on LCD byte by byte

i++;

\_delay\_ms(10);

}

}

void LCD\_set\_cursor(char a, char b)

{

if(a == 1)

LCD\_cmd(0x80 + b);

else if(a == 2)

LCD\_cmd(0xC0 + b);

}

1.LCD:

#include <avr/io.h>

#include <util/delay.h>

#include "lcd.h"

#define LCD\_DATA PORTD //LCD data port

#define LCD\_ctrl PORTB

#define F\_CPU 1000000UL

void LCD\_cmd(unsigned char cmd);

void init\_LCD(void);

void LCD\_write(unsigned char data);

void LCD\_Sting\_Write(unsigned char \*string) ;

int main(void)

{

DDRD|=0xFF; // setting the port B as output

DDRB|=0X07; // setting for port D pin 0,1,2 as output

init\_LCD(); // initialization of LCD

\_delay\_ms(100);

LCD\_String\_Write("Shreyas Pillu");

\_delay\_ms(500);

LCD\_cmd(LINE\_2); //Line 2 command

LCD\_String\_Write("Gr8 Anil Kumar");

\_delay\_ms(500);

/\*LCD\_cmd(0X94); //Line 3 command

LCD\_String\_Write("Prof. Varun M");

\_delay\_ms(500);

LCD\_cmd(0XD4); //Line 4 command

LCD\_String\_Write("Manager Vemula");

\_delay\_ms(100);\*/

LCD\_cmd(0X0F); // blink the cursor after printing is done

return 0;

}

void LCD\_cmd(unsigned char cmd)

{

LCD\_ctrl =(0<<RS)|(0<<RS)|(1<<EN);

//LCD\_ctrl = 0X04;

LCD\_DATA=cmd;

\_delay\_ms(50);

LCD\_ctrl =(0<<RS)|(0<<RW)|(0<<EN);

//LCD\_ctrl = 0X00;

\_delay\_ms(50);

}

void LCD\_write(unsigned char data) // Writing a character

{

LCD\_ctrl = (1<<RS)|(0<<RW)|(1<<EN);

//LCD\_ctrl = 0X05;

LCD\_DATA= data;

\_delay\_ms(50);

LCD\_ctrl = (1<<RS)|(0<<RW)|(0<<EN);

//LCD\_ctrl = 0x01;

\_delay\_ms(50);

}

void LCD\_String\_Write(unsigned char \*string) //store address value of the string in pointer \*string

{

int i=0;

while(string[i]!= '\0') // Loop will execute till it detects null character

{

LCD\_write(string[i]); // sending data on LCD byte by byte

i++;

\_delay\_ms(100);

}

}

2.BLINKING LED:

#include<avr/io.h>

#include<util/delay.h>

#define blink\_port\_D PORTD

#define blink\_port\_A PORTA

#define blink\_port\_B PORTB

#define blink\_port\_C PORTC

#define F\_CPU 1000000UL

int main(void)

{

DDRD = DDRD |(0X01<<PD6);

DDRA = DDRA |(0x01<<PA6);

DDRC = DDRC |(0X01<<PB6);

DDRB =DDRB |(0X01<<PC6);

while('!')

{

blink\_port\_D |= (0X01<<PD6);

\_delay\_ms(500);

blink\_port\_D &= ~(0X01<<PD6);

\_delay\_ms(500);

blink\_port\_A |=(0X01<<PA6);

\_delay\_ms(500);

blink\_port\_A &= ~(0X01<<PA6);

\_delay\_ms(500);

blink\_port\_B |=(0X01<<PB6);

\_delay\_ms(500);

blink\_port\_B &= ~(0X01<<PB6);

\_delay\_ms(500);

blink\_port\_C |=(0X01<<PC6);

\_delay\_ms(500);

blink\_port\_C &= ~(0X01<<PC6);

\_delay\_ms(500);

}

}

3.BLINKLED1:

#include<avr/io.h>

#include<util/delay.h>

#define blink\_port PORTD

#define F\_CPU 1000000UL

int main(void)

{

DDRD = DDRD |(0X01<<PD6);

while('!')

{

blink\_port |= (0X01<<PD6);

\_delay\_ms(500);

blink\_port &= ~(0X01<<PD6);

\_delay\_ms(500);

}

return 0;

}

4.PWM LIGHT:

#include<avr/io.h>

#include<util/delay.h>

#define FAST\_PWM\_MODE (0X09<<WGM01)

#define NON\_INV\_PWM (0X01<<COM01)

#define CLOCK\_TYPE (0X01<<CS00)

#define TIMER\_CONTROL\_REG0 TCCR0

#define F\_CPU 1000000UL

void PWM\_val(uint8\_t dutyvalue);

void PWM\_INT(void);

int main(void)

{

uint8\_8 brigtness =0;

PWM\_INIT();

//infinite loop

while(1)

{

//fade in

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

{

PWM\_val(brightness);

\_delay\_ms(20);

}

\_delay\_ms(500);

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

{

//fade out

PWM\_val(brightness);

\_delay\_ms(20);

}

\_delay\_ms(500);

}

}

5.TIMER:

#include <avr/io.h>

#include <util/delay.h>

#define LCD\_DATA PORTD //LCD data port

#define LCD\_ctrl PORTB

#define en PB2 // enable

#define rw PB1 // read/write

#define rs PB0 // register select

#define F\_CPU 1000000UL

void LCD\_cmd(unsigned char cmd);

void init\_LCD(void);

void LCD\_write(unsigned char data);

void LCD\_Sting\_Write(unsigned char \*string) ;

void LCD\_set\_cursor(char a, char b);

int main(void)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Prog Initialization\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int ones=0x30, tens=0x30, setclk =0, val;

DDRD|=0xFF; // setting the port B as output

DDRB|=0X07;

PORTC = 0x00;

DDRC=0x00;

// setting for port D pin 0,1,2 as output

init\_LCD();

\_delay\_ms(100); // initialization of LCD

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

LCD\_String\_Write("Press Start");

while(setclk == 0)

{

PORTC = 0x00;

if(bit\_is\_set(PINC,PC1))

{

setclk = 1;

LCD\_set\_cursor(1,0);

LCD\_String\_Write("Time Count =");

\_delay\_ms(500);

}

else

setclk = 0;

}

while(setclk == 1)

{ PORTC = 0x00;

LCD\_set\_cursor(1,13);

if(tens>0x30)

{

LCD\_write(tens);

if(tens == 0x3a)

{

LCD\_set\_cursor(1,0);

LCD\_String\_Write("time count ");

\_delay\_ms(1000);

}

}

LCD\_write(ones);

\_delay\_ms(1000);

ones++;

if(ones == 0x3a)

{

tens++;

ones=0x30;

}

if(bit\_is\_set(PINC,PC0))

{

setclk=0;

LCD\_set\_cursor(1,0);

LCD\_String\_Write("time count ");

}

}

return 0;

}

void init\_LCD(void)

{

LCD\_cmd(0x38); // initialization of 8bit mode

\_delay\_ms(50);

LCD\_cmd(0x01); // clear LCD

\_delay\_ms(50);

LCD\_cmd(0x0E); // Setting cursor ON

\_delay\_ms(50);

LCD\_cmd(0x80); // —8 go to first line and –0 is for 0th position

\_delay\_ms(50);

}

void LCD\_cmd(unsigned char cmd)

{

LCD\_ctrl =(0<<rs)|(0<<rw)|(1<<en);

//LCD\_ctrl = 0X04;

LCD\_DATA=cmd;

\_delay\_ms(1);

LCD\_ctrl =(0<<rs)|(0<<rw)|(0<<en);

//LCD\_ctrl = 0X00;

\_delay\_ms(1);

}

void LCD\_write(unsigned char data) // Writing a character

{

LCD\_ctrl = (1<<rs)|(0<<rw)|(1<<en);

LCD\_ctrl = 0X05;

LCD\_DATA= data;

\_delay\_ms(1);

LCD\_ctrl = (1<<rs)|(0<<rw)|(0<<en);

//LCD\_ctrl = 0x01;

\_delay\_ms(1);

}

void LCD\_String\_Write(unsigned char \*string) //store address value of the string in pointer \*string

{

int i=0;

while(string[i]!= '\0') // Loop will execute till it detects null character

{

LCD\_write(string[i]); // sending data on LCD byte by byte

i++;

\_delay\_ms(10);

}

}

void LCD\_set\_cursor(char a, char b)

{

if(a == 1)

LCD\_cmd(0x80 + b);

else if(a == 2)

LCD\_cmd(0xC0 + b);

}