**Pract 4 : LED Interfacing with ARM Controller**

#include <lpc21xx.h>

void delay (unsigned int);

int main()

{

IODIR1 = 0Xffffffff;

PINSEL0 = 0X00000000;

PINSEL1 = 0XFFFFFFFf;

while(1)

{

IOSET1 = 0X00ff0000;

delay(1000);

IOCLR1 = 0X00fF0000;

delay(1000);

}

}

void delay(unsigned int k)

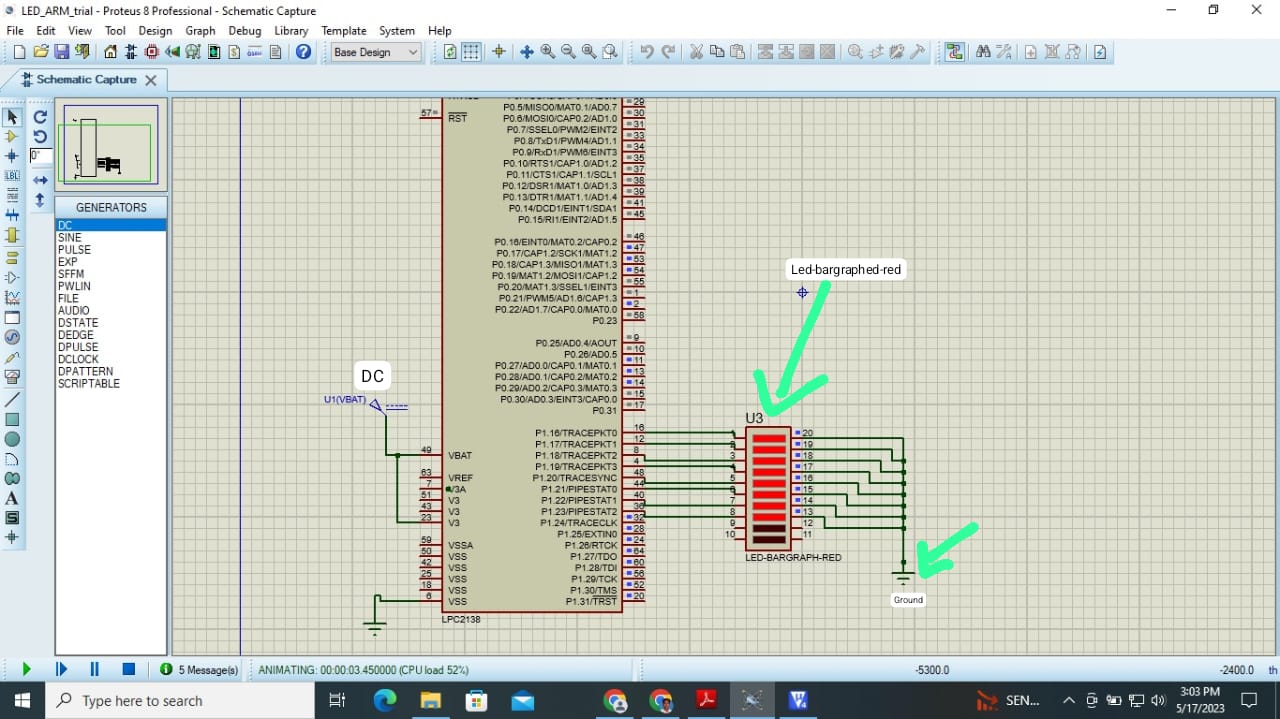
{

int i,j;

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

for(j=0;j<=100;j++);

}



// Pract 5 : Relay and LED Interfacing with ARM Controller

#include <lpc213x.h>

void delay (unsigned int);

int main()

{

IODIR0 = 0X00000001;

PINSEL0 = 0XFFFFFFFC;

PINSEL1 = 0XFFFFFFFf;

while(1)

{

IOSET0 = 0X00000001;

delay(100);

IOCLR0 = 0X00000001;

delay(100);

}

}

void delay(unsigned int k)

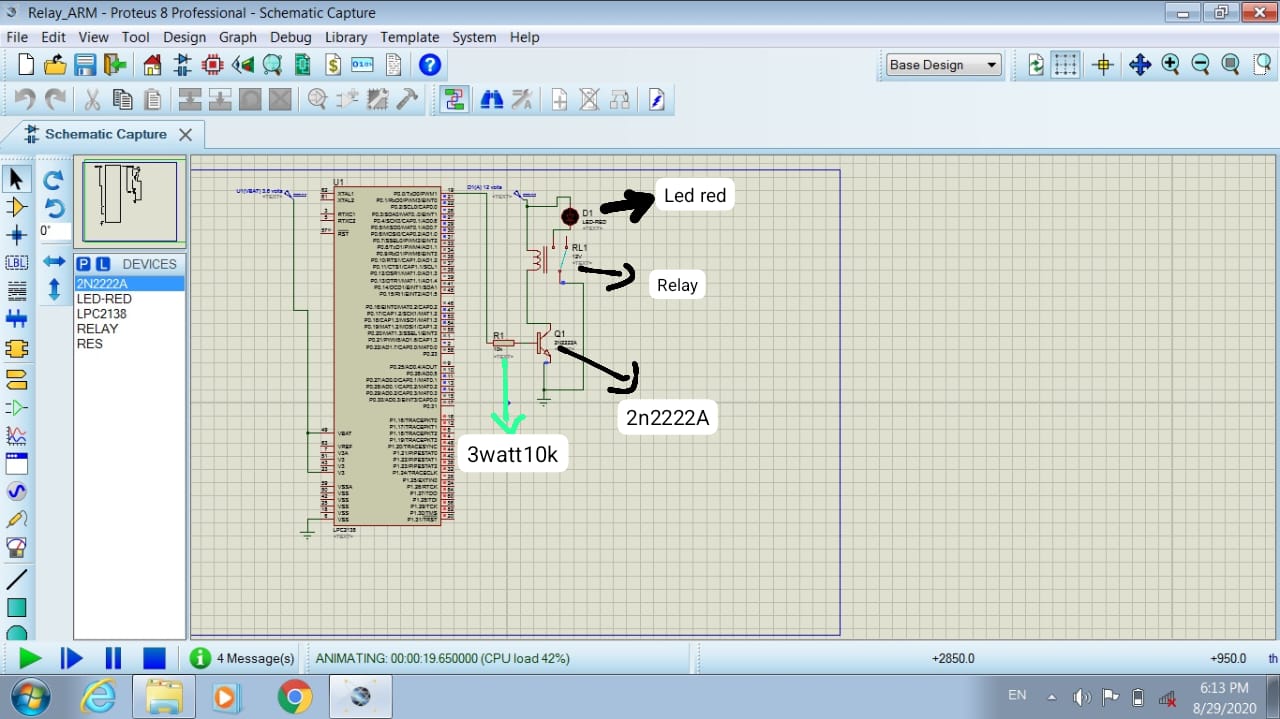
{

int i,j;

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

for(j=0;j<=1000;j++);

}



**Pract 6 : Interfacing stepper motor with ARM Controller Clockwise rotation**

#include <LPC214X.H>

void delay(unsigned int count)

{

unsigned int i ,j= 0;

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

for(j = 0; j <= 10; j ++);

}

unsigned long i;

int main()

{

IODIR1 = 0x00FF0000;

// i = 0x00110000;

while(1)

{

IOSET1 = 0x00880000;

delay(10000);

IOCLR1 = 0x00FF0000;

IOSET1 = 0x00440000;

delay(10000);

IOCLR1 = 0x00FF0000;

IOSET1 = 0x00220000;

delay(10000);

IOCLR1 = 0x00FF0000;

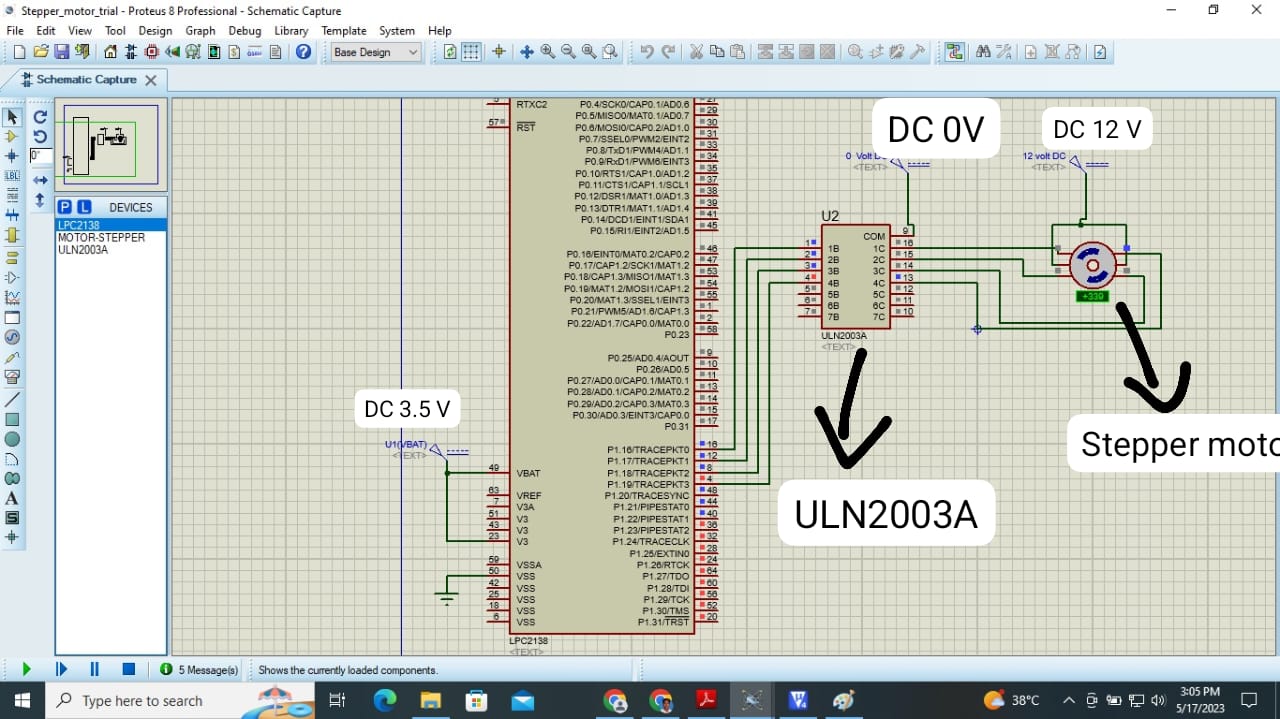
IOSET1 = 0x00110000;

delay(10000);

IOCLR1 = 0x00FF0000;

}

}



**Anticlockwise step rotation**

#include <LPC214X.H>

void delay(unsigned int count)

{

unsigned int i,j = 0;

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

{

for(j=0;j<=5;j++);

}

}

int main()

{

IODIR1 = 0x00FF0000;

while(1)

{

IOSET1 = 0x00FF0000;

delay(10000);

IOCLR1 = 0x00FF0000;

// IOSET1 = 0x00020000;

delay(10000);

// IOCLR1 = 0x00FF0000;

// IOSET1 = 0x00040000;

// delay(10000);

// IOCLR1 = 0x00FF0000;

// IOSET1 = 0x00080000;

// delay(10000);

// IOCLR1 = 0x00FF0000;

}

}

**Pract 7 : Interfacing switch & DC motor with ARM Controller**

#include <lpc213x.h>

// DC motor & Switch Interface

//DC motor connected to P0.0 & P0.1 , Switch to P0.5

int getPinState(int pinNumber);

int main()

{

IODIR0 |= 0X00000003; // P0.0 & P0.1 as Output pins and P0.5 as Input Pin

PINSEL0 &= 0XFFFFF3F0; // P0.0 , P0.1 , P0.5 as GPIO Pins

while(1)

{

if (getPinState(5))

{

IOSET0 = 0X00000001; // P0.0 Logic High (data bit 1 i.e. high voltage level)

IOCLR0 = 0X00000002; // P0.1 Logic LOW (data bit 0 i.e. low voltage level)

}

else

{

IOSET0 = 0X00000002; //P0.1 High ( data bit 1 i.e. high voltage level)

IOCLR0 = 0X00000001; // P0.0 LOW (data bit 0 i.e. low voltage level)

}

}

}

int getPinState(int pinNumber)

{

// Read the current state of all pins in GPIO block 0

int pinBlockState = IOPIN0;

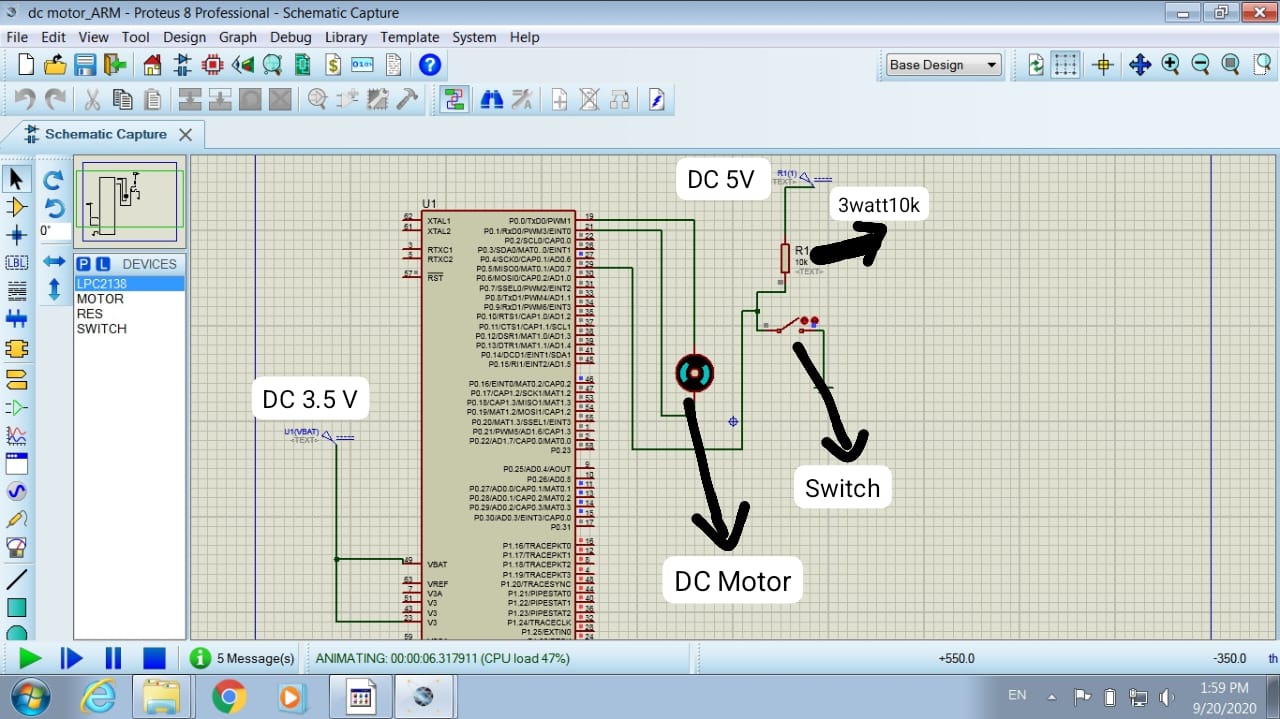
// Read the value of 'pinNumber P0.5 '

int pinState = (pinBlockState & (1 << pinNumber));// ? 1 : 0;

// Return the value of pinState

return pinState;

}

****

**Pract 8 : Interfacing DAC with ARM Controller**

#include <LPC213X.H> // DAC triangular wave genration

#include<math.h>

delay()

{int i=0;for(i=0;i<400;i++);}

void triangle() // func to generate triangular wave

{

unsigned int i;

for(i=0;i<0x0ff;i++)

{

DACR=0x00000000|(i<<6); // left shift to increase contents of DAC register

delay();

}

for(i=0x0ff;i>0;i--)

{

DACR=0x00000000|(i<<6);

delay();

}

}

void main(void)

{

PINSEL1 =0x00080000; // configure GPIO pin P0.25 as Aout for DAC operations

// by writing 10 at bit 19 & 18 of

PINSEL1 register

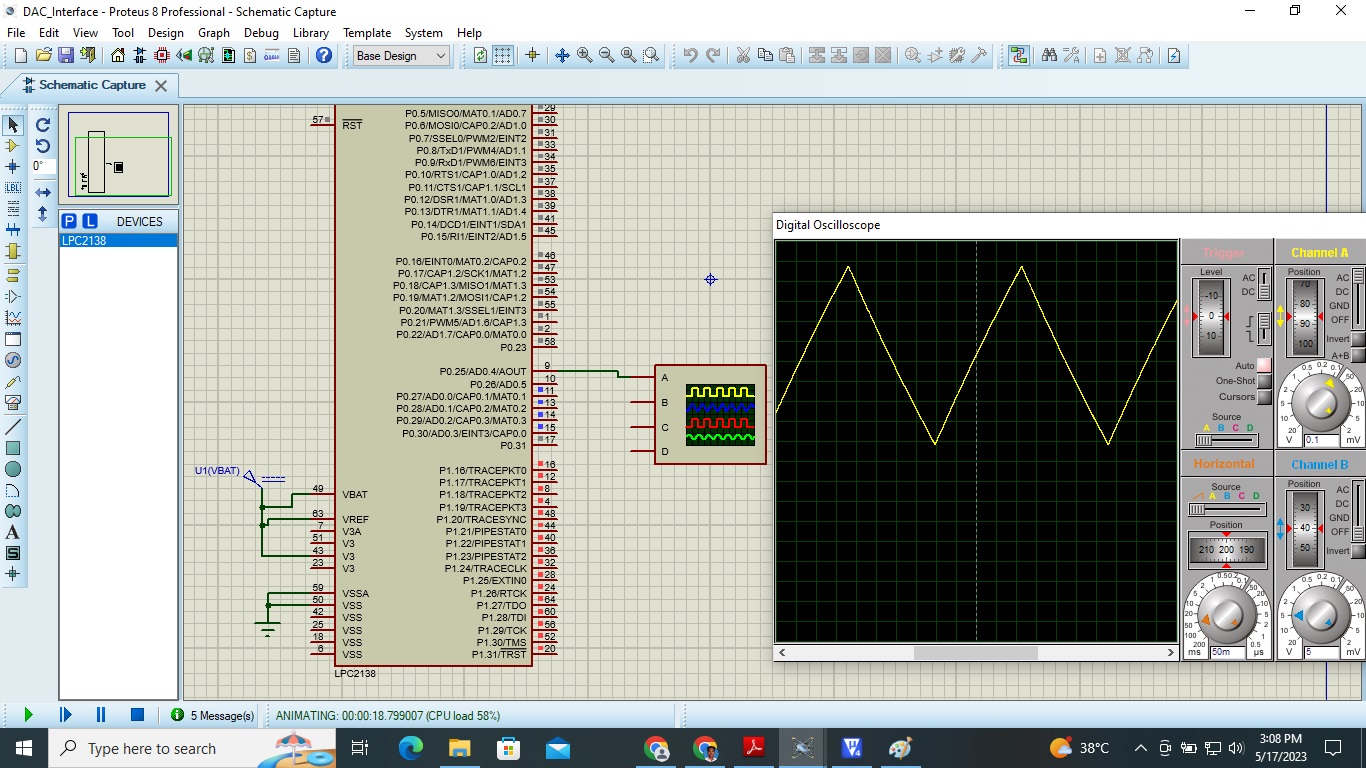
while(1)

{

triangle();

}

}

****

**Pract 9 : Interfacing Buzzer with ARM Controller**

#include <lpc214x.h>

void delay (unsigned int);

int main()

{

IODIR0 = 0X00000001;

PINSEL0 = 0XFFFFFFFC;

PINSEL1 = 0XFFFFFFFf;

while(1)

{

IOSET0 = 0X00000001;

delay(100);

IOCLR0 = 0X00000001;

delay(100);

}

}

void delay(unsigned int k)

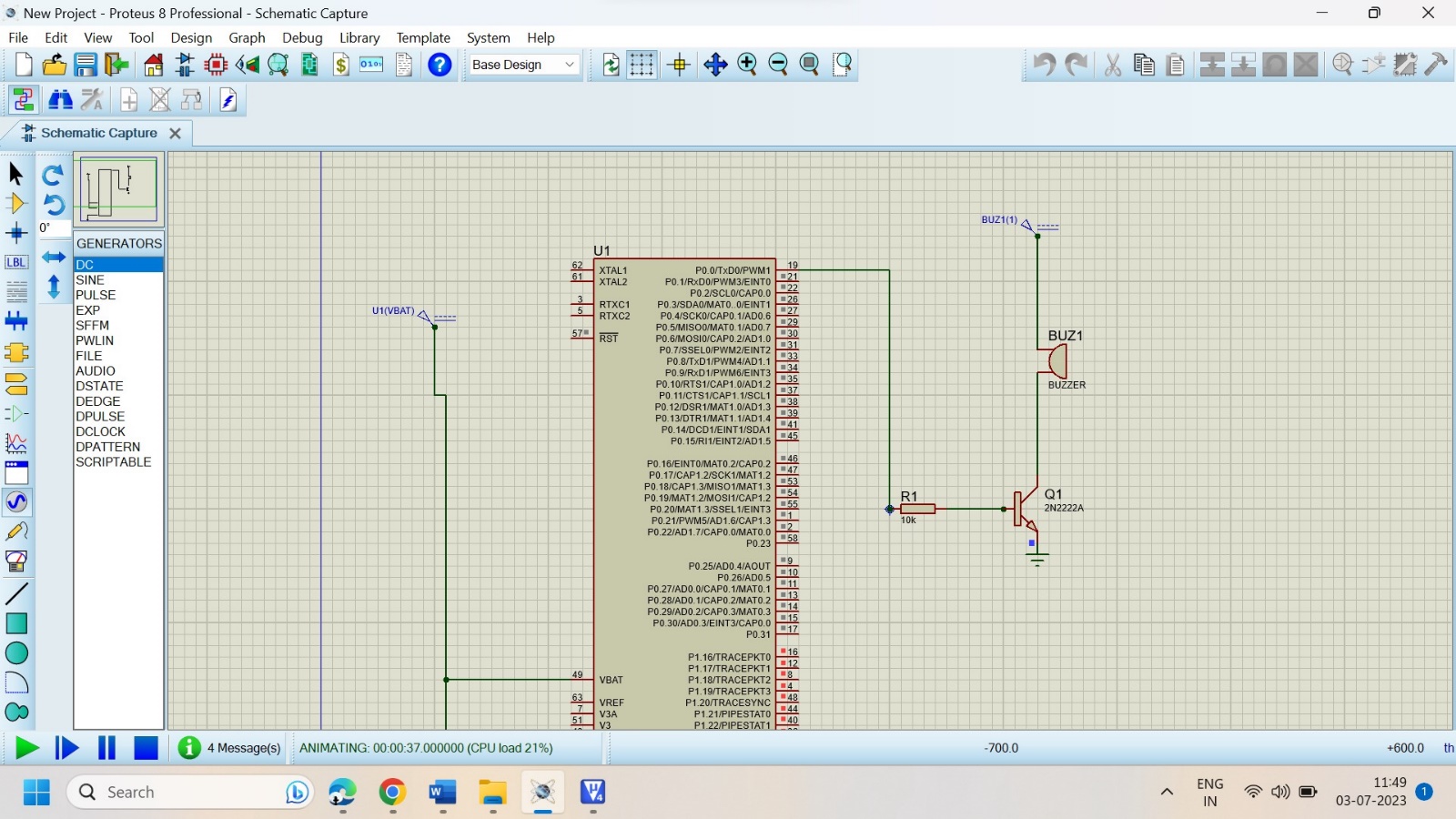
{

int i,j;

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

for(j=0;j<=1000;j++);

}



**Pract 1 : Add of 32 Bit No**

; Addition of 32 bit nos

AREA ADD32,CODE,READONLY ; indicates start of a new data or code section

;define program memory as ROM

ENTRY ; Declare an entry point where the program execution starts

main

ldr r0,=value1 ;load ro with address of value1

ldr r1,[r0] ; load r1 with value1 from memory

ldr r0,=value2 ;load ro with address of value2

ldr r2,[r0] ; load r2 with value2 from memory

adds r3,r1,r2 ;addition of r1 & r2 , result in r3

ldr r0,=result ;load ro with address of result

str r3,[r0] ; store r3 contents to result

stop b stop ;keep executing jump instrn.

value1 DCD 0x11111144 ; Define Constant Word , No's. in Little Endian format

value2 DCD 0x22222255 ; Allocate one or more words (32 bits) of data

AREA Result,DATA,READWRITE ;define data memory as RAM result DCD 0 ; initial sum

END ; Designate the end of a source file

**// Pract 2 : Add of 64 Bit No**

;addition of 64 bit nos

AREA ADD64,CODE,READONLY ; define program memory as ROMENTRY

main

ldr r0,=value1 ;load ro with address of value1

ldr r1,[r0] ; load r1 with value1 lower word from memory

ldr r2,[r0,#4] ; load r2 with value1 higher word from memory

ldr r0,=value2 ;load ro with address of value2

ldr r3,[r0] ; load r3 with value2 lower word from memory

ldr r4,[r0,#] ; load r4 with value2 higher word from memory

adds r6,r1,r3 ;add lower 32 bit data, result in r6

adc r5,r2,r4 ;add upper 32 bit data, result in r5

ldr r0,=result ;load ro with address of result

str r6,[r0] ; store lower 32 bit of result

str r5,[r0,#4] ; store higher 32 bit of result

stop b stop ;keep executing jump instrn.

value1 DCD 0x33333333,0x11111111 ;No's. in Little Endian format

value2 DCD 0x22222222,0x55555555

AREA Result,DATA,READWRITE ;define data memory as RAM

result DCD 0 ; initial sum

END

**Pract 3 : Count Length of String**

;Program to find Length of string

AREA P1,CODE,READONLY ; start of a new code section.

ENTRY ; entry point where the program execution starts

main

ldr r0,=datastring ; Initalize pointer to string of characters

ldr r3,=count ; Initalize pointer to result(Total length)

eor r1,r1,r1 ; clear R1 to count length (up counter)

l1 ldrb r2,[r0],#1 ;Load R2 bytewise ,R0 as pointer with post increment

cmp r2,#0 ;Check end of string for every charc.

beq l3 ;If it is end of string then goto L3

add r1,r1,#1 ;otherwise Increment length counter

bal l1 ; goto next charc. byte

l3 str r1,[r3] ;store length to memory location named as count

l2 bl l2 ;stop

datastring DCB "witsolapu",0 ; Allocate one or more bytes (8 bits) of data

AREA Result,DATA,READWRITE ;start of a new data section.

count DCB 0 ; Allocate one or more bytes (8 bits) of data

END ; Designate the end of a source file