Ex.1 Blinking of LEDed using 8051 Microcontroller using proteus

Aim:- To write an assembly language program to LED blink using 8051.

Software Required:

- proteus software

Program:-

```
    ORG 0000H
UP: SETB P2.0
    ACALL DELAY
    CLR P2.0
    ACALL DELAY
    SJMP UP
DELAY: MOV R4, #35
    H1: MOV R3 #255
    H2: DJNZ R3, H2
        DJNZ R4, H1
        RET
        END
```

Result:

Thus, the program has been successfully verified and executed.

EX.02. LED TOGGLE Using 8051 using proteus

Aim:- To write an assembly language for LED Toggle using 8051 using Keil and proteus.

Software Required:-

proteus 8 software

program:-

```
ORG 0000H
UP: MOV P2, #55H
ACALL DELAY
MOV P2,#0AAH
ACALL DELAY
SJMP UP

DELAY: MOV R4, #10
H1: MOV R3, #255
H2: DJNZ R3, H2
DJNZ R4, H1
RET
END
```

Result:-

Thus, the program has been successfully verified and executed.

EX.03. LED CHASER using 8051 using proteus

Aim:- To write an assembly language program to LED chaser using 8051 using Keil and proteus.

Software Required:

proteus & software

program:-

```
ORG 0000H
UP: MOV P2,#01H
ACALL DELAY
MOV P2,#02H
ACALL DELAY
MOV P2,#04H
ACALL DELAY
MOV P2,#08H
ACALL DELAY
MOV P2,#10H
ACALL DELAY
MOV P2,#20H
ACALL DELAY
MOV P2,#40H
ACALL DELAY
MOV P2,#80H
ACALL DELAY
SJMP UP
DELAY: MOV R4,#255
H1: DJNZ R4,H1
RET
END
```

Result: Thus, the program has been successfully executed.

EX.NO.04. FADE IN FADE OUT OF LED USING 8051 USING PROTEUS

Aim:- To write an assembly language program for fade in fade out of LED using 8051 using keil & proteus.

Software Required:

proteus 8 software

program:-

```
ORG 00H    ; start program at address 00H
MAIN: MOV P2, #00H ; Initialize port 2 (LED OFF)
    ACALL FADE_IN ; call Fade in subroutine
    ACALL FADE-OUT ; call Fade out subroutine
    SJMP MAIN ; Repeat forever
; Subroutine to Fade In the LED
FADE_IN:
    MOV R0, #00H ; start with 100% duty cycle (LED ON)
FADE_OUT_LOOP:
    ACALL PWM ; call the PWM Subroutine with the
current duty cycle
    DEC R0 ; Decrease the duty cycle
    CJNE R0, #00H, FADE_OUT_LOOP ; Repeat until min
    brightness (0% duty cycle)
    RET
; PWM Subroutine
PWM:
    MOV A R0 ; Load duty cycle value
```

```
MOV B, #FFH ; set maximum period
MOV P1, #00H ; LED ON (active-low, so writing 0
    turns on the LED) PWM-ON-LOOP:
DJNZ A, PWM-ON-LOOP ; Delay based on duty cycle
MOV P1, #01H ; LED OFF
PWM-OFF-LOOP:
  DJNZ B, PWM-OFF-LOOP
  RET ; Return from subroutine.
  END.
```

Result:- Thus, the program has been successfully verified and executed.

EX.NO.05. GENERATION OF SQUARE WAVES. PROTEUS

Aim:- To write an assembly language program to generate square wave using 8051.

Software Required:

Proteus & software

Program:-

```
    ORG 0000H
UP: SETB P2.0
    ACALL DELAY
    CLR P2.0
    ACALL DELAY
    SJMP UP
DELAY: MOV R4, #35
  H1: MOV R3, #255
  H2: DJNZ R3, H2
      DJNZ R4, H1
      RET
      END
```

Result:- Thus, the program has been successfully verified using executed.

EX.NO.06. GENERATION OF TRIANGULAR WAVE-PROTEUS

Aim:- To write an Assembly language program to generate triangular wave using 8052.

Software Required:

Proteus 8 software

Program:-

```
ORG 00H
MOV P2.0, #00H
MOV A, #00H
MOV R0, #00H
UPWARD:
INC A
MOV P1,A
ACALL DELAY
CJNE A, #0FFH, UPWARD
DOWNWARD:
DEC A
MOV P1,A
ACALL DELAY
CJNE A, #0FFH, DOWNWARD
SJMP UPWARD
DELAY:
 MOV R1, # 255
DELAY_LOOP1:
 MOV R2, #255
DELAY_LOOP2:
 DJNZ R2, DELAY_LOOP2
 DJNZ R1, DELAY_LOOP1
 RET
END
```

Result:- Thus, the program has been Successfully Verified and executed.

Ex.NO.07. Anticlockwise Rotation of stepper motor using 8051 using proteus

Aim:- To write an assembly language program to rotate the stepper motor in anti-clockwise direction in 8051 using proteus.

Software Required:

Proteus 8 software

Program:-

```
ORG 00H ;start the program to address 0x00
MAIN: MOV P2, #0F0H ; Initialize port 2 as output.
    ACALL COUNTERCLOCKWISE
    ACALL DELAY
    SJMP MAIN
    COUNTERCLOCKWISE:
     MOV A, #08H
     MOV P2, A
     ACALL DELAY
     MOV A, #04H
     ACALL DELAY
     MOV P2, A
     ACALL DELAY
     MOV A; #02H
     MOV P2, A
     ACALL DELAY
     MOV A, #01H
     MOV P2, A
     ACALL DELAY
     RET
    DELAY:
      MOV R1, #0FFH
```

```
DELAY-LOOP1:
    MOV R2, #0FFH
DELAY LOOP2:
    DJNZ R2, DELAY-LOOP2
    DJNZ R1, DELAY-LOOP1
    RET
    END
```

Output:-

The stepper motor is rotating in clockwise direction in steps.

Result:- Thus, the program has been successfully verified and executed.

EX.NO.08. CLOCKWISE ROTATION OF STEPPER MOTOR - PROTEUS

Aim:- To write an assembly language program to rotate the stepper motor in clockwise direction in 8051 using proteus.

Software Required:

Proteus 8 software

Program:

```
    ORG 0000H
UP: MOV P2,#09H
    ACALL DELAY
    MOV P2,#0CH
    ACALL DELAY
    MOV P2,#06H
    ACALL DELAY
    MOV P2,#03H
    ACALL DELAY
    SJMP UP
DELAY: MOV R4,#18
H1: MOV R3,#255
H2: DJNZ R3,H2
    DJNZ R4,H1
    RET
    END
```

Output: The stepper motor is rotating in clockwise direction in steps.

Result:- Thus, the program has been successfully verified and executed.

EX.NO.09. WIGITAL CLOCK ON LCD

Aim:- To write an assembly language program to display digital clock on LCD with using proteus.

Software Required:

proteus 8 software.

Program:-

```
ORG 0000H
MOV R7, #00H
MOV R6, #00H
MOV R5, #00H
ACALL INIT_LCD
MAIN_LOOP:
   ACALL UPDATE_LCD
   ACALL DELAY_1_SEC
   ACALL INCREMENT_TIME
   SJMP MAIN_LOOP
INIT_LCD:
   MOV A, #38H
   ACALL CMD_WRITE
   ACALL DELAY_SHORT
   MOV A, #0CH
   ACALL CMD_WRITE
   ACALL DELAY_SHORT
   MOV A #06H
   ACALL CMD_WRITE
   ACALL DELAY_SHORT
   MOV A, #01H
   ACALL CMD_WRITE
   ACALL DELAY_SHORT
   RET
```

```
UPDATE_LCD:
  MOV A, #80H
  ACALL CMD_WRITE
  MOV A, R7
  ACALL DISPLAY_TWO_DIGIT
  ACALL DISPLAY_COLON
  MOV A, R6
  ACALL DISPLAY_COLON
  ACALL DISPLAY_TWO_DIGIT
  MOV A, R5
  ACALL DISPLAY_TWO_DIGIT
  RET
DISPLAY_TWO_DIGIT:
  MOV B, #10
  DIV AB
  ADD A, #30H
  ACALL DISPLAY_CHAR
  MOV A, B
  ADD A, #30H
  ACALL DISPLAY_CHAR
  RET
DISPLAY_COLON:
  MOV A, #3AH
  ACALL DISPLAY_CHAR
  RET
DISPLAY_CHAR:
  MOV P2, A
  SETB P3.2
  CLR P3.3
  SETB P3.4
  NOP
  CLR P3.4
  ACALL DELAY_SHORT
```

```
RET

CMD-WRITE:
  MOV P2, A
  CLR P3.2
  CLR P3.3
  SETB P3.4
  NOP
  CLR P3.4
  ACALL DELAY_SHORT
  RET

DELAY_SHORT:
    MOV R0, #250
DELAY_SHORT_LOOP:
    DJNZ R0, DELAY_SHORT_LOOP
    RET
  DELAY_1_SEC:
    MOV R3, #50
  DELAY_LOOP:
      MOV R4, #255
  DELAY_LOOP_INNER:
    DJNZ R4, DELAY_LOOP_INNER
    DJNZ R3, DELAY_LOOP
    RET
  END
```

Result:- Thus, the assembly languageprogram to display digital clock on LCD with using proteus was executed.

EX.NO.10. Interfacing of Relay and LED with 8051 using proteus

Aim:- To write an assembly language program to interface relay and LED with 8051 using proteus.

Software required:

proteus 8 software

Program:-

```
ORG 0000H
MOV P1, #00H
MAIN-LOOP:
 SETB P1.0
 ACALL DELAY
 CLR P1.0
 ACALL DELAY
 SJMP MAIN-LOOP
DELAY:
  MOV R1, #255
DELAY1:
  MOV R2, #255
DELAY2:
  DJNZ R2, DELAY2
  DJNZ R1, DELAY1
  RET
END
```

Result:- Thus, the program has been Successfully verified Vomed Also executed...!

EX.NO.11. 7 Segment Display using 8051 - PROTEUS

Aim:- To write an assembly language program for 7 segment display using 8051 using keil and proteus.

Software Required:

proteus 8 software

Program:

```
ORG 000H
UP: MOV P2, #0C0H
ACALL DELAY
MOV P2: #0A4H
ACALL DELAY
MOV P2, #0F9H
ACALL DELAY
MOV P2, #0B0H
ACALL DELAY
MOV P2, #92H
ACALL DELAY
MOV P2, #82H
ACALL DELAY
MOV P2, #0F8H
ACALL DELAY
MOV P2, #80H
ACALL DELAY
MOV P2, #90H
ACALL DELAY
DELAY: MOV R5, #10
H1: MOV R4, #180
H2: MOV R3, #255
H3: DJNZ R3, H3
DJNZ R4, H2
DJNZ R5, H1
RET
END
```

Result:- Thus the program has been successfully verified and executed.

EX.NO.12. TRAFFIC SIGNALS USING 8051 - PROTEUS

Aim:- To write an assembly language program for traffic signals using 8051 using keil and proteus.

Software Required:

Proteus 8 software

Program:

```
ORG 00H
MOV P2, #00H
MOV P3, #00H
MAIN:
  SETB P2.2
  SETB P3.2
  SETB P2.3
  SETB P3.3
  ACALL DELAY2
  MOV P2, #00H
  MOV P3, #00H
  SETB P2.5
  SETB P3.5
  SETB P2.0
  SETB P3.0
  ACALL DELAY1
  SETB P2.1
  SETB P3.5
  SETB P2.0
  CLR P3.0
  ACALL DELAY2
  MOV P2, #00H
  MOV P3, #00H
  SJMP MAIN:
```

```
DELAY 1:
    MOV R0, #256D
D1-LOOP1:
    MOV R1, #255D
D1-LOOP 2:
    MOV R2, #1412D
D2-LOOP3:
    MOV R3, D2-LOOP3
    DJNZ R1, D2-LOOP 2
    DJNZ R0, D2-LOOP 1
    RET
    END
```

Result:- Thus, the program has been successfully verified and executed.

EX.NO.13. Blinking of an LED using Arduino

Aim: Blinking an LED is an introductory Arduino project in which we control an LED using Arduino. LED blinking refers to the process of continously turning an LED and off in repetitive pattern.

Components Required:

- 1X LED
- Bread board
- Arduino UNO
- Jumper wires
- Arduino IDE software

Program:-

```
int LEDpin = 13;
int delayT = 1000;
void setup (){
pinMode (LEDpin, OUTPUT);
}
void loop (){
digitalWrite (LEDpin, HIGH);
delay (delayT);
digitalWrite (LEDpin, LOW);
delay (delayT);
}
```

Result:- Hence, the blinking of LED using Arduino UNO is executed and the Output is verified successfully.

Ex.No.14. Fading of an LED using Arduino-UNO.

Aim:- This Experiment demonstrates the use of the analog write() function in fading an LED Off and On. Analog Wire uses pulse width modulation (PWM), turning a digital pin on and off very quickly with different ratio between on and off, to create a fading effect.

Components Required:-

- 1 X LED
- 220 ohm resistor
- Breadboard
- Arduino UNO
- Jumper wires
- Arduino IDE software

Program:-

```
int led = 9;
int brightness = 0;
int fadeAmount = 5;
void setup(){
  declare
  PinMode(led, OUTPUT);
}
void loop(){
  analogWrite(led, brightness);
  brightness = brightness + fadeAmount;
  if(brightness <= 0 || brightness >= 255){
     fadeAmount = - fadeAmount;
    }
  delay(30);
}
```

Result:- Hence, the fading of LED using Arduino UNO is executed and the output is verified successfully...!

EX.NO.15. Interfacing a water-level sensor-Arduino - UNO.

Aim:- This Experiment is to interface a water-level-sensor with Arduino to measure the water level.

Components Required:

- 1X water
- 220 ohm resistor
- Breadboard
- Arduino UNO
- Jumper wires
- Arduino UNO software, select Arduino UNO board, COM port.

Program:-

```
#define ledpin 6
#define Sensorpin A0
void setup(){
  Serial.begin(9600);
  pinMode(ledpin, OUTPUT);
   digitalwrite (ledpin, 1000);
  }
 Void loop(){
   int sensorValue = analogRead (sensorpin);
   if(Sensor value > 570) {
    int outputvalue = map( sensor value, 570, 800, 0, 255);
    Serial.println(OutputValue);
    analogwrite (ledpin, Output value);
    }
}
```

Result:- Hence, the Interfacing a water level sensor using Arduino UNO is executed and Verified the output Successfully.

EX.NO.16. Interfacing an ultrasonic sensor-Arduino

Aim:- This experiment is to interfacing an ultrasonic sensor using Arduino.

Components required:-

- 1X LED
- 220 ohm resistor
- Breadboard
- Arduino UNO
- Jumper wires
- Arduino IDE software

Program:-

```
#include "Newping.h"
#define TRIGGER_PIN 9
#define ECHO_PIN 10
#define MAX_DISTANCE 100
Newping sonar (TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);
void setup(){
    Serial.begin(9600);
}
void loop(){
    Serial.print("Distance= ");
    Serial.print(sonar.ping_cm());
    Serial.println("cm");
    delay(500);
}
```

Result:- Thus the experiment is executed and verified successfully.

EX.NO.17. MQ-6 gas sensor with Arduino UNO.

Aim:- This Experiment is to MQ-gas sensor using the Arduino UNO.

Components Required:-

- 1X LED
- 220 ohm resistor
- Breadboard
- Arduino UNO
- Jumper wires
- Arduino UNO software.

Program:-

```
int LED = 12;
int Buzzer = 13;
int LPG-SENSOR = 3;
int LPG-detected;
Void setup();
{
  Serial.begin(9600);
  PinMode(LED, OUTPUT);
  PinMode(LED, OUTPUT);
  PinMode(LPG-SENSOR, INPUT);
}
Void LOOP()
{
  LPG-detected = digitalRead(LPG-SENSOR);
  Serial.println(LPG-detected);
  if(LPG-detected == 1)
  {
    Serial.println("LPG-detected...");
    digitalwrite(LED, High);
    digitalwrite(LED, BUZZER);
  }
  else
  { serial println("NO LPG detected.");
    digitalwrite(LED, LOW);
    digitalwrite(BUZZER, LOW);
}
}
```

Result:-

Thus, the code was executed and verified successfully

EX.NO.18. Interfacing a buzzer with Arduino.

Aim: To write a program for interfacing a buzzer using Arduino UNO.

Components required:

- 1X LED
- 220 ohm resistor
- Breadboard
- Arduino UNO
- Jumper wires
- Arduino UNO software

Program:

```
const int buzzer = 9;
void setup(){
   pinMode(buzzer, OUTPUT);

}
void loop(){
    tone(buzzer, 1000);
    delay(1000);
    noTone(buzzer);
     delay(1000);
}
```

Result:- Thus, the program has been verified and executed successfully.

Ex.NO.19. Interfacing LED chaser with Arduino UNO

Aim:- To write a program for Interfacing LED chaser with Arduino UNO.

Components required:-

- 1X LED
- Breadboard
- 220 ohm resistor
- Arduino UNO
- Jumper wires
- Arduino UNO software

Program:-

```
int ledpin;
int pot = A0;
void setup(){
 for (ledpin=2; ledpin<=8; ledpin++){
    pinMode(ledpin, OUTPUT);
   }
}
void loop(){
   for(ledpin=2; ledpin<=8; ledpin++) {
   int value = analog.Read(pot);
   digitalWrite(ledpin, HIGH);
   delay(value);
  }
    for(ledpin=2; ledpin<=8, ledpin++){
       int value = analogRead(pot);
   digitalwrite(ledpin LOW);
   delay(Value);
   }
}
```

Results:- Thus, the program has been executed and verified Successfully....!

EX. NO. 20. RFID using module interfacing Arduino

Aim:- To write the program for RFID module interfacing using Arduino UNO.

components required:-

- 1 X LED
- Breadboard
- 220 ohm resistor
- Arduino UNO
- Jumper wires
- Arduino UNO software

Program:-

```
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN
#define SS_PIN
MFRC522 mfrc522(SS_PIN, RST_PIN);
void setup(){
Serial.begin.(115200);
while(!serial);
ATMEGA32U4
  SPI.begin();
  mfrc522.PCD_init();
  delay(4);
  mfrc522.PCD_init();
}
void loop(){
  if(!mfrc522.PICC_IsNewCardPresent()){
    return;
}
```

Result:- Thus the program has been executed and verified successfully.

Ex. 21. Study of PCB printing using Eagle CAD.

Aim:- To design and print a printed circuit Board layout using Eagle CAD software, incorporating schematic design and board layout for Electronic components.

Components Required:-

1. Hardware
2. software
3. Electronic components

Procedure:-

* Open Eagle CAD software
* Add components
* connect components in schematic view
* perform Electrical Rule check
* Generate Board layout
* Arrange components and create tracks
* optimize Board layout
* Generate Gerber files for printing
* Export PDF for Home printing.

Result: Thus, the Implementation of studying of PCB printing using Eagle CAD.

EX.NO.22. LED FLASHING USING LPC2148 -KIT.

Aim:- To write and Execute a program to blink LED using software delay routine in LPC 2148 kit.

Components Required:

- Keil uVision 5 software
- phillips Flash programmer
- LPC 2148 kit

Program:-

```
#include "lpc2148x.h"
void delay (unsinged int k);
Void main (void)
{
 IODIRO =0XFFFFFFFF;
 PINSELO =0;
 While (1)
 {
  IOSETO = 0X0000ff00;
  delay (100);
  IOCLRO =0X0000ff00;
  delay (1000);
 }
}
Void delay (unsinged int k)
{  unsinged int i,j;
   for( j=0;j<k;j++)
     for (i=0; i<800;i++);
}
```

Result:- Thus, the program to blink LED using software delay routine was written and executed in lpc2148 kit.

EX.NO.23. Accessing an Internal ADC - LPC2148

Aim:- To write and Execute a program for accessing an Internal ADC and display the binary output in LEDS in LPC2148 kit.

Components Required:

- Keil µ vision 5 software
- Philips Flash programmer
- LPC 2148 kit

Program:

```
#include <LPC2148X.H>
#define LEDS 0XFF<<8
#define ADO-1 1<<24
#define CLK_DIV 1<<8
#define PDN 1<<21
#define SOC 1<<24
#define BURST 1<<16
#define DONE 1<<31
void delay(unsigned int k)
{
   unsigned int i,j;
    for(j=0; j<k; j++)
        for(i=0; i<800; i++);
}
void adc_init()
{
unsigned long int ADC_CH;
    ADC_CH=0)1<<1;
```

Output:-

The potentiometer knob was adjusted to generate Analog input and Digital Display is observed.

```
ADDR = SCL | PDN | CLK-DIV | ADC-CH | BURST ;

}
{ unsinged int aval;
  unsinged long int Val;

if( channel == 1 ) Val = ADDDR1;
else if( channel == 2) Val = ADDDR2;
else if( channel == 3) Val = ADDDR3;

  Val = Val >> 6;
  Val = Val & 0x3FF;
  aVal = Val;
  return (aVal);
}
int main (void)
{
 unsinged int tp,1;

}
 while (1);
}
```

Result:- Thus, the c program was written and executed for accessing an internal ADC and display the binary output in LEDS in LPC 2148 kit.

Ex.NO.24. Square waveform generation with 10-bit DAC.

Aim:- To write the embedded for c program to generate a squarewaveform generation using internal 10 bit DAC using LPC2148 ARM Micro controller.

Apparatus required:

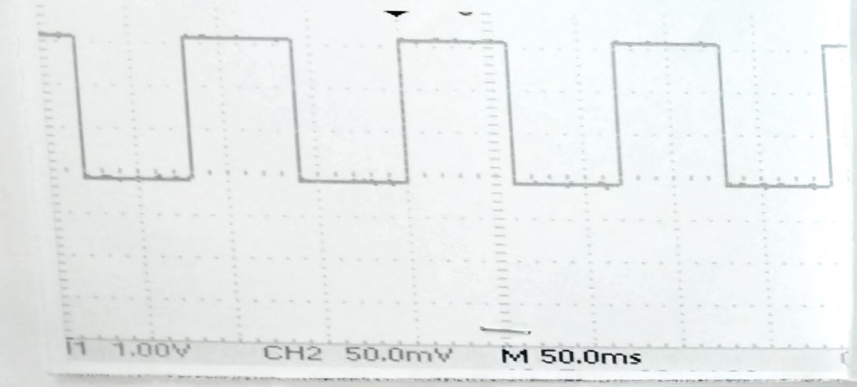
1. LPC2148 ARM microcontroller Development board.
2. Keil µvision version 5.
3. Flash magic.

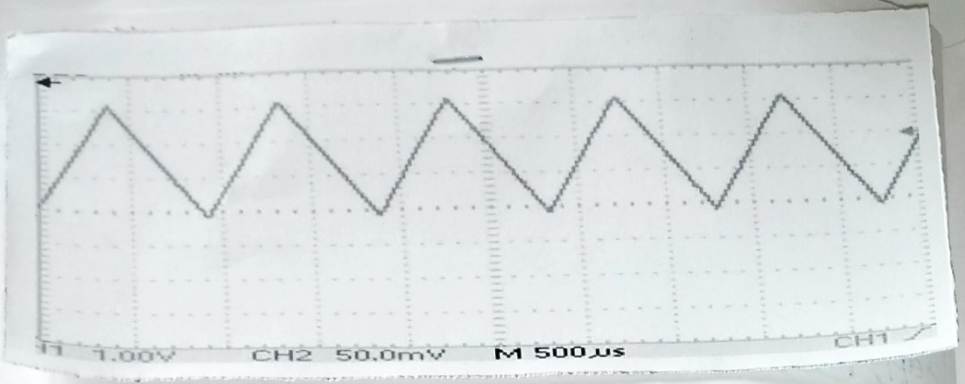
Program:-

```
#include <lpc2148X.h>
#include <stdint.h>
Void delay_ms(unit16_tj)
{
  unit16_tX,i;
 for(i=0; i<j; i++)
  {
    for(X=0; X<6000; X++);
  }
}
int main (void)
{
 Value=0;
 DACR = ((1<<16) | (value<<6));
 delay=ms(100);
  }
}
```

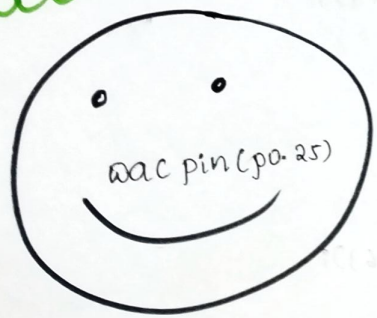
Result:- Thus the embedded c program to generate square waveform is successfully done.

Output:- DAC pin (P0.25).





Output:-



EX. NO. 25. Triangular wave generation using -DAC

Aim:- To write program for Triangular waveform generation using internal DAC of of LPC2148.

Apparatus Required:-

1. LPC 2148 ARM microcontroller development board.
2. Keil µ virjon version 5.
3. Flash magic.

Program:-

```
#include <lpc2148.h>
#include <stdint.h>
void delay_ms (unit16_tj)
{
  unit 16_t,j;
  for(i=0;i<j;i++)
      {
        for(x=0; x<6000; x++);
  }
}
int main(void)
{
 unit_16_t value;
 unit 8_t,i;
 i=0;
}
while (value!=0)
{
 DACR = ((1<<16)|(value<<6));
   value--;
 }
}
```

Result:- Thus the program has been executed and verified successfully.

Ex.No.26. Arithmetic operations using LPC2148.

Aim:- To write program for performing Arithmetic operations using LPC2148 kit.

Hardware Requirements:-

- LPC2148 Development Board
- 16x2 LCD Display
- Keyboard
- Power supply
- USB – UART cable.

Program:-

```
#include <lpc2148x.h>
void init_UART0(void);
void send_UART0(char *str);
int perform_operation(char op, int a, int b);
int main(void){
  char operation;
  int num1=10, num2=5;
  int result;
  int_UART0();
  operation='+';
  result=perform_operation(operation,num1,num2);
  char result_msg[50];
}
void send_UART0(char *str);{
  while(*str){
    while(!(U0LSR&0x20));
      U0THR=*str++;
  }
}
```

Result:- Thus, the program has been executed and verified successfully...!

EX.NO.27. Serial transmission and reception-UART.

Aim: To write and Execute c program for serial transmission and reception using on-chip UART in LPC2148 kit.

Apparatus Required:

* Keil μ virjon software
* philips flash programmer
* LPC 2148 kit

Program:

```
#include <lpc2148X.h>
void UART0_INit(void)
{
PLLDCON=0;
PLLOFEED=0XAA;
PLLOFEED=0X55;
VPBDIV=1;
PINSELL0 1=0X5;
U0FCR=0;
U0LCR=0X83;
U0DLL=0X27;
U0DLM=0;
U0LOCR=3;
}
Void sout (unsinged char dat)
{
 While!(U0LSR & 0X20));
 U0THR = dat;
}
```

OUTPUT:

Data was serially Transmitted

EX.NO.27. Serial transmission and reception-UART.

Aim:- To write and Execute c program for serial transmission and reception using on-chip UART in LPC2148 kit.

Apparatus Required:-

* Keil μ vision software
* philips flash programmer
* LPC 2148 kit

Program:-

```
#include <lpc2148x.h>
void UART0_Init(void)
{
PLL0CON=01;
PLL0FEED=0XAA;
PLL0FEED=0X55;
VPBDIV=1;
PINSEL0|=0X5;
U0FCR=0;
U0LCR=0X83;
U0DLL=0X27;
U0DLM=0;
U0LCR=3;
}
Void sout(unsinged char dat)
{
 while!(U0LSR & 0X20));
 U0THR = dat;
}
while(1);
}
```

Result:- Thus, the program has been executed and verified successfully.

EX.NO.28. Display a number in seven segment LED

Aim:- To write the program for displaying a number in seven segment LED in LPC2148 kit

Apparatus required:-

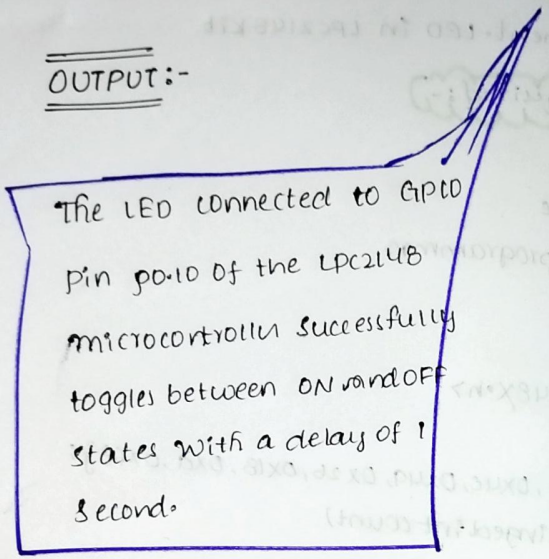
- LPC 2148 kit
- Keil software
- philips flash programmar

Program:-

```
#include <LPC2148X.h>
unsinged char
digit[] = {0x88, 0xeb, 0x4c, 0x49, 0x26, 0x18, 0xa0, 0x68};
 void delay (unsinged int count)
 {
   int j=0 ; i=0;
    for(j=0; j<count;j++)
    {  for(i=0;i<120;i++);
    }
}
int main(void)
{ unsinged char count=0;
  unsinged int i=0;
 while(1);
 {
  count++;
   if(count==16) count=0;
   for(i=0 ;i<800;i++)
   {
     IO0CLR =0x007F8000;
     IO0 SET = (digit[count]<<15);
     delay(200);
   }
}
```

Result:- Thus, the program has been executed and verified successfully.

OUTPUT:-



- Proper setup of the development environment and successfully programming of the LPC2148 using Keil µ vision and flash magic.
- correct functioning of the GPIO pin configuration and execution of the embedded 'c' code on the LPC2148 microcontroller.

EX.NO.29. Introduction to LPC2148 Development kit.

Aim:- To familiarize with the LPC2148 microcontroller development kit and Keil µ vision IDE for embedded programming and to implement a basic program to control an LED using GPIO.

Apparatus Required:-

1. LPC2148 kit
2. USB to UART cable
3. Keil µ vision IDE
4. flash magic software
5. PC
6. LED and resistor.

code:-

```
#include <lpc2148X.h>
void delay (unsigned int time){
   unsigned int i,j;
   for(i=0; i<time; i++){
      for(j=0; j<1000; j++);
   }
}
int main(){
   IO0DIR = 1<<10;
   while(1){
    IO0SET = 1<<10;
    delay(1000);
    IO0CLR = 1<<10;
    delay(1000);
   }
}
```

Result:- The LED connected to GPIO pin P0.10 of the LPC2148 microcontroller toggles ON and OFF continously with a delay.