

Led blink arduino:

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

Fading of an LED using AURDUINO:

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

Interfacing a water-level sensor with Arduino Uno:

```
#define ledPin 6
#define sensorPin A0
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);
}
void loop() {
  int sensorValue = analogRead(sensorPin);
  if (sensorValue > 570) {
    int outputValue = map(sensorValue, 570, 800, 0, 255);
    Serial.println(outputValue);
    analogWrite(ledPin, outputValue);
  }
}
```

Interfacing an ultrasonic sensor with Arduino Uno

```
#include "NewPing.h"
#define TRIGGER_PIN 9
#define ECHO_PIN 10
#define MAX_DISTANCE 400
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);
}
```

MQ GAS SENSOR USING AURDUINO:

```
int RedLED = 12;
int BUZZER = 9;
int MQ6 = 0;
int LPG;
void setup()
{
  Serial.begin(9600);
  pinMode(RedLED, OUTPUT);
  pinMode(BUZZER, OUTPUT);
  pinMode(MQ6, INPUT);
}
void loop()
{
  LPG = digitalRead(MQ6);
  Serial.println(LPG);
  if (LPG == 1)
  {
    Serial.println("LPG Gas in da house ! Bad Air");
    digitalWrite(RedLED, HIGH);
    digitalWrite(BUZZER, HIGH);
  }
  else
  {
    Serial.println("No LPG. Clean Air");
    digitalWrite(RedLED, LOW);
    digitalWrite(BUZZER, LOW);
  }
}
```

Interfacing a buzzer with Arduino Uno

```
const int buzzer = 9;
void setup(){
  pinMode(buzzer, OUTPUT);
}
void loop(){
  tone(buzzer, 1000);
  delay(1000);
  noTone(buzzer);
  delay(1000);
}
```

Interfacing LED chaser with Arduino Uno:

```
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 = analogRead(pot);
    digitalWrite(ledpin,HIGH);
    delay(value);
  }
  for(ledpin=2;ledpin<=8;ledpin++){
    int value = analogRead(pot);
    digitalWrite(ledpin,LOW);
    delay(value);
  }
}
```

RFID

```
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 = analogRead(pot);
    digitalWrite(ledpin,HIGH);
    delay(value);
  }
  for(ledpin=2;ledpin<=8;ledpin++){
    int value = analogRead(pot);
    digitalWrite(ledpin,LOW);
    delay(value);
  }
}
```

LPC2148 KIT

Square waveform generation

```
#include <lpc214x.h> #include <stdint.h>
void delay_ms(uint16_t j)
{
  uint16_t x,i; for(i=0;i<j;i++)
  {
    for(x=0; x<6000; x++); /* loop to generate 1
    milisecond delay with Cclk = 60MHz */
  }
}
int main (void)
{
  uint16_t value; uint8_t i;
  i = 0;
  PINSEL1 = 0x00080000; /* P0.25 as DAC output */
  while(1)
  {
    value = 1023;
    DACR = ( (1<<16) | (value<<6) );
    delay_ms(100); value = 0;
    DACR = ( (1<<16) | (value<<6) );
    delay_ms(100);
  }
}
```

TRIANGULAR WAVE FORM GENERATION

```
#include <lpc214x.h> #include <stdint.h>
void delay_ms(uint16_t j)
{
  uint16_t x,i; for(i=0;i<j;i++)
  {
    for(x=0; x<6000; x++); /* loop to generate 1 milisecond
    delay with Cclk = 60MHz */
  }
}
int main (void)
{
  uint16_t value; uint8_t i;
  i = 0;
  PINSEL1 = 0x00080000; /* P0.25 as DAC output */
  while(1)
  {
    value = 0;
    while ( value != 1023 )
    {
      DACR = ( (1<<16) | (value<<6) );
      value++;
    }
    while ( value != 0 )
    {
      DACR = ( (1<<16) | (value<<6) );
      value--;
    }
  }
}
```

Arithmetic operations using LPC 2148 kiT

```
#include <lpc214x.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;
    init_UART0();
    operation = '+';
    result = perform_operation(operation, num1, num2);
    char result_msg[50];
    sprintf(result_msg, "Result of %d %c %d = %d\r\n", num1,
operation, num2, result);
    send_UART0(result_msg);
    while (1);
}
int perform_operation(char op, int a, int b) {
    switch (op) {
        case '+': return a + b;
        case '-': return a - b;
        case '*': return a * b;
        case '/': return (b != 0) ? (a / b) : 0;
        default: return 0;
    }
}
void init_UART0(void) {
    PINSEL0 = 0x00000005;
    UOLCR = 0x83;
    UODLM = 0;
    UODLL = 97;
    UOLCR = 0x03;
}
void send_UART0(char *str) {
    while (*str) {
        while (!(UOLSR & 0x20));
        UOTHR = *str++;
    }
}
```

LED BLINKing using LPC 2148 kit

```
#include "lpc214x.h"
void delay (unsigned int k);
void main(void)
{
    IODIR0 = 0xFFFFFFFF;
    PINSEL0 = 0;
    while(1)
    {
        IOSET0 = 0x0000ff00;
        delay(1000);
        IOCLR0 = 0x0000ff00;
        delay(1000);
    }
}
void delay(unsigned int k)
{
    unsigned int i,j;
    for (j=0; j<k; j++)
        for(i = 0; i<=800; i++);
}
```

serial transmission LPC4128 KIT

```
#include <lpc214x.h>
void UART0_Init(void)
{
    PLLOCON = 0;
    PLL0FEED=0xAA;
    PLL0FEED=0x55;
    VPBDIV = 1;
    PINSEL0 |= 0x5;
    UOFCR = 0;
    UOLCR = 0x83;
    UODLL = 0x27;
    UODLM = 0;
    UOLCR = 3;
    void sout(unsigned char dat1){
        while(!(UOLSR & 0x20));
        UOTHR = dat1; }
int main (void)
{ int dat;
    UART0_Init();
    do
    {
        if(UOLSR & 1) {
            dat = UORBR;
            sout(dat);
        }
    }while(1);
}
```

accessing an internal ADC:

```
#include <LPC214X.H>
#define LEDS 0xFF<<8 //LED => P0.8 to P0.15
#define AD0_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; //Channel AD0.1
        AD0CR = SOC | PDN | CLK_DIV | ADC_CH | BURST ;
}
{
    unsigned int aval;
    unsigned long int val;
    if (channel == 1) val = AD0DR1;
    else if (channel == 2) val = AD0DR2;
    else if (channel == 3) val = AD0DR3;
    val = val >> 6;
    val = val & 0x3FF;
    aval = val;
    return (aval);
}
int main(void)
{
    unsigned int tp1;
    IODIRO = LEDS;
    PINSELO = 0;
    PINSEL1 = 0 | AD0_1;
    adc_init();
    do
    { tp1 = adc_read(1); // Channel AD0 0.1
        tp1 = tp1 >> 2;
        IOSET0 = LEDS;
        IOCLR0 = tp1 << 8;
        delay(1000);
    }while(1);
}
```

BLINKING OF LED- PROTEUS

```
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
```

LED TOGGLE- PROTEUS

```
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
```

```
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
```

GENERATION OF
SQUARE WAVE

ANTICLOCK-STEPPER MOTOR: PROTEUS

```
ORG 00H
MOV P2, #0F0H
    ACALL DELAY
    SJMP MAIN
    MOV A, #08H
    MOV P2, A
    ACALL DELAY
    MOV A, #04H
    MOV P2, A
    ACALL DELAY
    MOV A, #02H
    MOV P2, A
    ACALL DELAY
    MOV A, #01H
    MOV P2, A
    ACALL DELAY
    RET
    MOV R1, #0FFH
DELAY_LOOP1:
    MOV R2, #0FFH
DELAY_LOOP2:
    DJNZ R2,
DELAY_LOOP2
    DJNZ R1,
DELAY_LOOP1
    RET
END
```

**CLOCK-STEPPER
MOTOR: PROTEUS**

```
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
```

LED CHASER-PROTEUS

```
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
```

TRIANGULAR WAVE - PROTEUS

```
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, #00H, DOWNWARD
    SJMP UPWARD
DELAY:
    MOV R1, #255
DELAY_LOOP1:
    MOV R2, #255
DELAY_LOOP2:
    DJNZ R2, DELAY_LOOP2
    DJNZ R1, DELAY_LOOP1
    RET
END
```

7-SEGMENT DISPLAY

```
ORG 000H
UP:MOV
P2,#0C0H
ACALL DELAY
MOV P2,#0F9H
ACALL DELAY
MOV P2,#0A4H
ACALL DELAY
MOV P2,#0B0H
ACALL DELAY
MOV P2,#99H
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
```

INTERFACING OF RELAY- PROTEUS

```
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
```

FADE IN FADE OUT-PROTEUS

```
ORG 0000H
    SJMP START
DELAY:
    MOV R2, DPL
    MOV R3, DPH
DELAY_OUTER:
    MOV R1, #127
DELAY_INNER:
    NOP
    NOP
    DJNZ R1, DELAY_INNER
    DJNZ R2, DELAY_OUTER
    RET
START:
    MOV P1, #0FFH
MAIN_LOOP:
    MOV DPL, #100
    MOV DPH, #0
    ACALL DELAY
    CLR P1.0
    ACALL DELAY
    SETB P1.0
    SJMP MAIN_LOOP
END
```

--	--

7 SEGMENT DISPLAY LPC2148 :

```
#include <LPC214x.h>
unsigned char
dig[]={0x88,0xeb,0x4c,0x49,0x2b,0x19,0x18,0xcb,0x8,0x9,0xa,0x38,0x9c,0x68};
void delay(unsigned int count)
{
    int j=0,i=0;
    for(j=0;j<count;j++)
    {
        for(i=0;i<120;i++);
    }
}
int main(void)
{
    unsigned char count=0;
    unsigned int i=0;
    IO0DIR|=(1<<11); //Set Digit control lines as Outputs
    IO0SET|=(1<<11);
    IO0DIR|=0x007F8000;
    while(1)
    {
        count++;
        if(count==16)count=0;
        for(i=0;i<800;i++)//change to inc/dec speed of count
        {
            IO0CLR=0x007F8000;
            IO0SET=(dig[count]<<15);
            delay(200);
        }
    }
}
```