**Pr1**

**LED BLINK**

|  |
| --- |
| #include <reg52.h>  sbit LED = P2 ^ 0;  void Delay(void);  void main(void)  {      while (1)      {          LED = 0;          Delay();          LED = 1;          Delay();      }  }  void Delay(void)  {      int j;      int i;      for (i = 0; i < 10; i++)      {          for (j = 0; j < 10000; j++)              ;          {          }      }  } |

**LED FLASHING**

|  |
| --- |
| #include <reg51.h>  // delay function declaration  void delay(void);  void main(void)  {      // an infinite loop      while (1)      {          // Turn ON all LED's connected to Port1          P1 = 0xFF;          delay();          // Turn OFF all LED's connected to Port1          P1 = 0x00;          delay();      }  }  // delay function definition  void delay(void)  {      int i, j;      for (i = 0; i < 0xff; i++)          for (j = 0; j < 0xff; j++)              ;  } |

**BINARY**

|  |
| --- |
| #include <reg51.h>  void delay()  {      unsigned int i, j;      for (i = 0; i < 500; i++)          for (j = 0; j < 1000; j++)              ;  }  void main()  {      while (1)      {          delay();          P1 = 0x01;          delay();          P1 = 0x02;          delay();          P1 = 0x03;          delay();          P1 = 0x04;          delay();          P1 = 0x05;          delay();          P1 = 0x06;          delay();          P1 = 0x07;          delay();          P1 = 0x08;          delay();          P1 = 0x09;          delay();          P1 = 0x0A;          delay();          P1 = 0x0B;          delay();          P1 = 0x0C;          delay();          P1 = 0x0D;          delay();          P1 = 0x0E;          delay();          P1 = 0x0F;          delay();      }  } |

**Pr2**

**Square wave**

|  |
| --- |
| #include <reg51.h>  void Delay(unsigned int);  void main(void)  {      while (1)      {          P1 = 0x55;          Delay(250);          P1 = 0xAA;          Delay(500);      }  }  void MSDelay(unsigned int itime)  {      unsigned int i, j;      for (i = 0; i < itime; i++)          for (j = 0; j < 1275; j++)              ;  } |

**Ramp wave**

|  |
| --- |
| #include <reg51.h>  void main()  {      unsigned int i;      while (1)      {          for (i = 0; i < 256; i++)          {              P2 = i;          }      }  } |

**Triangle wave**

|  |
| --- |
| #include <reg51.h>  void delay(unsigned int);  void main(void)  {      while (1) // infinite loop      {          unsigned int x;          for (;;) // repeat forever          {              for (x = 0; x < 250; x++)              {                  P2 = x;                  delay(100);              }              for (x = 250; x > 0; x--)              {                  P2 = x;                  delay(100);              }          }      }  }  void delay(unsigned int time)  {      unsigned int i, j;      for (i = 0; i < time; i++)          for (j = 0; j < 1275; j++)              ;  } |

**PR3**

**Stepper motor**

|  |
| --- |
| #include <reg51.h>  void delay(void);  void main(void)  {      while (1)      {          P1 = 0xCC;          delay();          P = 0x66;          delay();          P1 = 0x33;          delay();          P1 = 0x99;          delay();      }  }  void delay()  {      TMOD = 0x01;      TL0 = 0xFD;      TH0 = 0x4B;      TR0 = 1;      while(TF0 == 0);      TR0 = 0;      TF0 = 0;  } |

**PR4**

|  |
| --- |
| #include <p18f4550.h>  #pragma config PLLDIV = 5  #pragma config CPUDIV = OSC1\_PLL2  #pragma config USBDIV = 2  #pragma config FOSC = HSPLL\_HS  #pragma config VREGEN = ON  #pragma config WDT = OFF  #pragma config PBADEN = OFF  #pragma config LVP = OFF  #include "vector\_relocate.h"  void MsDelay(unsigned int time)  {      unsigned int i, j;      for (i = 0; i < time; i++)          for (j = 0; j < 710; j++);  }  #define lrbit PORTBbits.RB1  #define rlbit PORTBbits.RB2  #define buzzer PORTCbits.RC2  #define relay PORTCbits.RC1  #define relay1 PORTCbits.RC0  void main()  {      unsigned char val = 0;      unsigned int k;      INTCON2bits.RBPU = 0;      ADCON1 = 0x0F;      TRISBbits.TRISB4 = 1;      TRISBbits.TRISB5 = 1;      TRISCbits.TRISC1 = 0;      TRISCbits.TRISC2 = 0;      TRISD = 0x00;      PORTD = 0x00;      buzzer = 0;      relay = 0;      relay1 = 0;      while (!(lrbit)) {          buzzer = 1;          relay = 1;          relay1 = 1;          PORTD = PORTD >> 1;          if (PORTD == 0x00)              PORTD == 0x80;          MsDelay(250);      }      while (!(rlbit)) {          buzzer = 0;          relay = 0;          relay1 = 0;          PORTD = PORTD << 1;          if (PORTD == 0x00)              PORTD = 0x01;          MsDelay(250);      } |

**Pr5**

|  |
| --- |
| / This program demonstrates the interfacing of LCD to PIC18F4550 microcontroller /  #include <p18f4550.h>  #include "vector\_relocate.h" //Vector Remapping for USB HID Bootloader  #pragma config PLLDIV = 5    // (20 MHz crystal on PICDEM FS USB board)  #pragma config CPUDIV = OSC1\_PLL2  #pragma config USBDIV = 2 // Clock source from 96MHz PLL/2  #pragma config FOSC = HSPLL\_HS  #pragma config VREGEN = ON // USB Voltage Regulator  #pragma config WDT = OFF  #pragma config PBADEN = OFF  #pragma config LVP = OFF  // LCD data pins connected to PORTD and control pins connected to PORTE  #define LCD\_DATA PORTD   // LCD data port  #define ctrl PORTE       // LCD signal port  #define en PORTEbits.RE2 // enable signal  #define rw PORTEbits.RE1 // read/write signal  #define rs PORTEbits.RE0 // register select signal  #define BUSY PORTDbits.RD7      // LCD function definitions      void LCD\_Busy(void);  void LCD\_cmd(unsigned char cmd);  void init\_LCD(void);  void LCD\_write(unsigned char data);  void LCD\_write\_string(static char \*str);  // Function to generate delay  void myMsDelay(unsigned int time)  {      unsigned int i, j;      for (i = 0; i < time; i++)          for (j = 0; j < 710; j++)              ;/Calibrated for a 1 ms delay in MPLAB/  }  // Function to configure the ports connected to LCD and call functions to initalise, write data and command to LCD  void display\_string\_LCD(static char \*pstring1, static char \*pstring2)  {      ADCON1 = 0x0F; // Configuring the PORTE pins as digital I/O      TRISD = 0x00;  // Configuring PORTD as output      TRISE = 0x00;  // Configuring PORTE as output      init\_LCD();    // initialization of LCD      myMsDelay(50); // delay of 50 mili seconds      LCD\_write\_string(pstring1);      myMsDelay(15);      LCD\_cmd(0xC0); // initiate cursor to second line      LCD\_write\_string(pstring2);      myMsDelay(1000);      return;  }  // Function to initialise the LCD  void init\_LCD(void)  {      LCD\_cmd(0x38); // initialization of 16X2 LCD in 8bit mode      myMsDelay(15);      LCD\_cmd(0x01); // clear LCD      myMsDelay(15);      LCD\_cmd(0x0C); // cursor off      myMsDelay(15);      LCD\_cmd(0x80); // ---8 go to first line and --0 is for 0th position      myMsDelay(15);      // ---8 go to first line and --0 is for 0th position      return;  }  // Function to pass command to the LCD  void LCD\_cmd(unsigned char cmd)  {      LCD\_DATA = cmd;      rs = 0;      rw = 0;      en = 1;      myMsDelay(15);      en = 0;      myMsDelay(15);      return;  }  // Function to write data to the LCD  void LCD\_write(unsigned char data)  {      LCD\_DATA = data;      rs = 1;      rw = 0;      en = 1;      myMsDelay(15);      en = 0;      myMsDelay(15);      return;  }  // Function to split the string into individual characters and call the LCD\_write function  void LCD\_write\_string(static char \*str) // store address value of the string in pointer \*str  {      int i = 0;      while (str[i] != 0)      {          LCD\_write(str[i]); // sending data on LCD byte by byte          myMsDelay(15);          i++;      }      return;  }  void main(void)  {      char var1[] = "RMDSSOE Warje";      char var2[] = "TE E&TC";      display\_string\_LCD(var1, var2);      while (1)          ;  } |

**Pr6**

|  |
| --- |
| #include <p18f4550.h>  #pragma config PLIDIV = 5  #pragma config CPUDIV = OSC1\_PLL2  #pragma config UNDDIV = 2  #pragma config FOSC = HSPLL\_HS  #pragma config VREGEN = ON  #pragma config WDT = OFF  #pragma config PBADEN = OFF  #pragma config LVP = OFF  #include <stdlib.h>  void myDelay(unsigned int time);  #pragma code  #pragma interrupt timer\_isr  #void timer\_isr(void)  {      TMR0H = 0X48;      TMR0L = 0XE5;      PORTB = ~PORTB;      INTCONbits.TMR0IF = 0;  }  TMR0H = 0X48;  TMR0L = 0XE5;  PORTB = ~PORTB;  INTCONbits.TMR0IF = 0;  }  void main()  {      ADCON1 = 0x0f;      TRISB = 0;      PORTB = 0x0f;      T0CON = 0x07;      TMRH = 0x48;      TMR0L = 0xE5;      INTCONbits.TMR0IF = 0;      INTCONbits.TMROIE = 1;      TCONbits.TMR0ON = 1;      INTCONbits.GIE = 1;      while (1)      {          while (!INTCONbits.TME0IF);          TMR0H = 0X48;          TMR0L = OXE5;          PORTB = ~PORTB;          INTCONbits.TMR0IF = 0;      }  } |

**Pr7**

|  |
| --- |
| #include <p18f4550.h>  #pragma config PLLDIV = 5 // (20 MHz crystal on PICDEM FS USB board)  #pragma config CPUDIV = OSC1\_PLL2  #pragma config USBDIV = 2 // Clock source from 96MHz PLL/2  #pragma config FOSC = HSPLL\_HS  #pragma config VREGEN = ON // USB Voltage Regulator  #pragma config WDT = OFF  #pragma config PBADEN = OFF  #pragma config LVP = OFF  #include "vector\_relocate.h"  #define LCD\_DATA PORTD   // LCD data port  #define en PORTEbits.RE2 // enable signal  #define rw PORTEbits.RE1 // read/write signal  #define rs PORTEbits.RE0 // register select signal  void ADC\_Init(void);                     // Function to initialize the ADC  unsigned int Get\_ADC\_Result(void);  void Start\_Conversion(void);             // Function to Start of Conversion  void msdelay(unsigned int time);         // Function to generate delay  void init\_LCD(void);                     // Function to initialise the LCD  void LCD\_command(unsigned char cmd);  void LCD\_data(unsigned char data);  void LCD\_write\_string(static char \*str);  void main()  {      char msg1[] = "RMDSSOE,Warje";      char msg2[] = "ADC O/P:";      unsigned char i, Thousands, Hundreds, Tens, Ones;      unsigned int adc\_val;      ADCON1 = 0x0F; // Configuring the PORT pins as digital I/O      TRISD = 0x00;  // Configuring PORTD as output      TRISE = 0x00;  // Configuring PORTE as output      ADC\_Init();             // Init ADC peripheral      init\_LCD();             // Init LCD Module      LCD\_write\_string(msg1); // Display Welcome Message      LCD\_command(0xC0);      // Goto second line, 0th place of LCD      LCD\_write\_string(msg2); // Display Message "ADC O/P"      while (1)      {          Start\_Conversion();         // Trigger conversion          adc\_val = Get\_ADC\_Result(); // Get the ADC output by polling GO bit          LCD\_command(0xC8); // Goto 9th place on second line of LCD          i = adc\_val / 1000;   // Get the thousands place          Thousands = i + 0x30; // Convert it to ASCII          LCD\_data(Thousands);  // Display thousands place          i = (adc\_val % 1000) / 100; // Get the Hundreds place          Hundreds = i + 0x30;        // Convert it to ASCII          LCD\_data(Hundreds);         // Display Hundreds place          i = ((adc\_val % 1000) % 100) / 10; // Get the Tens place          Tens = i + 0x30;                   // Convert it to ASCII          LCD\_data(Tens);                    // Display Tens place          i = adc\_val % 10;   // Get the Ones place          Ones = i + 30;      // Convert it to ASCII          LCD\_data(i + 0x30); // Display Ones place          msdelay(300); // Delay between conversions. It is a library function,refer delays.h file in MCC18 installation directory      }  }  // Function Definitions  void ADC\_Init()  {      ADCON0 = 0b00000100; // A/D Module is OFF and Channel 1 is selected      ADCON1 = 0b00001101; // Reference as VDD & VSS, AN1 set as analog pins      ADCON2 = 0b10001110; // Result is right Justified                           // Acquisition Time 2TAD                           // ADC Clk FOSC/64      ADCON0bits.ADON = 1; // Turn ON ADC module  }  void Start\_Conversion()  {      ADCON0bits.GO = 1;  }  // If you do not wish to use adc conversion interrupt you can use this  // to do conversion manually. It assumes conversion format is right adjusted  unsigned int Get\_ADC\_Result()  {      unsigned int ADC\_Result = 0;      while (ADCON0bits.GO)          ;      ADC\_Result = ADRESL;      ADC\_Result |= ((unsigned int)ADRESH) << 8;      return ADC\_Result;  }  void msdelay(unsigned int time) // Function to generate delay  {      unsigned int i, j;      for (i = 0; i < time; i++)          for (j = 0; j < 710; j++)              ; // Calibrated for a 1 ms delay in MPLAB  }  void init\_LCD(void) // Function to initialise the LCD  {      LCD\_command(0x38); // initialization of 16X2 LCD in 8bit mode      msdelay(15);      LCD\_command(0x01); // clear LCD      msdelay(15);      LCD\_command(0x0C); // cursor off      msdelay(15);      LCD\_command(0x80); // go to first line and 0th position      msdelay(15);  }  void LCD\_command(unsigned char cmd) // Function to pass command to the LCD  {      LCD\_DATA = cmd; // Send data on LCD data bus      rs = 0;         // RS = 0 since command to LCD      rw = 0;         // RW = 0 since writing to LCD      en = 1;         // Generate High to low pulse on EN      msdelay(15);      en = 0;  }  void LCD\_data(unsigned char data) // Function to write data to the LCD  {      LCD\_DATA = data; // Send data on LCD data bus      rs = 1;          // RS = 1 since data to LCD      rw = 0;          // RW = 0 since writing to LCD      en = 1;          // Generate High to low pulse on EN      msdelay(15);      en = 0;  }  // Function to write string to LCD  void LCD\_write\_string(static char \*str)  {      int i = 0;      while (str[i] != 0)      {          LCD\_data(str[i]); // sending data on LCD byte by byte          msdelay(15);          i++;      }  } |

**PR8**

|  |
| --- |
| #include <p18f4550.h>  #include <stdlib.h>  #pragma config PLLDIV = 5 // (20 MHz crystal on PICDEM FS USB board)  #pragma config CPUDIV = OSC1\_PLL2  #pragma config USBDIV = 2 // Clock source from 96MHz PLL/2  #pragma config FOSC = HSPLL\_HS  #pragma config VREGEN = ON // USB Voltage Regulator  #pragma config WDT = OFF  #pragma config PBADEN = OFF  #pragma config LVP = OFF  void myMsDelay(unsigned int time) // Definition of delay subroutine  {      unsigned int i, j;      for (i = 0; i < time; i++) // Loop for itime          for (j = 0; j < 710; j++)              ; // Calibrated for a 1 ms delay in MPLAB  }  void main()  {      TRISCbits.TRISC2 = 0; // Set PORTC, RC2 as output (CCP1)      TRISCbits.TRISC6 = 0; // Set PORTC, RC6 as output (DCM IN3)      TRISCbits.TRISC7 = 0; // Set PORTC, RC6 as output (DCM IN4)      PR2 = 0xBA;           // set PWM Frequency 4KHz      CCP1CON = 0x0C;       // Configure CCP1CON as PWM mode.      T2CON = 0x07;         // Start timer 2 with prescaler 1:16      PORTCbits.RC6 = 1;    // Turn ON the Motor      PORTCbits.RC7 = 0;      while (1) // Endless Loop      {          // ----------Duty Cycle 80%-----------          CCP1CONbits.DC1B0 = 0;          CCP1CONbits.DC1B1 = 0;          CCPR1L = 0x96;          myMsDelay(2000);          // -----------------------------------          // ----------Duty Cycle 60%-----------          CCP1CONbits.DC1B0 = 0;          CCP1CONbits.DC1B1 = 1;          CCPR1L = 0x70;          myMsDelay(2000);          // -----------------------------------          // ----------Duty Cycle 40%-----------          CCP1CONbits.DC1B0 = 0;          CCP1CONbits.DC1B1 = 0;          CCPR1L = 0x4B;          myMsDelay(2000);          // -----------------------------------          // ----------Duty Cycle 20%-----------          CCP1CONbits.DC1B0 = 0;          CCP1CONbits.DC1B1 = 1;          CCPR1L = 0x25;          myMsDelay(2000);          // -----------------------------------      }  } |