Experiment 8: Serial Communication & ADC / DAC Implementation in ViARM 2378 Development Board (through C - Interface)

Santhosh S P ee21b119

Problem 1: Serial communication

Problem statement

Write a program (in C) to display the ASCII code in LEDs, corresponding to the key pressed in the key board of the PC interfaced to ViARM-2378. Use the RS232 serial cable interfaced to the Vi Microsystem's ViARM 2378 development board.

Code

```
#include "LPC23xx.h"

/*
Routine to set processor and pheripheral clock
*/

void  TargetResetInit(void)
{
      // 72 Mhz Frequency
      if ((PLLSTAT & 0x02000000) > 0)
      {
          //If the PLL is already running
          PLLCON  &= ~0x02;        //Disconnect the PLL
          PLLFEED  =  0xAA;       //PLL register update sequence, 0xAA,0x55
          PLLFEED  =  0x55;
      }
      PLLCON   &= ~0x01;         //Disable the PLL
      PLLFEED   =  0xAA;         //PLL register update sequence, 0xAA, 0x55
      PLLFEED   =  0x55;
      SCS      &= ~0x10;       //OSCRANGE = 0, Main OSC is between 1 and 20 Mhz
      SCS      |=  0x20;       // OSCEN = 1, Enable the main oscillator
```

```
      while ((SCS &  0x40) == 0);
      CLKSRCSEL = 0x01;        //Select main OSC, 12MHz, as the PLL clock source
      //Configure the PLL multiplier and divider
      PLLCFG    = (24 << 0) | (1 << 16);
      PLLFEED   = 0xAA;       //PLL register update sequence, 0xAA, 0x55
      PLLFEED   = 0x55;
      PLLCON   |= 0x01;       //Enable the PLL
      PLLFEED   = 0xAA;       //PLL register update sequence, 0xAA, 0x55
      PLLFEED   = 0x55;
      CCLKCFG   = 3;          //Configure the ARM Core Processor clock divider
      USBCLKCFG = 5;          //Configure the USB clock divider


      while ((PLLSTAT & 0x04000000) == 0);
      //Set peripheral clocks to be half of the  main clock
      PCLKSEL0  = 0xAAAAAAAA;
      PCLKSEL1  = 0x22AAA8AA;
      PLLCON   |= 0x02;        //Connect the PLL. It's the active clock source
      PLLFEED   = 0xAA;        //PLL register update sequence, 0xAA, 0x55
      PLLFEED   = 0x55;

      while ((PLLSTAT & 0x02000000) == 0);
      PCLKSEL0 = 0x55555555;       //PCLK is the same as CCLK
      PCLKSEL1 = 0x55555555;
}


// serial reception routine


int serial_rx(void)
{
    while (!(U0LSR & 0x01));
    return (U0RBR);
}


//serial transmission routine


void serial_tx(int ch)
{
    //while ((U0LSR & 0x20)!=0x20);
    while ((U0LSR & 0x20)==0);
    U0THR = ch;
```

```
}


// serial transmission routine for a string of characters


void string_tx(char *a)
{
    while(*a!='\0')
    {
        while((U0LSR&0X20)!=0X20);
        U0THR=*a;
        a++;
    }
}


//main routine


int  main ()
{
    unsigned int Fdiv;
    char value;
    TargetResetInit();

    //uart1 initialization


    PINSEL0 = 0x00000050;

    U0LCR = 0x83;
    // 8 bits, no Parity, 1 Stop bit

    Fdiv = ( 72000000 / 16 ) / 19200 ;  //baud rate

    U0DLM = Fdiv / 256;
    U0DLL = Fdiv % 256;
    U0LCR = 0x03;              // DLAB = 0

    FIO3DIR=0xFF;

    while(1)
    {
        value=serial_rx();
        serial_tx(value);
```

```
        FIO3PIN = value;
    }
    return 0;
}
```

Output

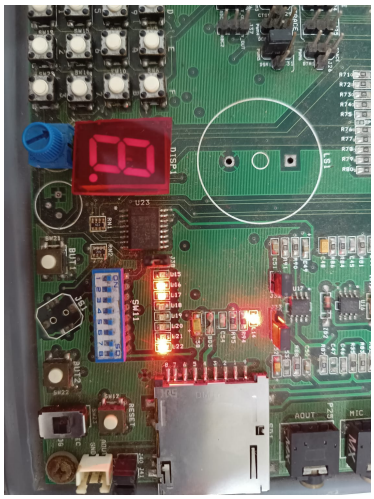

Figure 1: LEDs displaying the ASCII code of the character 'a', (97_{10}).

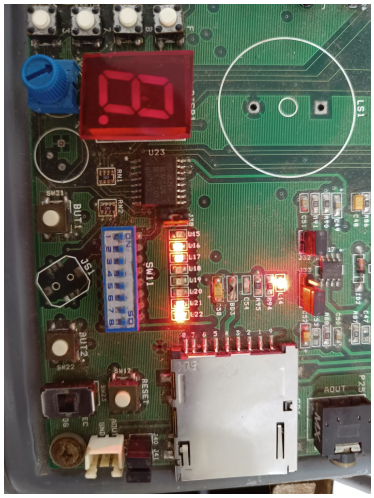

Figure 2: LEDs displaying the ASCII code of the character 'c', (99_{10}).

Problem 2: ADC

Problem statement

Given a real-time (analog) signal from a sensor, convert it into a digital signal (Implement an ADC). Decrease the step size? Do you see any change in the bits used to represent the whole range? What is the quantization error?

Code

```
#include "LPC23xx.h"

void  TargetResetInit(void)
{
      // 72 Mhz Frequency
      if ((PLLSTAT & 0x02000000) > 0)
      {
          //If the PLL is already running
          PLLCON  &= ~0x02;      //Disconnect the PLL
          PLLFEED  =  0xAA;      //PLL register update sequence, 0xAA, 0x55
          PLLFEED  =  0x55;
      }

      PLLCON   &= ~0x01;     //Disable the PLL
      PLLFEED   =  0xAA;     //PLL register update sequence, 0xAA, 0x55
      PLLFEED   =  0x55;

      SCS      &= ~0x10;     //OSCRANGE = 0, Main OSC is between 1 and 20 Mhz
      SCS      |=  0x20;     //OSCEN = 1, Enable the main oscillator

      while ((SCS &  0x40) == 0);
      CLKSRCSEL = 0x01;
      //Select main OSC, 12MHz, as the PLL clock source

      //Configure the PLL multiplier and divider
      PLLCFG    = (24 << 0) | (1 << 16);

      PLLFEED   = 0xAA;     //PLL register update sequence, 0xAA, 0x55
      PLLFEED   = 0x55;
      PLLCON   |= 0x01;     //Enable the PLL

      PLLFEED   = 0xAA;     //PLL register update sequence, 0xAA, 0x55
      PLLFEED   = 0x55;

      CCLKCFG   = 3;        //Configure the ARM Core Processor clock divider
      USBCLKCFG = 5;        //Configure the USB clock divider
```

```
      while ((PLLSTAT & 0x04000000) == 0);
      PCLKSEL0  = 0xAAAAAAAA;    // Set peripheral clocks to be half of main clock
      PCLKSEL1  = 0x22AAA8AA;
      PLLCON   |= 0x02;    //Connect the PLL. It's now the active clock
      PLLFEED   = 0xAA;               //PLL register update sequence, 0xAA, 0x55
      PLLFEED   = 0x55;

      while ((PLLSTAT & 0x02000000) == 0);
      PCLKSEL0 = 0x55555555;          //PCLK is the same as CCLK
      PCLKSEL1 = 0x55555555;
}



//serial transmission routine


void serial_tx(int ch)
{
    while ((U0LSR & 0x20)!=0x20);
    U0THR = ch;
}


//Routine for converting hex value to ASCII value


int atoh(int ch)
{
    if(ch<=0x09)
        ch = ch + 0x30;
    else
        ch = ch + 0x37;
    return(ch);
}


//main routine


int  main ()
{
    unsigned int Fdiv, value,i,j;
    // char value;
```

```
    TargetResetInit();
    // init_timer( ((72000000/100) - 1) );

    PCONP |=0X00001000;   //switch ADC from disable state to enable state
    PINSEL0 = 0x00000050; //Pinselection for uart tx and rx lines
    PINSEL1 = 0X01554000; //Pinselection for adc0.0


    //uart initialization


    U0LCR = 0x83;          // 8 bits, no Parity, 1 Stop bit

    Fdiv = ( 72000000 / 16 ) / 19200 ;  //baud rate

    //Fdiv = ( 72000000 / 16 ) / 2400 ;  //baud rate

    U0DLM = Fdiv / 256;
    U0DLL = Fdiv % 256;
    U0LCR = 0x03;            // DLAB = 0

    AD0CR = 0X01210F01; // Adc initialization
    while(1)
    {
        // Wait here until adc completes the conversion
        while((AD0DR0 & 0X80000000)!=0X80000000){};

        //to get the  converted value and display it on the serial port

        value = (AD0DR0>>6)& 0x3ff ;    //ADC value
        //serial_tx(value);

        serial_tx('\t');
        serial_tx(atoh((value&0x300)>>8));
        serial_tx(atoh((value&0xf0)>>4));
        serial_tx(atoh(value&0x0f));
        serial_tx(0x0d);
        serial_tx(0x0a);

        for(i=0;i<=0xFF;i++)
        {
            for(j=0;j<=0xFF;j++);
        }
    }
    return 0;
}
```

Output

Video: Converting analog input from sensor to a digital value and displaying on the serial monitor.

Problem 3: DAC

Problem statement

Given the ViARM2378 ARM development board, generate

1. Square wave
2. Triangular wave
3. Sine wave (using a lookup table)

Code

Generating a square wave

```
#include "LPC23xx.h"

void large_delay(unsigned int k)
{
    unsigned int i,j;
    for(i=0;i<=k;i++);
        for(j=0;j<=0xFF;j++);
}

int  main ()
{
    unsigned int value=0;
    unsigned int max=0x3FF;
    PINSEL1 = 0x00200000;   //Pinselection for uart tx and rx lines

    //UART initialization

    PCLKSEL0 = 0x00C00000;
    PINMODE1=0x00300000;
    FIO4DIR=0x00;

    //generating a square wave

    while(1)
    {
        DACR = value<<6;
        large_delay(0xff);
        DACR = max<<6;
        large_delay(0xff);
    }

    return 0;
```

```
}
```

Generating a triangular wave

```
#include "LPC23xx.h"

int main (void)
{
    PCLKSEL0=0x00C00000;
    PINMODE1=0x00300000;
    PINSEL1=0x00200000;
    int value;
    int i=0;

    while(1)
    {
        //generating a triangular wave

        value=0;

        while(value!=1023)
        {
            DACR=((1<<16)|(value<<6));
            value++;
        }

        while (value!= 0)
        {
            DACR=((1<<16)|(value<<6));
            value--;
        }

    }
    return 0;
}
```

Generating a sine wave

```
#include "LPC23xx.h"

// Lookup Table for sine values
int sin_wave[101]={0x200,0x220,0x240,0x25f,0x27f,
    0x29e,0x2bc,0x2d9,0x2f6,0x312,0x32c,0x346,0x35e,0x374,
    0x38a,0x39d,0x3af,0x3c0,0x3ce,0x3db,0x3e6,0x3ef,0x3f6,
    0x3fb,0x3fe,0x3ff,0x3fe,0x3fb,0x3f6,0x3ef,0x3e6,0x3db,
    0x3ce,0x3c0,0x3af,0x39d,0x38a,0x374,0x35e,0x346,0x32c,
```

```
    0x312,0x2f6,0x2d9,0x2bc,0x29e,0x27f,0x25f,0x240,0x220,
    0x200,0x1df,0x1bf,0x1a0,0x180,0x161,0x143,0x126,0x109,
    0xed,0xd3,0xb9,0xa1,0x8b,0x75,0x62,0x50,0x3f,0x31,0x24,
    0x19,0x10,0x9,0x4,0x1,0x0,0x1,0x4,0x9,0x10,0x19,0x24,
    0x31,0x3f,0x50,0x62,0x75,0x8b,0xa1,0xb9,0xd3,0xed,0x109,
    0x126,0x143,0x161,0x180,0x1a0,0x1bf,0x1df,0x200};


void large_delay(int n)
{
    int i,j;
    for(i=0;i<n;i++)
        for(j=0;j<0x0F;j++);
}


int main (void)
{
    PCLKSEL0=0x00C00000;
    PINMODE1=0x00300000;
    PINSEL1=0x00200000;
    int value;
    int i=0;


    while(1)
    {
        //generating a sine wave
        i=0;

        while(i<101)
        {
            value=sin_wave[i];
            DACR=(value<<6);
            large_delay(100);
            i++;
        }
    }
    return 0;
}
```

Output

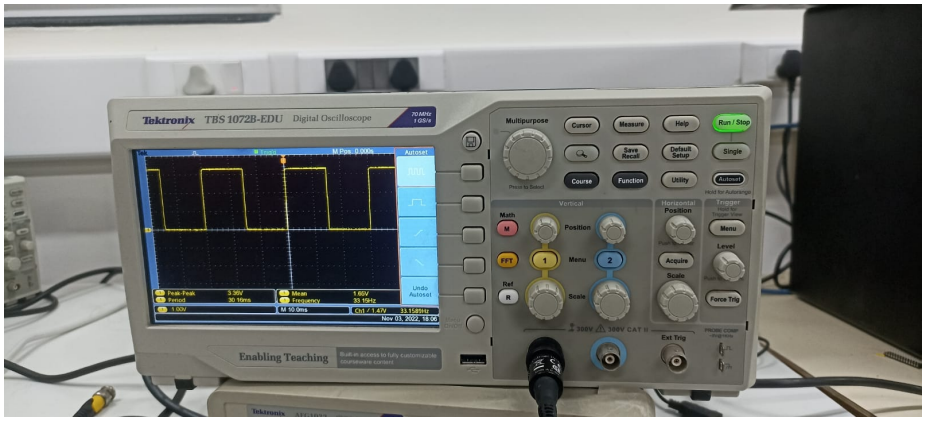

Figure 3: Generating a square wave

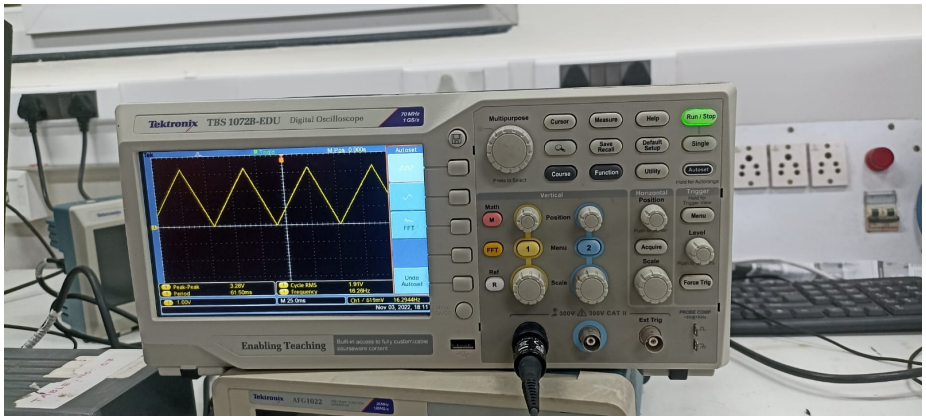

Figure 4: Generating a triangular wave

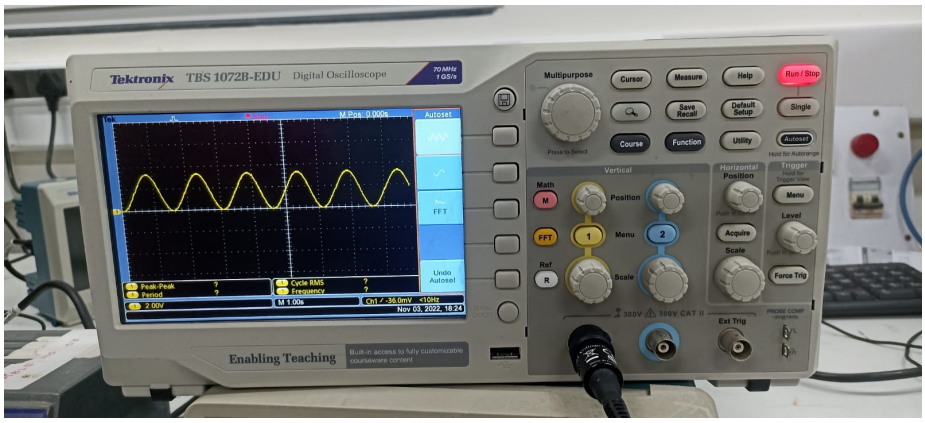

Figure 5: Generating a sine wave