Khalifa University for Sciences Technology and Research

Department of electrical and electronic engineering

ELCE333 :Microprocessor Systems Laboratory

Laboratory Experiment No. 8
Experiment Title: Analog to Digital Converter

Group Members
Afra I. Bin Fares 100033139
Alia Alalili 100037087
Anoud Alshamsi 100035514

Submitted to
Dr. Mahmoud Khonji
Dr. Mohammed Al Zaabi

Spring 2015

Table of contents

List of figures

Abstract

The laboratory experiment eight discusses about analogue to digital converter and it divide it into three essential tasks.Also, there are three control registers will be used inorder to control the ATD operation. The first task is about compiling a program and modifying to exam two different bit resolutions. Next, the second task which is about testing the temperature sensor using 10-bit resolution, whereas the third task is about testing the light sensor using the proximity of our hand to the sensor. All outputs and codes are provided in this report

1. Introduction

The HCS12 microcontroller supports analog to digital conversion where it has 8-channel and 16 channel implementations. The device has an automotive, industrial and tempreture functions. They operate at a value of 5V input voltage.

The analog to digital converter (ATD) highly autonomousand has flexiable conversion sequences. It provides all of the timing and controls for both sample and conversion periods. The resolution of the converter indicates the number of discrete values it can produce over the range of analog values. Sequences is first written to a single register first by a valid signal to synchronize the ATD conversion process with external events. Conversions in each sequence can be configured for sample time, resolution and result data format.

I. Aim:

To introduce the students to use of analog to digital (ATD) converters.

II. Objectives:

1. Understand the concept of analog to digital (ATD) converters.
2. Get introduced to the concept of ATD subsystem initialization and conversion.
3. Gain the experience using the 16 channels A/D converter of the DRAGON12.
4. Use onboard peripherals related to ATD
5. Use the CodeWarrior IDE for the development of HCS12 microcontroller C programs.
6. To compile, download and debug/test a C program using CodeWarrior C compiler and Dragon12 Plus Trainer board.

2. Design, result and analysis

In this part of the report, the tasks performed in the experiment will be discussed by explaining the steps of these tasks, listing, explaining and analyzing the results obtained from this experiment. The experiment is basically divided into three tasks. All three tasks were performed using C language written in the code warrior software, and the results were obtained using the dragon12 plus trainer.

2.1. Task-1: 10- Bit Resolution

In this part of the experiment the following program (provided in lab 7 script introduction) was built and executed in a code warrior project:

```
#include <hidef.h> /* common defines and macros */
#include "derivative.h" /* derivative-specific definitions */
#include "lcd.h"
void ATD_init(void)
{ ATD0CTL2_ADPU = 1; // power up ATD channel 0, disable interrupts
delay(1); // wait for ADC to warm up
ATD0CTL4 = 0x85; // 8-bit,sample time 2 ADC clock, prescale of 5,
}
int ATD_CONVERT()
{ATD0CTL5 = 0x87; // channel no. 7 and right justified
while(!(ATD0STAT0 & 0x80)); // wait for conversion to finish
return(ATD0DR0); // get and return the value to the caller
}
void main(void) {
int val;
float out;
LCD_Init();
ATD_init();
for(;;) {
val=ATD_CONVERT();
out=((val*1.0)/51); //output in Voltage = conversion result *step size
//out =conversion result *((VrefH-VrefL)/(2^resolution-1))
LCDWriteLine(1,"Value= ");
LCDWriteFloat(val);
LCDWriteLine(2,"Voltage= ");
LCDWriteFloat(out);
delay(100);
LCD_clear_disp();
_FEED_COP(); /* feeds the dog */
} /* loop forever */
/* please make sure that you never leave main */
}
```

Analysis: The program basically reads the value of the potentiometer while it is running on the dragon 12 board. The program reads the value as the potentiometer is moved using a screwdriver and displays it on the LCD screen. The used resolution in this case is 8-bits.

Results:The LCD readings were as shown in Figure 1 & 2.(note that the value changes as the potentiometer is moved).

Figure 5: reading1

Figure 6: reading 2

In the second part of this task, the program was modified to have an ATD resolution of 10-bits. This was done by setting ATD0CTL4 = 0x05; as explained in the introduction, the program is modified as the following:

```
#include <hidef.h> /* common defines and macros */
#include "derivative.h" /* derivative-specific definitions */
#include "lcd.h"
void ATD_init(void)
{ ATD0CTL2_ADPU = 1; // power up ATD channel 0, disable interrupts
delay(1); // wait for ADC to warm up
ATD0CTL4 = 0x05; // 10-bit,sample time 2 ADC clock, prescale of 5,
}
int ATD_CONVERT()
{ATD0CTL5 = 0x87; // channel no. 7 and right justified
while(!(ATD0STAT0 & 0x80)); // wait for conversion to finish
return(ATD0DR0); // get and return the value to the caller
}
void main(void) {
int val;
float out;
LCD_Init();
ATD_init();
for(;;) {
val=ATD_CONVERT();
out=((val*5.0)/1023); //output in Voltage = conversion result *step size
//out =conversion result *((VrefH-VrefL)/(2^resolution-1))
LCDWriteLine(1,"Value= ");
LCDWriteFloat(val);
LCDWriteLine(2,"Voltage= ");
LCDWriteFloat(out);
delay(100);
LCD_clear_disp();
_FEED_COP(); /* feeds the dog */
} /* loop forever */
/* please make sure that you never leave main */
}
```

Observations: For the second program Results appeared in the LCD screen were similar to the ones shown in Figure 1&2, however since we increased the resolution , we got better and more accurate results.

2.2. Task-2: Temperature sensor

In this part of the experiment, a C language program was written in order to read the temperature sensor output and display it as a voltage on the first line of the LCD and as a temperature in Celsius on the second line of the LCD. a 10 bit resolution was used(the sensor's resolution is 10 mV/C).the program was as follows:

```
#include <hidef.h> /* common defines and macros */
#include "derivative.h" /* derivative-specific definitions */
#include "lcd.h"
void ATD_init(void)
{ ATD0CTL2_ADPU = 1; // power up ATD channel 0, disable interrupts
delay(1); // wait for ADC to warm up
ATD0CTL4 = 0x05; // 10-bit,sample time 2 ADC clock, prescale of 5,
}
int ATD_CONVERT()
{ATD0CTL5 = 0x85; // channel no. 5 and right justified
while(!(ATD0STAT0 & 0x80)); // wait for conversion to finish
return(ATD0DR0); // get and return the value to the caller
}
void main(void) {
int val;
float out,temp;
LCD_Init();
ATD_init();
for(;;) {
val=ATD_CONVERT();
out=((val*5.0)/1023); //output in Voltage = conversion result *step size
temp=out/0.01;
//out =conversion result *((VrefH-VrefL)/(2^resolution-1))
LCDWriteLine(1,"Voltage= ");
LCDWriteFloat(out);
LCDWriteLine(2,"Temp: ");
LCDWriteFloat(temp);
delay(100);
LCD_clear_disp();
_FEED_COP(); /* feeds the dog */
} /* loop forever */

}
```

Observations: The results in the LCD screen was as shown in Figure 3.

Figure 7: temperature sensor readings

2.3. Task-3: Light sensor

In the third part of the experiment, a C language program was written in order to read the light sensor output and alter the speed of the flash of the PORTB LEDs based on the proximity of our hand to the sensor.the program was as follows:

```
#include <hidef.h> /* common defines and macros */
#include "derivative.h" /* derivative-specific definitions */
#include "lcd.h"
void ATD_init(void)
{ ATD0CTL2_ADPU = 1; // power up ATD channel 0, disable interrupts
delay(1); // wait for ADC to warm up
ATD0CTL4 = 0x05; // 10-bit,sample time 2 ADC clock, prescale of 5,
}
int ATD_CONVERT()
{ATD0CTL5 = 0x84; // channel no. 5 and right justified
while(!(ATD0STAT0 & 0x80)); // wait for conversion to finish
return(ATD0DR0); // get and return the value to the caller
}
void main(void) {
DDRB=0xFF;                    //MAKE PORTB OUTPUT
DDRJ=0xFF;
PTJ=0x00;
DDRP=0xFF;
PTP=0x0F;                    //TURN OFF 7SEG LED
ATD_init();

for(;;) {
int val;
val=ATD_CONVERT();
PORTB=0xFF;
delay(val);
PORTB=0x00;
delay(val);
_FEED_COP(); /* feeds the dog */
} /* loop forever */
}
```

Observation:the LED flashed on a speed depending on the sensor, as the light sensor reads a specific value the speed of the flashes change, as shown in Figure 4.

Figure 8: LEDs observation

3. Conclusion and Recommendation

In conclusion, we understood the main concept of the analog to digital (ATD) converters.Also, we learnt how to use the 16 channels A/D converter (DRAGON12 board) . Not only this, also we learnt how to use the onboard peripherals.Finally, we end up with pleasant results that's because we believe that we done the lab experiment properly and we recommend that's students should obtain proper knowledge and try to link it with what they achieved in laboratory experiment inorder to understand the main goal of doing this experiment .

4. Assignment Question

1) Write a C language program to change the speed of flashing of the PORTB LEDs based on the potentiometer value.

```
#include <hidef.h>
#include "derivative.h"
#include "lcd.h"
void ATD_init(void)
{
    ATD0CTL2_ADPU = 1;        // power up ATD channel  0, disable interrupts
    delay(1);            // wait for ADC to warm up
    ATD0CTL4 = 0x85;          // 8-bit,sample time 2 ADC clock, prescale of 5,
}
int ATD_CONVERT()
{ATD0CTL5 = 0x87;         // channel no. 7 and right justified
    while(!(ATD0STAT0 & 0x80));     // wait for conversion to finish
    return(ATD0DR0);         // get and return the valueue to the caller
}
void main(void) {
    int value;
    float out;
    ATD_init();
    LCD_Init();
    DDRB = 0xFF;                 //PTB as output for LEDs
    DDRJ = 0xFF;                   //PTJ as output
    DDRT = 0xFF;                //PORTT as output
    PTJ = 0x00;                   //Let PTB show data. Needed by Dragon12+ board
    PTP=  0x0F;

    for(;;) {

        value=ATD_CONVERT();
        out=((value*1.0)/51);   //output in Voltage = conversion result *step size
        LCDWriteLine(1,"Value= ");
        LCDWriteFloat(val);
        LCDWriteLine(2,"Voltage= ");
        LCDWriteFloat(out);

        delay(500);

        LCD_clear_disp();
        PORTB = 0xFF ;
        delay(out) ;
        PORTB = 0x00 ;
        delay(out) ;

    }
}
```

2) Modify the program in task 2 to display the temperature on the second line of the LCD in Kelvin if SW4 is pressed and in Fahrenheit if SW3 is pressed.

```
#include <hidef.h>
#include "derivative.h"
#include "lcd.h"
void ATD_init(void)
{
    ATD0CTL2_ADPU = 1;      // power up ATD channel 0, disable interrupts
    delay(1);           // wait for ADC to warm up
    ATD0CTL4 = 0x05;        // 10-bit,sample time 2 ADC clock, prescale of 5,
}

int ATD_CONVERT()
{
    ATD0CTL5 = 0x85;         // channel no. 5 and right justified
    while(!(ATD0STAT0 & 0x80));  // wait for conversion to finish

    return(ATD0DR0);        // get and return the value to the caller
}
void main(void)
{
    int val;
    float out;
    float temp ;
    float Fahrenheit;
    float Kelvin;

    LCD_Init();
    ATD_init();

    DDRH = 0x00;
    PTP=  0x0F;
    PUCR=0x01;

for(;;) {
    val=ATD_CONVERT();
    out=((val*5.0)/1023);
    temp= ((out*1000)/10);
    Fahrenheit =((1.8 * temp)+32);
    Kelvin = temp + 273;
    LCDWriteLine(1,"Tempreture= ");
    LCDWriteFloat(temp);
    delay(500);
    LCD_clear_disp();
    if (PTH_PTH1==1) {

        LCDWriteLine(2,"in Kelvin= ");
        LCDWriteFloat(Kelvin);

    }
    else
        if (PTH_PTH2==1 )
        {
            LCDWriteLine(2,"in Fahrenheit= ");
            LCDWriteFloat(Fahrenheit);        }}}
```