Microprocessor Systems Laboratory (ELCE333)

Laboratory Experiment No.8

MARKING SHEET

Student Names and ID No's: Amal AlQasimi 100036830
Shaikha AlBesher 100037064
Dana Bazazeh 100038654

Laboratory Section: Wednesday **Experiment Date: March 25, 2015**

No.	**Criteria**	**Description**	**Weight %**	**Mark**	**Comments**
1	Pre-lab	A mark will be allocated to each student that reflects his preparations for the lab.	20		
2	Performance In the lab	A mark will be allocated to each student individually that reflects his performance in the lab	10		
3	Results and Analysis	Documentation and analysis of the results for each task performed in the lab	30		
4	Summary/ Conclusions	Conclusions for each task performed in the lab	10		
5	Assignment Questions	Answers to assignment questions	20		
6	Report Presentation	Overall presentation of the report including proper layout and clarity of figures, tables, and graphs. Correct use of English language.	10		
	Total		**100**		

ELCE 333: Microprocessor Systems Laboratory

Lab Report- Experiment No.8

Experiment Title:

Completed by: Analogue to Digital Converter

Amal Al Qasimi (100036830)

Shaikha Al Besher (100037064)

Dana Bazazeh (100038654)

Lab instructors:

Mahmoud Khonji

Mohammed Al Zaabi

Spring 2015

Table of Contents

Summary

In this lab, students were introduced to the concept of conversion and how the ADC Analogue to Digital converter is used. The operation of the ADC device is discussed and several registers and functions that are used to manipulate ADC settings are introduced. This lab consists of 3 main tasks that are described and analyzed in detail in this report. The first task talks about how to read the potentiometer output value using 8 and 10 bits resolution and to then display the ADC conversion output on the LCD. Task 2 is about how to read the temperature from the temperature sensor output and display it in Celsius as well as a voltage on the LCD. The 3rd task asks us to read the light sensor output and based on our hand proximity to the sensor, alter the flashing speed of LEDs of PORTB. At the end of this report, a detailed analysis and a conclusion are presented to sum up the results obtained.

1. Introduction

Computers and microcontrollers can only operate on digital data and therefore need their input to be in digital format. However, many physical signals in the real world such as temperature and light quantities are analogue. We must therefore have some kind of converter to convert these analogue quantities to digital signals. An Analogue to Digital Converter ADC is a device that is part of the HCS12 microcontroller device that can take an analog signal and digitize it into a binary format. The HCS12 chips come with two sets called ATD0 and ATD1, each have 8 channels. With a 2MHz module clock, the ADC can perform an 8-bit single sample in 6µs or a 10-bit single conversion in 7µs (including sample times).

Factors that influence conversion result

1) Converter resolution (8, 10, 12, 16, 24 bits)
2) Converter reference voltage supplied

The analog subsystem consists of

1) A multiplexer,
2) Input sample amplifier,
3) Resistor-capacitor digital-to-analog converter (RC DAC) array
4) High-gain comparator.

ADC's main registers and their uses are:

1) **ATD control register 2 (ATDxCTL2)**

   Bit (ADPU) which is used to enable the ATD converter

2) **ATD control register 4 (ATDxCTL4)**

   Bit 7 used for selecting the conversion resolution value.

   Bits 6 &5 are used for setting clock cycles for sample time.

   Bits 3-0 are used for setting the prescaler value.

3) **ATD control register 5 (ATDxCTL5)**

   Bits 2-0 used for selecting the channel to read from

4) **ATD Status Register**

   Bit 7 is set when conversion result is completed and ready to use.

5) **Conversion Result Register (ATDxDRy)**

   Stores the conversion result of the ADC

1. 1.1 Aim:

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

2. 1.2 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 and Results

3. Task 1: 10- Bit Resolution

(a) This tasks asks us to execute the program given in the labscript on the Dragon12 Plus Trainer and then vary the position of the microcontrollers potentiometer position and in order to verify the LCD readings using a DMM on the NI ELVIS Instruments.

The full program code is given below:

```
#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 */
}
```

Figure 1: Program code to display the potentiometer output on the LCD

The code starts by enabling the ADC by setting the ADPU bit of the second control register ATD0CTL2. In order to set the resolution to 8 bits, bit 7 of the ATD0CTL4 must be set and also the 4 lower bits register bits that represent the prescaler should be set to 5. Next, channel 7 is selected by setting the 4 lower bits of register ATD0CTL5 to 7. The ATD_CONVERT() function will wait until bit7 of the status register is set and will then return the conversion results found in register ATD0DR0. In the main function, we initialize the LCD connection as we have done in previous labs and then initialize a variable val to store the result of the conversion. In order to get the value in volts, a conversion needs to be done. For 8 bit resolution, this is (val*1.0)/51. Finally, the output is displayed on the LCD. After loading and running the program, the results of both the LCD areads the potentiometer output value using 8 bits resolution and displays the output on the LCD: nd the Digital Multimeter are shown below:

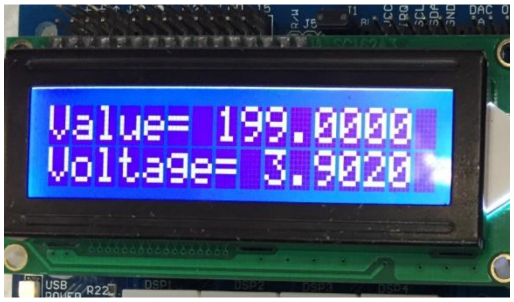
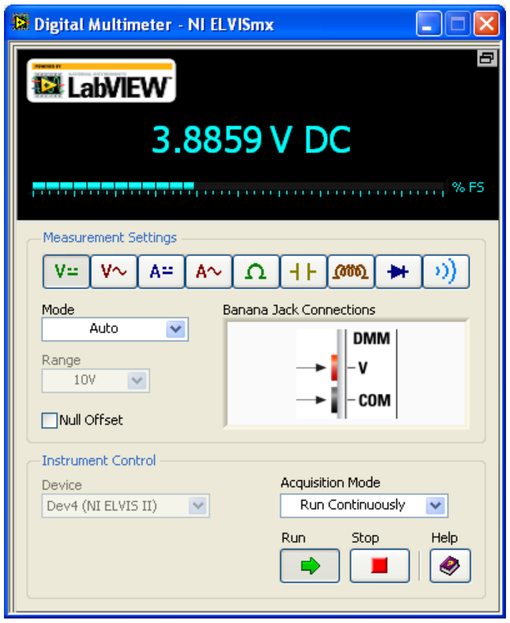


Figure 3: DMM result for 8 bit resolution


Digital Multimeter - NI ELVISmx

3.8859 V DC

Measurement Settings

Mode: Auto

Range: 10V

Null Offset

Banana Jack Connections

Instrument Control

Device: Dev4 (NI ELVIS II)

Acquisition Mode: Run Continuously

Figure 2: LCD output for 8 bit resolution

We can see that there is a very small difference between the 2 values. A difference of 3.9020-3.8859 = 0.0161 volts is noted.

Using a screw driver, we change the potentiometer position and notice how this changes the value of the voltage noted with a maximum of 5 volts.

(b) Next, we need to modify the given program to set the ADC resolution to 10 bits. Here we only need to modify 2 lines of code.

ATD0CTL4 = 0x05;

Here we clear bit 7 to represent a resolution of 10 bits while keeping the prescaler value same (5).

out=((val*5.0)/1023);

Here we multiply by the maximum voltage which is 5 and divide by $2^{10} - 1 = 1023$ where 10 is the resolution.

Again we load and run the program and obtain the results below:

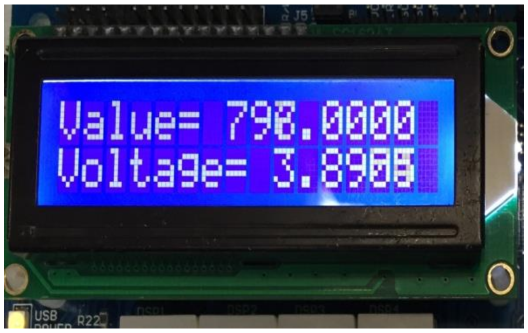
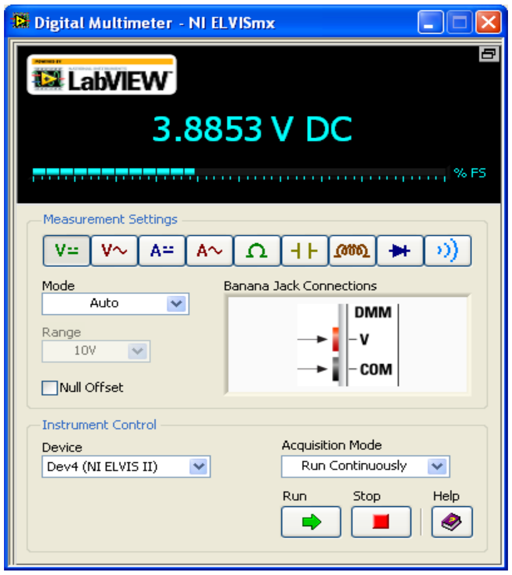


Figure 4: LCD output for 10 bit resolution



Digital Multimeter - NI ELVISmx

3.8853 V DC

Measurement Settings

Mode: Auto

Range: 10V

Null Offset

Banana Jack Connections

Instrument Control

Device: Dev4 (NI ELVIS II)

Acquisition Mode: Run Continuously

Figure 5: DMM result for 10 bit resolution

The difference here is 3.8905-3.8853 = 0.0052 which is smaller than the difference obtained in part a where the resolution was 8 bits.

Therefore we can conclude that increasing the resolution to 10bits provides us with more precise results.

4. Task 2: Temperature sensor

This task asks us to write a C language program that will allow us to read the temperature from the temperature sensor output and then display it both as a voltage and as a temperature in Celsius on the LCD. We modify the program from the first task and the full program is shown below:

```
#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;
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(1,"Voltage= ");
LCDWriteFloat(out);
LCDWriteLine(2,"Temperature= ");
val = out/0.01;
LCDWriteFloat(val);
delay(100);
LCD_clear_disp();
_FEED_COP(); /* feeds the dog */
} /* loop forever */
/* please make sure that you never leave main */
}
```

Figure 6: Program code to display the temperature sensor output on the LCD

We set the resolution to 10 bits as required by setting the register ATD0CTL4 = 0x05. Also, we know that the temperature sensor is connected to analogue channel 5 of ADC and therefore we have to change control register 5 ATD0CTL5 = 0x85, where the 4 lower bits represent the channel number. The voltage output is calculated as done in task 1. We know that the sensor's resolution is 10 mV/C and therefore the temperature can be obtained by dividing this voltage by 0.01. We test the program by loading and running it on the Dragon12 Plus Trainer and the output is shown below:

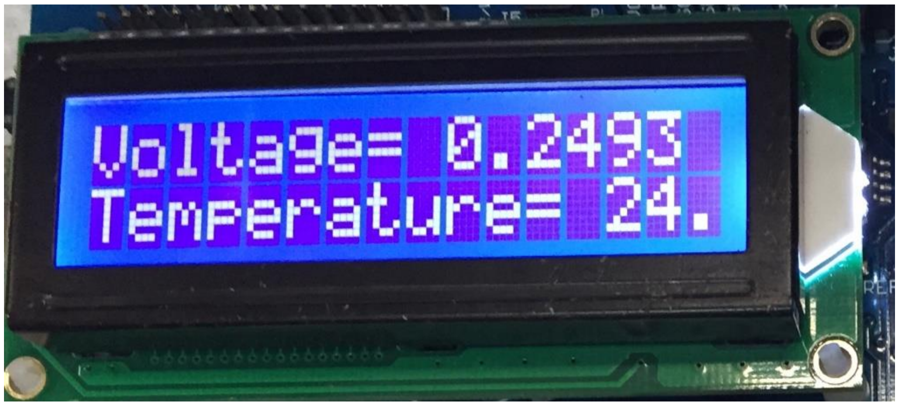


Figure 7: LCD output from temperature sensor

The temperature 24 degrees Celsius represents the room temperature and therefore the program must be correct. By covering the temperature sensor, we noticed that the temperature recorded increased.

5. *Task 3: Light Sensor*

This task requires writing a C program that can read the light sensor output. Accordingly the speed of the flash of the PORTB LEDs will vary depending on the proximity of the hand to the sensor.

Refer to the code below:

```
#include <hidef.h> /* common defines and macros */
#include "derivative.h" /* derivative-specific definitions */
#include "lcd.h"
void delay2(unsigned long int ms)
{   unsigned long int i,j;
for(i=0;i<(ms*100);i++)for(j=0;j<14;j++) asm("nop\n");


}
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,
//ATD0CTL4 = 0x85; // 8-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) {
int val;
float out;
DDRB = 0xFF;
DDRJ = 0xFF;
DDRP = 0xFF;
DDRT = 0xFF;
PTP = 0x0F;
PTJ = 0x00;


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);

PORTB = 0x00;
delay2(out*1000);

PORTB = 0xFF;
delay2(out*1000);




_FEED_COP(); /* feeds the dog */
} /* loop forever */
/* please make sure that you never leave main */
}
```

3. Conclusion:

Laboratory experiment 8 consists of 3 tasks, which are:

- Task 1: That is related to changing the resolution of particular code.
- Task 2: Uses the temperature sensor to show the temperature associated with the voltage and the Celsius value of the temperature on the LCD.
- Task 3: Uses a light sensor to measure the brightness around it, and accordingly change the speed of the flashing LEDs.

The laboratory experiment is an introduction to the Analog to Digital (ATD) convertors, the concept of ATD subsystem initialization and conversion. Upon the completion of the lab session the tasks were successfully accomplished and fulfill the objectives without difficulties.

4. References

1. CourseTextBook
2. http://www.evbplus.com/
3. MC9S12DP256User'sManual