CPE301 – SPRING 2025

Design Assignment 4

Student Name: Ryan Sewell

Student #: 8000473785

Student Email: sewelr2@unlv.nevada.edu

Primary Github address: https://github.com/sewelr2

Directory: DA-Submissions/DA4

Video Playlist: <https://youtube.com/playlist?list=PLt45mEFhRV6ffOYRcGHhoI5aDeP3Zgqt5&feature=shared>

The goal of this assignment is to write, implement and demonstrate using Microchip Studio 7 a C code for the AVR ATMEGA328pb

* Read the ADC value from the POT connected to AC0/PC0. Keep displaying the voltage value UART terminal every 0.01 sec. The resolution of the oscilloscope should be 0.1V. Use Timer auto-trigger for this implementation.
* Using a GUI Python script display the ADC values as waveform (using tkinter).

1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

Microchip Studio Atmega328PB-Xmini PC Multi-Function Shield Tauno Serial Plotter

* Assembler - Potentiometer
* Simulator
* Debugger

A computer circuit board with many letters and numbers

AI-generated content may be incorrect.

1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1**

#define F\_CPU 16000000UL

#define BAUD\_RATE 9600

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

void usart\_init ();

void adc\_init();

void timer\_init();

void usart\_send (unsigned char ch);

int main (void)

{

timer\_init ();

usart\_init ();

adc\_init ();

while (1)

{

/\*

ADCSRA|=(1<<ADSC); //start conversion

while((ADCSRA&(1<<ADIF))==0);//wait for conversion to finish

ADCSRA |= (1<<ADIF);

int a = ADCL;

a = a | (ADCH<<8);

a = (a/1024.0) \* 5000/10;

usart\_send((a/100)+'0');

a = a % 100;

usart\_send((a/10)+'0');

a = a % 10;

//a = a \* (9/5) + 32; // celsius to fahrenheit

usart\_send((a)+'0');

usart\_send('\r');

\_delay\_ms(100);

\*/

}

return 0;

}

ISR (TIMER1\_OVF\_vect)

{

ADCSRA|=(1<<ADSC); //start conversion

while((ADCSRA&(1<<ADIF))==0);//wait for conversion to finish

ADCSRA |= (1<<ADIF);

int a = ADCL;

a = a | (ADCH<<8);

a = (a/1024.0) \* 5000/10;

usart\_send((a/100)+'0');

a = a % 100;

usart\_send((a/10)+'0');

a = a % 10;

//a = a \* (9/5) + 32; // celsius to fahrenheit

usart\_send((a)+'0');

usart\_send('\r');

\_delay\_ms(100);

TCNT1 = 49911; // Reset timer

}

void usart\_init (void)

{

UCSR0B = (1<<TXEN0);

UCSR0C = (1<< UCSZ01)|(1<<UCSZ00);

UBRR0L = F\_CPU/16/BAUD\_RATE-1;

}

void adc\_init (void)

{

/\*\* Setup and enable ADC \*\*/

ADMUX = (0<<REFS1)| // Reference Selection Bits

(1<<REFS0)| // AVcc - external cap at AREF

(0<<ADLAR)| // ADC Left Adjust Result

(1<<MUX2)| // Analog Channel Selection Bits

(0<<MUX1)| // ADC4 (PC4 PIN27)

(1<<MUX0);

ADCSRA = (1<<ADEN)| // ADC ENable

(0<<ADSC)| // ADC Start Conversion

(0<<ADATE)| // ADC Auto Trigger Enable

(0<<ADIF)| // ADC Interrupt Flag

(0<<ADIE)| // ADC Interrupt Enable

(1<<ADPS2)| // ADC Prescaler Select Bits

(0<<ADPS1)|

(1<<ADPS0);

}

void timer\_init (void)

{

TCCR1B |= 5; //(1 << CS12) | (1 << CS10); // Sets prescaler to 1024

TIMSK1 = (1 << TOIE1); // Enables overflow flag

TCNT1 = 49911; // 1 second delay = (0xFFFF) - TCNT = 65535 - 15624 = 49911

sei();

}

void usart\_send (unsigned char ch)

{

while (! (UCSR0A & (1<<UDRE0))); //wait until UDR0 is empty

UDR0 = ch; //transmit ch

}

void usart\_print(char\* str)

{

int i = 0;

while(str[i] != 0)

usart\_send(str[i]);

}

1. **DEVELOPED/MODIFIED CODE OF TASK 2/A from TASK 1**

#include <stdio.h>

void usart\_print(const char \*str);

int main(void)

{

while(1)

{

// Main loop waits for ADC ISR triggered by Timer1

}

return 0;

}

ISR(ADC\_vect)

{

uint16\_t adcVal = ADC; // Read ADC result

unsigned int volts\_tenth = (adcVal \* 50UL + 511) / 1023; // Convert ADC value to 0.1V for 5V ref

char buffer[10]; // Size of string 10 bits

snprintf(buffer, sizeof(buffer), "%u.%u V\r\n", volts\_tenth / 10, volts\_tenth % 10); // Formatting of displayed voltage

usart\_print(buffer); // Print string

}

void adc\_init()

{

ADMUX = (1 << REFS0); // Select AVCC as ref and AC0 as input

// Enable ADC with auto trigger and set prescaler to 128

ADCSRA = (1 << ADEN) | (1 << ADATE) | (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0);

ADCSRA |= (2 << ADIE); // Enable ADC interrupt

ADCSRB = (1 << ADTS2) | (0 << ADTS1) | (1 << ADTS0); // Timer1 compare match B auto trigger source

ADCSRA |= (1 << ADSC); // Start first ADC conversion

}

void timer\_init()

{

TCCR1B = (1 << WGM12); // Set Timer1 to CTC mode

OCR1A = 2499; // Config for 10ms timer @ 16MHz

OCR1B = 2499; // Set trigger point for ADC @ 10ms interval

TCCR1B |= (1 << CS11) | (1 << CS10); // Start Timer1 with prescaler 64

}

void usart\_print(const char\* str)

{

while (\*str)

{

usart\_send(\*str++);

}

}

1. **SCHEMATICS**

A diagram of a circuit

AI-generated content may be incorrect.

1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

A screenshot of a computer

AI-generated content may be incorrect.

1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**

Task 1:

Task 2:

…

1. **VIDEO LINKS OF EACH DEMO**

Task 1:

Task 2:

…

1. **GITHUB LINK OF THIS DA**

Task 1:

Task 2:

…

**Student Academic Misconduct Policy**

<http://studentconduct.unlv.edu/misconduct/policy.html>

“This assignment submission is my own, original work”.

Ryan Sewell