CPE301 – SPRING 2019

Design Assignment 5A

Student Name: Tanner Tindall

Student #: 8000733733

Student Email: tindat1@unlv.nevada.edu

Primary Github address: <https://github.com/TannerTindall51>

Directory:

<https://github.com/TannerTindall51/tindalltannerm_submission/tree/master/Design_Assignments/DA5A>

Design Assignment 5 The goal of the assignment is to develop the above code to do the following:

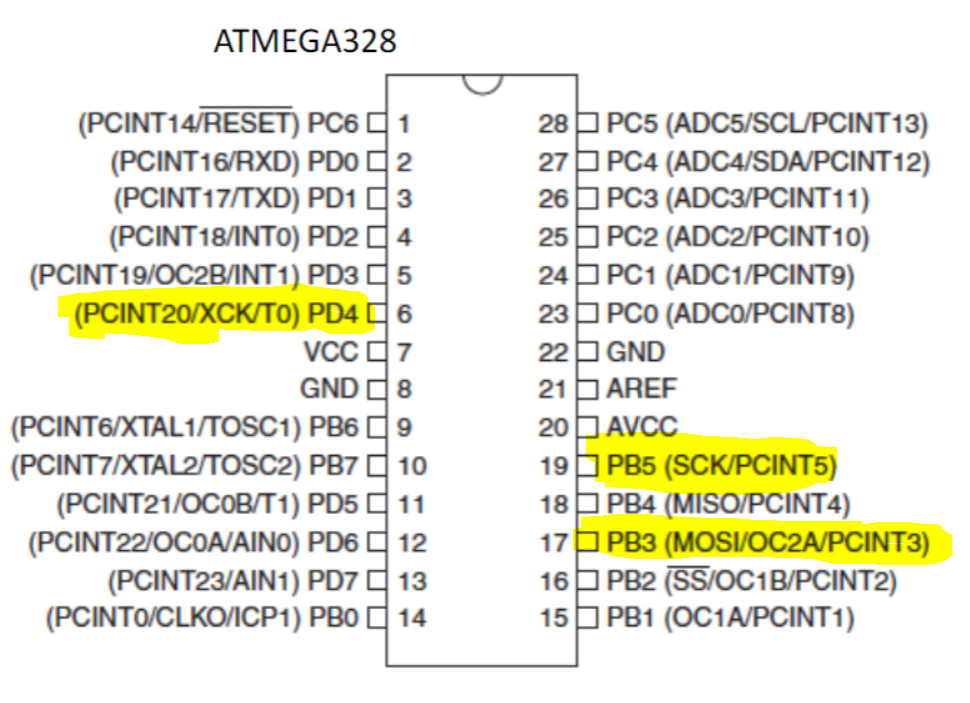
1. Using the Multi-function shield, using a custom serial transfer protocol display the temperature sensor value (int only) of LM34/35 on to the seven-segment display.

2. Using the Multi-function shield, using the SPI protocol display the temperature sensor value (int only) of LM34/35 on to the seven-segment display.

3. Using one-wire interface display the temperature sensor for DS18B20 on the UART terminal.

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

* Atmel Studio 7.0 (Assembler, Simulator, & Debugger)
* Atmega328PB-Xmini
* Micro USB
* Shield



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

//

//CPE301 - DA5A Task 1

//Tanner Tindall

//

#define BAUD 9600

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <util/setbaud.h>

#include <stdlib.h>

#include <stdio.h>

#include <util/delay.h>

#define latch 4

#define clock 7

#define data 0

const *uint8\_t* SEGMENT\_MAP[] = {0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0X80,0X90};

const uint8\_tSEGMENT\_SELECT[] = {0xF1,0xF2,0xF4,0xF8};

volatile *uint16\_t* tempC\_value;

void shift\_out\_init(void)

{

DDRB |= (1<<data);

DDRD|=(1<<clock)|(1<<latch); //PD7 = SCLK & PD4 = latch

}

*uint16\_t* adcRead()

{

ADCSRA |= (1<<6); //enable ADC conversion

while (!(ADCSRA & (1<<latch))); //waits for conversion to complete by ADIF flag

ADCSRA |= (1<<latch); //reset when complete

return ADC; //return value once complete

}

void startADC()

{

DDRC &= ~(1<<4); //set PC4 to input

ADMUX = (1<<REFS1)| (1<<REFS0)| (1<<MUX2); //set Vref and set ADC4 as input

ADCSRA = (1<<ADEN) | (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0); //enable ADC and set prescalar to 128

ADCSRB = 0x00; //set ADCSRB to free running

}

void shift\_out(*uint8\_t* indata)

{

for (*uint8\_t* i = 0; i < 8; i++)

{

/\* Write bit to data port. \*/

if (0 == (indata & \_BV(7 - i)))

{

// digital\_write(SHIFT\_OUT\_DATA, LOW);

PORTB &= (0 << data);

}

else

{

// digital\_write(SHIFT\_OUT\_DATA, HIGH);

PORTB |= (1 << data);

}

/\* Pulse clock to write next bit. \*/

PORTD |= (1<<clock);

PORTD &= (0<<clock);

}

}

*uint16\_t* tempC()

{

*uint16\_t* temperature = adcRead(); //record value from adc

tempC\_value = ((temperature)\*(110.0/1023.0)); //convert to temp value

shift\_out(tempC\_value); //send the value out to display

}

int main(void)

{

shift\_out\_init();

startADC();

while (1)

{

adcRead();

tempC();

PORTD |= (1<<latch);

}

}

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

//

//CPE301 - DA5A Task 2

//Tanner Tindall

//

#define BAUD 9600

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <util/setbaud.h>

#include <stdlib.h>

#include <stdio.h>

#include <util/delay.h>

#define latch 4

const *uint8\_t* segMapArray[]= {0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0X80,0X90}; //taken from lect 13 summary

*uint8\_t* segSelectArray[];

volatile int x;

volatile *uint16\_t* tempC\_value;

void startADC()

{

DDRC &= ~(1<<4); //set PC4 to input

ADMUX = (1<<REFS1)| (1<<REFS0)| (1<<MUX2); //set Vref and set ADC4 as input

ADCSRA = (1<<ADEN) | (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0); //enable ADC and set prescalar to 128

ADCSRB = 0x00; //set ADCSRB to free running

}

*uint16\_t* adcRead()

{

ADCSRA |= (1<<6); //enable ADC conversion

while (!(ADCSRA & (1<<latch))); //waits for conversion to complete by ADIF flag

ADCSRA |= (1<<latch); //reset when complete

return ADC; //return value once complete

}

void initPorts()

{

DDRB |= (1<<3)|(1<<5); //set MOSI and SCK output

DDRD |= (1<<latch); //latch clock output

PORTD &= ~(1<<latch); //initialize clock to 0

SPCR0 = 0x07; //set SPI to master mode

}

void transmitSPI(*uint8\_t* data)

{

SPDR0 = data; //send data

while (!(SPSR0&(1<<SPIF))); //wait until data register in emptied and all is sent

SPSR0 |= (1<<SPIF); //reset flag

}

*uint16\_t* tempC()

{

*uint16\_t* temperature = adcRead(); //record value from adc

tempC\_value = ((temperature)\*(110.0/1023.0)); //convert to temp value

}

void Latch()

{

PORTD |= (1<<latch); //set

*\_delay\_ms*(20);

PORTD &= ~(1<<latch); //reset

}

int main()

{

initPorts(); //initialize ports

startADC(); //initialize adc

while(1)

{

tempC(); //collect a temp value from ADC

segConvert(); //convert to segment display and transmit

}

}

void segConvert()

{

for (x = 0; x < 4; x++) //loop for each segment select

{

segSelectArray[x] = ((*uint8\_t*)((tempC\_value/*pow*(10, x))) % 10); //scale temp value according to for loop

transmitSPI(segMapArray[segSelectArray[x]]); //send the position value to defined array

transmitSPI(1<<(4+x)); //shift transmitted value

Latch(); //latch the clock before next iteration

}

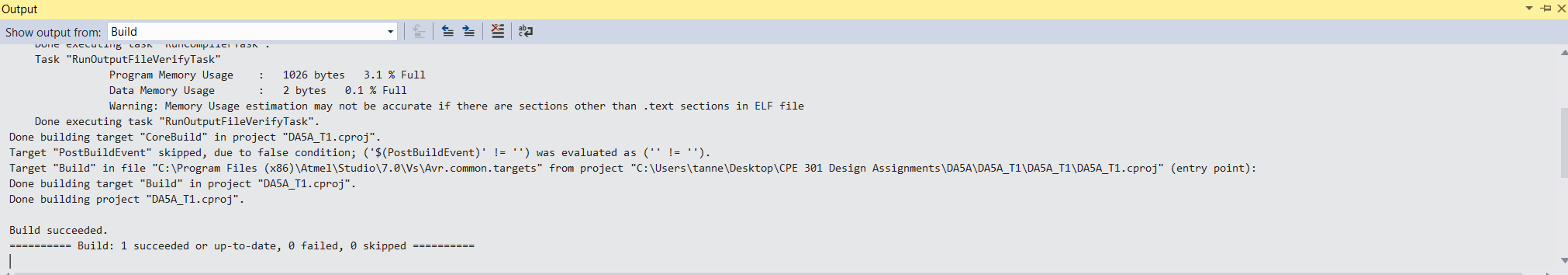
}

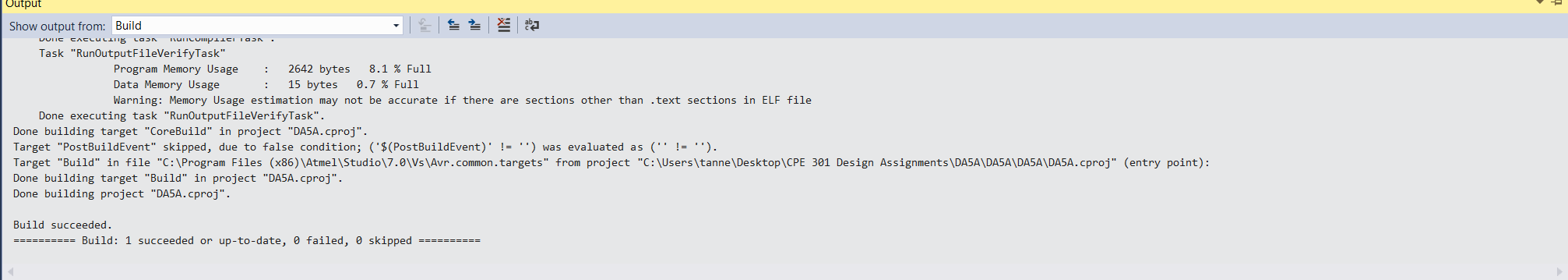
1. **SCHEMATICS**

N/A

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

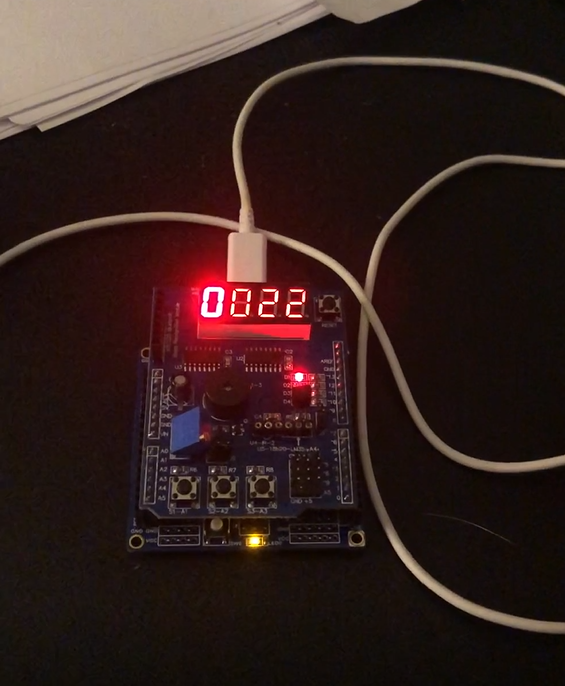
Task 1:



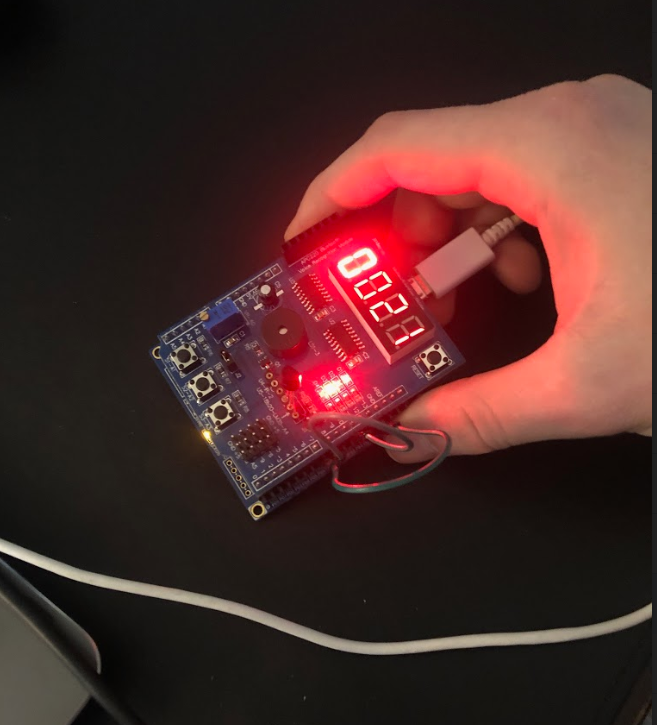
Task 2:

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

Task 1:



Task 2:



1. **VIDEO LINKS OF EACH DEMO**

Task 1: https://youtu.be/UloqvbBSXoI

Task 2: https://youtu.be/h4Y5UKhhAm8

1. **GITHUB LINK OF THIS DA**

<https://github.com/TannerTindall51/tindalltannerm_submission/tree/master/Design_Assignments/DA5A>

**Student Academic Misconduct Policy**

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

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

Tanner Tindall