CPE301 – SPRING 2021

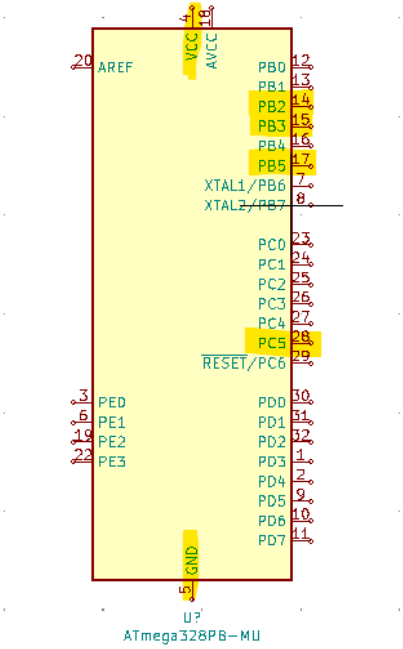
Midterm I

Student Name: Brian Wolak
Student #: 2000509437
Student Email: wolak@unlv.nevada.edu
Primary Github address: https://github.com/brianwolak/submission_da.git
Directory: submission_da/MIDTERM_1 at main · brianwolak/submission_da (github.com)

Task 1:

Write a C Code that will transmit the following information and actions using the following key inputs using the host terminal:

1. On-reboot of 'h' key – help screen listing all functionality/keys
2. 't' display temperature in ^{0}C
3. 'f' display temperature in ^{0}F
4. 'o' turn ON LED at PB5, 'O' turn OF LED at PB5
5. 'd' duty cycle value modification of LED located at PB3
6. 'i' frequency modification of LED at PB2



ATMEGA328pb Ports Used in Midterm I

Video Link:

https://youtu.be/hLQ4jqF9xxw

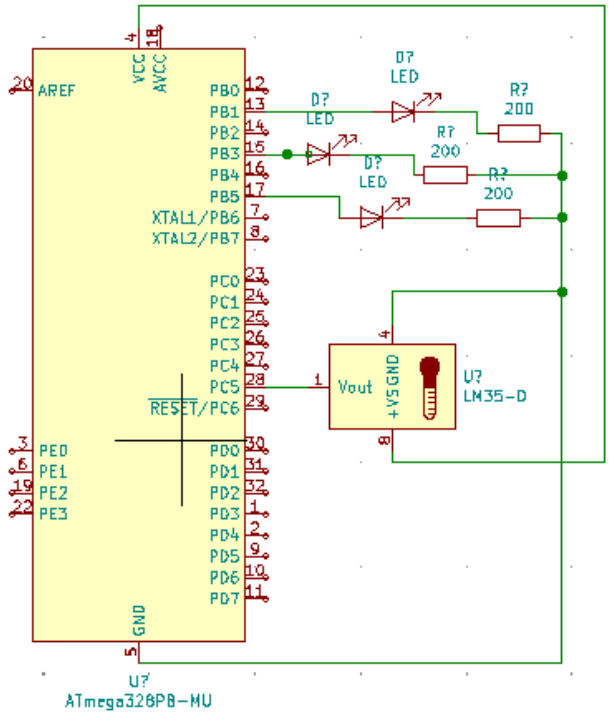
C Code:

```
#include <avr/io.h>
#include <stdio.h>

#ifndef F_CPU
#define F_CPU 16000000UL
#endif

#ifndef BAUD
#define BAUD 9600
#endif
#include <util/setbaud.h>
#include <avr/interrupt.h>
#include <util/delay.h>

volatile uint8_t tempread;                      //temp read value from ADC
char display1[20];                              //Fahrenheit string
char display2[20];                              //Celsius string


void uart_putchar(char c, FILE *stream);
char uart_getchar(FILE *stream);
void uart_init(void);

FILE uart_output = FDEV_SETUP_STREAM(uart_putchar, NULL, _FDEV_SETUP_WRITE);
FILE uart_input = FDEV_SETUP_STREAM(NULL, uart_getchar, _FDEV_SETUP_READ);
//UNSART initialize function
void uart_init(void) {
	UBRR0H = UBRRH_VALUE;
	UBRR0L = UBRRL_VALUE;

	#if USE_2X
	UCSR0A |= _BV(U2X0);
	#else
	UCSR0A &= ~(_BV(U2X0));
	#endif

	UCSR0C = _BV(UCSZ01) | _BV(UCSZ00); // 8-bit data
	UCSR0B = _BV(RXEN0) | _BV(TXEN0);   // Enable RX and TX
}

void uart_putchar(char c, FILE *stream) {
	if (c == '\n') {
		uart_putchar('\r', stream);
	}
	loop_until_bit_is_set(UCSR0A, UDRE0);
	UDR0 = c;
}

//input retrieve function
char uart_getchar(FILE *stream) {
	loop_until_bit_is_set(UCSR0A, RXC0);
	return UDR0;
```

```
}

//help screen menu display function
void help_menu(void){
	printf("\nWelcome to the help menu\n");
	printf("Press 't' to display temperature in Celsius\n");
	printf("Press 'f' to display temperature in Fahrenheit\n");
	printf("Press 'o' to turn on PB5 LED\n");
	printf("Press 'O' to turn off PB5 LED\n");
	printf("Press 'd' to modify PB3 LED duty cycle\n");
	printf("Press 'i' to modify PB2 LED frequency\n");
}
//read ADC function
void ADC_READ(void){
	ADCSRA |= (1<<ADSC);                    //start transfer
	while((ADCSRA & (1<<ADIF)) == 0);       //wait for ADIF flag
	ADCSRA |= (1 << ADIF);                  //clear ADIF flag
	tempread = ADC;                         //store temp value
}

//convert for serialplot output
void USART_TX_FLOAT(char data){
	UDR0 = data;
}

void USART_TX_string(char *data){
	while (*data != '\0'){                  //while data DNE 0
		while (!(UCSR0A & (1<<UDRE0)));     //while UNDRE0 DNE 1
		UDR0 = *data;                       //UDR0 gets data value
		data++;                             //next data value
	}
}

//ADC setup function
void adc_set(void){
	//reference Vcc, ADC5 input, PINC.5
	ADMUX |= (0<<REFS1) | (1<<REFS0) | (0<<ADLAR) | (0<<MUX3) | (1<<MUX2) | (0<<MUX1)
| (1<<MUX0);
	//enable ADC, 128 prescale
	ADCSRA = (1<<ADEN) | (0<<ADSC) | (0<<ADATE) | (0<<ADIF) | (0<<ADIE) | (1<<ADPS2) |
(1<<ADPS1) | (1<<ADPS0);
}


int main(void) {
	sei();
	DDRB |= (1<<5) | (1<<3) | (1<<2);       //set PB5,PB3, PB2 as outputs
	float tempc;                            //celsius temp variable
	float tempf;                            //farenheit temp variable
	int dutycycle;                          //duty cycle varaible
	int frequency;                          //frequency variable
	uart_init();                            //call USART initialize function
	adc_set();                              //call ADC setup function
	stdout = &uart_output;
	stdin  = &uart_input;

	char input;                             //keyboard input variable
	help_menu();                            //display help menu on startup
```

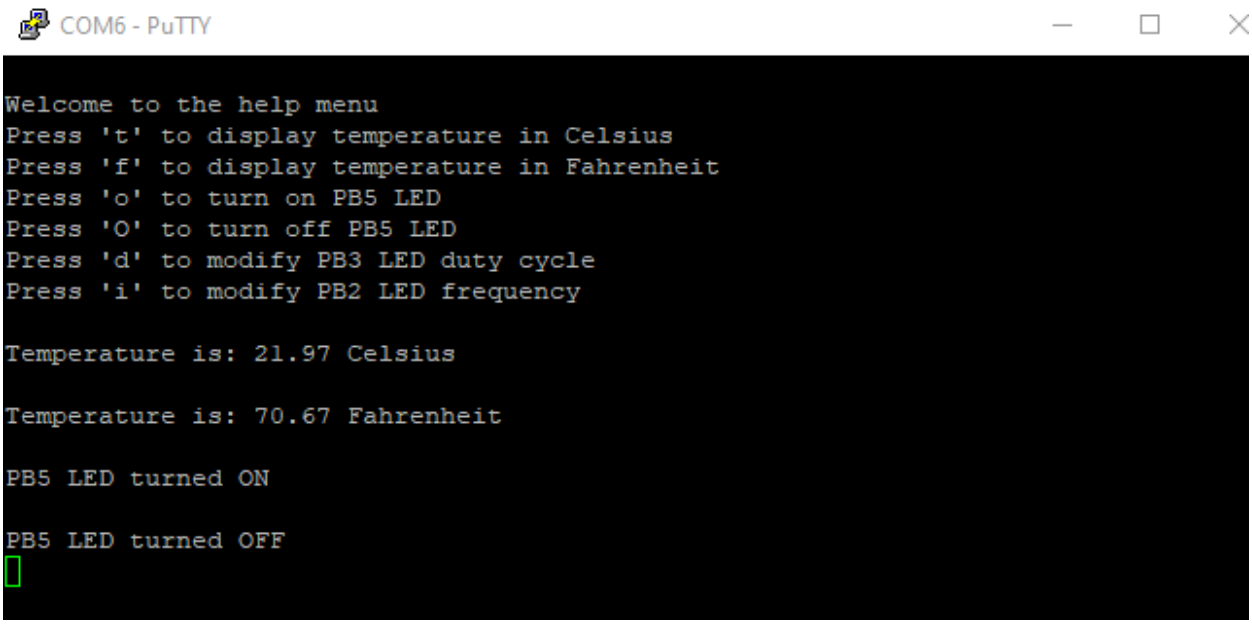
```
	while(1) {
		//printf("You wrote %c\n", input);
		input = getchar();				//keyboard input variable
		switch (input) {
			case 'h':
			help_menu();				//display help menu on 'h' key
			break;

			case 't':
			ADC_READ();				//read ADC on 't' key
			tempc = ((tempread * 500.0) / 1024);//tempc from binary to decimal
			sprintf(display2, "%.2f", tempc);  //convert tempc to string
			USART_TX_string("\nTemperature is: ");
			USART_TX_string(display2);
			USART_TX_string(" Celsius");
			USART_TX_string("\r\n");
			break;

			case 'f':
			ADC_READ();			//read ADC on 'f' key
			tempc = ((tempread * 500.0) / 1024);//tempc from binary to decimal
			tempf = (tempc * 1.8) + 32;	//convert to farenheit
			sprintf(display1, "%.2f", tempf);  //convert tempc to string
			USART_TX_string("\nTemperature is: ");
			USART_TX_string(display1);
			USART_TX_string(" Fahrenheit");
			USART_TX_string("\r\n");
			break;

			case 'o':
			PORTB |= (1<<PB5);		//turn PB5 LED ON with 'o' key
			printf("\nPB5 LED turned ON\r\n");
			break;

			case 'd':			//duty cycle change for 'd' key
			printf("\nEnter a 0-100 decimal value to modify PB3 duty cycle \n");
			scanf("%d", &dutycycle);
			//bounds check for cuty cycle
			while((dutycycle > 100) || (dutycycle < 0)){
				printf("Re-enter duty cycle between 0-100\n");
				scanf("%d", &dutycycle);
			}
			printf("\nPB3 LED duty cycle has been changed to %d percent \n",
dutycycle);
			int period3 = 15624;					//preset period
			int duty3 = (dutycycle/100.0) * period3; //calculate duty cycle
			OCR3B = duty3;				//compare A value set for DC
			OCR3A = period3;		//set OCR3A top value
			TCCR3A = 0x03;				//set normal mode
			TCCR3B = 0x1D;				//set 1024 prescale, mode 15
			TIMSK3 = 0x05;
			break;
			case 'i':				//modify period for 'i' key input
			printf("\nEnter frequency value for PB2 between 2-100Hz\n");
			scanf("%d", &frequency);	//read input for frequency
			//bounds check for frequency input
			while((frequency > 100) || (frequency < 2)){
				printf("Please re-enter Frequency between 2-100Hz\n");
```

```
                        scanf("%d", &frequency);
                    }
                    printf("\nPB2 LED frequency has been changed to %d Hz\n",
frequency);
                    //calculating period
                    int period2 = ((16000000.0 / 1024.0) * (1.0 / frequency) - 1.0);
                    int duty2 = .5 * period2; //calculate duty cycle
                    OCR1B = duty2;              //compare A value set for DC
                    ICR1 = period2;             //set ICR1 top value
                    TCCR1A = 0xA2;              //set PWM mode 14, OCA & OCB non-invert
                    TCCR1B = 0x1D;              //set PWM mode 14, 1024 prescale
                    break;
                    case 'O':
                    PORTB &= ~(1<<PB5);         //turn PB5 LED OFF on 'O' key input
                    printf("\nPB5 LED turned OFF\r\n");
                    break;
                    default:
                    //printf("\nCommand not recognized\r\n");
                    break;
            }
        }
        return 0;
}
//toggle PB3 on compare B
ISR(TIMER3_COMPB_vect){
        PORTB &= ~(1<<PB3);
}
//toggle PB3 on overflow
ISR(TIMER3_OVF_vect){
        PORTB |= (1<<PB3);
}
```

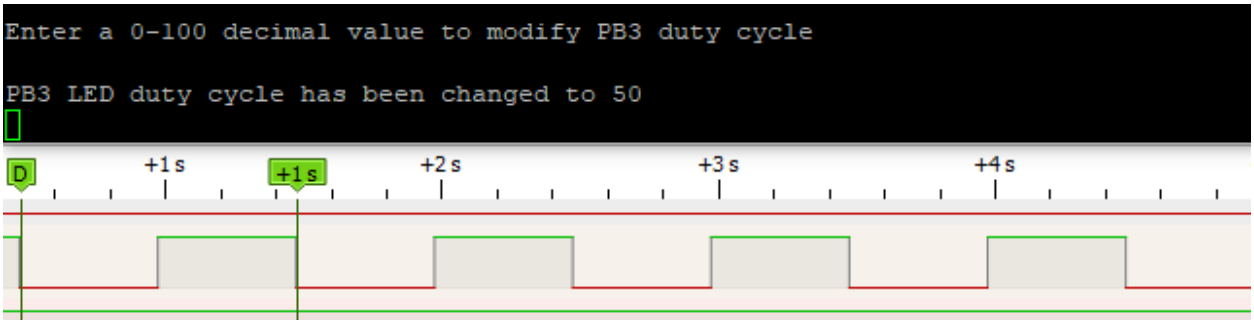


Midterm I Circuit

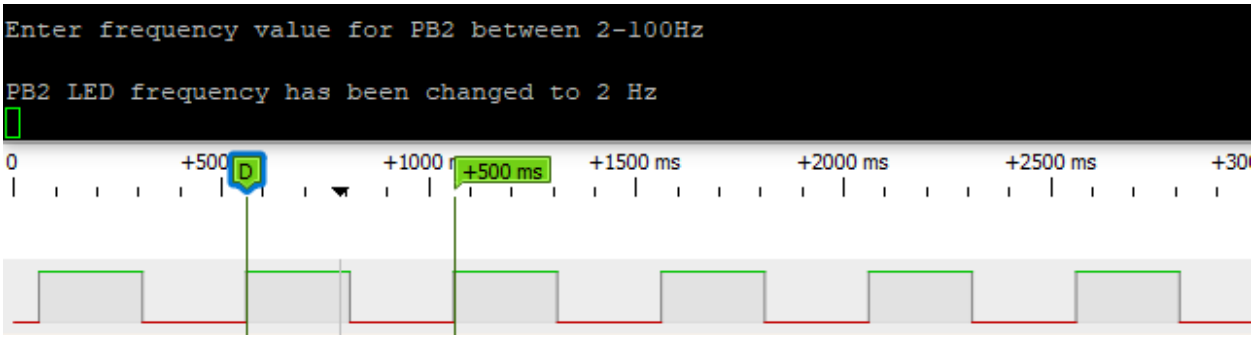
PuTTY Terminal / Waveform Outputs:



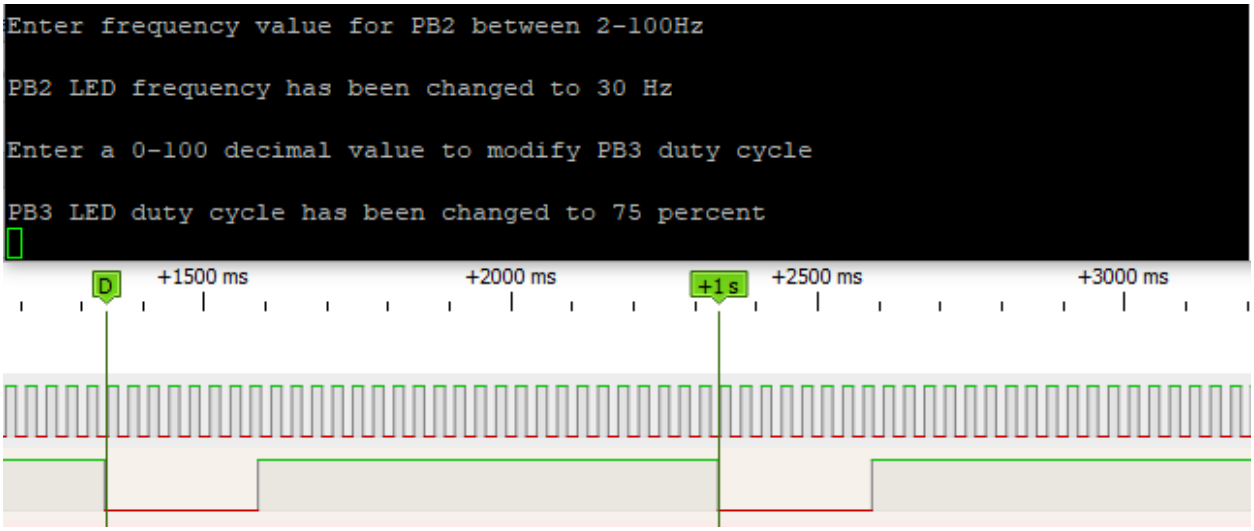
Initial Startup help screen with 't' and 'f' key inputs to display temperatures as well as the functionality of 'o' and 'O' to control LED at PB5



Duty Cycle of PB3 changed to 50%



PB2 LED frequency changed to 2Hz showing 500ms period



Both PB2 and PB3 LEDs operating at 30Hz and 75% duty cycle respectively