CPE301 – SPRING 2019

Design Assignment 5

Student Name: Yannick Gael Kengne Tatcha
Student #: 5003294512
Student Email: kengneta@unlv.nevada.edu
Primary Github address: https://github.com/Vasty1995/submission_da
Directory: DA5

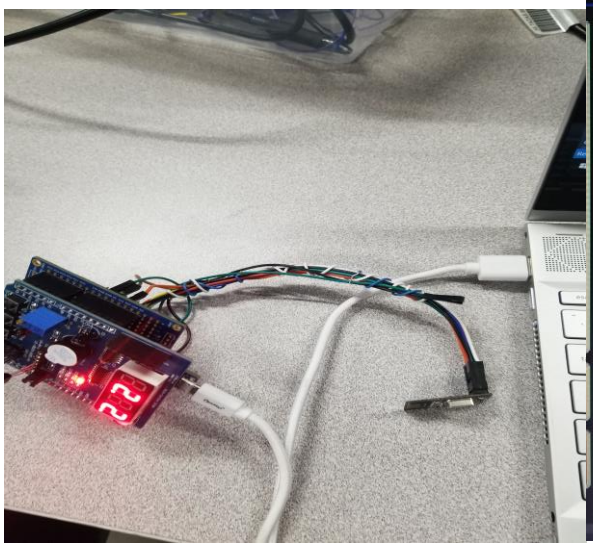
1. INITIAL/MODIFIED/DEVELOPED CODE

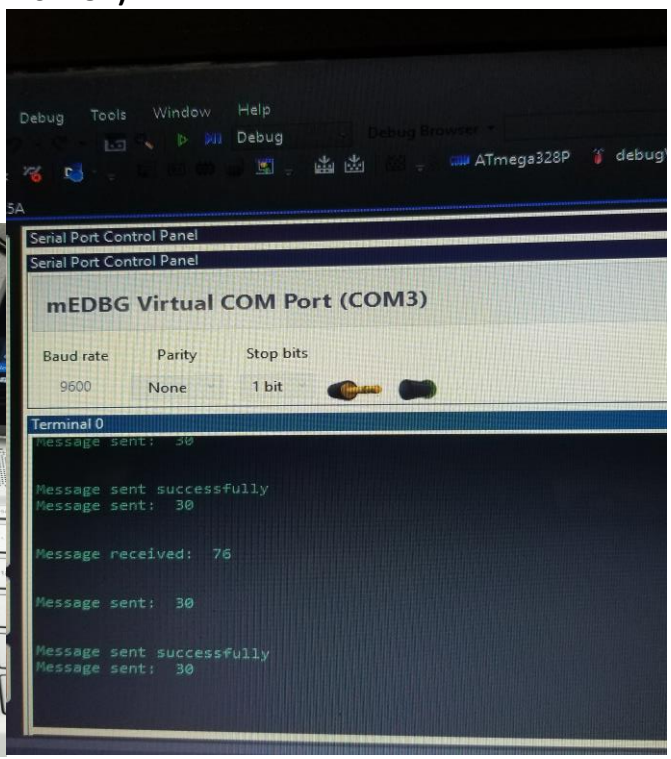
```
#ifndef F_CPU
#define F_CPU 16000000UL
#endif

#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <stdbool.h>
#include <stdio.h>
#include <string.h>

//     Set up UART for printf();
#ifndef BAUD
#define BAUD 9600
#endif
#include "inc/STDIO_UART.h"

//     Include nRF24L01+ library
#include "inc/nrf24l01.h"
#include "inc/nrf24l01-mnemonics.h"
#include "inc/spi.h"
void print_config(void);
void read_adc(void);        // Function Declarations
void adc_init(void);
volatile unsigned int adc_temp;
char outs[20];

//     Used in IRQ ISR
volatile bool message_received = false;
volatile bool status = false;

int main(void)
{
       //     Set cliche message to send (message cannot exceed 32 characters)
       char tx_message[32];                    // Define string array
       strcpy(tx_message,"hello i am cody");     // Copy string into array

       //     Initialize UART
       uart_init();
       adc_init();
       //     Initialize nRF24L01+ and print configuration info
       nrf24_init();
```

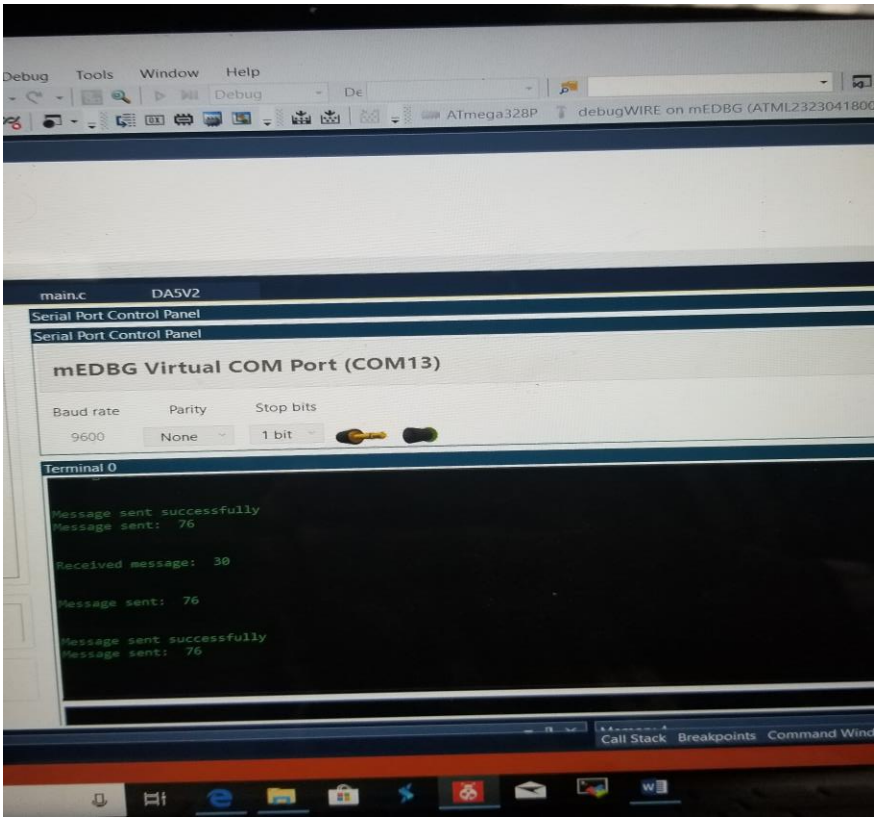
```
	print_config();

	//	Start listening to incoming messages
	nrf24_start_listening();

	while (1)
	{
		read_adc();
		adc_temp = (adc_temp*500)/1023 + 20;
		snprintf(outs,sizeof(outs),"%3d\r\n", adc_temp);  // print it
		strcpy(tx_message,outs);    // Copy string into array
		nrf24_send_message(tx_message);
		_delay_ms(1500);
		if (message_received)
		{
			//	Message received, print it
			message_received = false;
			printf("Received message: %s\n",nrf24_read_message());
			//	Send message as response
			_delay_ms(500);
			status = nrf24_send_message(tx_message);
			if (status == true) printf("Message sent successfully\n");
		}
	}
}

//	Interrupt on IRQ pin
ISR(INT0_vect)
{
	message_received = true;
}

void print_config(void)
{
	uint8_t data;
	printf("Startup successful\n\n nRF24L01+ configured as:\n");
	printf("-----------------------------------------\n");
	nrf24_read(CONFIG,&data,1);
	printf("CONFIG              0x%02X\n",data);
	nrf24_read(EN_AA,&data,1);
	printf("EN_AA               0x%02X\n",data);
	nrf24_read(EN_RXADDR,&data,1);
	printf("EN_RXADDR           0x%02X\n",data);
	nrf24_read(SETUP_RETR,&data,1);
	printf("SETUP_RETR          0x%02X\n",data);
	nrf24_read(RF_CH,&data,1);
	printf("RF_CH               0x%02X\n",data);
	nrf24_read(RF_SETUP,&data,1);
	printf("RF_SETUP            0x%02X\n",data);
	nrf24_read(STATUS,&data,1);
	printf("STATUS              0x%02X\n",data);
	nrf24_read(FEATURE,&data,1);
	printf("FEATURE             0x%02X\n",data);
	printf("-----------------------------------------\n\n");
}

/* INIT ADC */
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)
	(0<<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);
}
/* READ ADC PINS */
void read_adc(void)
{
	unsigned char i = 4;
	adc_temp = 0;
	while (i--)
	{
		ADCSRA |= (1<<ADSC);
		while(ADCSRA & (1<<ADSC));
		adc_temp+= ADC;
		_delay_ms(50);
	}
	adc_temp = adc_temp / 4;  // Average a few samples
}
```

2. SCREENSHOT OF EACH DEMO (BOARD SETUP)







3. VIDEO LINKS OF EACH DEMO

https://youtu.be/IiTHcic-boU

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"*This assignment submission is my own, original work*".
Yannick Gael Kengne Tatcha