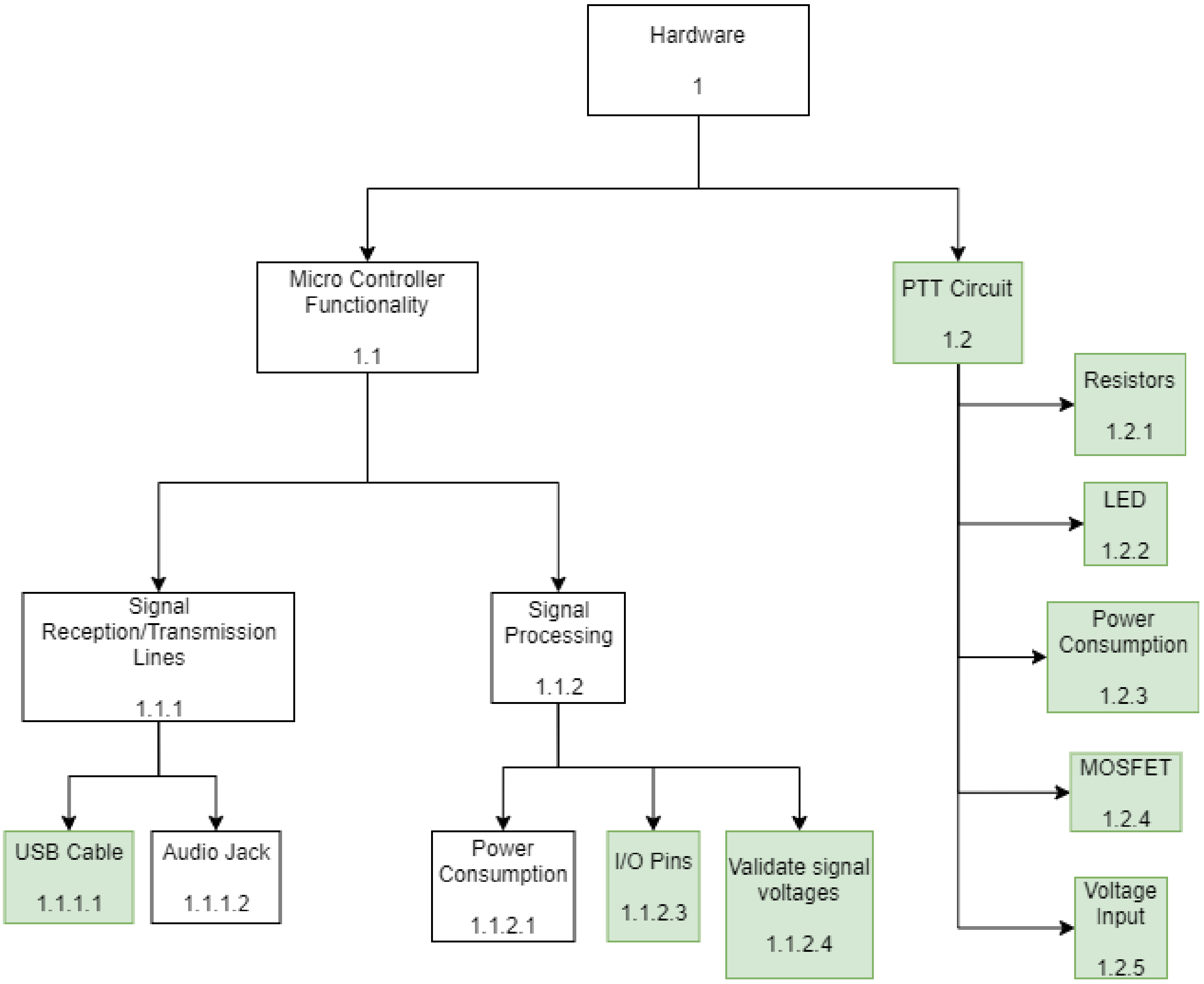
MCU TNC
Deliverable 5

David Cain, Kobe Keopraseuth, and Kaleb Leon

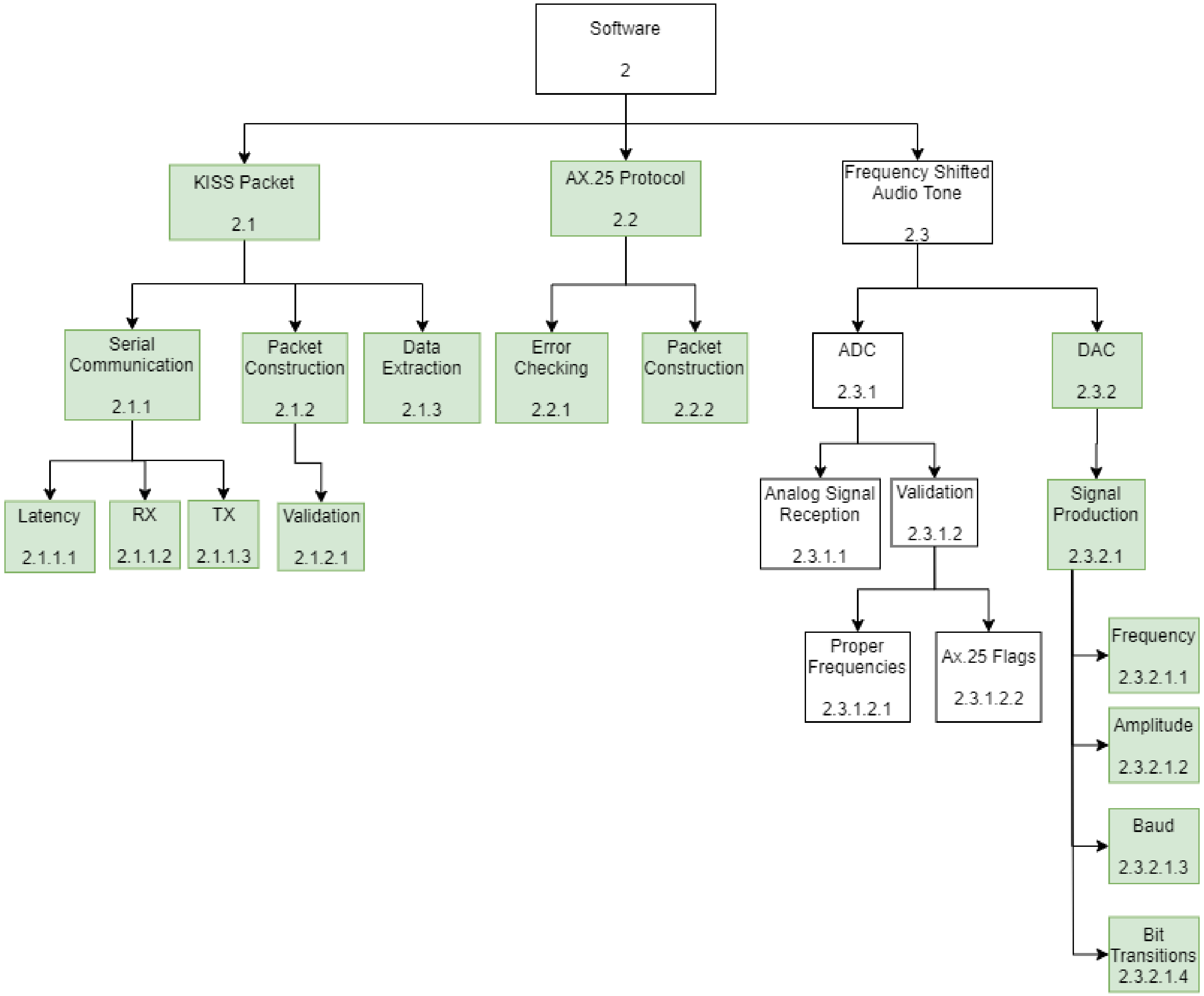
Testing Tree Progress

Hardware



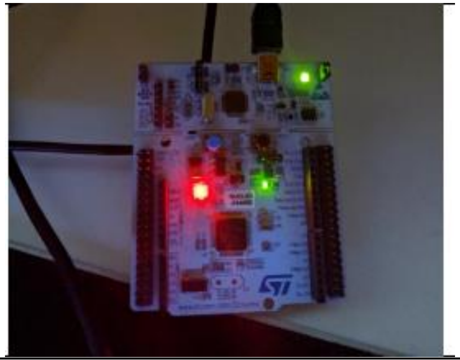
Hardware
1
Micro Controller Functionality
1.1
PTT Circuit
1.2
Resistors
1.2.1
LED
1.2.2
Power Consumption
1.2.3
MOSFET
1.2.4
Voltage Input
1.2.5
Signal Reception/Transmission Lines
1.1.1
Signal Processing
1.1.2
USB Cable
1.1.1.1
Audio Jack
1.1.1.2
Power Consumption
1.1.2.1
I/O Pins
1.1.2.3
Validate signal voltages
1.1.2.4

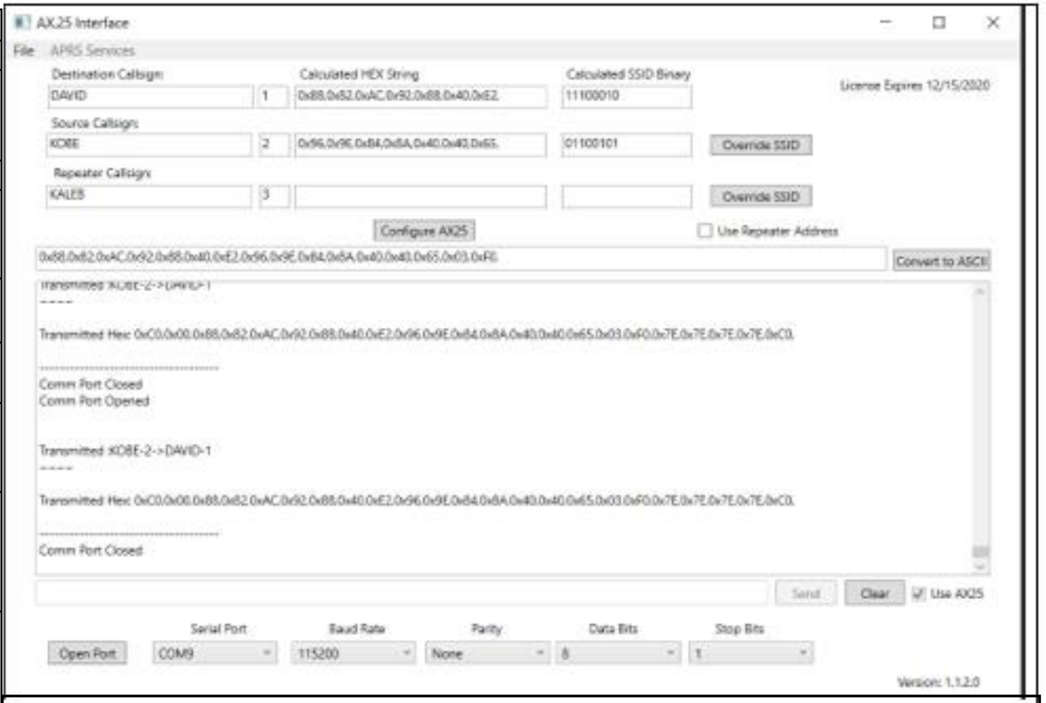
Software



Software
2
KISS Packet
2.1
AX.25 Protocol
2.2
Frequency Shifted Audio Tone
2.3
Serial Communication
2.1.1
Packet Construction
2.1.2
Data Extraction
2.1.3
Error Checking
2.2.1
Packet Construction
2.2.2
ADC
2.3.1
DAC
2.3.2
Latency
2.1.1.1
RX
2.1.1.2
TX
2.1.1.3
Validation
2.1.2.1
Analog Signal Reception
2.3.1.1
Validation
2.3.1.2
Signal Production
2.3.2.1
Proper Frequencies
2.3.1.2.1
Ax.25 Flags
2.3.1.2.2
Frequency
2.3.2.1.1
Amplitude
2.3.2.1.2
Baud
2.3.2.1.3
Bit Transitions
2.3.2.1.4

Testing Forms

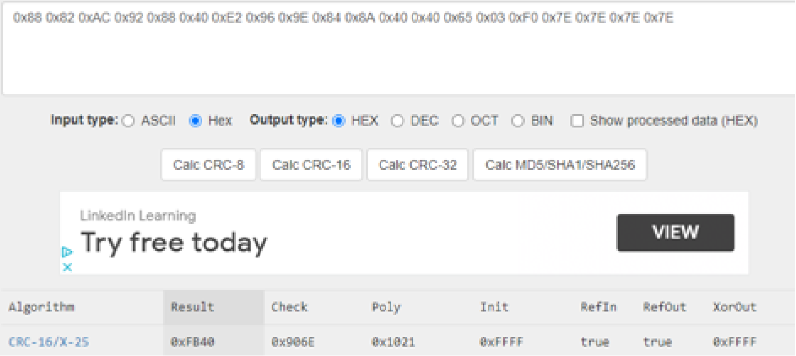
TNC Testing Form (REV1)	
Leaf on the Tree	AX.25 Protocol
Device Under Test (Testing Tree Number):	2.2.1
Date:	11/1/20
Person(s) Conducting Experiment:	Kobe Keopraseuth
Signature:	
Experiment Purpose:	The purpose of this experiment of this experiment is to verify that our microcontroller can take in a KISS packet and format the AX.25 Packet Correctly.
Experiment Procedure:	Take in a KISS packet from computer and display the fields of the AX.25 packet. We will also show the calculated crc to show that the KISS packet for properly extracted.
Equipment Settings / Software Settings (w Revision):	Use Rizwan's software to send a KISS packet and display the AX.25 packet on serial monitor.
Testing Diagram / Picture:	
Data Points:	



Rizwan's software for transmitting KISS packets

```
Start flag       = 1  1  0  0  0  0  0  0
Address Field 1  = 1  0  0  0  1  0  0  0
Address Field 2  = 1  0  0  0  0  0  1  0
Address Field 3  = 1  0  1  0  1  1  0  0
Address Field 4  = 1  0  0  1  0  0  1  0
Address Field 5  = 1  0  0  0  1  0  0  0
Address Field 6  = 0  1  0  0  0  0  0  0
Address Field 7  = 1  1  1  0  0  0  1  0
Address Field 8  = 1  0  0  1  0  1  1  0
Address Field 9  = 1  0  0  1  1  1  1  0
Address Field 10 = 1  0  0  0  0  1  0  0
Address Field 11 = 1  0  0  0  1  0  1  0
Address Field 12 = 0  1  0  0  0  0  0  0
Address Field 13 = 0  1  0  0  0  0  0  0
Address Field 14 = 0  1  1  0  0  1  0  1
Control Field    = 0  0  0  0  0  0  1  1
PID Field        = 1  1  1  1  0  0  0  0
Info Field 1     = 0  1  1  1  1  1  1  0
Info Field 2     = 0  1  1  1  1  1  1  0
Info Field 3     = 0  1  1  1  1  1  1  0
Info Field 4     = 0  1  1  1  1  1  1  0
Stop flag        = 1  1  0  0  0  0  0  0
```

Bitstream output of received KISS packet



<u>Online crc calculation for the KISS packet</u>

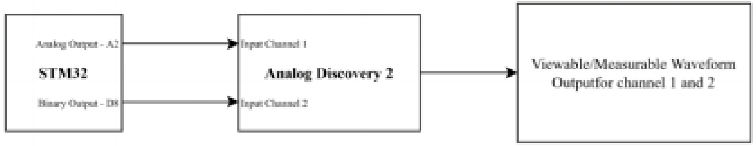
```
Convert CRC to FCS (hex) = fb40
```

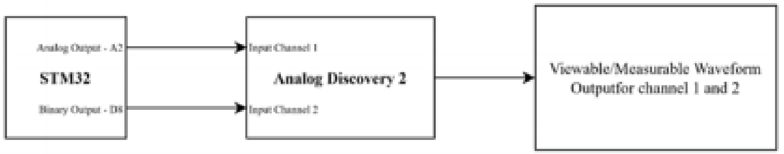
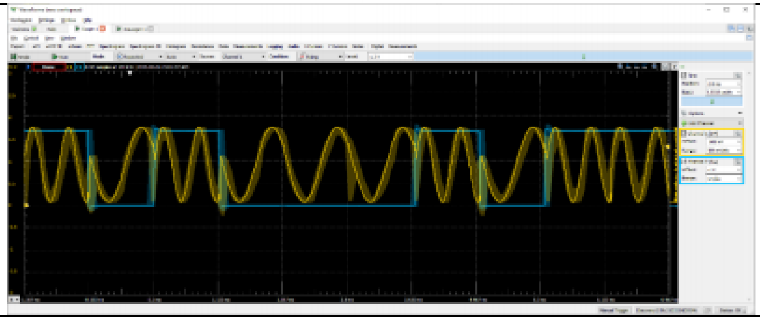
<u>Microcontroller’s crc calculation for KISS packet</u>

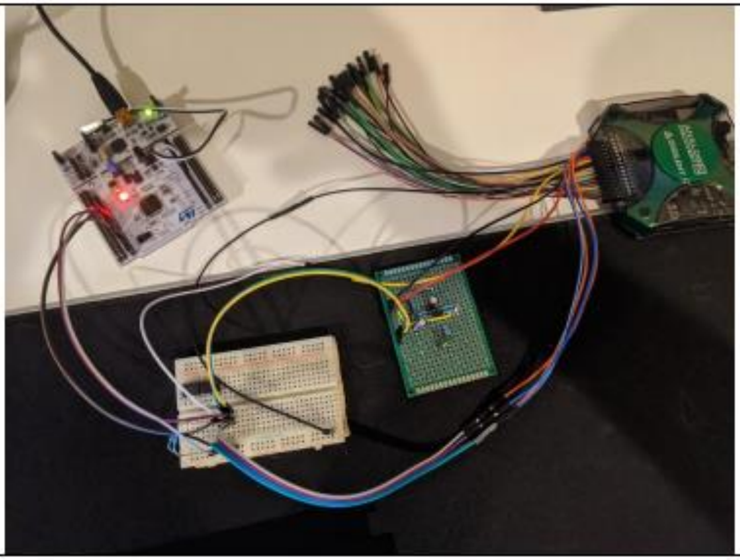
```
Printing AX25_PACKET being sent to radio
AX25 FLAG =  0  1  1  1  1  1  1  0
Address Field 1 = 1  0  1  0  0  1  1  0
Address Field 2 = 0  0  0  0  0  0  1  0
Address Field 3 = 0  0  0  0  0  0  1  0
Address Field 4 = 0  1  0  1  0  0  0  1
Address Field 5 = 0  0  1  0  0  0  0  1
Address Field 6 = 0  1  1  1  1  0  0  1
Address Field 7 = 0  1  1  0  1  0  0  1
Address Field 8 = 0  1  0  0  0  1  1  1
Address Field 9 = 0  0  0  0  0  0  1  0
Address Field 10 = 0  0  0  1  0  0  0  1
Address Field 11 = 0  1  0  0  1  0  0  1
Address Field 12 = 0  0  1  1  0  1  0  1
Address Field 13 = 0  1  0  0  0  0  0  1
Address Field 14 = 0  0  0  1  0  0  0  1
Address Field extra =
Control Field   = 1  1  0  0  0  0  0  0
PID Field       = 0  0  0  0  1  1  1  1
Info Field =  0  1  1  1  1  1  0  1  0  0  1  1  1  1  1  0  1  0  0  1  1  1  1  1  0  1  0  0  1  1  1  1  1  0  1  0
FCS Field     = 1  1  1  1  1  0  0  1  1  0  1  0  0  0  0  0  0  0
AX25 FLAG =  0  1  1  1  1  1  1  0
```

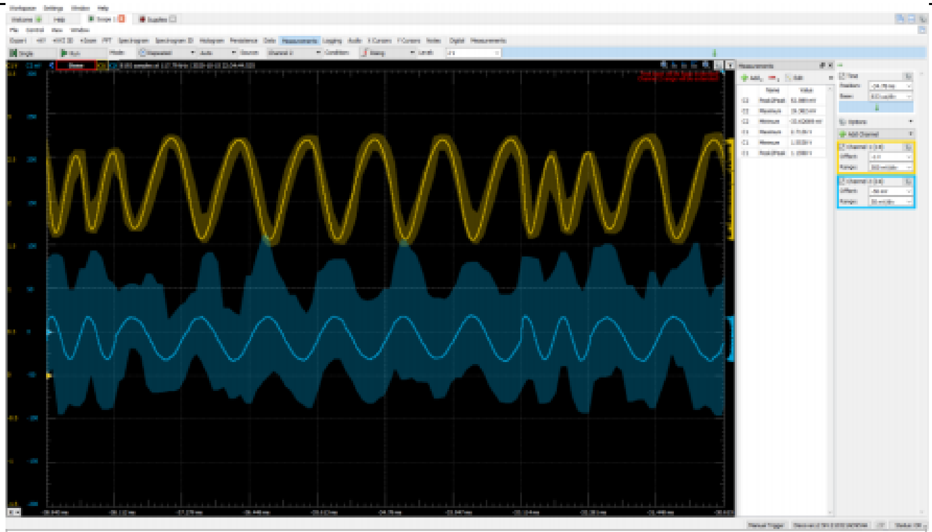
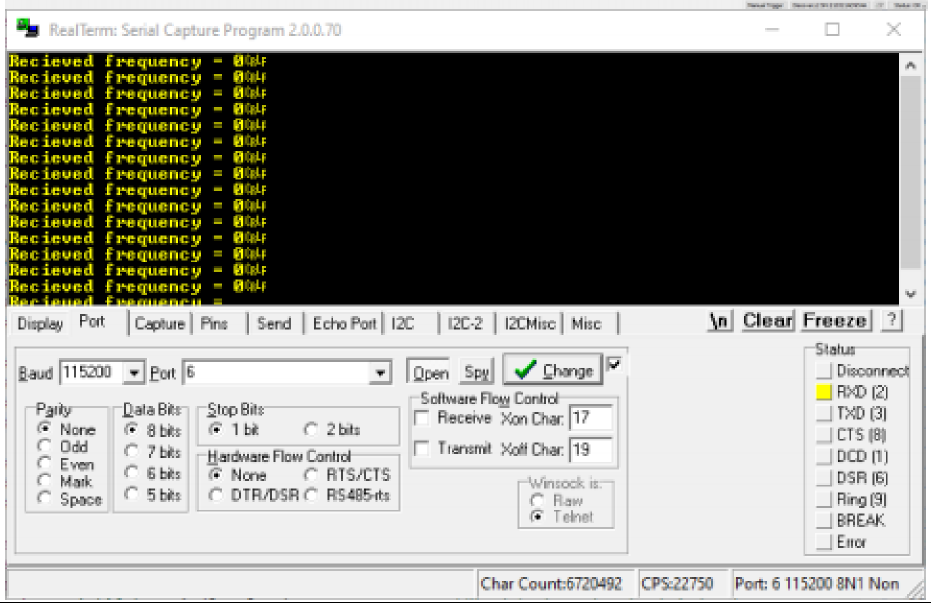
<u>AX.25’s bit sequence when sent to radio</u>

Pass / Fail:	Pass
Interpreted Notes:	Our microcontroller can properly extract a KISS packet and format it into AX.25, along with correctly calculating the crc for the FCS field. It is also able to place the bits to be sent to over radio in the correct order.
Recommendations for Modifications:	None

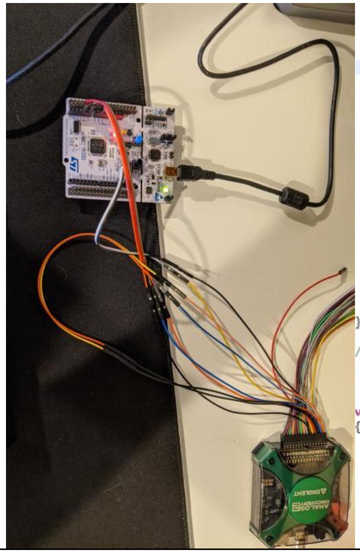
TNC Testing Form (REV1)	
Leaf on the Tree	Baud
Device Under Test (Testing Tree Number):	2.3.2.1.3.1
Date:	10/4/2020
Person(s) Conducting Experiment:	David Cain
Signature:	
Experiment Purpose:	The purpose of this experiment is to measure and ensure the number of signaling events per second (or baud rate) is correctly established as 1200Hz
Experiment Procedure:	To verify the baud rate, a diagnostic signal will be enabled in software to output the current transmission bit value represented in binary. This binary wave form can easily have baud rate measured.
Equipment Settings / Software Settings (w Revision):	Analog Discovery 2 input channel 1 and 2 will be connected to the STM32 output pins D8(PA9) and A2(PA4)
Testing Diagram / Picture:	
Data Points:	
Pass / Fail:	Pass
Interpreted Notes:	Waveform is sustaining a baud rate of 1200Hz. This was tested with multiple wave forms but easily viewed with alternating bit pattern.
Recommendations for Modifications:	None

TNC Testing Form (REV1)	
Leaf on the Tree	Bit Transitions
Device Under Test (Testing Tree Number):	2.3.2.1.4
Date:	10/31/2020
Person(s) Conducting Experiment:	David Cain
Signature:	
Experiment Purpose:	The purpose of this experiment is to ensure that the waveforms of our output do not suffer due to bit transitions.
Experiment Procedure:	Force the TNC into a debugging broadcast mode, then use the Digilent discovery 2 to measure the waveform frequency at many points.
Equipment Settings / Software Settings (w Revision):	The Digilent will be set to record the waveform and an optical inspection will be used.
Testing Diagram / Picture:	
Data Points:	
Pass / Fail:	Fail
Interpreted Notes:	There is an obvious phase shift when the bits are transitioning.
Recommendations for Modifications:	Correct code that generates the output to calculate the next expected starting point of each bit.

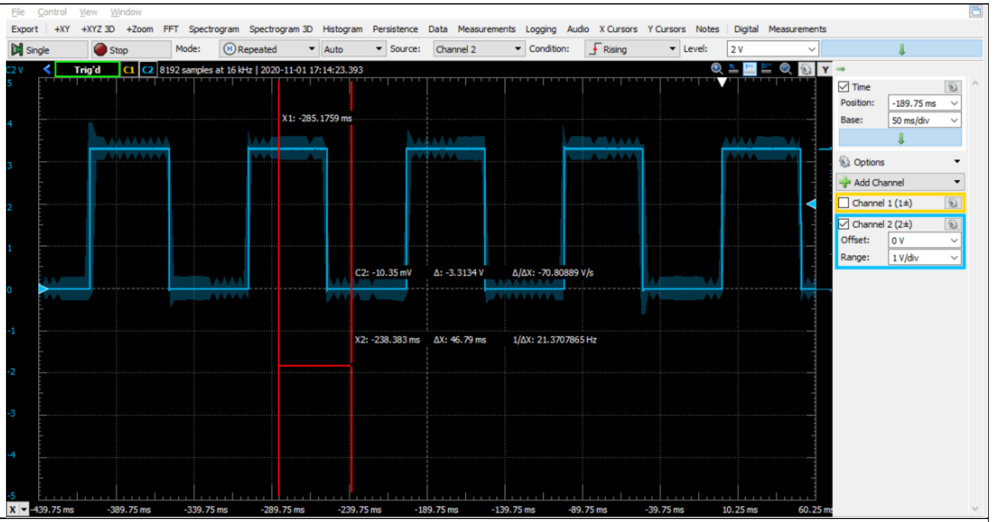
TNC Testing Form (REV1)	
Leaf on the Tree	Amplifier
Device Under Test (Testing Tree Number):	1.3.1.1
Date:	10/15/2020
Person(s) Conducting Experiment:	David Cain, Kobe Keopraseuth
Signature:	
Experiment Purpose:	The purpose of this experiment is to ensure the amplifier output can trigger the external interrupt on the STM32 for reading incoming AFSK waveform frequency components.
Experiment Procedure:	Using waveform generator from Analog Discovery 2, input a 50mV ptp AFSK waveform to amplifier circuit, checking readings reported by microcontroller on serial port.
Equipment Settings / Software Settings (w Revision):	Micro controller is set to receiving mode Using WaveForms software(version 3.12.2), output generated AFSK signal
Testing Diagram / Picture:	

Data Points:	 
Pass / Fail:	Fail
Interpreted Notes:	Frequency output remains zero due to gain not being high enough to trigger interrupt.
Recommendations for Modifications:	Modify amplifier circuit to increase gain.

TNC Testing Form (REV1)	
Leaf on the Tree	Validate Signal Voltages
Device Under Test (Testing Tree Number):	1.1.2.4
Date:	11/1/20
Person(s) Conducting Experiment:	Kobe Keopraseuth, Kaleb Leon, David Cain
Signature:	
Experiment Purpose:	The purpose of this experiment is to test to make sure the voltages are accurate for our analog signal on the microcontroller.
Experiment Procedure:	We will send a square wave analog signal from the Nucleo, that is generated by waiting a number of milliseconds and change the GPIO pin to output a 0 or 1. We set the amplitude to a specific value to be measured in our scope analog discovery.
Equipment Settings / Software Settings (w Revision):	We use the nucleo board with the code shown below to produce the signal. Then we read the signal using our analog discovery shown on channel 2 in the data points.
Testing Diagram / Picture:	



```
/* Initialize all configured peripherals */
MX_GPIO_Init();
MX_USART2_UART_Init();
MX_TIM3_Init();
/* USER CODE BEGIN 2 */
HAL_TIM_IC_Start_IT(&htim3, TIM_CHANNEL_1);
uint32_t captureVal;
uint32_t time = HAL_GetTick();
uint32_t millis_wait = 50;
/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */

    if(HAL_GetTick() - time > millis_wait){
        HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_0);
        time = HAL_GetTick();
    }
}
/* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Configure the main internal regulator output vol
  */
  __HAL_RCC_PWR_CLK_ENABLE();
  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAG
```

Data Points:	
Pass / Fail:	Pass
Interpreted Notes:	The signal being displayed is correct because the amplitude of the wave meets the input set in the Nucleo code.
Recommendations for Modifications:	N/A

Changes Made to Testing Plan

- Removing Audio Input Circuit (1.3)
  - Amplifier (1.3.1) also removed
  - LP Filter (1.3.2) also removed
- Testing clock recovery and data decoding with new board provided by the mentors
- Removed RS232 Jack (1.1.1.3)
- Removed Validation (2.2.2.1) under packet construction because it was redundant

Change Order Forms

Change Order Form (COF) Rev 1	
Date	10/17/2020
Subsystem	1.3 Audio Input Circuit (Audio Input Circuit)
Component	1.3.1 Receiving Circuit (Amplifier)
Change	Need to change to resistor values and need to supply a lower DC voltage to the amplifier.
Reasoning	Before the amplifier was not connected to a low pass filter, but because of adding the low pass filter the amplifier's gain has decrease. The gain needs to be high enough for signals, coming from the radio, to be read by the STM32 microcontroller. The microcontroller reads in 70 % of 3.3 V as a high input and 30 % of 3.3 V as a low input. Unfortunately, after testing the signal, outputted from the amplifier, does not reach those amplitudes.
Rationale	A way to fix this issue is to perform a small signal analysis on the current circuit and solve for the necessary resistor values and DC supply voltage, knowing the desired gain and DC offset to achieve.
Mod. Details	Although we yet to test the circuit in the lab, after solving for the necessary resistors and correct DC supply voltage, the circuit was able to output a sine wave that reaches the necessary voltages through simulation, using LTspice.

Change Order Form (COF) Rev 1	
Date	11/1/2020
Subsystem	Signal Reception/Transmission Lines 1.1.1
Component	RS-232 port
Change	Removal from design
Reasoning	Design will not support this port anymore.
Rationale	Due to the time constraints of the semester, we will not have time to implement a RS-232 jack.
Mod. Details	Removal

Change Order Form (COF) Rev 1	
Date	11/1/2020
Subsystem	1.3 (Audio Input Circuit)
Component	1.3 (Audio Input Circuit)
Change	Will remove audio input circuit all together.
Reasoning	After consulting with our mentors, we realized that the process of recovering the clock and data from an AFSK signal using solely digital logic was quite difficult and they overlooked this when assigning the task.
Rationale	Nolan is providing a modem board that will simply output binary values for us to interpret the data and clock from. We will complete our project utilizing this board but will lay a foundation for the C.A.P.E. research group to build on to achieve a minimal hardware solution for recovering the data in the future.
Mod. Details	Remove amplifier and filtering circuit from the design and will now utilize a modem board provided by Nolan.

Tests to be done by Project End

- Analog to Digital Conversion aka Receiving
  - Needs additional tweaking to have initial functionality
- Audio tone RX and TX full subsystem test
- Full software test
- Micro Controller functionality testing
  - Power Consumption (1.1.2.1) in Signal Processing
  - Audio Jack (1.1.1.2) in transmission lines
- Full Final Design Testing Coming soon

Questions?