[[1]](#footnote-1)

MCU TNC Design

Kaleb P Leon, *Member, IEEE*, Kobe Keopraseuth, Member, IEEE and David Cain, Member, IEEE

*Abstract*— *Abstract*— Developed by Kaleb Leon, Kobe Keopraseuth, and David Cain with All Rights Reserved. The objective of this paper is to document and describe the design process of developing a software based Terminal Node Controller (TNC). The TNC will be able to perform all the basic functions of a hardware TNC but with little to no hardware necessary. It will be capable of receiving an audio tone FM modulated signal, convert it into binary, gather the payload, check for bit errors and send the payload to the PC via KISS (keep it simple, stupid) mode. I will also be able to receive a packet via KISS from the PC, follow the AX.25 protocol to form it into a valid data packet for radio communication, translate it into an FM modulated audio signal tone and send the tone to the radio. This paper has multiple parts that show the design process: the scope of work, objective tree, feasibility analysis, functional specifications and design alternatives.

*Index Terms*— controller, payload, packet, modulation, radio communication

# INTRODUCTION

T

HE MCU TNC Design team is in the process of designing a software based TNC. This project’s purpose is to make an easy to use and more compact version of a TNC. Hardware TNCs are usually very bulky electronics and come with a hefty price tag. With the use of a microcontroller, the design of a TNC can be condensed down to around four inch surface area at the price of a standard microcontroller. The team will implement code in C on a microcontroller that will represent all the analog functions of a hardware TNC in the form of software. The software’s job will be to process all the incoming data into packet form suitable for the PC when receiving and the radio when transmitting. When in transmitting mode, the microcontroller will receive a KISS packet from the PC and translate the payload. It will then take the payload and transfer it into the AX.25 format using bit stuffing techniques which will then be modulated onto a signal in the form of frequency modulation. The frequency modulation will be done with different frequencies representing a zero or a one. When in receiving mode, the microcontroller will receive a FM modulated audio tone and translate it down into a KISS packet to be sent to the PC. These processes done normally in the form of hardware can be done all in code. The future sections of this paper include research done that has relevance to the protocols and systems that will be used in this design and a detailed project analysis which will contain a feasibility analysis in addition to design alternatives and tradeoffs. Following the main sections of this paper, are a set of appendices A, B, and C. Appendix A contains a more streamline scope of work as well as functional requirements. It will also contain a level 0 diagram and an objective tree to visually describe our designs basic functionality. Appendix B will contain a feasibility analysis in which we discuss whether the project is technologically, timely, and economically feasible in addition to our plan for the incoming year. Appendix C will show our analysis on our alternatives and tradeoffs to the systems and protocols used in this design.

# Research of work done by others

In the world of radio communication, there are many projects involving TNC design but most of them involve hardware redesign or optimization. Our design takes the analog systems of the previous age and takes them into the digital modern age of software. However, there has been one other project that has accomplished this task of a software TNC and they are call their software TNC Direwolf. We hope that our research and design will help improve on their design and make it more compact and efficient.

## KISS Mode

One of the first obstacles to overcome is how to send data from the PC to the TNC. According to the mentors of this project, the most common protocol used to communicate is the KISS protocol. According to Chepponis and Karn presentation [2], the KISS protocol provides direct computer to TNC communication. It provides the host software of the PC with the ability to control all the TNC functions. It allows the pc to send a packet with a payload controlled by stop and start flags. These flags are represented by a hex value of C0. The data in the payload can be as large as 1024 bytes but this size can also increase based on the TNC’s specs. Our project will use this protocol to send and receive data from the PC. However, we will be converting our data into the KISS form using software on the microcontroller.

## HDLC (High-level Data Link Control) Protocol

After the KISS packet is received by the TNC it will have to translate it into another packet format to prepare it form transmission and this was the next obstacle. This protocol, HDLC, is a data link layer described by Estevez [3] in combination with AX.25 to solve this packet formation that works mainly with KISS. The HDLC protocol is made from the payload sent from KISS, start and stop flags, a control frame, and an error checking frame. Estevez states that HDLC is NRZ-I encoded meaning that a logical bit 0 is marked by a change in state and a logical bit 1 is marked by no change in state. Similar to KISS, it also has an end flag or maker which is represented by a Hex 7E. To identify the difference between these end flags and the rest of the message HDLC forbids more than 5 consecutive 1s. To do this they use a technique called bit stuffing which means when they see 5 consecutive 1s a 0 bit is added in and this zero is ignored. The last main function that Estevez mentions is that it has a form of error checking. It has a frame for a 16 bit check sum which is compared at the transmitter and receiver. This checksum is computed using CRC-16CCITT. If this frame does not match at the receiver the packet is dropped and a retransmission is requested.

## AX.25 Protocol

Since the framing and error checking is done by HDLC the AX.25 frames do not include the checksum. When transmitting these packets a source and destination as well as a control frame is necessary. According to the spec sheet of AX.25 by Beech, Nielson, and Taylor [1], it handles these frames. In reference to the address frame, it contains the source address, destination address and one or more repeater addresses up to eight. The control frame describes what the data is: information, numbered, unnumbered, or supervisory.

## AFSK (Audio Frequency Shift Keying)

Now that these frames are formed it is important to know how the packets will be translated into audio signals and transmitted over the air. Bits are to be represented by two separate frequencies. Based on Estevez’s documentation [2], FM AFSK normally uses a baud rate of 1200 on a frequency shifts between tones of 1200 Hz and 2200 Hz. To translate these signals when receiving, frequency counting could be used as well as a comparator or Schmitt trigger.

# Project Analysis

## Project Feasibility

The scope of work and required functional specifications are feasible.

The main goal of the project is aimed at replacing the hardware that was originally developed in the early 1980s, this means the technical requirements are very minimal. In addition, a previous design has been successfully achieved by a project called Direwolf meaning the design is feasible on a technological standpoint. Also, the hardware and protocols we plan on using is well documented and already implemented in hardware so it is very feasible to make the shift to software related logic.

Given the simplicity of packet management required to have a functioning TNC it seems it will be feasible to have close to a working design in the first half year. In addition, the second half year could be used to develop our own more optimized microcontroller board as well as add some features. According to the Gnatt Chat referenced in Appendix B, our planned schedule projects the projects feasibility in the time span of one year.

In the terms of cost, the design required little to no physical devices or hardware so the price will be relatively low considering most of the work will be free software. In addition, we currently possess many of the necessary components personally and from our mentors so the total projected cost is less than one hundred dollars making the project economically feasible.

## Alternatives and Tradeoffs Considerations

With a project based mostly around software there are small amounts of alternatives and tradeoffs to consider. In Appendix C, Tables are shown referencing our comparison of different alternatives to many parts of our design and our decisions. The alternatives are mostly based around specs given to us by our mentors and we chose the ones that would best meet those specs. Our first decision to make was on microcontrollers. This would play a major role in our overall design and would have to meet many of our required specs. Our three choices were the STM32L4433, the Teensey 4.0 board, and an Arduino Mega. The next decision was how would we implement and build our PTT (push to talk). There are a number of ways this logic could be implemented like using an Inverter Circuit, a P-Channel MOSFET, or a Comparator IC. Also, we had to make a decision on what we would use for our signal conversion methods when receiving and transmitting. For receiving we will need to take in an analog signal and convert it to digital binary so we looked into different analog to digital converters such as: using Fourier analysis, Schmitt Trigger, or Zero Crossing. Lastly, we had to look into transmitting one we have a binary packet how to translate that into FM audio tone. This can be accomplished using the STM’s built in digital to analog controller, a resistor switching network, or a Digital to analog integrated circuit. Most of the decisions were made based on whether they were already on the micro controller and power consumption. Ultimately, for microcontroller choice we chose STM32L4433 to use due to its already integrated DACs and ADCs as well as its CRC calculation unit which can detect bit errors. However, the downside is we are not familiar with C and will have to learn. The STM32 fills all the alternatives so it makes it the obvious choice.

References

1. Beech, W. A., Nielsen, D. E., & Taylor, J. AX.25 Link Access Protocol for Amateur Packet Radio. (1997). PDF. Tucson .
2. Chepponis, Mike, and Phil Karn. “The KISS TNC: A Simple Host-to-TNC-Communications-Protocol,”January1987. http://www.ax.25.net/kiss.aspx.
3. Destevez. “KISS, HDLC, AX.25 and Friends.” Daniel Estvez, May 6, 2018.https://destevez.net/2016/06/kiss-hdlc-ax-25-and-friends/.

**Kaleb P. Leon** was born in New Iberia, Louisiana in 1998. Attended New Iberia Senior High for high school and graduated Valedictorian. Currently a senior attending University of Louisiana at Lafayette concentrating in Computer Engineering and minoring Computer Science. Mr. Leon is also part of a team with a publication in on Cyber Physical Security of Electric Vehicles and was part of a team awarded Louisiana Clean Fuel Leader Award for most innovative project of the year in 2017. He anticipates graduating in Fall 2020 and is looking for government jobs in Cyber Security or Security Engineering.

**Kobe T. Keopraseuth** was born in New Iberia, Louisiana in 1998 and attended Westgate High School. After graduating he enrolled in the University of Louisiana at Lafayette in the fall of 2016. Since 2018, he’s been a member of IEEE. At the time of writing this paper he is a senior pursuing a Bachelor of Science in Electrical Engineering. He is expecting to graduate in the Fall of 2020.

**David L. Cain** was born in Hallsville, Texas in 1997 and attended New Iberia High School. Currently a senior at the University of Louisiana at Lafayette, majoring in Electrical Engineering and minoring in Computer Science. Mr. Cain has been a Member of IEEE since 2017. Expected to graduate in the Fall 2020.

**Appendix A**

## Scope of Work

The purpose of this project is to design a TNC that has its basic functions implemented mostly if not solely in the form of software. The TNC must be able to receive a FM audio tone signal and translate it into binary (HDLC packet). Then, take that binary and perform an error check to see whether the binary packet is valid. If the packet is valid it must extract the payload and form the packet into another format (KISS) to send to the PC. The TNC must also be able to receive a packet in the form of KISS from the PC and form the appropriate HDLC packet. It will also need to form the error checking bits and translate the whole package into a FM audio tone to be sent over the air. This must be done primarily using software with little to no hardware. This design should be easily used by any experience hardware TNC user and provide convenience in its size and lack of power consumption.

## Functional Requirements

Receiving

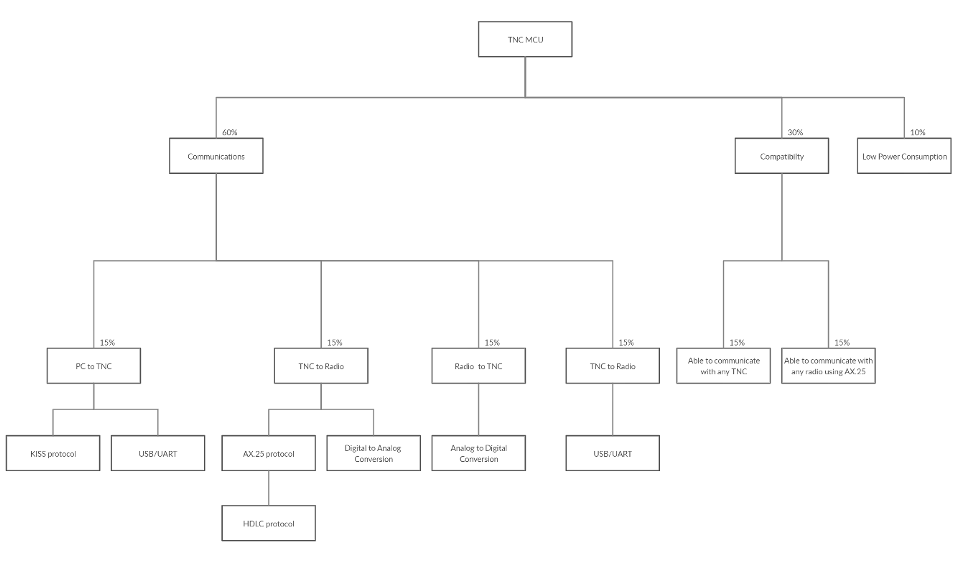
1. Receive Audio Tone Signal
2. Analog to digital convert into binary
3. Gather payload
4. Check for Errors
5. Send payload via KISS to PC

Transmitting

1. Receive KISS formatted data from PC
2. Follow AX.25 protocol to form data packet
3. Translate into analog audio tone signal
4. Send Audio tone to radio

## Objective Tree

We have created an objective tree shown in Figure A-1 below. This is used to visualize our main priorities in our design. Communications is massive part of this design. This is due to the TNC needing to communicate with two different systems each requiring their own packet format controlled by protocols. The compatibility is second but not to be neglected because the TNC needs to be able to communicate with other TNCs and radios to be properly functional. Lastly, due to specifications from the sponsors low power consumption is desired but not required for total TNC functionality.

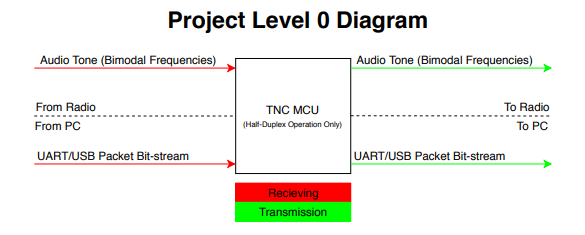


TNC to PC

## Figure A-1: Objective Tree MCU TNC

## Level 0 Functional Block Diagram

Shown below is our Level 0 Functional Block Diagram, Figure A-2. This diagram shows the inputs and outputs in our base design. As shown our design functions is half duplex meaning it can only receive at one time and transmit at a separate time. Never at the same time. When receiving different tasks are performed as well as when transmitting.



## Figure A-2: Level 0 Diagram

**Appendix B**

## Technological Analysis

With the goal of the project aiming to replace hardware that was originally developed in the early 1980s, this means the technical requirements are minimal. The project is stated in a fashion that allows us to get the hardware working well, then begin adding useful features and ensuring documentation is sufficient/competent for successive developers to branch off. The hardware for the project has only a few requirements:

* Interface with a terminal program via UART/USB
* Packetize data into AX.25 Protocol requirements
* Detect the frequency of an incoming audio signal
* Output an audio signal representing the packet bitstream
* Or output the packet bitstream via UART/USB

Tasks such as interfacing with UART or detecting the frequency of an audio signal not only have many options but also are very well documented. The process of packetizing the data will be an exercise of reading the documentation on how AX.25 packets are formed.

## Time Analysis

To assess the feasibility of time, a rough Gantt Chart has been attached to outline the goals of the team at various points of the design process. It is important to emphasize the design occurs over two semesters. For our project, the physical designing and testing will begin after a month of research on AX.25 and the accomplishments of other research groups. A goal of our project is to ensure documentation is sufficient for future groups to have a strong understanding of our system; to ensure this outcome, every three weeks we will dedicate time to ensuring the documentation current at that time is thorough and informative.

## Cost Analysis

Table A displays our list of components, along with their prices, and if we currently possess them. We possess most of the components personally and from our mentors, so the total cost of this project will be under $100, which is very feasible for our circumstances.

## Table B-1: Cost Table

|  |  |  |
| --- | --- | --- |
| Item | Rented/Posses | Cost |
| STM32F446RE Nucleo board | No | $14.90 |
| Arduino Uno | Yes | $0.00 |
| LEDs | Yes | $0.00 |
| MOSFET | No | $2.95 |
| Resistors(22 kΩ, 4.7 kΩ, 470 Ω) | Yes | $0.00 |
| Capacitor (100 nF) | Yes | $0.00 |
| USB logic analyzer | Yes | $0.00 |
| breadboard | Yes | $0.00 |
| 3.5 mm/2.5 mm jack | Yes | $2.09/$2.99 |
| jumper wires(M-M, M-F,F-F) | Yes | $0.00 |
| RS232 Cable | No | 2 X ($4.29) |
| Total |  | $31.51 |

## Regulatory Considerations

Our design will be using an audio jack to communicate with radios, which will be tested with ham and handheld radios. This is so that there is no interference between the TNC and radio’s frequencies and outside frequencies.

## https://i.groupme.com/407x930.jpeg.66210572828540fca29c514d204ecc74.largeGnatt Chart



## Figure B-2: Gnatt Chart Semester One

## Figure B-3: Gnatt Chart Semester Two

**Appendix C**

## **Alternatives and Tradeoffs Analysis**

### Microcontroller Comparison

After comparing the options listed below, the team decided on the STM32L4433 Nucleo board for the microcontroller that would format the receiving and transmitting data. Although we are all with the utilization of the Arduino Mega, it does not contain most of the functionalities we need to transmit and receive analog/digital data. We would have to implement more external components, which could possibly increase the design’s size. The Teensey board, which had very similar features to the STM32L4433, was also declined since it did contain the CRC calculation unit. The STM32L4433 is the perfect fit for our design since it has DACs and ADCs for transmitting and receiving data. It also contains a CRC calculation unit which would help detect errors, if any, when transmitting or receiving. The only downside is that we are not familiar with coding in embedded C, but our can aid us in that aspect.

|  |  |  |  |
| --- | --- | --- | --- |
| **MICROCONTROLLER** | **STM32L4433** | **Teensey 4.0** | **Arduino Mega** |
|  | https://lh5.googleusercontent.com/vZoX3w8K_sCQFoTykNPWfqf3xuJQAqylwHUTkPFhDTEnoA9SrEhm_5hSl4DOz-9z-AqcM_aHLVc-x9__vypHhTDtjOEnzjpENWCAxdcQCYVvTjrcEF0XSWRIL6IKZngmLNirKRKl | https://lh5.googleusercontent.com/9CowNBNGzVfuy8S1OtbJJrCOMLitN6CkCD6gEinUmoD4XYqm927odGAn_yz3qfkPq1foV96YcPf4OMm860gn85-sgNL9zGpOhYl8atMsjS2mj4iRcmcpTVIkke0OCGgy4E8Kle_H | https://lh6.googleusercontent.com/L7i54ezO_1n5mmOs-hwhkTP5p-vZLRxPfx2y02TwMBOpBQbTIZTCdI9y4w52BB-kIcUOKe79NStopPR150HVxHrARX5TTshv3aEo6uAgU0KVhrb3EhjkMn2f0mlhcLSIot6XD2eR |
| Description | This microcontroller contains 16 external ADC channels, 1 12-bit ADC, 2 12-bit DAC output channels, an on board RTC, 2 CAN buses, 2 ultra-low-power comparators, CRC calculation unit, and a Schmitt trigger I/O. | This microcontroller contains 40 digital pins (all interrupt capable), 14 analog pins, 2 ADCs on chip, a RTC for date/time, an ARM Cortex-M7 at 600 MHz, 1024K RAM (512K is tightly coupled), and a 2048K Flash (64K reserved for recovery & EEPROM emulation). | This microcontroller contains 16 Analog read pins, 53 Digital pins,and  6 interrupt pins. |
| Cost | 14.90 | 19.99 | 18.99 |
| Pros | * Contains CRC calculation unit * Low Cost * Many GPIOs | * Fast clock speed * Has RTC | * Easy to use * Many GPIOs |
| Cons | * Embedded C programming | * Highest Cost * No CRC calculation unit | * Does not contain RTC * Does not contain DACs or ADCs |

### PTT

The radios that we are testing with will be outputting fifteen volts, and we need our automatic push to talk to be able input that voltage and send low signal to activate. The team decided to use a MOSFET to switch the mode on, since it is flexible in the voltage it takes in. Although the inverter IC and comparator IC are very simple to use, they may not be able to pass in twenty milliamps. The MOSFET on the hand definitely meets our specifications.

|  |  |  |  |
| --- | --- | --- | --- |
| Circuit Design | Inverter Circuit | [P-Channel MOSFET](https://www.falstad.com/circuit/circuitjs.html?cct=$+1+0.000005+11.086722712598126+36+5+43%0Af+176+176+240+176+33+1.5+0.02%0Av+176+96+240+96+0+0+40+15+0+0+0.5%0Aw+240+96+240+160+2%0Ar+240+192+240+272+0+1000%0As+112+176+176+176+0+0+false%0Aw+240+96+240+64+0%0Aw+240+64+112+64+0%0Aw+112+64+112+176+0%0Ag+176+96+176+128+0%0Ag+240+272+240+288+0%0Ar+176+176+176+272+0+1000%0Aw+176+272+240+272+0%0Aw+240+192+352+192+2%0Ac+240+96+352+96+0+1e-7+15%0Ac+352+192+352+96+0+1e-7+0.0003214320676074135%0Ag+352+96+368+80+0%0Ao+14+64+0+4099+0.001220703125+0.00009765625+0+2+14+3%0A38+1+0+1+15+Voltage%0A) | [Comparator IC](http://www.ti.com/product/LM393) |
|  | https://lh5.googleusercontent.com/Ez9nSa_enZCL84zQaSSjkZ0s9zcuHRymivi4vE-t1-vYG1dtiOhezrcqmIuusZ-ktl5Zyt42C0efUMzidWHjRVzTA530zCjzEnjKBTFhLw_QY7zQzoUjNDHcYajjtPRdZKluY19W | https://lh6.googleusercontent.com/XDWB-bMpjrCLuQ3PzPGxKq8x6okLyu5NI1LKe_1L1xsXcTZLviXh7jvfsRNmhLYOj3xyNp2XbzAVY3irOYU377jXth5dbwCmDDcOlO0Y7IIgyRJgNGG2CGW-SB66f6FYWqyAO9ZJ | https://lh4.googleusercontent.com/7D0NWypkSsM4kSsj4pFUumgPwPunbdxGYyupSGdlkl1aByJXVAehluhTxP07p8gTGtAUoEY0az72tGVu_t4IajHfccMPzvps-rufmwy6EUr7rbJ1mlDQdOxuOmfgnTOxcy0iUVTI |
| Description | This IC outputs an inverted signal of the input. | Simple transistor gate to create the desired active low needed for radio circuits. | Many of these ICs feature several outputs: A<B, A>B and A=B. Using the greater than output, this would be an easy way to generate an inverting signal. |
| Pros | Since the device has built in digital logic, this means setting up a circuit for it to operate in would be much simpler than the MOSFET. | Using a MOSFET allows for flexible input voltage and does not add to power constraints. | Since the device has built in digital logic, this means setting up a circuit for it to operate in would be much simpler than the MOSFET. |
| Cons | Device may not be capable of supplying needed current for the system. Device should be capable of passing ~20mA | MOSFETs are capable of generating excess heat that may cause issues when design is compressed to <2x2in | Device may not be capable of supplying needed current for the system. Device should be capable of passing ~20mA |

### ADC

Since we are going for the STM32L4433 microcontroller we are using the schmitt trigger as our ADC. The zero crossing implementation would be a simple circuit to use, but it would be an extra component we would have to purchase and it would take up extra space on our final board design. Although applying fourier analysis would not involve using any hardware, applying it would be too much for the scope of this design. The TNC is only working with two different frequencies. The schmitt trigger would save us money and space on our final design.

|  |  |  |  |
| --- | --- | --- | --- |
| Signal Analysis Method | Fourier Analysis | Schmitt trigger | Zero Crossing |
|  | https://lh3.googleusercontent.com/sLzpU_T0F9-GSnzUXa71Jv5GBBn3SJzTdJPnxWULuXOO-W01GnXEwYRfGIGVG1Hig2N0dq_M1nwlimbTwNF5k7Nb3kye6pvzylKoDIJAWhlEEJRr2vYCHqdgLC_iOI2r5iMDyaTC | https://lh6.googleusercontent.com/u5YuDBGZ6pxH8QbC_hipf-lNKlfhJq-JFy1bJh08HQ2Dw-U1HhR71tm_6Pcb8-vfzbFx-_ZrZFNTk92CrgZeyCpaHZzlYDYKUP4KJ0WAcbz2DvFMpO66ipYRErjcaQK6O4ubrdPb | https://lh5.googleusercontent.com/YeWC7OlgQ_GwtUyanL28WS9CD4xqXe0YP2GNhwqOvsidIvKDxyxnU5rTM2PfU6TXxskPXTqNwJlyEOxwBZQmDfJqWUd5UNcjhmrwJRf1yZJfJH1ZiUF9CASXXJuvzDVKaqEDhZRM |
| Description | In our case, this project would use this method to add multiple analog signals of different frequencies to generate digital signals | Simple transistor gate to create the desired active low needed for radio circuits. | Uses comparator to output a toggling logic signal when analog voltage goes to zero. |
| Pros | It doesn’t involve any hardware. | Built in on STM 32 boards, great for filtering out oscillation in digital signal, when noise is in audio/analog signa | Easy implementation with a comparator |
| Cons | Not efficient because we would need multiple waves of different frequencies to generate a digital signal when we are only working with two different frequencies | If not built on microcontroller we would have to buy a comparator IC | Device may not be capable of supplying needed current for the system. Device should be capable of passing ~20mA |

### DAC

After comparing the options for the DACs, we decided to use the DAC built in our microcontroller. The resistor switching network would be more components we need to implement on our final design and we would have to spend more time coding the DAC. The DAC IC would be very simple to use in our design, but it would take more space on final design and add more to our cost. The DAC is already built in to our microcontroller so it would be more convenient.

|  |  |  |  |
| --- | --- | --- | --- |
| Circuit Design | Built into Controller | Resistor Switching Network | DAC IC |
|  | https://lh5.googleusercontent.com/2B7QmVERSSFp5JTksCIPu51ITsoDohngg5mGgHY1aT2O__FcOuzQg5qeLKpQxhibyvJsk7RXMIU8StlZE0XN1-D6RWcSGC52MFb_ZIk2dzSRtk2m8dnYSwU__nRE01F0arsAe9dN | https://lh5.googleusercontent.com/WuaXKkqGKXJpaLUGEUCQoZVK9Vz4f6-vovEcoTJEXfqrdYvp8tRBpDAEckdlgcIQQQgBVKC2J1PzWQN8MmKkl1kW38QPg0s-z67vgtcWe3wl7fCWRG97dWfK0zsLKwSaf33Dan9Z | https://lh6.googleusercontent.com/eSPGi4k4aQhuUAKpIGtxSygssukgaxtbHYOIPtKKQu8W73BNpAs7I9sNY4ScSk0nZL9VD20Wbno8sQ6xpMzm5WWCyPNHs6M1ZchvKm4tZVzWFQP06ypkg9isuJEu-45QMQMOEfiW |
| Description | If the design were to include any of the STM32 line, the MCUs have builts in DACs. | Would only consist of using ~4-6 GPIO, connected to different resistor values to represent variable step voltage output. This output would be passed through an LPF to generate a smooth sinusoid. | This would be using a dedicated High-Speed DAC ICs (such as [DAC38RF82](http://www.ti.com/product/DAC38RF82)) that only requires digital input translated to an analog wave for us. |
| Pros | Similarly to the dedicated IC, the benefit is there will only be a need to generate digital values. | to this option is the simplicity and lack of components needed to generate waveform at low power cost. | is the ease of use, only needing to generate digital values that will quickly be converted to sinusoidal waveform. |
| Cons | Often built in DACs are slow and this may not work within the strict timing constraints of AX.25 | would be the requirement to create code to drive a resistor network meaning more time would be spent on the DAC | With the dedicated silicon, this will raise the price and power consumption of the board. |

1. This paper was submitted for review February 18, 2020. The project’s mentors are Mr. Nolan Edwards, Mr. James Palmer, Mr. Nick Pugh, and Mr. Rizwan Merchant.

   K. P. Leon is with Electrical and Computer Engineering Department, University of Louisiana at Lafayette, Lafayette, LA 70504 USA (email: [C00094357@louisiana.edu](mailto:C00094357@louisiana.edu))

   K. T. Keopraseuth is with Electrical and Computer Engineering Department, University of Louisiana at Lafayette, Lafayette, LA 70504 USA (email: [C00092349@louisiana.edu](mailto:C00092349@louisiana.edu))

   D. L. Cain is with Electrical and Computer Engineering Department, University of Louisiana at Lafayette, Lafayette, LA 70504 USA (email:C00043561@louisiana.edu) [↑](#footnote-ref-1)