

Mini Project Report- Applied SDLC and Software Testing

Team 15

Digital Modulation Tools

L&T Technology and Services

Contents

Introduction	3
Digital Modulation Concepts and Types:	3
Analog to Digital Modulation Techniques	3
Digital to Digital Modulation Techniques.....	4
SWOT analysis.....	6
4W's and 1H.....	6
Who:	6
What:	6
When:	6
Where:.....	6
How:	6
Detail requirements.....	7
High Level Requirements:.....	7
Low level Requirements:.....	7
Design	8
High Level Requirements UML Diagrams	8
Package Diagram.....	8
Component Diagram.....	9
Use Case	10
Activity Diagram.....	10
Low Level Requirements UML Diagrams	10
NRZ and RZ Activity Diagram	10
NRZ and RZ Component Diagram	10
NRZ and RZ Package Diagram	10
NRZ and RZ Use Case Diagram	11
ASK Use Case Diagram.....	11
ASK Use Case Diagram.....	11
ASK Component Diagram.....	11
Display Activity Diagram.....	11
Display Structural Diagram	11
Manchester Activity Diagram.....	11
Manchester Component Diagram.....	11

Manchester Package Diagram.....	11
Manchester Use Case Diagram.....	12
FSK Component Diagram	13
FSK Use Case Diagram	13
PSK Component Diagram.....	14
FSK Activity Diagram	15
FSK Sequence Diagram PSK Activity Diagram	16
PSK Sequence Diagram	18
TEST PLAN:.....	19
Table: High level test plan.....	19
Table: Low level test plan	19
Results	22
References	27

Introduction

Digital Modulation Tools is a simple console application designed to give the user a simulation of their data and the possible modulations and encoding. The bit stream given can be modulated in various modulation techniques like - ASK, FSK, PSK, etc. It is developed using C programming language.

Digital Modulation Concepts and Types:

Analog to Digital Modulation Techniques

Amplitude Shift Keying

ASK is a type of modulation where the digital signal is represented as a change in amplitude. In order to carry out amplitude shift keying, we require a carrier signal and a binary sequence signal. It is also known as On-Off keying. This is because the carrier waves switch between 0 and 1 according to the high and low level of the input signal.

Frequency Shift Keying

Frequency Shift Keying FSK is the digital modulation technique in which the frequency of the carrier signal varies according to the digital signal changes. FSK is a scheme of frequency modulation.

The output of a FSK modulated wave is high in frequency for a binary High input and is low in frequency for a binary Low input. The binary 1s and 0s are called Mark and Space frequencies.

Binary Phase Shift Keying

Phase Shift Keying PSK is the digital modulation technique in which the phase of the carrier signal is changed by varying the sine and cosine inputs at a particular time.

This is also called as 2-phase PSK or Phase Reversal Keying. In this technique, the sine wave carrier takes two phase reversals such as 0° and 180° .

BPSK is basically a Double Side Band Suppressed Carrier DSBSC modulation scheme, for message being the digital information.

Quadrature Amplitude Modulation

The creation of symbols that are some combination of amplitude and phase can carry the concept of transmitting more bits per symbol further. This method is called quadrature amplitude modulation (QAM).

While QAM is enormously efficient of spectrum, it is more difficult to demodulate in the presence of noise, which is mostly random amplitude variations.

Linear power amplification is also required.

Phase Shift Keying

Digital to Digital Modulation Techniques

NRZ

- NRZ stands for Non-return zero.
- In NRZ encoding, the level of the signal can be represented either positive or negative.

NRZ-L

- In NRZ-L encoding, the level of the signal depends on the type of the bit that it represents.
- If a bit is 0 or 1, then their voltages will be positive and negative respectively. Therefore, we can say that the level of the signal is dependent on the state of the bit.

NRZ-I

- NRZ-I is an inversion of the voltage level that represents 1 bit.
- In the NRZ-I encoding scheme, a transition occurs between the positive and negative voltage that represents 1 bit.
- In this scheme, 0 bit represents no change and 1 bit represents a change in voltage level.

RZ

- RZ stands for Return to zero.
- There must be a signal change for each bit to achieve synchronization. However, to change with every bit, we need to have three values: positive, negative and zero.
- RZ is an encoding scheme that provides three values, positive voltage represents 1, the negative voltage represents 0, and zero voltage represents none.
- In the RZ scheme, halfway through each interval, the signal returns to zero.
- In RZ scheme, 1 bit is represented by positive-to-zero and 0 bit is represented by negative-to-zero.

Bi-phase

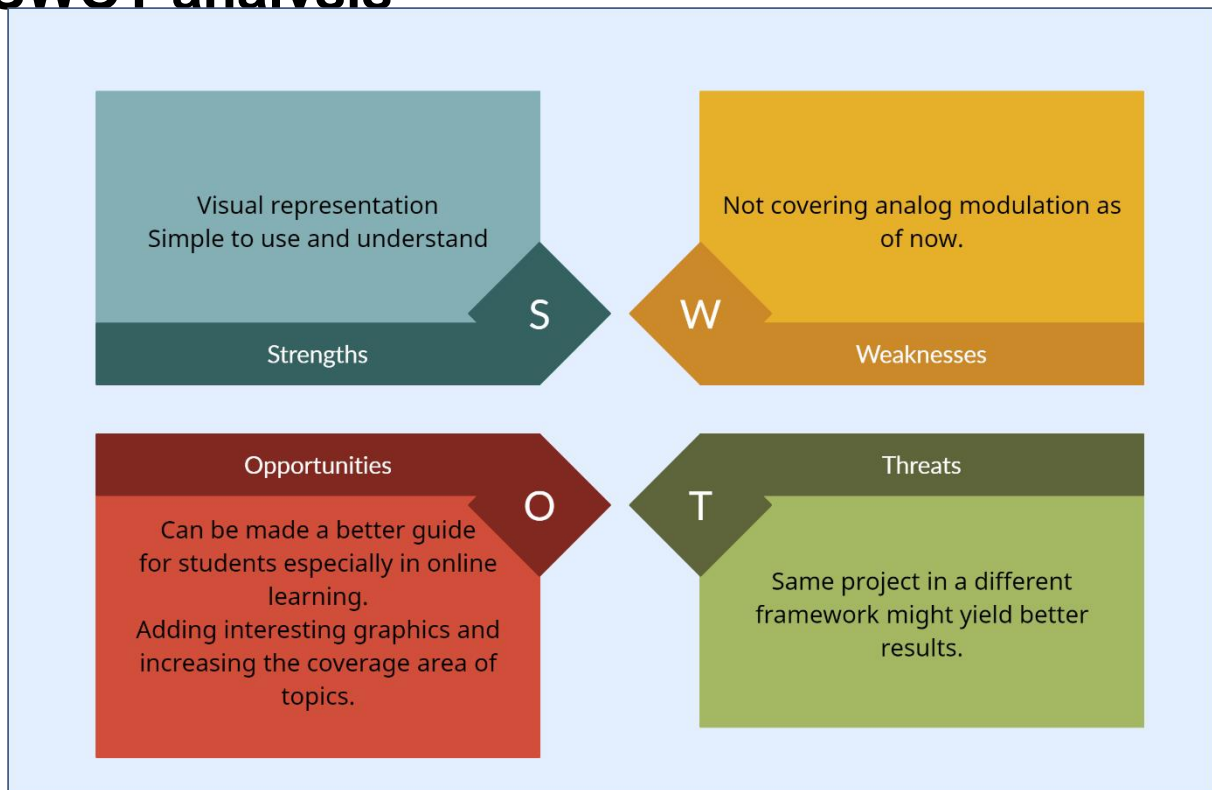
Manchester Encoding

- It changes the signal at the middle of the bit interval but does not return to zero for synchronization.
- In Manchester encoding, a negative-to-positive transition represents binary 1, and positive-to-negative transition represents 0.
- Manchester has the same level of synchronization as RZ scheme except that it has two levels of amplitude.

Differential Manchester Encoding

- It changes the signal at the middle of the bit interval and here the presence or absence of the transition at the beginning of the interval determines the bit.
- A transition means binary 0 and no transition means binary 1.
- Two signal changes represent 0 and one signal change represent 1.

SWOT analysis



4W's and 1H

Who:

- Students and curious learners who wish to see how digital signals are encoded.

What:

- A console based simulating tool covering digital modulation concepts.

When:

- On encountering troubles in understanding digital modulation concepts.

Where:

- A handy tool that can be accessed from anywhere .

How:

- One-to-one mapping will be used for displaying the modulated waveforms and bitstreams and also a database of circuits used for the respective purpose will be prepared which would be displayed as and when needed and thus guiding the user to a complete extent.

Detail requirements

High Level Requirements:

ID	Description	Category	Status
HLR01	User Interface	Technical	Done
HLR02	Digital to Analog Modulation Techniques	Technical	Done
HLR02	Digital to Digital Modulation Techniques	Technical	Done

Low level Requirements:

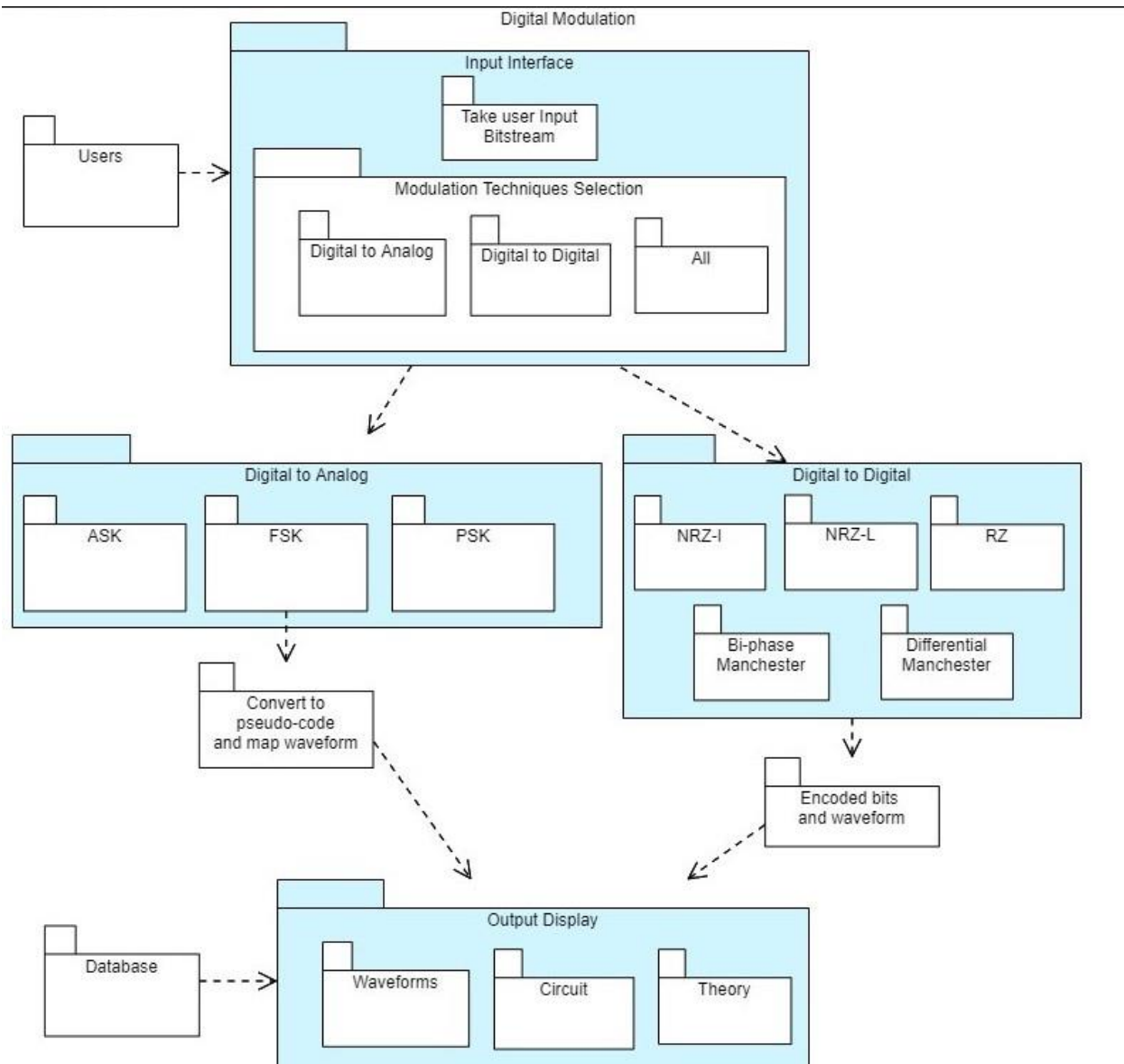
ID	Description	HLR ID	Status
LR00	Input Interface	HLR01	Done
LR01	Output Display	HLR01	Done
LR02	Displaying theory of circuits	HLR01	Done
LR03	Amplitude Shift Keying	HLR02	Done

ID	Description	HLR ID	Status
LR04	Frequency Shift Keying	HLR02	Done
LR05	Phase Shift Keying	HLR02	Done
LR06	Quadrature Amplitude Modulation	HLR02	Done
LR07	Non Return to Zero - I	HLR03	Done
LR08	Non Return to Zero - L	HLR03	Done
LR09	Return to Zero	HLR03	Done
LR10	Biphase Manchester	HLR03	Done
LR11	Differential Manchester	HLR03	Done

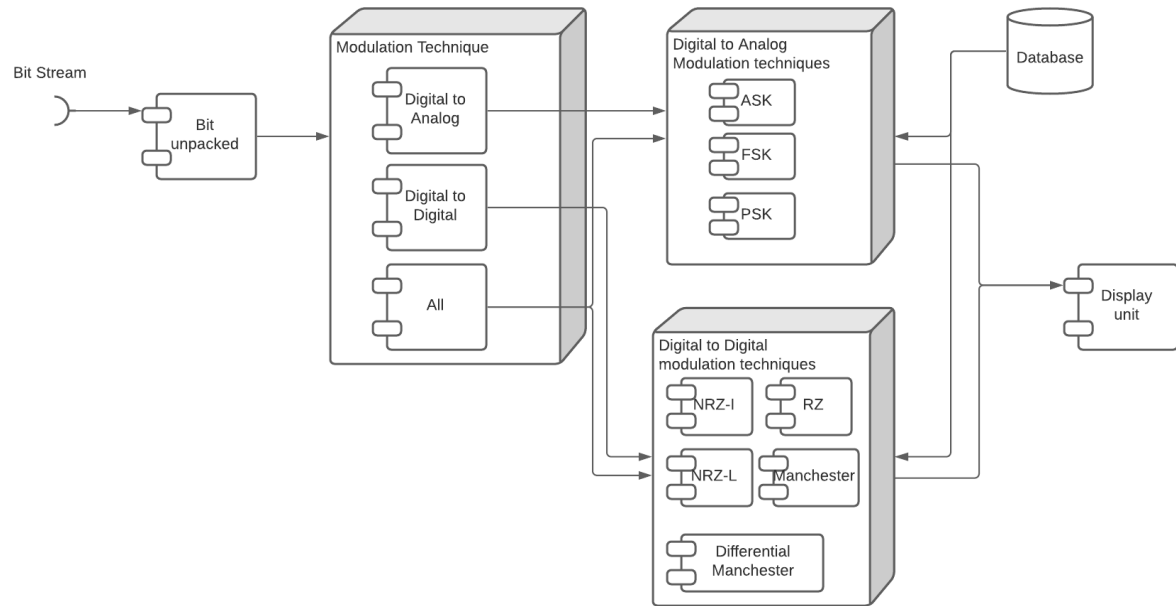
Design

High Level Requirements UML Diagrams

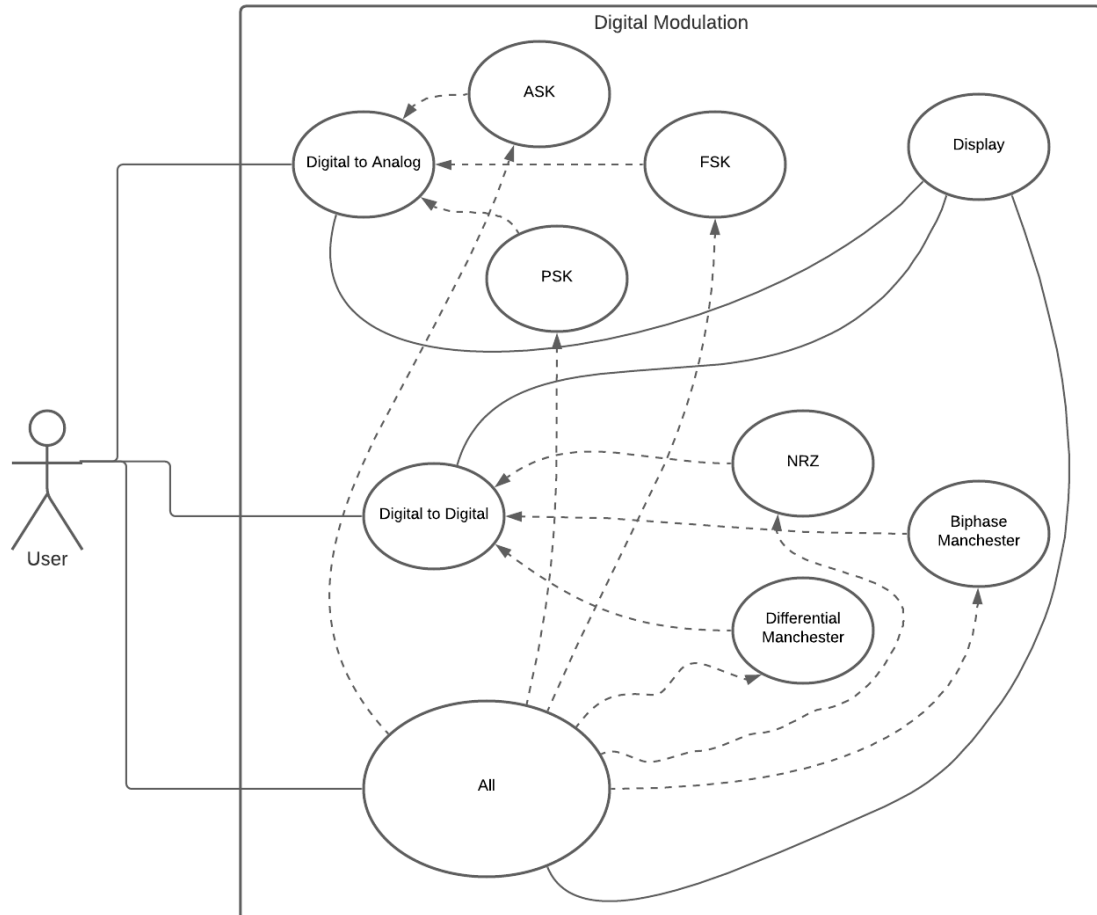
Package Diagram



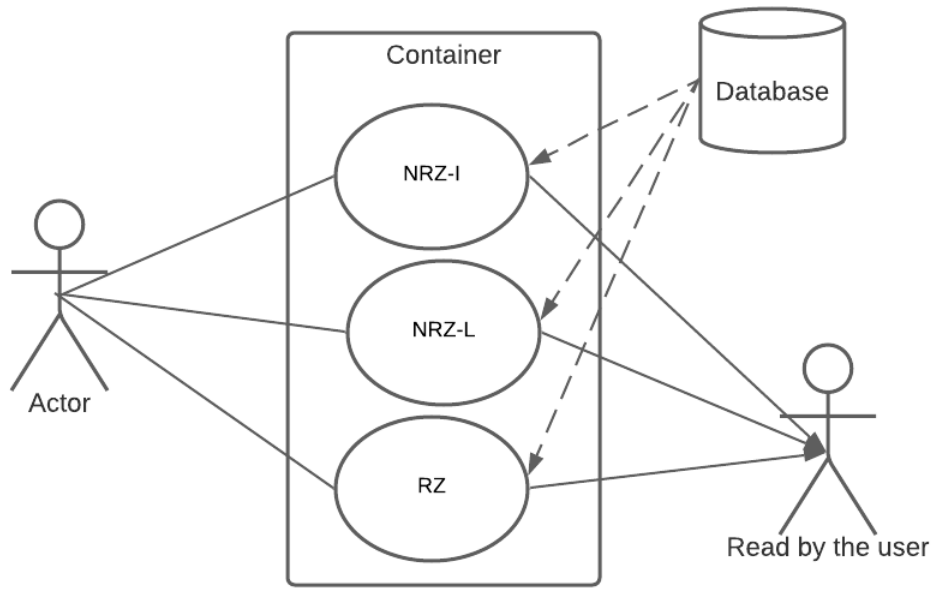
Component Diagram



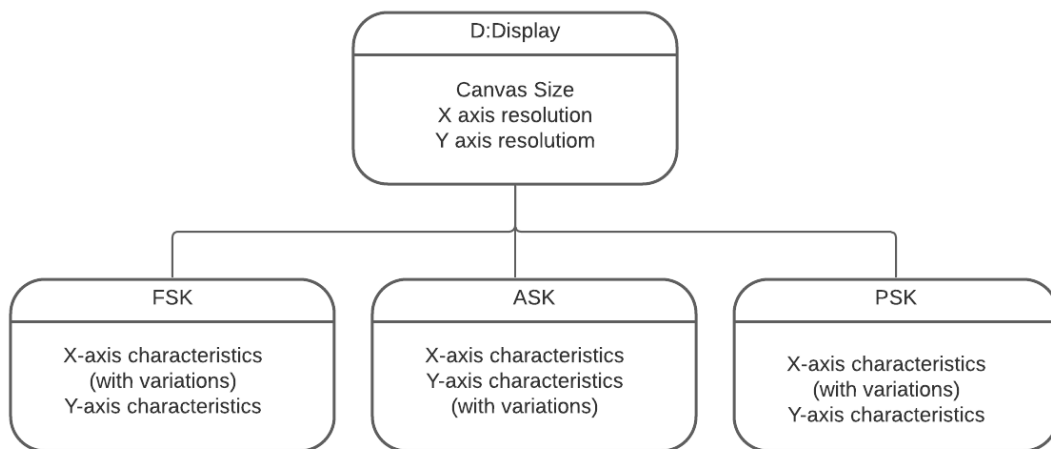
Use Case



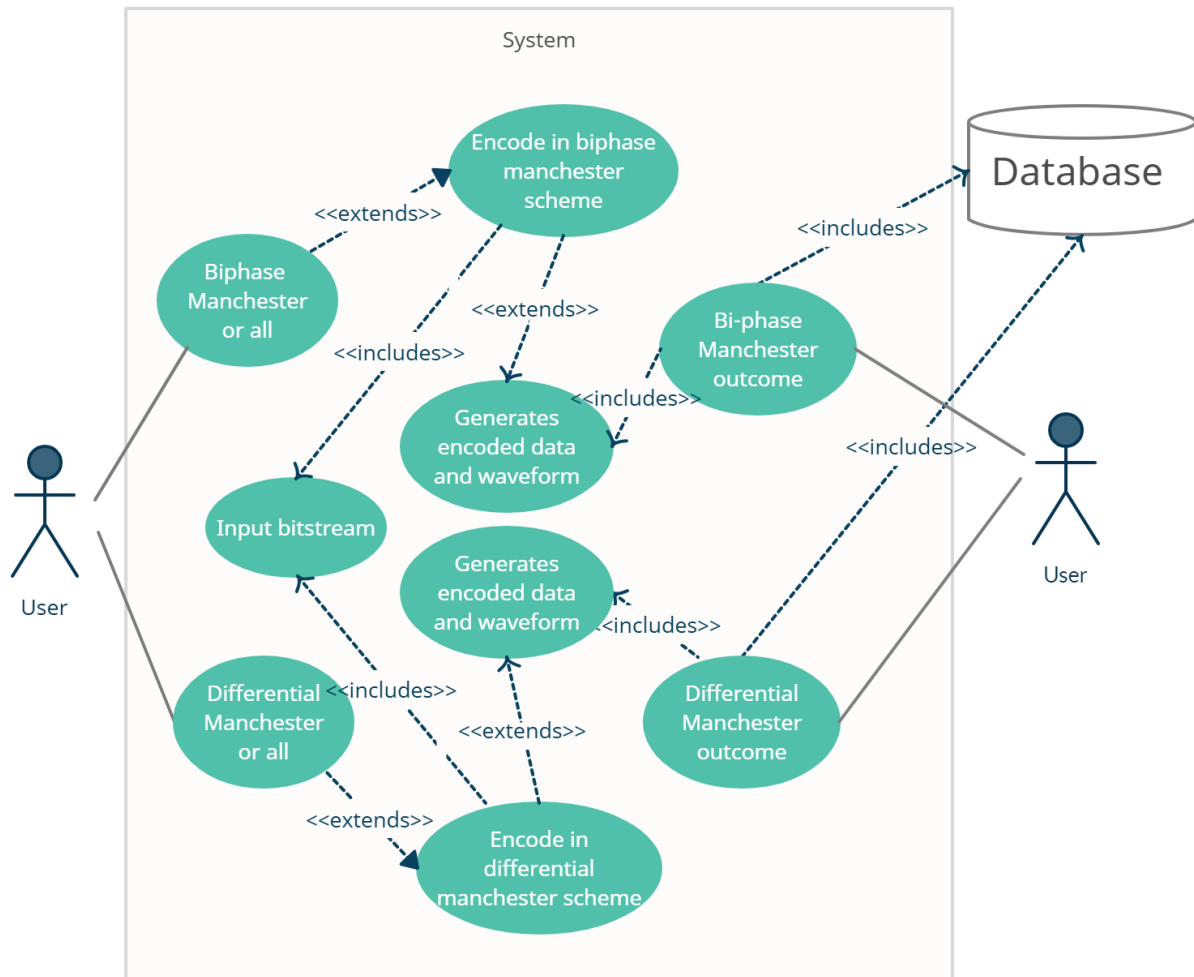
NRZ and RZ Use Case Diagram



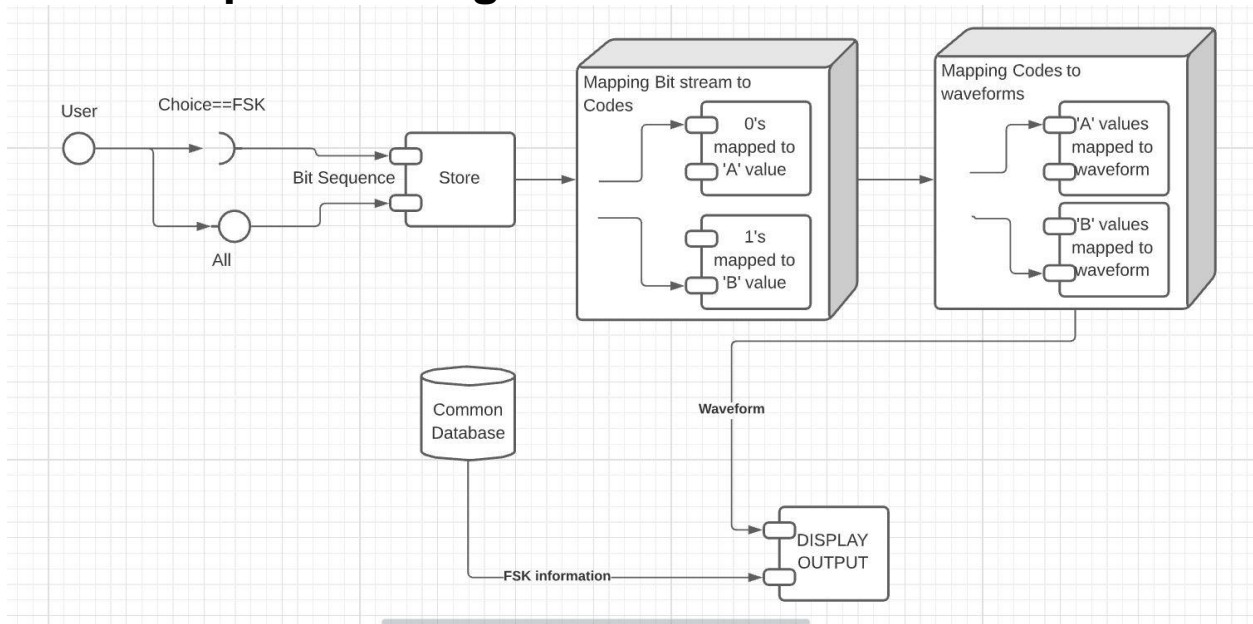
Display Structural Diagram



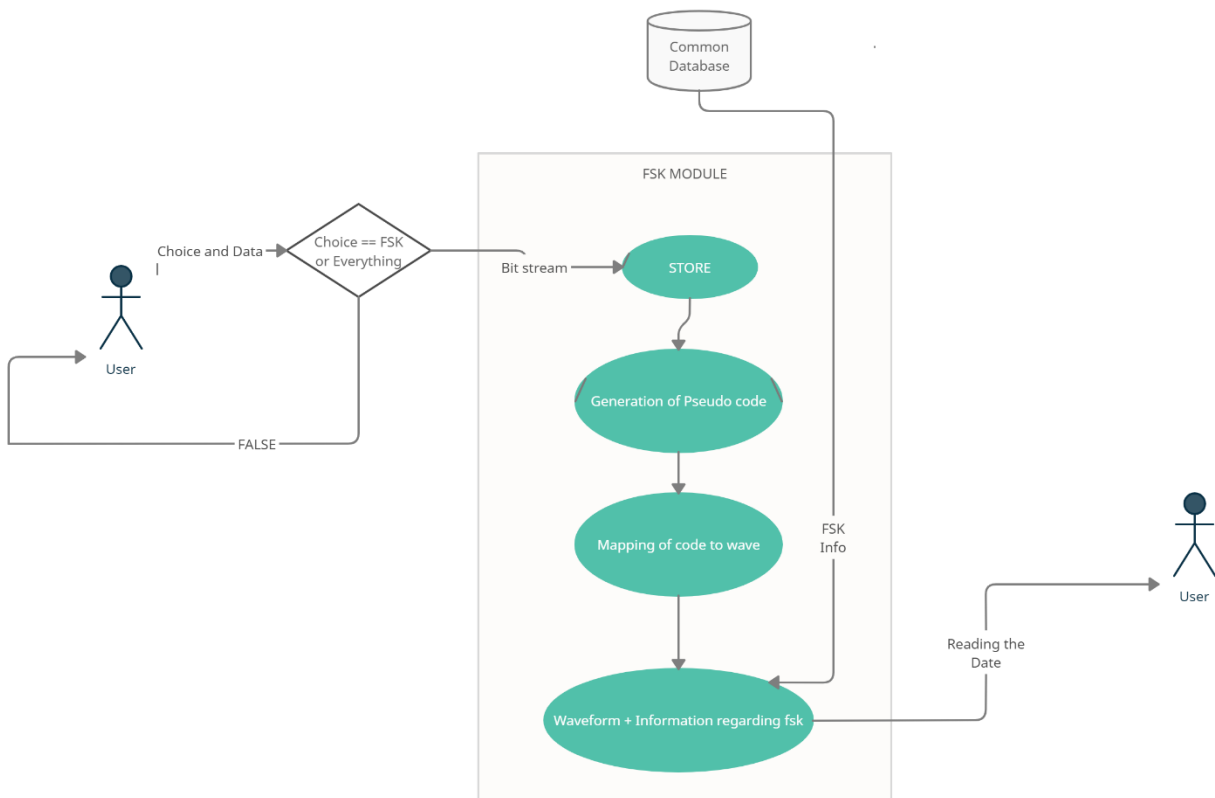
Manchester Use Case Diagram



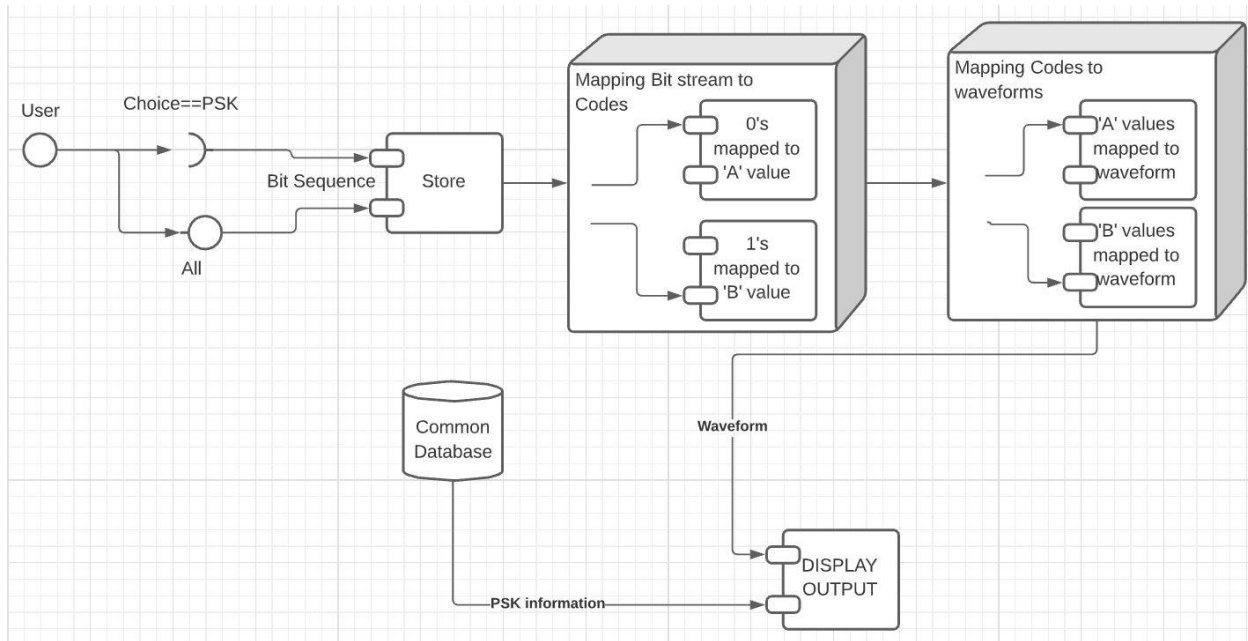
FSK Component Diagram



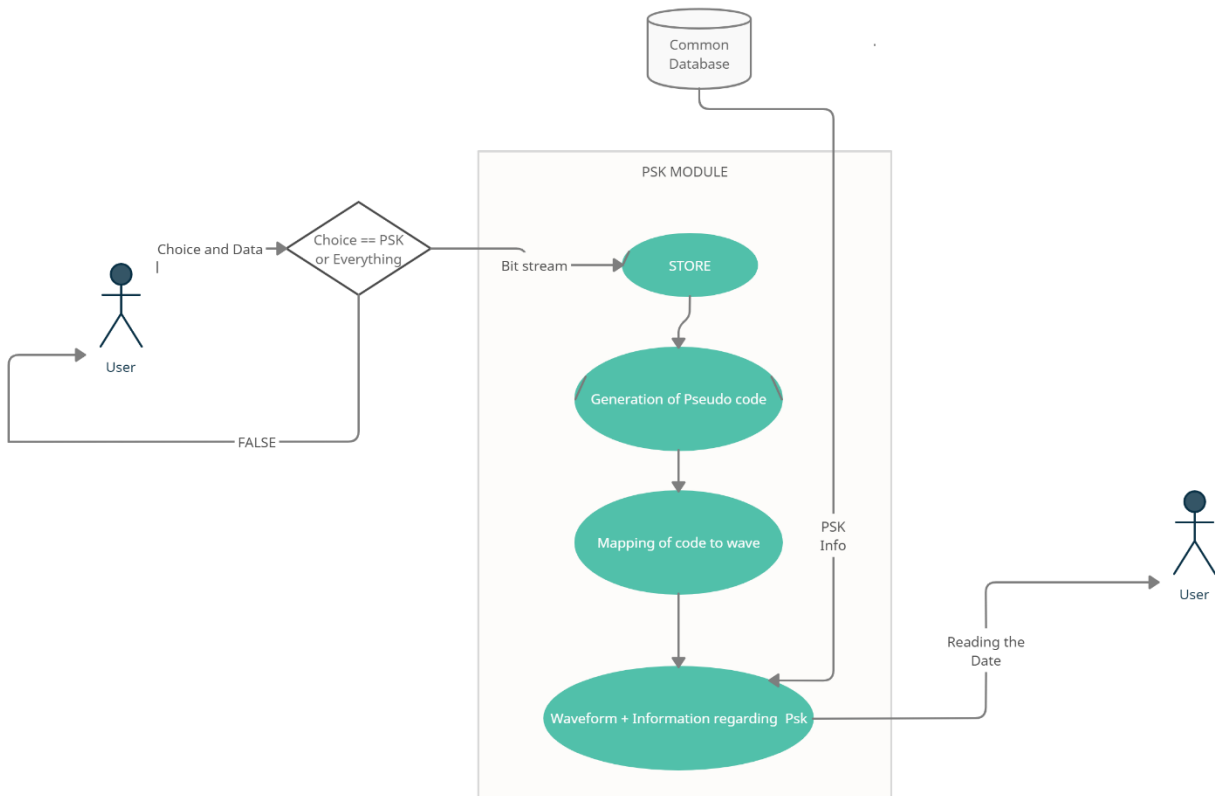
FSK Use Case Diagram



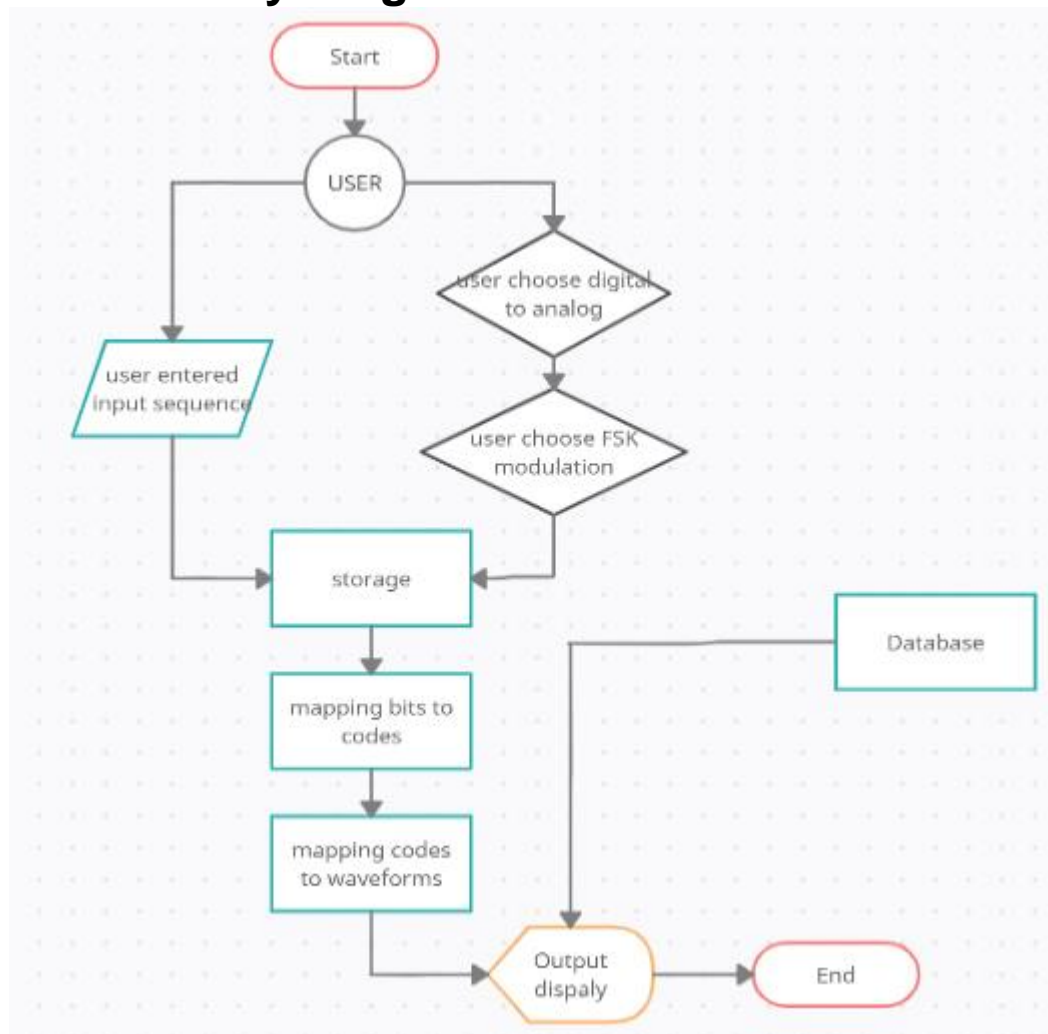
PSK Component Diagram



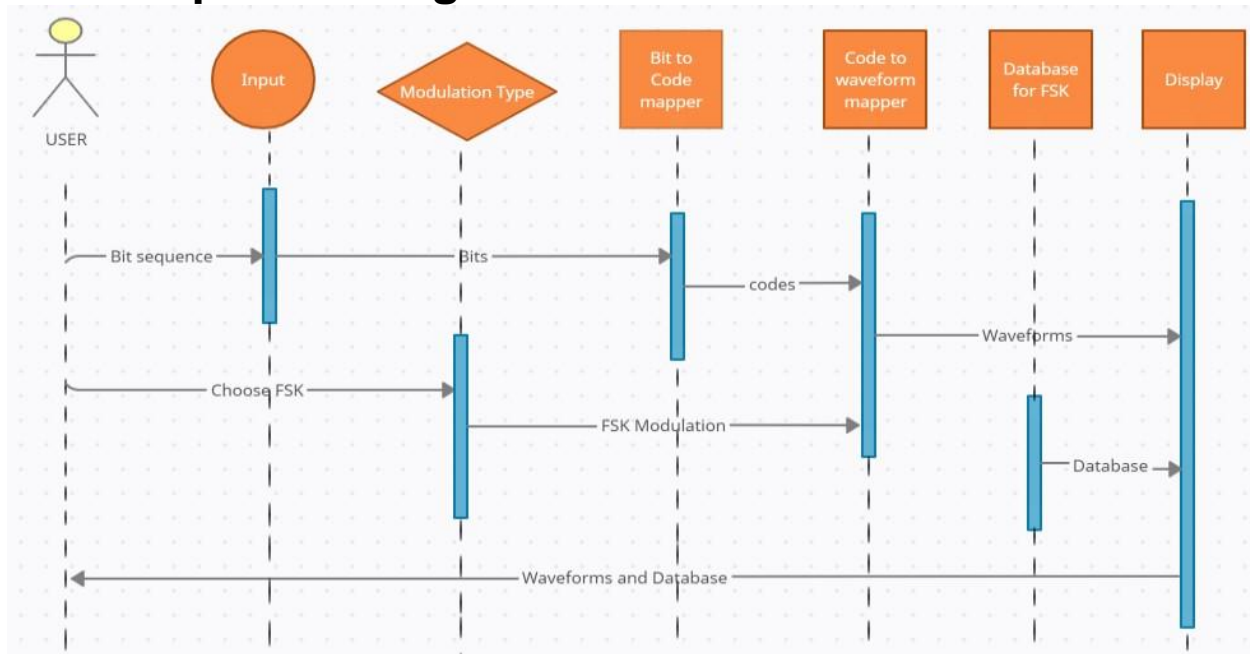
PSK Use Case Diagram



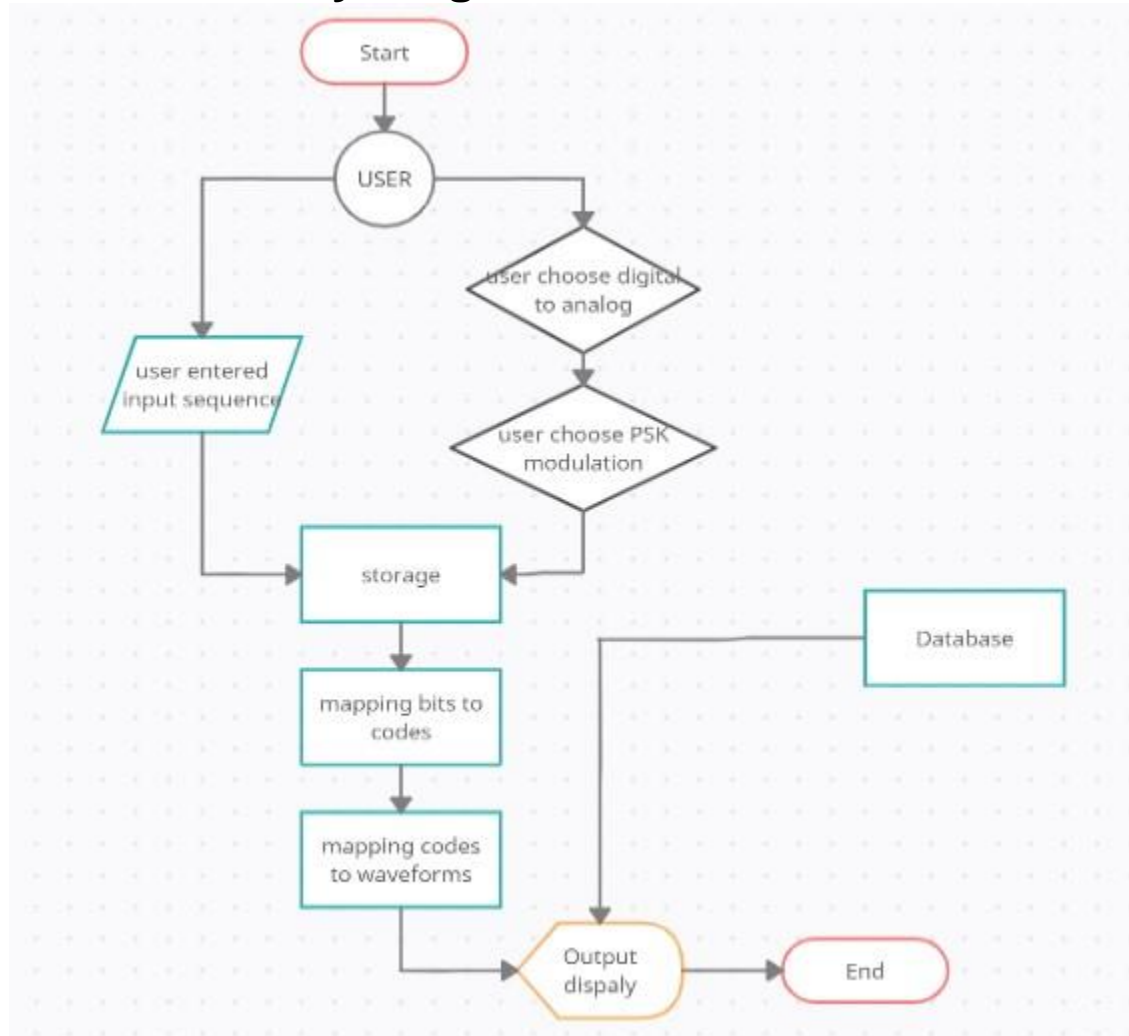
FSK Activity Diagram



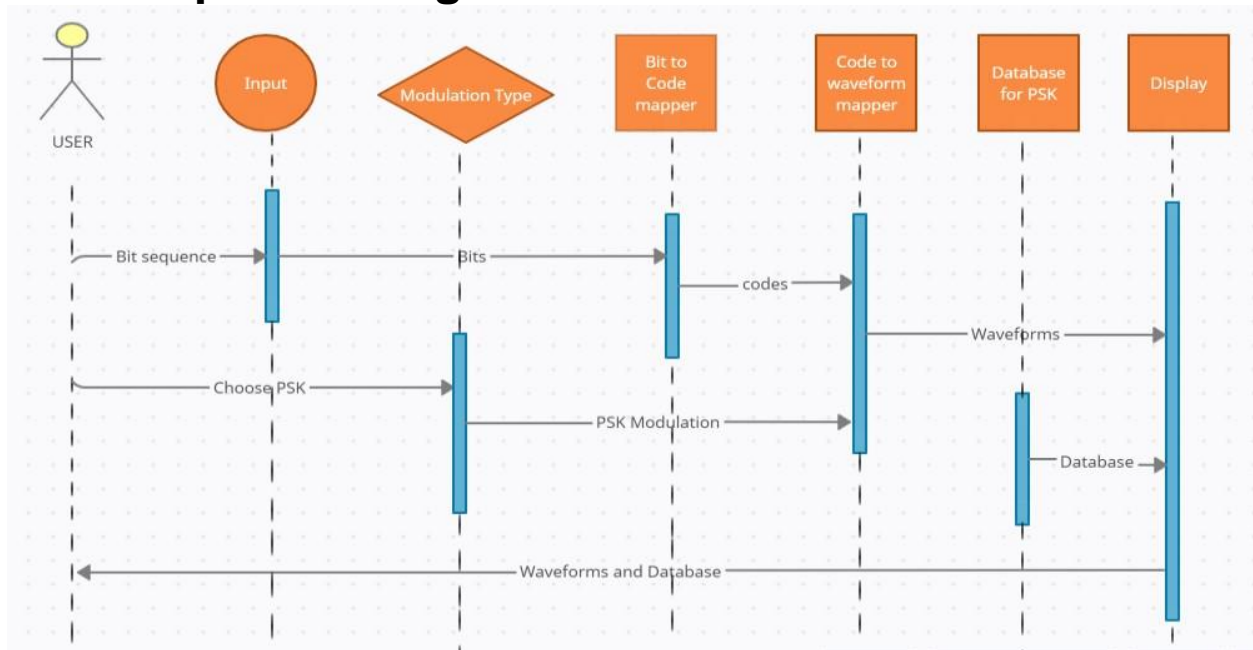
FSK Sequence Diagram



PSK Activity Diagram



PSK Sequence Diagram



TEST PLAN:

Table: High level test plan

Test ID	Description	Exp IN	Exp OUT	Actual Out	Means of Test	Type Of Test
H_01	Display should work	---	display	OPdisp	Tester	Requirement Based
H_02	Waveforms should turn up	---	waveform	Opwaveform	Tester	Scenario Based
H_03	Input Interface	---	interfacedisplay	OPintdisp	Tester	Requirement Based
H_04	Access Database	Integer(1-5)	database	Opdatabase	Tester	Scenario Based

Table: Low level test plan

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Means of Test
L_01	ASK	Enter 8 bit binary number: 10001101	ASK OP	ASK OP	Tester

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Means of Test
L_02	FSK	Enter 8 bit binary number: 10101011	<u>FSK OP</u>	<u>FSK OP</u>	Tester
L_03	PSK	Enter 8 bit binary number: 11101001	<u>PSK OP</u>	<u>PSK OP</u>	Tester
L_04	QAM	Enter 8 bit binary number: 11101001	<u>QAM OP</u>	<u>QAMP OP</u>	Tester
L_05	NRZ-L	Enter 8 bit binary number: 11011110	<u>NRZL OP</u>	<u>NRZL OP</u>	Tester
L_06	NRZ-L	Enter 8 bit binary number: 10010011	<u>NRZL OP</u>	<u>NRZL OP</u>	Tester
L_07	NRZ-L	Enter 8 bit binary number: 11001	<u>NRZL OP</u>	<u>NRZL OP</u>	Tester
L_08	NRZ-I	Enter 8 bit binary number: 11011110	<u>NRZI OP</u>	<u>NRZI OP</u>	Tester

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Means of Test
L_09	NRZ-I	Enter 8 bit binary number: 10010011	<u>NRZI OP</u>	<u>NRZI OP</u>	Tester
L_10	NRZ-I	Enter 8 bit binary number: 11001	<u>NRZI OP</u>	<u>NRZI OP</u>	Tester
L_11	RZ	Enter 8 bit binary number: 11011110	<u>RZ OP</u>	<u>RZ OP</u>	Tester
L_12	RZ	Enter 8 bit binary number: 10010011	<u>RZ OP</u>	<u>RZ OP</u>	Tester
L_13	RZ	Enter 8 bit binary number: 11001	<u>RZ OP</u>	<u>RZ OP</u>	Tester
L_14	Biphase Manchester	Enter 8 bit binary number: 11011110	<u>Man EOP</u>	<u>Man AOP</u>	Tester
L_15	Differential Manchester	Enter 8 bit binary number: 11011110	<u>Diff EOP</u>	<u>Diff AOP</u>	Tester

Results

Implementation Screen

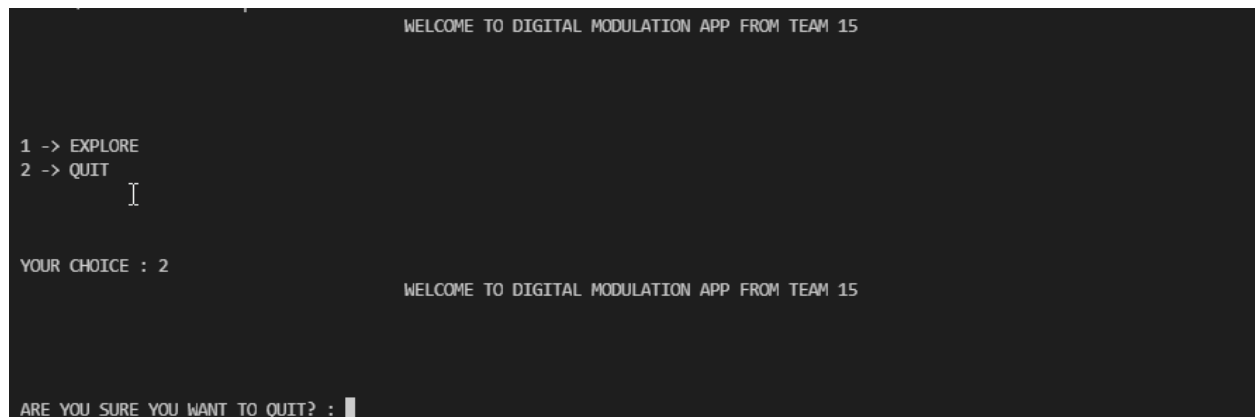
```
WELCOME TO DIGITAL MODULATION APP FROM TEAM 15

I

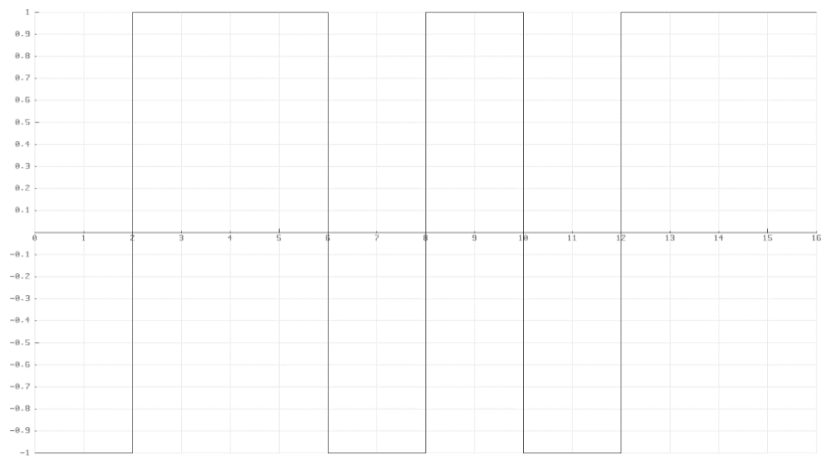
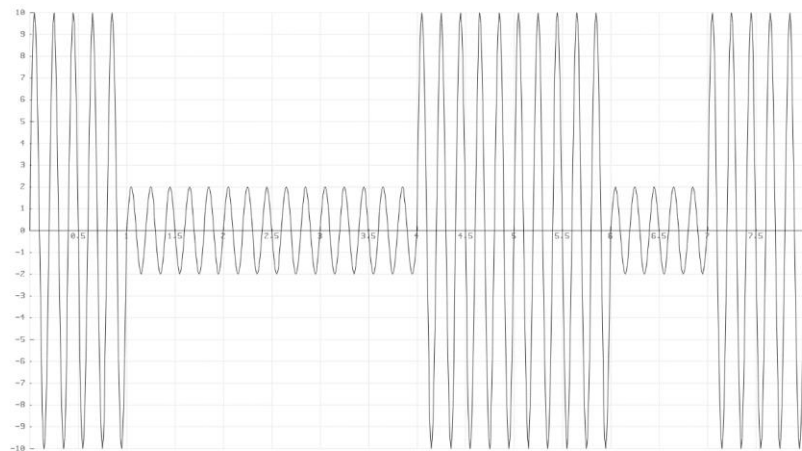
1 -> EXPLORE
2 -> QUIT

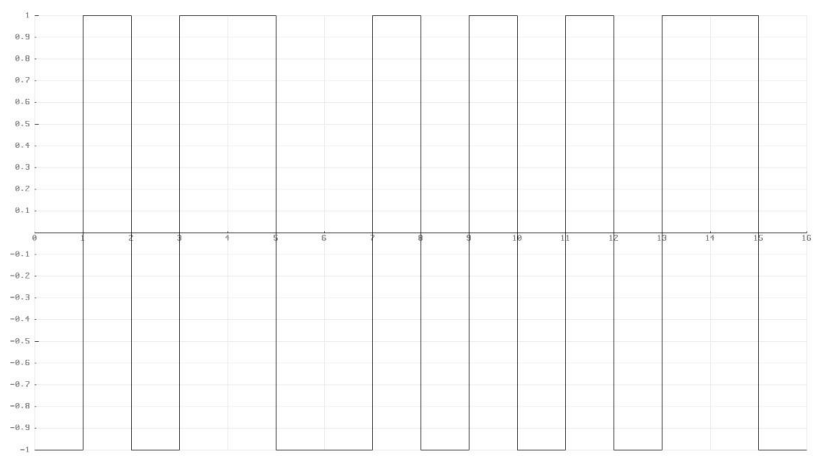
YOUR CHOICE : 1
Enter 8 bit binary number: 1001
Valid binary number 1001 .
Select the modulation type:
1. Digital to Analog
2. Digital to Digital

YOUR CHOICE : 1
Enter 8 bit binary number: 1001
Valid binary number 1001 .
Select the modulation type:
1. Digital to Analog
2. Digital to Digital
2
Select a modulation technique:
1. Non Return to Zero - I
2. Non Return to Zero - L
3. Return to Zero
4. Biphase:Manchester
5. Biphase:Differential Manchester
2
-----
MODULATION: Non-return to-zero-Level
-----
DEFINATION: It is the same as NRZ, however, the first bit of the input signal should have a change of polarity.
-----
PROCESS: There is a change in the polarity of the signal, only when the incoming signal changes from 1 to 0 or from 0 to 1.
-----
ADVANTAGES:
1. It is simple.
2. A lesser bandwidth is required.
-----
DISADVANTAGES:
1. No error correction done.
2. Presence of low frequency components may cause the signal droop.
-----
APPLICATIONS:
1. Non-return to zero encoding is commonly used in slow speed communications interfaces for both synchronous and asynchronous transmission.
2. Used in CAN protocol.
-----
The output string is: 0001001
```

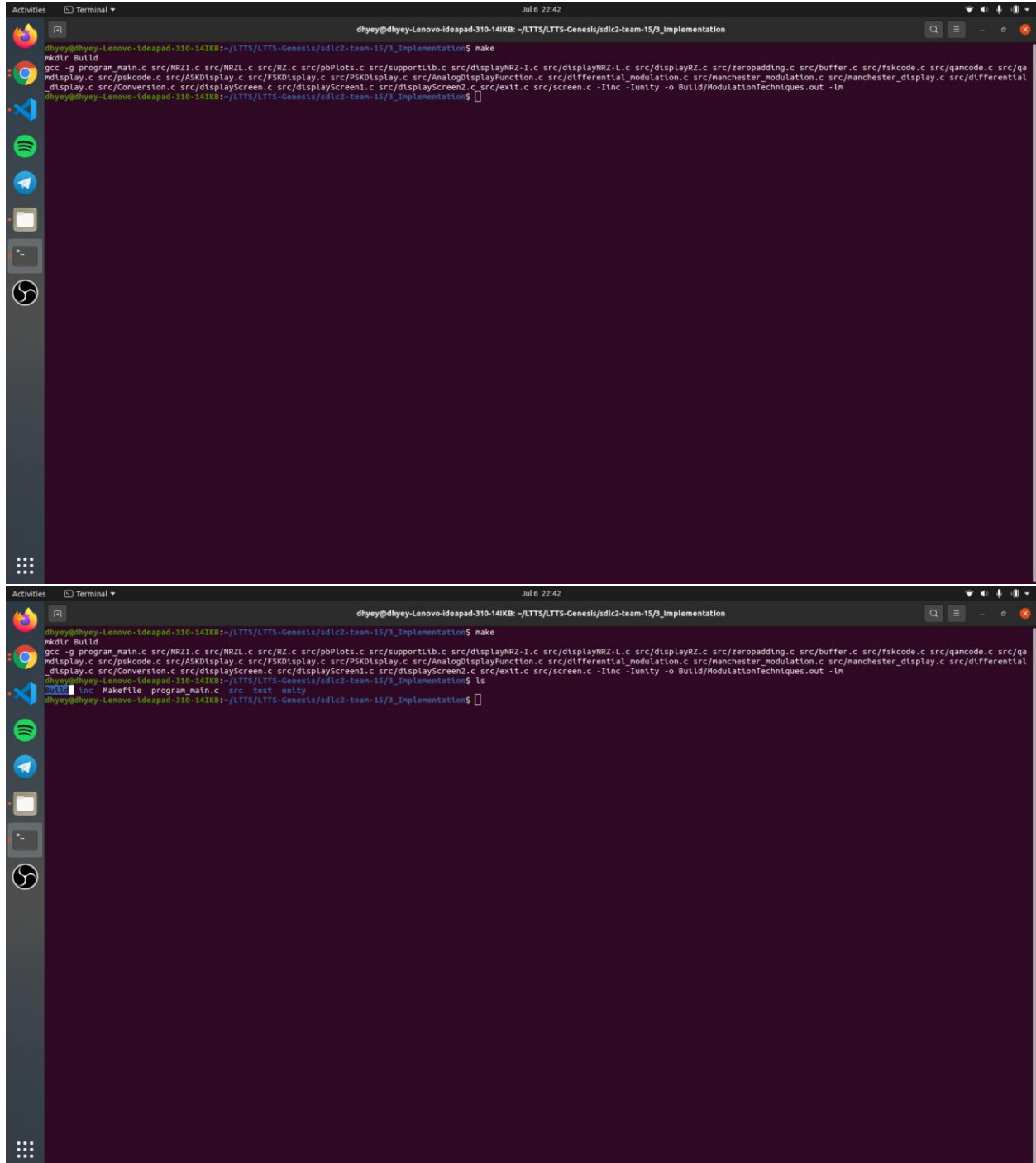


Display Waveforms Examples





Build



```
dhyy@dhyey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$ make
mkdir Build
gcc -g program_main.c src/NRZI.c src/NRZI.c src/RZ.c src/pbPlots.c src/supportlib.c src/displayNRZ-I.c src/displayNRZ-L.c src/displayRZ.c src/zeropadding.c src/buffer.c src/fskcode.c src/qamcode.c src/qamdisplay.c src/pskcode.c src/ASKDisplay.c src/FSKDisplay.c src/PSKDisplay.c src/AnalogDisplayFunction.c src/differential_modulation.c src/manchester_modulation.c src/manchester_display.c src/differential_display.c src/Conversion.c src/displayScreen.c src/displayScreen.c src/displayScreen2.c src/exit.c src/screen.c -lnc -lunity -o Build/ModulationTechniques.out -ln
dhyy@dhyey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$

dhyy@dhyey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$ make
mkdir Build
gcc -g program_main.c src/NRZI.c src/NRZI.c src/RZ.c src/pbPlots.c src/supportlib.c src/displayNRZ-I.c src/displayNRZ-L.c src/displayRZ.c src/zeropadding.c src/buffer.c src/fskcode.c src/qamcode.c src/qamdisplay.c src/pskcode.c src/ASKDisplay.c src/FSKDisplay.c src/PSKDisplay.c src/AnalogDisplayFunction.c src/differential_modulation.c src/manchester_modulation.c src/manchester_display.c src/differential_display.c src/Conversion.c src/displayScreen.c src/displayScreen.c src/displayScreen2.c src/exit.c src/screen.c -lnc -lunity -o Build/ModulationTechniques.out -ln
dhyy@dhyey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$ ls
Build  inc  Makefile  program_main.c  src  test  unity
dhyy@dhyey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$
```

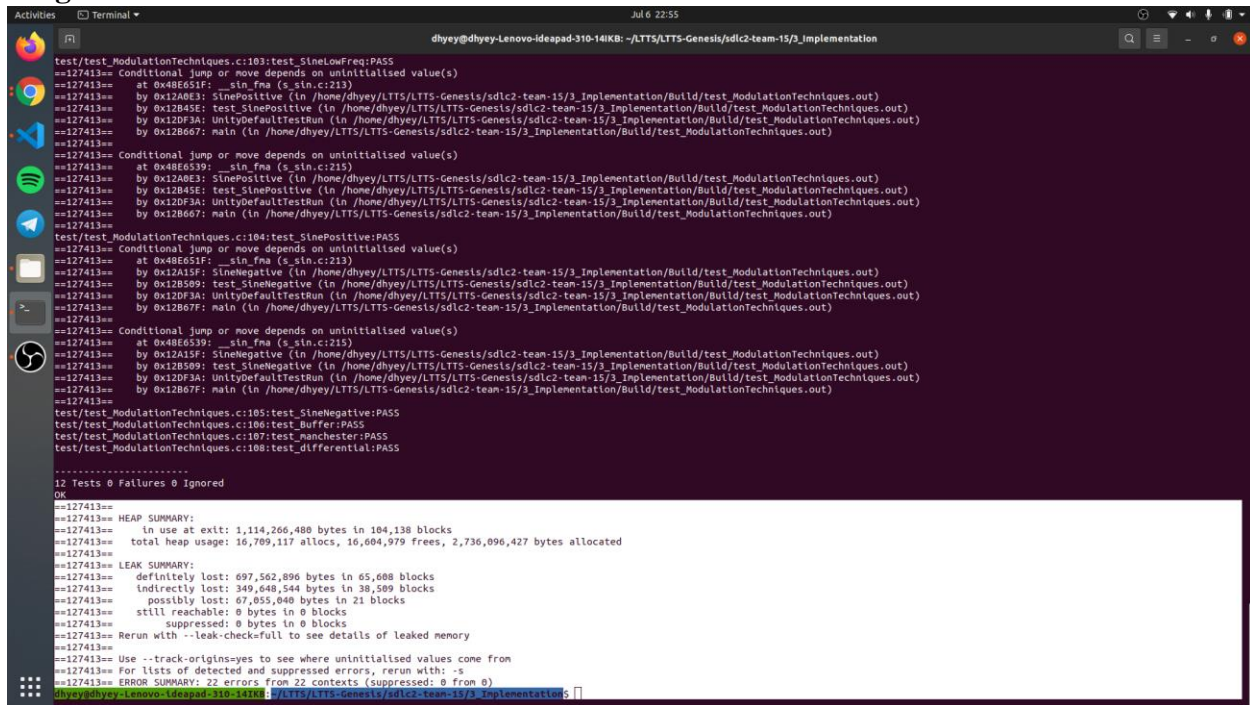
Unit Testing

```
dhvey@dhvey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$ make
mkdir Build
gcc -g program_main.c src/NRZI.c src/NRZL.c src/RZ.c src/pbPlots.c src/supportLib.c src/displayNRZ-I.c src/displayNRZ-L.c src/displayRZ.c src/zeropadding.c src/buffer.c src/fskcode.c src/qamcode.c src/qamdisplay.c src/pskcode.c src/ASKdisplay.c src/PSKdisplay.c src/AnalogDisplayFunction.c src/differential_modulation.c src/manchester_modulation.c src/manchester_display.c src/differential_display.c src/Conversion.c src/displayScreen.c src/displayScreen1.c src/displayScreen2.c src/exit.c src/screen.c -Iinc -Iunity -o Build/test_ModulationTechniques.out
dhvey@dhvey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$ ls
Build inc Makefile program_main.c src test unity
dhvey@dhvey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$ make test
gcc src/NRZI.c src/NRZL.c src/RZ.c src/pbPlots.c src/supportLib.c src/displayNRZ-I.c src/displayNRZ-L.c src/displayRZ.c src/zeropadding.c src/buffer.c src/fskcode.c src/qamcode.c src/qamdisplay.c src/pskcode.c src/ASKdisplay.c src/PSKdisplay.c src/AnalogDisplayFunction.c src/differential_modulation.c src/manchester_modulation.c src/manchester_display.c src/differential_display.c src/Conversion.c src/displayScreen.c src/displayScreen1.c src/displayScreen2.c src/exit.c src/screen.c test/test_ModulationTechniques.c unity/unity.c -Iinc -Iunity -o Build/test_ModulationTechniques.out
test/test_ModulationTechniques.c:96:test_NRZI:PASS
test/test_ModulationTechniques.c:97:test_NRZL:PASS
test/test_ModulationTechniques.c:99:test_zeropadding:PASS
test/test_ModulationTechniques.c:100:test_SineHighAmp:PASS
test/test_ModulationTechniques.c:101:test_SineLowAmp:PASS
test/test_ModulationTechniques.c:102:test_SineHighFreq:PASS
test/test_ModulationTechniques.c:103:test_SineLowFreq:PASS
test/test_ModulationTechniques.c:104:test_SinePositive:PASS
test/test_ModulationTechniques.c:105:test_SineNegative:PASS
test/test_ModulationTechniques.c:106:test_Buffer:PASS
test/test_ModulationTechniques.c:107:test_manchester:PASS
test/test_ModulationTechniques.c:108:test_differential:PASS
-----
12 Tests 0 Failures 0 Ignored
OK
dhvey@dhvey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$
```

GCC Coverage

```
dhvey@dhvey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$ make coverage
gcc -fprofile-arcs -ftest-coverage -fPIC -O0 test/test_ModulationTechniques.c unity/unity.c -Iinc -Iunity src/NRZI.c src/NRZL.c src/RZ.c src/pbPlots.c src/supportLib.c src/displayNRZ-I.c src/displayNRZ-L.c src/displayRZ.c src/zeropadding.c src/buffer.c src/fskcode.c src/qamcode.c src/qamdisplay.c src/pskcode.c src/ASKdisplay.c src/PSKdisplay.c src/AnalogDisplayFunction.c src/differential_modulation.c src/manchester_modulation.c src/manchester_display.c src/differential_display.c src/Conversion.c src/displayScreen.c src/displayScreen1.c src/displayScreen2.c src/exit.c src/screen.c -o Build/test_ModulationTechniques.out
dhvey@dhvey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$ ls
Build inc Makefile program_main.c src test unity
dhvey@dhvey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$ make test
gcc src/NRZI.c src/NRZL.c src/RZ.c src/pbPlots.c src/supportLib.c src/displayNRZ-I.c src/displayNRZ-L.c src/displayRZ.c src/zeropadding.c src/buffer.c src/fskcode.c src/qamcode.c src/qamdisplay.c src/pskcode.c src/ASKdisplay.c src/PSKdisplay.c src/AnalogDisplayFunction.c src/differential_modulation.c src/manchester_modulation.c src/manchester_display.c src/differential_display.c src/Conversion.c src/displayScreen.c src/displayScreen1.c src/displayScreen2.c src/exit.c src/screen.c test/test_ModulationTechniques.c unity/unity.c -Iinc -Iunity -o Build/test_ModulationTechniques.out
test/test_ModulationTechniques.c:96:test_NRZI:PASS
test/test_ModulationTechniques.c:97:test_NRZL:PASS
test/test_ModulationTechniques.c:99:test_zeropadding:PASS
test/test_ModulationTechniques.c:100:test_SineHighAmp:PASS
test/test_ModulationTechniques.c:101:test_SineLowAmp:PASS
test/test_ModulationTechniques.c:102:test_SineHighFreq:PASS
test/test_ModulationTechniques.c:103:test_SineLowFreq:PASS
test/test_ModulationTechniques.c:104:test_SinePositive:PASS
test/test_ModulationTechniques.c:105:test_SineNegative:PASS
test/test_ModulationTechniques.c:106:test_Buffer:PASS
test/test_ModulationTechniques.c:107:test_manchester:PASS
test/test_ModulationTechniques.c:108:test_differential:PASS
-----
12 Tests 0 Failures 0 Ignored
OK
dhvey@dhvey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$ gcov test_ModulationTechniques.gc
File 'test/test_ModulationTechniques.c'
Lines executed:100.00% of 81
Creating 'test_ModulationTechniques.c.gcov'
dhvey@dhvey-Lenovo-Ideapad-310-14IKB: ~/LTTS/LTTS-Genesis/sdlc2-team-15/3_Implementation$
```

Valgrind



```
dhvey@dhvey-Lenovo-ideapad-310-14IKB: ~/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation
test/test_ModulationTechniques.c:103:test_SineLowFreq:PASS
==127413== Conditional jump or move depends on uninitialised value(s)
==127413== at 0x48E651F: __sin_fma (s_sin.c:213)
==127413== by 0x12A0E3: SinePositive (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12B45E: test_SinePositive (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12DF3A: UnityDefaultTestRun (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12B667: main (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413==
==127413== Conditional jump or move depends on uninitialised value(s)
==127413== at 0x48E6539: __sin_fma (s_sin.c:215)
==127413== by 0x12A0E3: SinePositive (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12B45E: test_SinePositive (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12DF3A: UnityDefaultTestRun (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12B667: main (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413==
test/test_ModulationTechniques.c:104:test_SinePositive:PASS
==127413== Conditional jump or move depends on uninitialised value(s)
==127413== at 0x48E651F: __sin_fma (s_sin.c:213)
==127413== by 0x12A15F: SineNegative (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12B595: test_SineNegative (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12DF3A: UnityDefaultTestRun (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12B67F: main (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413==
==127413== Conditional jump or move depends on uninitialised value(s)
==127413== at 0x48E6539: __sin_fma (s_sin.c:215)
==127413== by 0x12A15F: SineNegative (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12B595: test_SineNegative (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12DF3A: UnityDefaultTestRun (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413== by 0x12B67F: main (ln /home/dhvey/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation/Build/test_ModulationTechniques.out)
==127413==
test/test_ModulationTechniques.c:105:test_SineNegative:PASS
test/test_ModulationTechniques.c:106:test_Buffer:PASS
test/test_ModulationTechniques.c:107:test_manchester:PASS
test/test_ModulationTechniques.c:108:test_differential:PASS
-----
12 Tests 0 Failures 0 Ignored
OK
==127413==
==127413== HEAP SUMMARY:
==127413==   in use at exit: 1,114,266,480 bytes in 104,130 blocks
==127413==   total heap usage: 16,709,117 allocs, 16,604,979 frees, 2,736,096,427 bytes allocated
==127413==
==127413== LEAK SUMMARY:
==127413==   definitely lost: 697,562,896 bytes in 65,608 blocks
==127413==   indirectly lost: 349,048,544 bytes in 38,599 blocks
==127413==   possibly lost: 67,055,040 bytes in 21 blocks
==127413==   still reachable: 0 bytes in 0 blocks
==127413==   suppressed: 0 bytes in 0 blocks
==127413== Rerun with --leak-check=full to see details of leaked memory
==127413==
==127413== Use --track-origins=yes to see where uninitialised values come from
==127413== For lists of detected and suppressed errors, rerun with: -s
==127413== ERROR SUMMARY: 22 errors from 22 contexts (suppressed: 0 from 0)
dhvey@dhvey-Lenovo-ideapad-310-14IKB: ~/LTS/LTS-Genesis/sdlc2-team-15/3_Implementation
```

References

- [Different Modulation Techniques](#)
- [How to plot in C](#)