**Digital Communication System Lab File**

**Group Members:**

Vedant Milind Athavale ( 181090071 ) (+91 9004188281) (vmathavale\_b18@et.vjti.ac.in)

Aditi Patil (181091003) (+91 9326566734) (aapatil\_b18@et.vjti.ac.in)

Tejas Adhikari (181090068) (+91 9987117186) (tsadhikari\_b18@et.vjti.ac.in)

Dhruv Singh ( 181090064 ) (+91 9527723605) (dysingh\_b18@et.vjti.ac.in)

Anchal Thool (181091009) (+919359788404) (aathool\_b18@et.vjti.ac.in)

**List of experiments:**

**1**.Implementation of (n, k) Hamming code encoder/ decoder.

**2**.Check whether a given set of n-tuple vectors are linearly dependent or independent.

**3**.Check if the given degree m polynomial is primitive or not.

**4**.Find the primitive elements from the given set of mod-n.

**5**.Construct generator polynomial for (n, k) cyclic code.

**6**.Construct generator matrix from Generator polynomial in systematic form.

**7**.Implement BASK, BPSK, BFSK modulation techniques.

**8**.Implement BASK, BPSK, BFSK demodulation techniques.

**9**.Implement M-aryASK, PSK, FSK modulation techniques.

**10**.Implement M-aryASK, PSK, FSK demodulation techniques.

**Lab Codes:  
  
Click Here on the link:** [**Link to GitHub For DCS Codes**](https://github.com/vedant-milind/Digital-Communication-Systems-Lab-Codes)  
**Click Here on the link:** [**Link to Google Drive For DCS Codes**](https://drive.google.com/drive/folders/1VYgAQbLejp9MX4x-OowgNAs5vsscjJ_6)

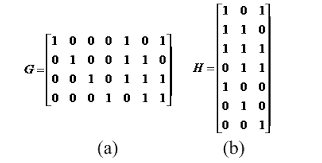
**Experiment No: 1**

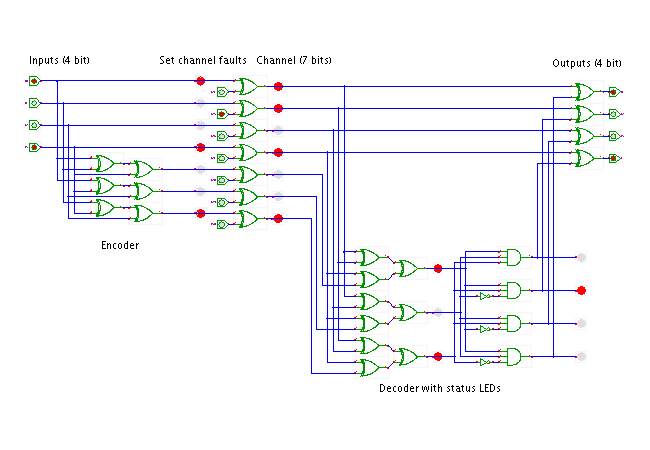
**Aim**:- Implementation of (n,k) Hamming code encoder and decoder.

**Software Requirements**:-Matlab or Scilab.

**Introduction**: In telecommunication, Hamming codes are a family of linear error-correcting codes. Hamming codes can detect up to two-bit errors or correct one-bit errors without detection of uncorrected errors. By contrast, the simple parity code cannot correct errors, and can detect only an odd number of bits in error. Hamming codes are perfect codes, that is, they achieve the highest possible rate for codes with their block length and minimum distance of three. Richard W. Hamming invented Hamming codes in 1950 as a way of automatically correcting errors introduced by punched card readers.  
 .

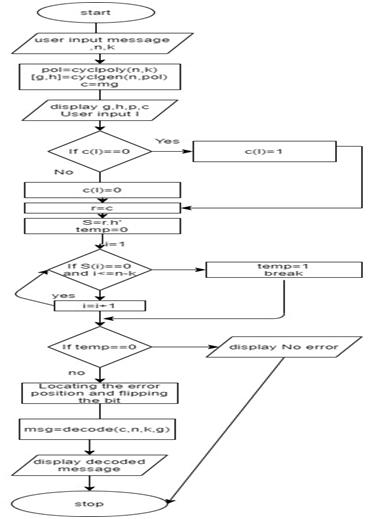
Traditional Hamming codes are (7, 4) codes, encoding four bits of data into seven bit blocks (a Hamming code word). The extra three bits are parity bits. Each of the three parity bits are parity for three of the four data bits, and no two parity bits are for the same three data bits. All of the parity bits are even parity.

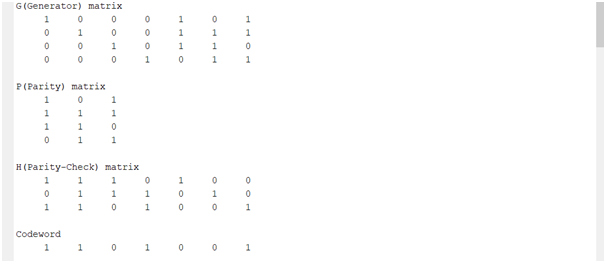


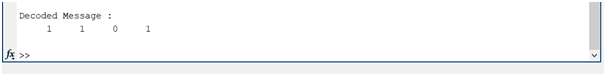
This matrix G is called the generating matrix. Given G, a message m then gets encoded as c = mG. To check whether the received message is correct or not, if CHT=0, then the received message is correct, else it's faulty.  
  
**Block Diagram:**

**Algorithm**:   
  
1. Take the length of codeword(n) and number of parity bits(k) and the message m as input from user.  
2. The coefficients of generator polynomial are obtained by pol=cyclpoly(n,k).  
3. The generator and parity-check matrices are obtained with cyclgen(n,pol).  
4. Parity matrix is obtained by slicing the generator matrix.  
5. The codeword (c) is obtained by c=m\*G.  
6. Take the position of the bit in codeword to be flipped as input from user(l).  
7. Flip the bit at position l in codeword and denote the result by r.  
8. Get syndrome S by S=r\*h’.  
9. Check if the elements of S are equal to any column of h’ and when found note the position of that column as errors.  
10. Flip the bit at errors in r to get the codeword without error.  
11. The initial message is obtained by using the function msg=decode(c,n,k,'linear/binary',g).

**Flowchart**:

****

**Output:**  
  
  
  




**Result:** (n,k) Hamming code encoder and decoder have been implemented using the matlab code.  
 **Proposed Hardware:** Laptop or Computer Desktop.

**Conclusion**: Thus, we understood the Hamming code encoding and decoding techniques. We also wrote the Matlab code to implement the (n,k) Hamming code encoder and decoder.

**Experiment No: 2**

**Aim:**- Check whether a given set of n-tuple vectors are linearly dependent or independent.

**Requirements:**- Python3 /Matlab.

**Introduction of topic with block diagram:** Let v1, v2, v3 …. vk, be k vectors in vector space over a field F. The set of all linear combinations of v1, v2, v3 …. vk forms a subspace of V. A set of vectors v1, v2, v3 …. vk in a vector space V over a field F is said to be linearly dependent if and only if there exist k scalars a1, a2, a3, …., ak from field F, not all zero such that,

A set of vectors v1, v2, v3 …. vk is said to be linearly independent if it is not linearly dependent.

That is v1, v2, v3 …. vk are linearly independent if and only if Unless a1 = a2 = … = ak = 0

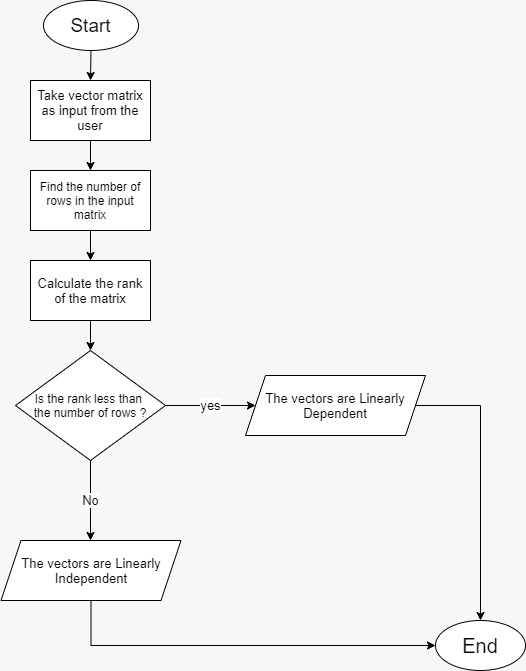
Example – Consider the vector space of all 5-tuple over GF(2) the linear combinations of (00111) & (11101) are –

Set of vectors are linearly independent.

**Algorithm:**

1. Import the necessary libraries
2. Input the number of vectors in the tuple and then input the vectors
3. Generate the sets of coefficients i.e. a1, a2, a3, …., an
4. Iterate through all the coefficients.
5. Multiply the coefficient with the last bit of each vector. Next iteration multiply with the next left bit.
6. Perform mod-2 operation on the sum and append it to the string of the final sum result.
7. Keep repeating till we complete the adding the bits.
8. If the sum equals “0000” set flag to 1.
9. If the flag is equal to 1, the set of vectors is linearly dependent. Otherwise it is linearly independent.

**Flowchart:**



**Output:**   
  
**CASE #1**

Enter the vectors in the form[v1;v2...;vn]

[1 1 1 1

1 0 0 1

1 0 1 1]

vectors =

1 1 1 1

1 0 0 1

1 0 1 1

rank = 3

The vectors are Linearly Independent

**CASE #2**

Enter the vectors in the form[v1;v2...;vn]  
  
[1 1 1 1

1 0 0 1

1 1 1 1]

vectors =

1 1 1 1

1 0 0 1

1 1 1 1

rank = 2

The vectors are Linearly Dependent

**Result:** Thus our program checks whether a given set of n-tuple vectors are linearly dependent or independent.

**Proposed Hardware**: Laptop or Computer Desktop.

**Conclusion**: We have understood the difference between a linearly dependent and independent set of vectors and written the script in Matlab to determine whether a set of vectors is linearly dependent or independent.

**Experiment No: 3**

**Aim**:- To check if the given degree m polynomial is primitive or not..

**Requirements**:- Matlab or Scilab.

**Introduction of topic with block diagram**:

A polynomial F(X) with coefficients in GF (p) = Z/pZ is a primitive polynomial if its degree is m and it has a root α in GF (pm) such that {0, 1, α, α2, α3, ..., αpm-2} is the entire field GF (pm). This means also that α is a primitive (pm − 1)-root of unity in GF (pm).

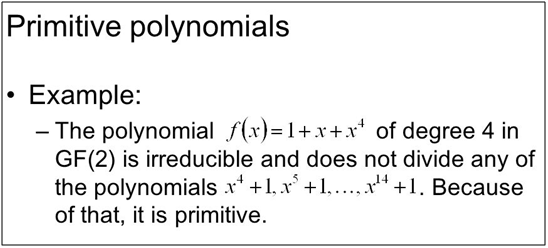
An irreducible polynomial F(x) of degree m over GF(p), where p is prime, is a primitive polynomial if the smallest positive integer n such that F(x) divides xn − 1 is n = pm − 1.

Because all minimal polynomials are irreducible, all primitive polynomials are also irreducible.

Primitive polynomials are used in the representation of elements of a finite field. If α in GF (pm) is a root of a primitive polynomial F(x) then since the order of α is pm − 1 that means that all nonzero elements of GF (pm) can be represented as successive powers of α:

GF (pm) = {0, 1, α, α2, α3, ..., αpm-2} {\displaystyle \mathrm {GF} (p^{m})=\{0,1,\alpha ,\alpha ^{2},\ldots ,\alpha ^{p^{m}-2}\}.}

When these elements are reduced modulo F(x), they provide the polynomial basis representation of all the elements of the field.



**Algorithm:**

· Take input for degree of the polynomial and store in variable m.

· Create an array a to store the coefficients of the polynomial of length m+1.

· Run a for loop to take the polynomial coefficients as input from the user with m+1 iterations and store them in array a.

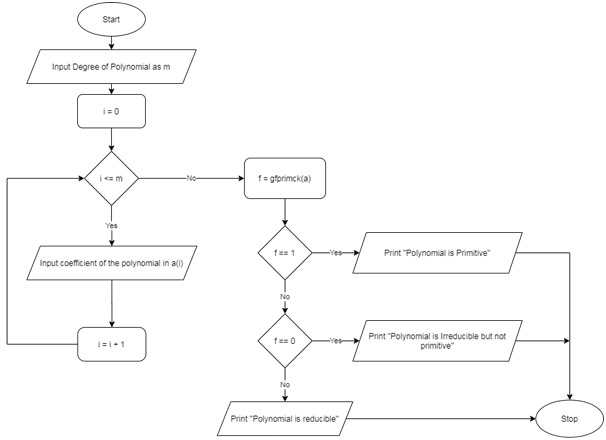
· Pass the array a as input argument in the gfprimck() and store its output in the variable f.

· If the value of f is 1 then the polynomial is primitive and print this statement.

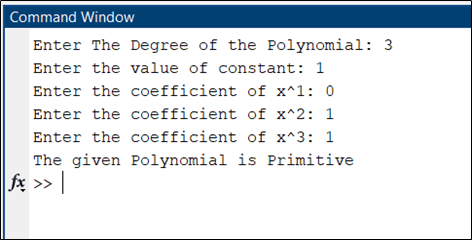
· Else if the value of f is 0 then the polynomial is irreducible but not primitive and print this statement.

· Else if f is -1 or has any other value then the polynomial is reducible and print this statement.

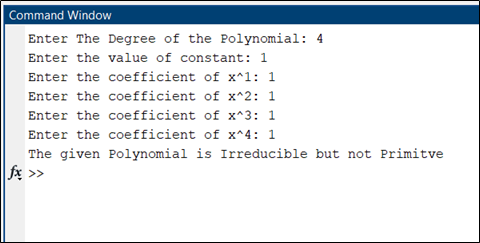
**Flowchart:**



**Output 1:**



**Output 2:**



**Result:** Thus for given polynomial, the program detects whether the polynomial is primitive or not .

**Proposed Hardware:** Laptop or Computer Desktop. **Conclusion:** Thus, we studied primitive polynomials and successfully implemented it in MATLAB.

**Experiment No: 4**

**Aim**:- To find the primitive elements from the given set of mod-n.  
**Requirements**: Matlab or Scilab.

**Introduction of topic with block diagram:**

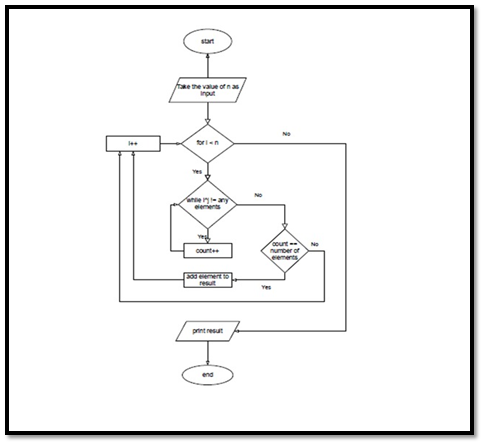
Primitive elements are elements which when raised to power can give all the elements present in the set except zero.

In field theory, a primitive element of a finite field GF(*q*) is a generator of the multiplicative group of the field. In other words, *α* ∈ GF(*q*) is called a primitive element if it is a primitive (q − 1)th root of unity in GF(*q*); this means that each non-zero element of GF(*q*) can be written as *αi* for some integer *i*.

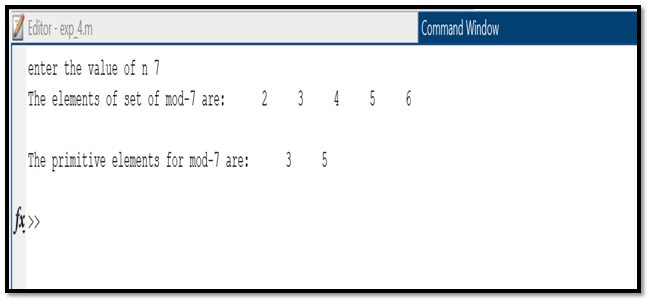
Example of Finding a Primitive Polynomial: Find the primitive elements from the GF(16)

α 5 = α 3 + α 2 = 1 + α + α 2  
α 6 = α 3 + α 2 + α= α + 1 + α 2 + α = 1 + α 2  
α 7 = α 3 + α = α + 1 + α = 1  
GF(23 )=GF(8) = {0, 1, α, α 2 , α + 1 , α 2 + α, α 2 + α + 1, 1 + α 2 }  
GF(23 )=GF(8) = {0, 1, α, α 2, ......., α 6 }  
  
**Algorithm:**1. Input the required set  
2. Input the n for mod n operation  
3. Use isPrimitiveRoot(G,n) function, which takes set G and n as input and returns a logical array of primitive elements (1 for true 0 for false)  
4. Print the primitive elements

**Flowchart:**



**Output:**



**Result:** Thus, if we consider the set {2,3,4,5,6}, the primitive elements under mod-7 operation are 3&7 which is the program output.

**Proposed Hardware:** Laptop or Computer Desktop.

**Conclusion:**.Thus, we understood the theory behind Primitive Elements, drew a flowchart for a code to generate primitive elements under mod-7 along with an algorithm. We demonstrated the output by writing a code in Matlab for the same.

**Experiment No: 5**

**Aim:**- To construct generator polynomials for (n, k) cyclic code.

**Requirements**: Matlab or Scilab

**Introduction of topic with block diagram:**

Cyclic codes are a subset of linear codes.

A binary code is said to be cyclic if it follows 2 rules:

**Linearity Property** : the sum of any 2 code words is a code itself

**Cyclic Property** : A code word when shifter is a code word too

A polynomial code is a linear code having a set of valid code words that comprises polynomials divisible by a shorter fixed polynomial is known as generator polynomial.

They are used for error detection and correction during the transmission of data as well as storage of data.

Addition and Subtraction: According to the rules of finite field theory, in modulo 2 arithmetic, addition and subtraction has no carries or borrows. Thus, both the operations are same as XOR (exclusive OR) operations

When messages are encoded using polynomial code, a fixed polynomial called generator polynomial,(𝑥) is used. The length of (𝑥) should be less than the length of the messages it encodes.

**Algorithm:**

1) Take value of n as input from user

2) Take value of k as input from user

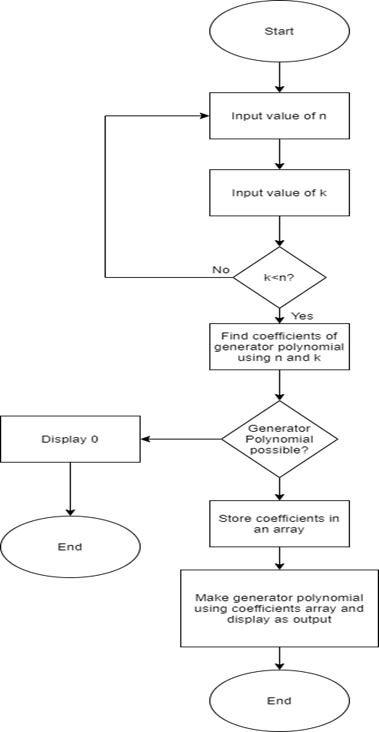
3) If n<k ask user to re-enter the values

4) Find coefficients of generator polynomial using the values of n and k

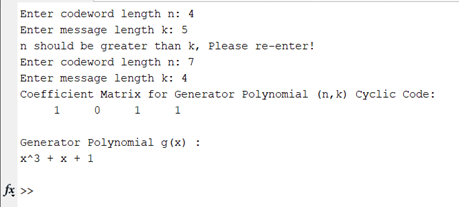
5) Store values in an array if found else, give output as 0

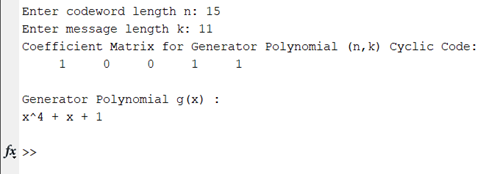
6) Convert the given array of coefficients to polynomial and display as output

**Flowchart:**



**Output:**





**Result**: Thus, the generator polynomial is obtained.

**Hardware Proposed:** Laptop or Computer Desktop.

**Conclusion**: Through this experiment we understood how to construct Generator polynomial for (n,k) cyclic code.

**Experiment No: 6**

**Aim:-** To construct generator matrix from Generator polynomial in systematic form.

**Requirements:-** Matlab.

**Theory:-**

Given the generator polynomial G(X) of an (n,k) cyclic code, we can construct a Generator Matrix.

This generator matrix G is constructed by noting that the k polynomials G(X), XG(X)....,X^(k-1).

Hence,the n-tuples corresponding to these polynomials is used as the rows of k-by-n matrix called Generator matrix.

**Algorithm:-**

· Take input for the codeword length and for the message bits.

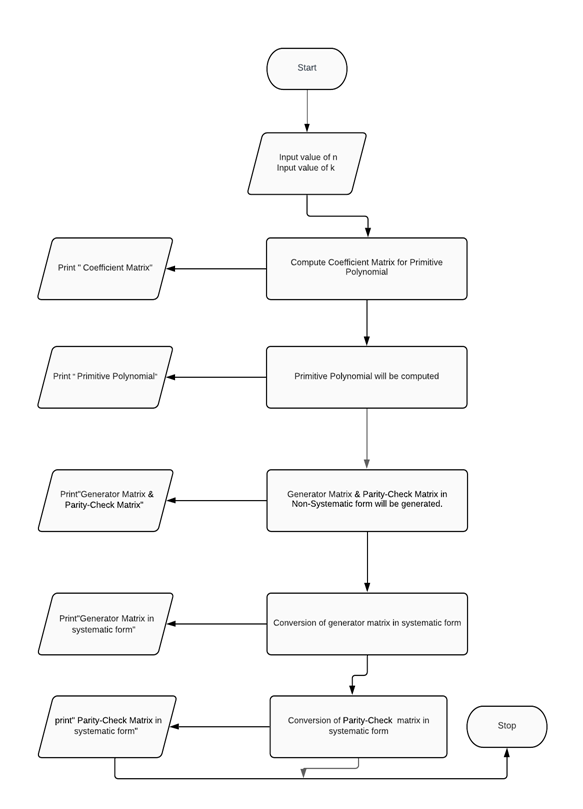
· We first compute Coefficient Matrix for Primitive Polynomial (n,k) Cyclic Code.

· Coefficient Matrix will be then computed to give Primitive Polynomial for (n,k) Cyclic Code.

· Generator Matrix & Parity-Check Matrix for (n,k) Cyclic Code in Non-Systematic form will be further generated.

· Finally, Generator Matrix & Parity-Check Matrix for (n,k) Cyclic Code in Systematic form will computed using their non-systematic form respectively.

**Flowchart:**

****

**Output:**

**1)**

Enter value of n: 7

Enter value of k: 4

Coefficient Matrix for Primitive Polynomial (n,k) Cyclic Code:

1 0 1 1

Primitive Polynomial for (n,k) Cyclic Code:

x^3 + x + 1

Generator Matrix for (n,k) Cyclic Code (Non-Systematic form):

1 0 1 1 0 0 0

0 1 0 1 1 0 0

0 0 1 0 1 1 0

0 0 0 1 0 1 1

Parity-Check Matrix for (n,k) Cyclic Code (Non-Systematic form):

1 1 1 0 1 0 0

0 1 1 1 0 1 0

0 0 1 1 1 0 1

Generator Matrix for (n,k) Cyclic Code (Systematic form):

1 0 0 0 1 0 1

0 1 0 0 1 1 1

0 0 1 0 1 1 0

0 0 0 1 0 1 1

Parity-Check Matrix for (n,k) Cyclic Code (Systematic form):

1 0 0 1 1 1 0

0 1 0 0 1 1 1

0 0 1 1 1 0 1

**2)**

Enter value of n: 6

Enter value of k: 3

Coefficient Matrix for Primitive Polynomial (n,k) Cyclic Code:

1 0 0 1

Primitive Polynomial for (n,k) Cyclic Code:

x^3 + 1

Generator Matrix for (n,k) Cyclic Code (Non-Systematic form):

1 0 0 1 0 0

0 1 0 0 1 0

0 0 1 0 0 1

Parity-Check Matrix for (n,k) Cyclic Code (Non-Systematic form):

1 0 0 1 0 0

0 1 0 0 1 0

0 0 1 0 0 1

Generator Matrix for (n,k) Cyclic Code (Systematic form):

1 0 0 1 0 0

0 1 0 0 1 0

0 0 1 0 0 1

Parity-Check Matrix for (n,k) Cyclic Code (Systematic form):

1 0 0 1 0 0

0 1 0 0 1 0

0 0 1 0 0 1

**Result:** Thus, we get the generator matrix from the generator polynomial.

**Propose your hardware for the same experiments:** Laptop or Computer Desktop.

**Conclusion:** We studied the concept of generator polynomial and the process to obtain Generator matrix using Generator polynomial.

**Experiment No: 7&8**

**Aim:**- To implement BASK, BPSK, BFSK modulation techniques and demodulation techniques.

**Requirements:**- Matlab or Scilab.

**Theory:-**

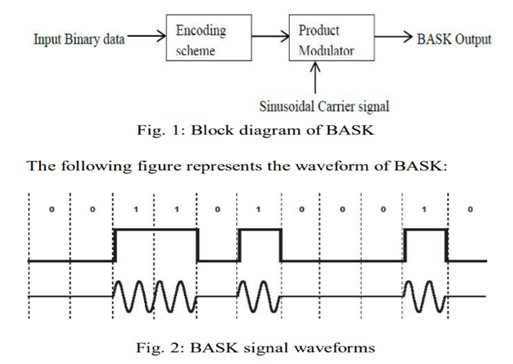
**BASK:**

Amplitude-shift keying(ASK) is a form of modulation that represents digital data as variations in the amplitude of a carrier wave.Any digital modulation scheme uses a finite number of distinct signals to represent digital data. ASK uses a finite number of amplitudes, each assigned a unique pattern of binary digits. Usually, each amplitude encodes an equal number of bits. Each pattern of bits forms the symbol that is represented by the particular amplitude. Thedemodulator, determines the amplitude of the received signal and maps it back to the symbol it represents, thus recovering the original data.Here, frequency and phase of the carrier are kept constant.A binary amplitude-shift keying(BASK) signal can be defined by

s(t) = A m(t) cos 2∏fct, 0 < t< T

where A is a constant, m(t) = 1 or 0, fc is the carrier frequency, and T is the bit duration.

The following figure represents the block diagram of BASK:



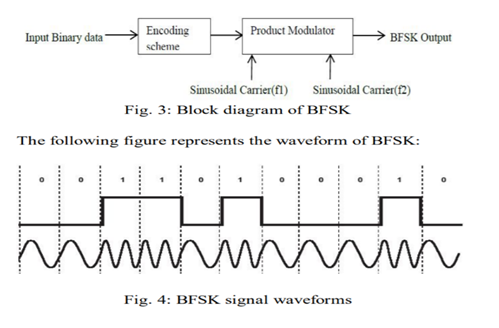
**BFSK:**

Frequency-shift keying (FSK) is a frequency modulation scheme in which digital information is transmitted through discrete frequency changes of a carrier wave. The simplest FSK isbinaryFSK(BFSK). BFSK uses a pair of discrete frequencies to transmit binary (0s and 1s) information.With this scheme, the "1" is called the mark frequency and the "0" is called the space frequency. A BFSK signal can be expressed as

s(t) = A sin 2∏[(fc+m(t)fm)+tⱷ0], 0 < t< T

Where, m(t) =0 or 1, binary messageT =bit durationA, fc, ⱷ0are the amplitude, frequency and phase of the sinusoidal carrier signal

The following figure represents the block diagram of BFSK:

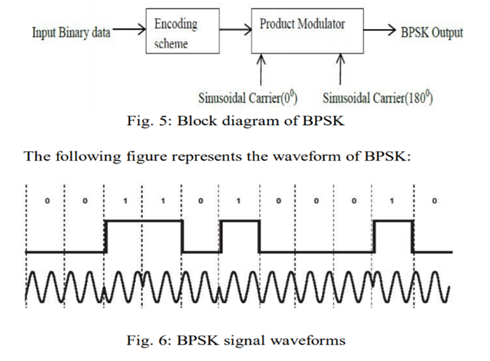


**BPSK:**

Phase Shift Keying (PSK) is a digital modulation scheme that conveys data by changing or modulating phase of carrier signal.BPSK is also called as (PRK) Phase Reversal Keying or 2PSK. It is the simplest form of PSK. It uses two phases which are separated by 180° and so can also be termed 2-PSK. It does not particularly matter exactly where the constellation points are positioned and they can be shown on the real axis at 0° and 180°.The general form for BPSK follows the equation:

s(t) =A cos (2∏fct+∏(1-n)), 0 < t< T

The following figure represents the block diagram of BFSK:



**BASK:**

**Algorithm**:

Procedure followed to obtain the BASK signal is as follows:

1) Define the transmitted signal

a) Variables to be defined are:- number of bits, size of transmitted signal,

b) binary signal 0 or 1 as a message to be transmitted, bit period (second)

2) Represent the input signal as digital signal

a) If input signal is 1 append in an array else if input signal is 0 append in array

b) Define time of the signal in MATLAB

c) Plot the transmitting signal

3) Define BASK modulation

a) Declare amplitude of carrier signal for bit 1 and 0

b) Declare carrier frequency

c) If input signal is 1 multiply with carrier signal and append in an array else if input signal is 0 multiply with carrier signal append in array

d) Plot modulated signal

4) Transmit Signal

**(Demodulation)**

a) Receive signal with/without noise

b) Define BASK demodulation

i) Define carrier signal as earlier

ii) Multiply received signal with carrier signal

iii) Integrate the received signal with lower frequency i.e pass signal through LPF

iv) Define logic level 1 or 0 depending upon the output amplitude

v) If received amplitude is greater than average amplitude then logic 1 else if received amplitude is less than average amplitude then logic 0

vi) Append solution in array

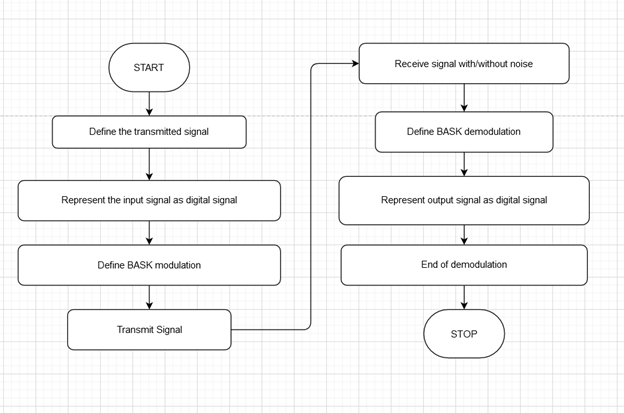
c) Represent output signal as digital signal

i) Define received signal as array

ii) Plot received signal

d) End of demodulation

**Flowchart:**



**BASK**

**BPSK:**

**Algorithm:**

Procedure followed to obtain the BPSK signal is as follows:

1) Define the transmitted signal

a) Variables to be defined are:- number of bits, size of transmitted signal,

b) binary signal 0 or 1 as a message to be transmitted, bit period (second).

2) Represent the input signal as digital signal

a) If input signal is 1 append in an array else if input signal is 0 append in array

b) Define time of the signal in MATLAB

c) Plot the transmitting signal

3) Define BPSK modulation

a) Declare amplitude of carrier signal

b) Declare carrier frequencies for bit 1

c) Declare carrier phase for 1 and 0 respectively

d) If input signal is 1 add carrier phase with carrier signal and append in an array else if input signal is 0 add carrier phase with carrier signal and append in array

e) Plot modulated signal

4) Transmit Signal

**(Demodulation)**

a) Receive signal with/without noise

b) Define BPSK demodulation

i) Define carrier signal as earlier

ii) Multiply received signal with carrier signal

iii) Integrate the received signal with lower frequency i.e pass signal through LPF

iv) Define logic level 1 or 0 depending upon the output amplitude

v) If received amplitude is greater than average amplitude then logic 1 else if received amplitude is less than average amplitude then logic 0

vi) As phase change of 90 was introduced for each bit, the signals either add or cancel each other out

vii) Append solution in array

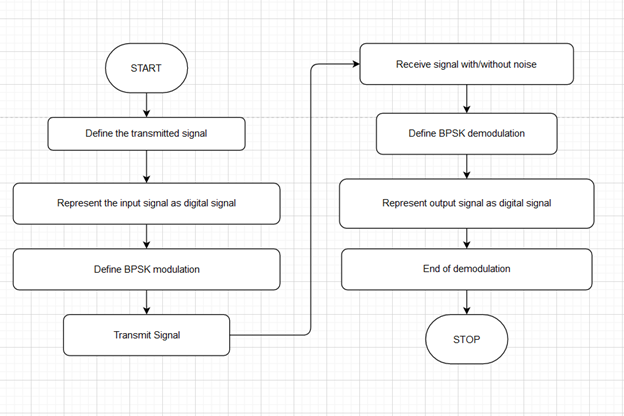
c) Represent output signal as digital signal

i) Define received signal as array

ii) Plot received signal

d) End of demodulation

**Flowchart;**



**BPSK**

**BFSK:**

**Algorithm:**

1) Procedure followed to obtain the BfSK signal is as follows:

2) Define the transmitted signal

a) Variables to be defined are:- number of bits, size of transmitted signal,

b) binary signal 0 or 1 as a message to be transmitted, bit period (second).

3) Represent the input signal as digital signal

a) If input signal is 1 append in an array

b) else if input signal is 0 append in array

c) Define time of the signal in MATLAB

d) Plot the transmitting signal

4) Define BFSK modulation

a) Declare amplitude of carrier signal

b) Declare carrier frequencies for bit 1 and 0

c) If input signal is 1 multiply with carrier signal frequency-1 and append in an array else if input signal is 0 multiply with carrier signal frequency-2 append in array

d) Plot modulated signal

5) Transmit Signal

**(Demodulation)**

a) Receive signal with/without noise

b) Define BFSK demodulation

i) Define carrier signal as earlier with respective frequencies

ii) Multiply received signal with carrier signal for both frequencies

iii) Integrate the received signal with frequencies 1 & 2 i.e pass signal through LPF

iv) of frequencies 1 & 2

v) Define logic level 1 or 0 depending upon the output amplitude by subtracting

vi) the amplitude for received signal of frequency 2 from frequency 1 OR If received amplitude is greater than half of carrier signal amplitude of frequency 1 then logic 1 else if received amplitude is greater than half of carrier signal amplitude of frequency 2 then logic 0

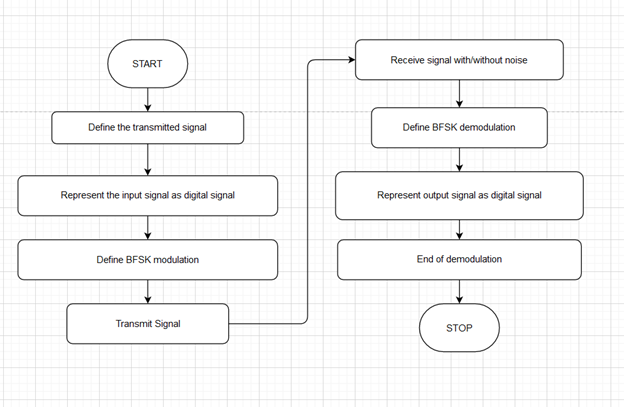
c) Represent output signal as digital signal

i) Define received signal as array

ii) Plot received signal

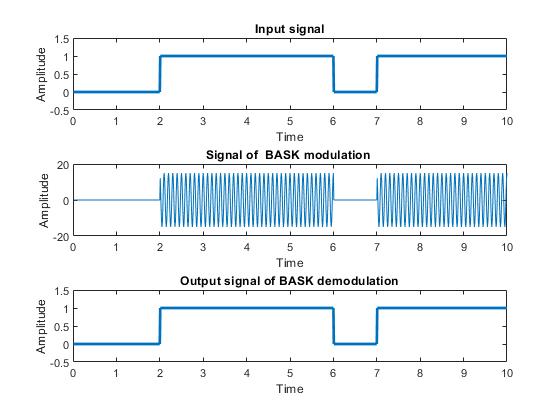
d) End of demodulation

**Flowchart:**

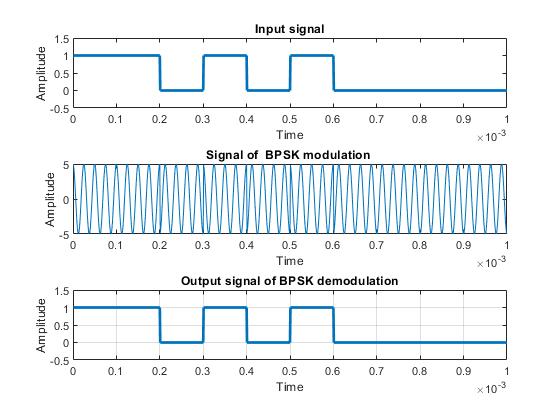


**BFSK**

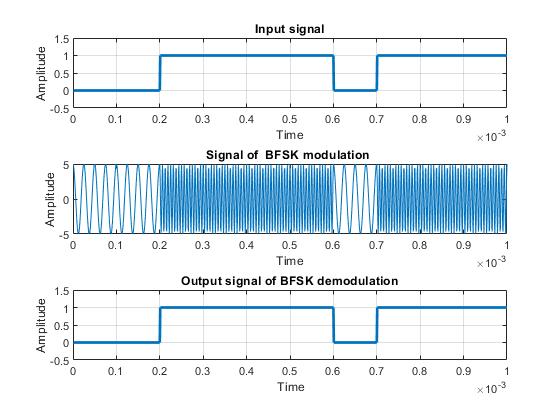
**Output:**

****

**BASK**

****

**BPSK**

****

**BFSK**

**Result:**

Thus, different types of digital modulation techniques i.e. Binary Amplitude shift Keying (BASK), Binary Phase Shift Keying (BPSK), Binary Frequency Shift Keying (BFSK)Have been studied using MATLAB.

**Propose your hardware for the same experiments:**

BASK:

Modulation:Input Binary Data, Sinusoidal carrier Signal,Product modulator

Demodulation: Received signal, Product demodulator, carrier signal

BFSK:

Modulation:Input Binary Data, Sinusoidal carrier 1 and 2 Signal, Product modulator

Demodulation: Received signal, Product demodulator, carrier signal

BPSK:

Modulation:Input Binary Data, Sinusoidal carrier 1 and phase shifted 2 Signal, Product modulator

Demodulation: Received signal, Product demodulator, carrier signal

**Conclusion:** Hence BASK, BPSK, BFSK modulation/demodulation techniques were implemented.

**Experiment No: 9&10**

**Aim:-** Implement M-aryASK, PSK, FSK modulation and demodulation techniques.

**Requirements:-** Multisim and Matlab

**Theory:-**

The word binary represents two bits. M represents a digit that corresponds to the number of conditions, levels, or combinations possible for a given number of binary variables.This is the type of digital modulation technique used for data transmission in which instead of one bit, two or more bits are transmitted at a time. As a single signal is used for multiple bit transmission, the channel bandwidth is reduced.

M-ary Equation

If a digital signal is given under four conditions, such as voltage levels, frequencies, phases, and amplitude, then M = 4.

The number of bits necessary to produce a given number of conditions is expressed mathematically as

N=log2M

Where

N is the number of bits necessary

M is the number of conditions, levels, or combinations possible with N bits.

The above equation can be rearranged as

2N=M

For example, with two bits, 22 = 4 conditions are possible.

Types of M-ary Techniques

In general, Multi-level M−ary modulation techniques are used in digital communications as the digital inputs with more than two modulation levels are allowed on the transmitter’s input. Hence, these techniques are bandwidth efficient.

There are many M-ary modulation techniques. Some of these techniques, modulate one parameter of the carrier signal, such as amplitude, phase, and frequency.

M-ary ASK

This is called M-ary Amplitude Shift Keying M−ASK or M-ary Pulse Amplitude Modulation :- PAM

The amplitude of the carrier signal, takes on M different levels.

Representation of M-ary ASK

Sm(t)=Amcos(2πfct)Amϵ(2m−1−M)Δ,m=1,2....M and 0≤t≤Ts

Some prominent features of M-ary ASK are −

This method is also used in PAM.

Its implementation is simple.

M-ary ASK is susceptible to noise and distortion.

M-ary FSK

This is called as M-ary Frequency Shift Keying M−ary FSK. The frequency of the carrier signal takes on M different levels.

Representation of M-ary FSK

Si(t)=2EsT−−−√cos(πTs(nc+i)t)0≤t≤Ts and i=1,2,3.....M

Where fc=nc2Ts

for some fixed integer n.

Some prominent features of M-ary FSK are −

Not susceptible to noise as much as ASK.

The transmitted M number of signals are equal in energy and duration.

The signals are separated by 1/2Ts Hz making the signals orthogonal to each other.

Since M signals are orthogonal, there is no crowding in the signal space.

The bandwidth efficiency of M-ary FSK decreases and the power efficiency increases with the increase in M.

M-ary PSK

This is called as M-ary Phase Shift Keying M−ary PSK. The phase of the carrier signal, takes on M different levels.

Representation of M-ary PSK

Si(t)=2ET−−√cos(wot+ϕit)0≤t≤T and i=1,2...M and ϕi(t)=2πiM where i=1,2,3......M

Some prominent features of M-ary PSK are −

The envelope is constant with more phase possibilities.

This method was used during the early days of space communication.

Better performance than ASK and FSK.

Minimal phase estimation error at the receiver.

The bandwidth efficiency of M-ary PSK decreases and the power efficiency increases with the increase in M.

So far, we have discussed different modulation techniques. The output of all these techniques is a binary sequence, represented as 1s and 0s. This binary or digital information has many types and forms, which are discussed further.

**ASK Algorithm:**

· The message signal is generated in binary using the rand() function and displayed. Bit array is generated using bit period as 0.000001 for plotting square wave graph.

· The message array is re-shaped using the reshape() function according to the given M. The reshaped message is converted into symbolic form and displayed.

· The symbolic form information is also displayed as a stem graph. The qammod() is used to obtain the constellation diagram for the given M.

· The constellation diagram is plotted using scatterplot()

· The qammod() is again used to obtain the modulated output of the symbols and stored in the variable p. The p variable is split into real part and imaginary part.

· The real part is multiplied by cosine wave and the imaginary part is multiplied by sine wave and added together.

· The modulated waveform is displayed as a graph plot.

· Noise is added to the modulated signal using the awgn() with signal to noise ratio as 15.

· The graph with noise is plotted on a new figure.

· The original signal is then obtained from the waveform using a for loop.

· The signal array is converted into the symbols using the qamdemod() and stored in the variable ax.

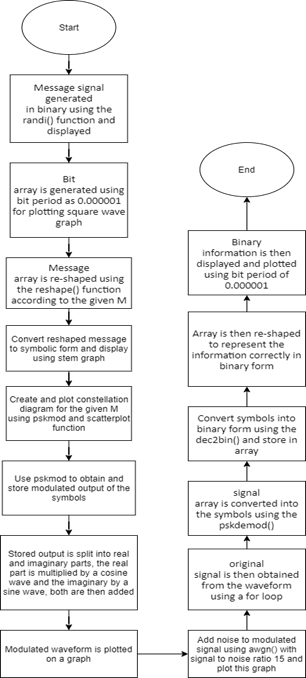
· The symbols are converted into binary form using the dec2bin() and stored in array bi\_in.

· The array bi\_in is then re-shaped to represent the information correctly in binary form.

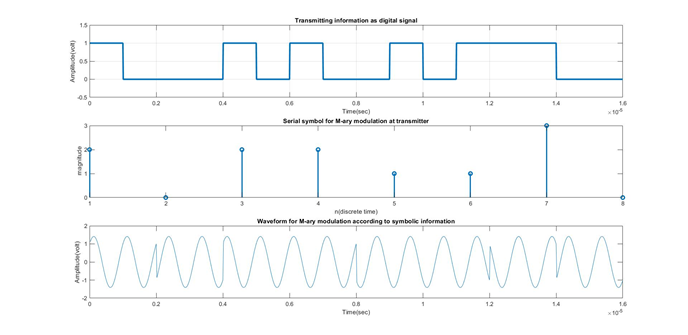
· The binary information is then displayed.

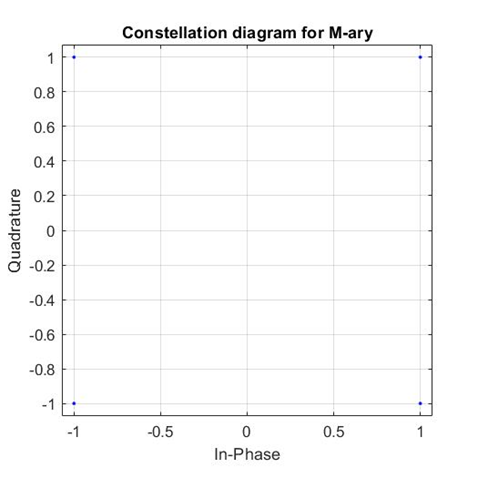
· The binary information is then plotted using bit period as 0.000001 for plotting square wave graph.

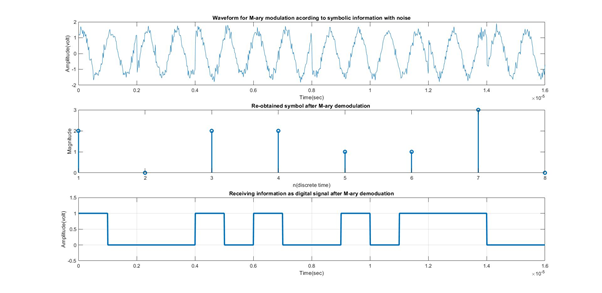
**ASK Flowchart:**



**ASK Output:**







**PSK Algorithm:**

· The message signal is generated in binary using the rand() function and displayed.

· Bit array is generated using bit period as 0.000001 for plotting square wave graph.

· The message array is re-shaped using the reshape() function according to the given M.

· The reshaped message is converted into symbolic form and displayed.

· The symbolic form information is also displayed as a stem graph.

· The pskmod() is used to obtain the constellation diagram for the given M.

· The constellation diagram is plotted using scatterplot()

· The pskmod() is again used to obtain the modulated output of the symbols and stored in the variable p.

· The p variable is split into real part and imaginary part.

· The real part is multiplied by cosine wave and the imaginary part is multiplied by sine wave and added together.

· The modulated waveform is displayed as a graph plot.

· Noise is added to the modulated signal using the awgn() with signal to noise ratio as 15.

· The graph with noise is plotted on a new figure.

· The original signal is then obtained from the waveform using a for loop.

· The signal array is converted into the symbols using the pskdemod() and stored in the variable ax.

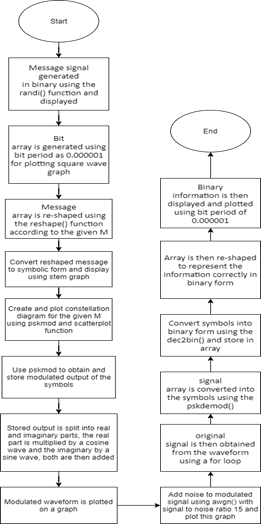
· The symbols are converted into binary form using the dec2bin() and stored in array bi\_in.

· The array bi\_in is then re-shaped to represent the information correctly in binary form.

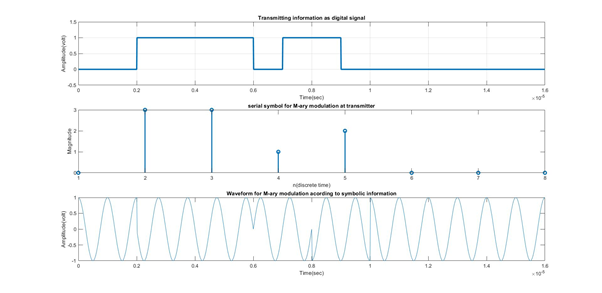
· The binary information is then displayed.

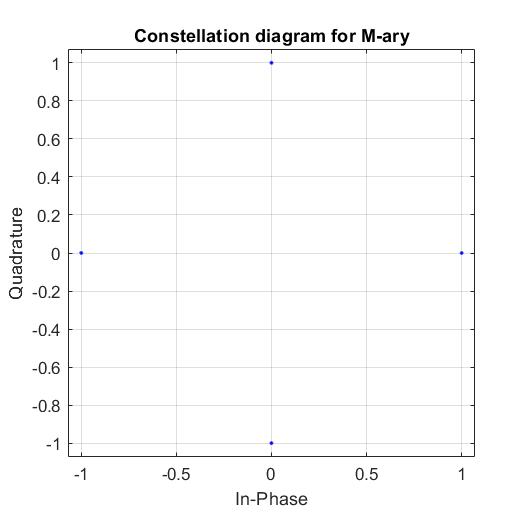
· The binary information is then plotted using bit period as 0.000001 for plotting square wave graph.

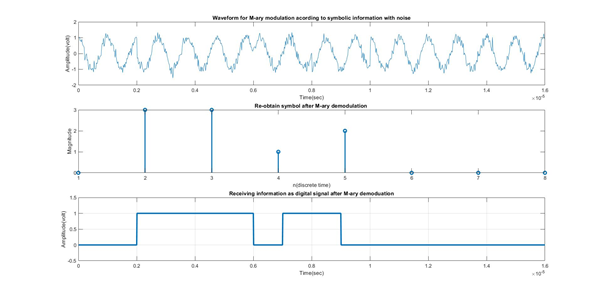
**PSK Flowchart:**



**PSK Output:**







**FSK Algorithm:**

· The message signal is generated in binary using the rand() function and displayed.

· Bit array is generated using bit period as 0.000001 for plotting square wave graph.

· The message array is re-shaped using the reshape() function according to the given M.

· The reshaped message is converted into symbolic form and displayed.

· The symbolic form information is also displayed as a stem graph.

· The fskmod() is used to obtain the constellation diagram for the given M.

· The constellation diagram is plotted using scatterplot()

· The fskmod() is again used to obtain the modulated output of the symbols and stored in the variable p.

· The p variable is split into real part and imaginary part.

· The real part is multiplied by cosine wave and the imaginary part is multiplied by sine wave and added together.

· The modulated waveform is displayed as a graph plot.

· Noise is added to the modulated signal using the awgn() with signal to noise ratio as 15.

· The graph with noise is plotted on a new figure.

· The original signal is then obtained from the waveform using a for loop.

· The signal array is converted into the symbols using the fskdemod() and stored in the variable ax.

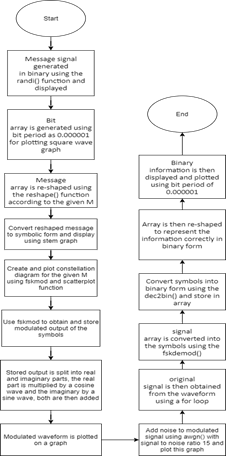
· The symbols are converted into binary form using the dec2bin() and stored in array bi\_in.

· The array bi\_in is then re-shaped to represent the information correctly in binary form.

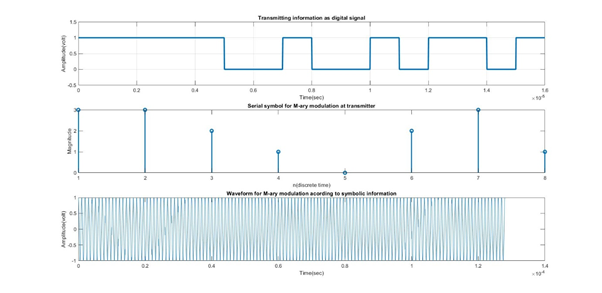
· The binary information is then displayed.

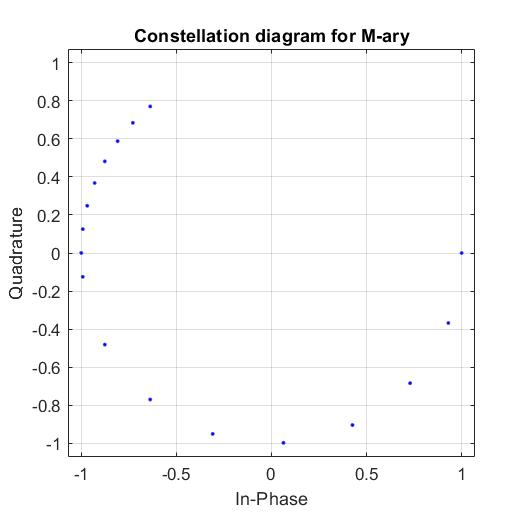
· The binary information is then plotted using bit period as 0.000001 for plotting square wave graph.

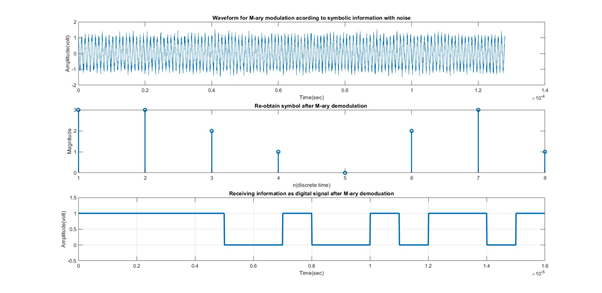
**FSK Flowchart:**



**FSK Output:**







**Result:** Implementing M-aryASK, PSK, FSK modulation and demodulation techniques was successful.

**Propose your hardware for the same experiments:**

M-ary ASK

Modulator:Input Binary Data, Sinusoidal carrier Signal, Encoding Sequence Product modulator

Demodulator:Received signal, Product demodulator, carrier signal

M-ary FSK

Modulator:Input Binary Data, Sinusoidal carrier 1,2... Signal,, Encoding Sequence Product modulator

Demodulator:Received signal, Product demodulator, carrier signal

M-ary PSK

Modulator:Input Binary Data, Sinusoidal carrier 1 phase sifted 2,3... Signal, Encoding Sequence, Product modulator(8PSK)

Demodulator:Received signal, Product demodulator(8PSK), carrier signal

**Conclusion:**Hence M-ARY ASK, M-ARY PSK, M-ARY FSK modulation/demodulation techniques were implemented.