Digital Modulation Techniques using a Graphic User Interface

Prof. Avani P. Patel[1], Prof. Vibhuti P. Patel[2]
[1, 2] Computer Engineering Department
[1, 2] GIDC Degree Engineering College, Abrama, Navsari

Abstract- *This paper presents Digital Modulation Techniques using a Graphic User Interface. In this four digital modulation techniques are used – Amplitude Shift Keying, Frequency Shift Keying, Binary Phase Shift Keying, Quadrature Phase Shift Keying. Digital Modulation Techniques play a very important role in all the communication systems used today. To fully understand the modulation techniques, they are implemented and can be easily understood by looking at GUI made using Matlab. Digital signal is taken as input and using carrier wave and the type of modulation the modulated signal is generated. Through the Graphical User Interface, it becomes easier to understand the techniques.*

Keywords- Modulation, amplitude, frequency, Quadrature, communication, intergration

I. INTRODUCTION

Digital Modulation Techniques using Graphic User Interface. In this project we have taken six digital modulation techniques – Amplitude Shift Keying, On Off Keying, Frequency Shift Keying, Binary Phase Shift Keying, Quadrature Phase Shift Keying, 8 Quadrature Amplitude Keying.

Digital Modulation Techniques play a very important role in all the communication systems used today. So it has become necessary to study the digital modulation techniques thoroughly. MATLAB is very useful software for engineers. It is widely used in all the engineering fields. To fully understand the digital modulation techniques that we are implementing and to become well versed with the MATLAB software and the procedure of making GUIs. When one sees the actual output waveform of the modulated signals, they can understand the concepts very easily.

II. MODULATION TECHNIQUES

Amplitude-shift keying: Amplitude-shift keying is a modulation technique in which there is variation in amplitude of the carrier waves. The amplitude varies but does not affect frequency and phase. The logic 0 and logic 1 are used to represent level of amplitude.

On-off keying: It is the simplest form of amplitude-shift keying modulation that represents digital data that is 1 or 0 as the presence and absence of carrier wave. It is mostly used to transmit radio frequencies and Morse code.

Frequency-shift keying: In this modulation scheme digital information is transferred to represent frequency changes of carrier waves. There is binary frequency shift keying in which binary 0 and binary 1 are used to represent mark frequency and space frequency.

Phase-shift keying: In this modulation scheme the data are represented by changing the phase of the reference signal that is carrier wave. PSK uses a finite number of phase, each been assigned a unique pattern of binary digits. Each phase encodes equal number of bits. Each pattern of bits represents a particular symbol that is unique for each phase.

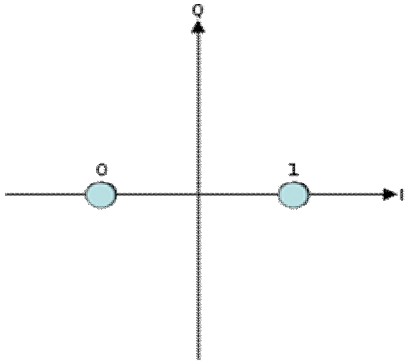


Figure 1. Constellation diagram of BPSK

BPSK is the simplest form of phase shift keying. There are two phases separated by 180◦ and so called as 2-PSK. This modulation is most robust as it takes the highest level of distortions. It is not suitable for high data-rate applications.

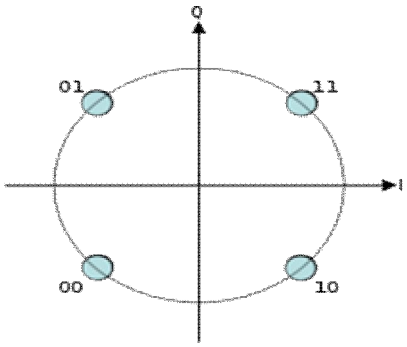


Figure 2. Constellation diagram of QPSK

Quadrature phase-shift keying refers to PSK with 4 states. Phase modulation is a version of frequency modulation where phase of carrier wave is modulated to encode bits of digital information in each phase change. The four phases in QPSK are: 450,1350,2250 and 2250 and 3150 degrees.

Table 1.

Phase	Data
45^0	Binary 00
135^0	Binary 01
225^0	Binary 10
315^0	Binary 11

Quadrature Amplitude modulation: Quadrature amplitude modulation is both analog and digital modulation scheme. Here two analog signals are modulated by changing the amplitude of two carrier waves using ASK scheme. These two carrier waves are out of phase with each other at an angle of 900 and so called quadrature components. The resultant waveform is a combination of both amplitude-shift keying and phase-shift keying.

III. GRAPHICAL USER INTERFACE IN MATLAB

What Is a GUI ?

A graphical user interface is a graphical display in one or more windows containing controls called components that help the user to perform interactive work. The user of GUI does not have to type commands at the command line to accomplish the task. The user needs not to understand the details of the coding and how work is performed. GUI components like menus, toolbars, radio buttons, list boxes and sliders are used and user needs to just drag and drop the components and display data as tables or as plots.

GUI Layout: The graphical user interface development environment provides a set of tools for creating graphical user interfaces (GUIs). These tools simplify the process of programing GUIs. One can create GUI by clicking and dragging the components such as axes, panels, text fields, sliders, and buttons into the layout area. Also menus can be created for the GUI.

GUI Programming: GUIDE automatically generates a matlab file containing the code of the how the GUI operates and the components placed. The code contains the GUI callbacks- these are routines that execute when a user interacts with a GUI component. So the code of how the actions are to be performed in GUI is placed in callback routines.

Starting GUIDE: There are many ways to start GUIDE. Following steps are to be followed:

- Command line by typing guide.
- Start menu by selecting MATLAB>GUIDE(GUI Builder)
- MATLAB File menu by selecting NEW> GUI
- MATLAB toolbar by clicking the GUIDE button

It will display the Quick start dialog box as shown in the following figure.

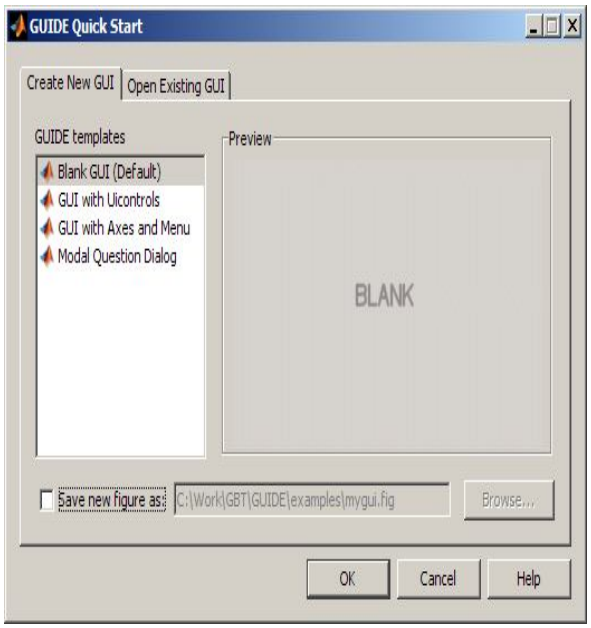


Figure 3.

The GUIDE Quick start dialog box contains two tabs:

- Create New GUI- Asks you to start creating new GUI by choosing template for it. You can specify the name by which the GUI is to be saved.
- Open Existing GUI- Helps you to open an existing GUI in the GUIDE.

Available Components: The palette at the left side of the layout editor contains components that you can add to your GUI. Each component can be given names.

Push Button

Push button generates an action when clicked. It appears depressed when clicked and appears raised when released.

Slider

Sliders accept data by enabling the user to move a sliding bar, which is called a slider. User moves the slider by dragging it, or by clicking an arrow. The location of the slider indicates the location within the specified range.

Radio Button

Radio buttons are like check boxes but they are mutually exclusive. Only one option can be selected at a time. To activate it, click the mouse buttons on the object. The display indicates the state of the button.

Check Box

Check boxes generate an action when checked and they are useful for providing the user with independent choices. For example: displaying a toolbar.

Edit Text

The edit text component provides the user with the space to enter text or string. Users can also input numbers but they must be converted to numeric equivalents.

Static Text

Static text is generally used to label other controls as well as provide instruction to the user. User cannot change the text interactively.

Pop-Up Menu

Pop-up menus open to display a list of choices when users click the arrow.

List Box

List boxes help the user to select one or more items from the list of items.

Toggle Button

Toggle buttons generate an action to indicate whether they are on or off. It appears depressed when clicked and will remain depressed until you click it second time. When you do so, the button remains in raised state showing off.

Table

Use the table button to create a table component.

Axes

Axes enable a user to display graphics such as graphs and images. Like all graphics objects, axes have properties that can be set to control many aspects of its behaviour and appearance.

Panel

Panels help to arrange GUI components into groups. It helps to group related controls and can make interface easier to use. It can have title and borders. A panel can have title and borders. The position of each component within the panel is interpreted relative to the panel. If the panel is moved then the components within it are also moved on the panel.

Toolbar

Toolbars containing push buttons and toggle buttons can be created. One can choose between predefined buttons, such as save and print and can customize with one's own icons and callback.

ActiveX® Component

ActiveX component enables to display ActiveX controls in your GUI. They are available only on Microsoft Windows platform. It can be the child of the figure but not of panel or button group.

The GUI Files: By default, GUIDE stores a GUI in two files which are generated the first time you save or run the GUI:

- A MATLAB FIG-file, with extension .fig, that contains a complete description of the GUI layout and the GUI components, such as push buttons, axes, panels, menus, and so on. The FIG-file is a binary file and you cannot modify it except by changing the layout in GUIDE. Note that a FIG-file is a kind of MAT-file.
- A MATLAB function file, with extension .m, that contains the code that controls the GUI, including the callbacks for its components.

These two files have the same name and usually reside in the same folder. They correspond to the tasks of laying out and programming the GUI. When you lay out the GUI in the Layout Editor, your work is stored in the FIG-file. When you program the GUI, your work is stored in the corresponding code file.

File and GUI Names:

The code file and the fig file must have the same name. This is also the name of GUI. Names are assigned to the GUI when you save it the first time.

For example, if the file is named project.fig and project.m then the name of GUI is project and the GUI can be executed by typing project at the command line. The code and fig file are in the same folder and the folder is in your path.

Renaming GUIs and GUI Files:

Renaming a GUI can be done using save as from the Layout Editor File menu. When you do this, Guide renames both the code and the fig file. Also can update the callback properties that contain the old names to use the new name and update all instances of the file name in the body of the code.

Making GUI

For this topic, we have made the project2.fig file as shown below. We included three axes – one to show the input signal given by the user, one to show the carrier wave and one to show the modulated signal. There is one edit text to take the input binary signal from the user also one pop-up menu so that the user can select which modulation technique the user wants to use. Then we used six edit texts to label the three axes, the pop-up menu and the edit text.

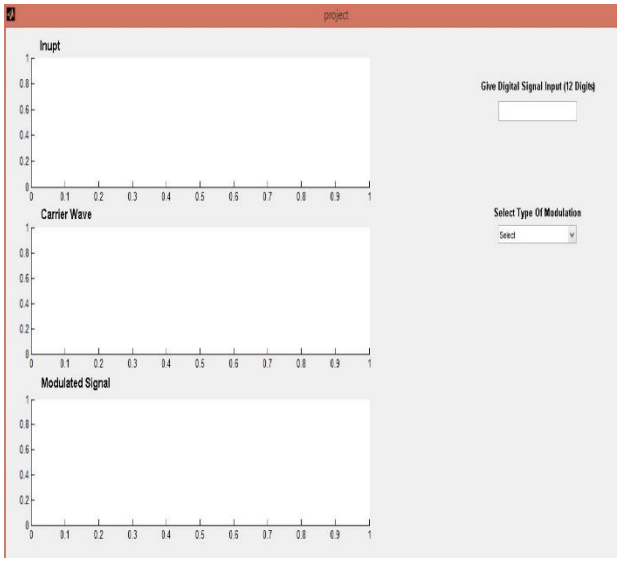


Figure 4. GUI for Modulation

When we save the 'project2.fig' file, 'project2.m' file is automatically created. We do our programming as in the file 'project2.m'.

IV. FUNCTIONS USED FOR MODULATION USING GUI

B = get (object, ' property)

Retrieves the value of property for object and assigns value to B.

X= str2double('str')

Converts the string str which should be an ASCII character representation of real or complex scalar value to the MATLAB double-precision representation. The string can contain digits, a comma, a decimal point, a leading + or – sign an e preceding power of 10 scale factor and an i for a complex unit.

R= rem(X, Y)

If Y=0, returns X-n*Y where n=fix(X/Y). If Y is not an integer and the quotient X/Y is within roundoff error of an integer, then n is that integer. The inputs X and Y must be real arrays of the same size, or real scalars.

B= floor(A)

Rounds the elements of A to the nearest integers less than or equal to A. For complex A, the imaginary and real parts are rounded independently.

axes(‘ PropertyName’, propertyvalue, ..)

creates an axes object having the specified property values. MATLAB uses default values for any properties that you do not explicitly defines as arguments.

hold on

Retains the current plot and certain axes properties so that subsequent graphing commands add to the existing graph.

hold off

Resets axes properties to their default before drawing new plots. hold off is the default.

n = length(X)

returns the size of the longest dimensions of X. If X is a vector, this is the same as its length.

B = zeros(m,n) or B = zeros([m n])

Returns an m-by-n matrix of zeros.

Y = ones (m,n) or Y = ones([m,n])

Returns an m-by-n matrix of ones.

Plot(Y)

Plot the columns of Y versus their indes if Y is a real number. If Y is complex, plot(Y) is equivalent to plot(real(Y), imag(Y)). In all other uses of plot, the imaginary component is ignored.

axis([xmin xmax ymin ymax])

sets the limits for the x- and y-axis of the current axes.

Y = sin(X)

returns the circular sine of the elements of X.

grid on

adds major gridlines to the current axes.

grid off

removes major and minor grid lines from the current axes.

out = randint(m,n,rg)

generates an m-by-n integer matrix. If rg is zero, out is a zero matrix. Otherwise, the entries are uniformly distributed and independently chosen from the range
[0, rg-1] if rg is a positive integer
[rg+1, 0] if rg is a negative integer

Between min and max, inclusive, if rg = [min,max] or [max,min]

scatterplot(x)

produces a scatter plot for the signal x. The interpretation of x depends on its shape and complexity.

If x is a real two-column matrix, scatterplot interprets the first column as in-phase component and the second column as quadrature components.

If x is a complex vector, scatterplot interprets the real part as in-phase components and the imaginary part as quadrature component.

If x is a real vector, scatterplot interprets it as a real signal.

y = awgn (x, snr)

adds white Gaussian noise to the vector signal x. The scalar snr specifies the signal-to-noise ratio per sample, in db. If x is complex, awgn adds complex noise. This syntax assumes that the power of x is 0 dBW.

y = awgn(x,snr,'measured')

is the same as y = awgn(x,snr), except that awgn measures the power of x before adding noise.

title (‘string’)

outputs the string at the top in the center of current axes.

h = modem.pskmod(M)

constructs a PSK modulator object h for M-ary modulation.

h = modem.qammod(M)

constructs a QAM modulator object h for M-ary modulation.

V. RESULT

(a) ASK

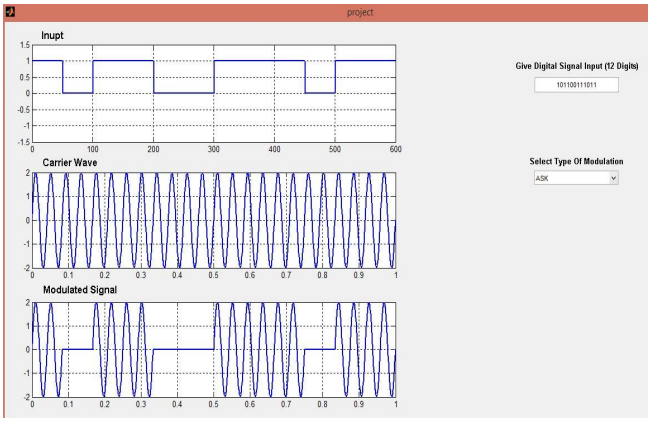


Figure 5.

(a) FSK

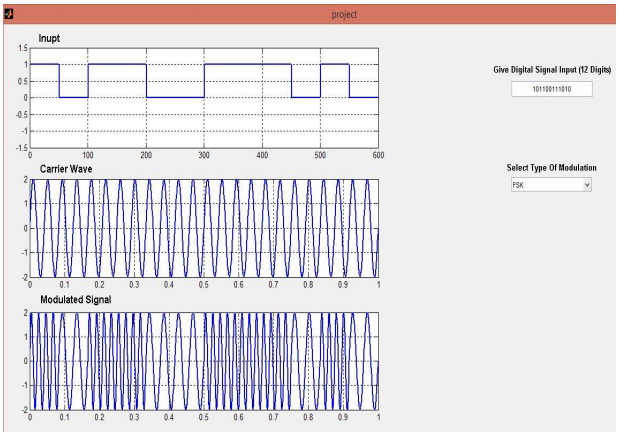


Figure 6.

(b) (i)BPSK

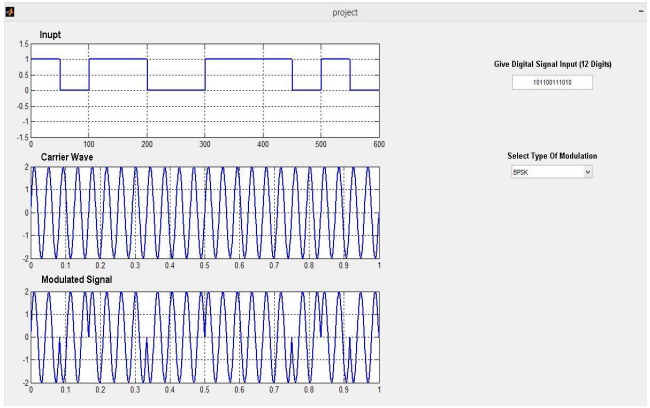
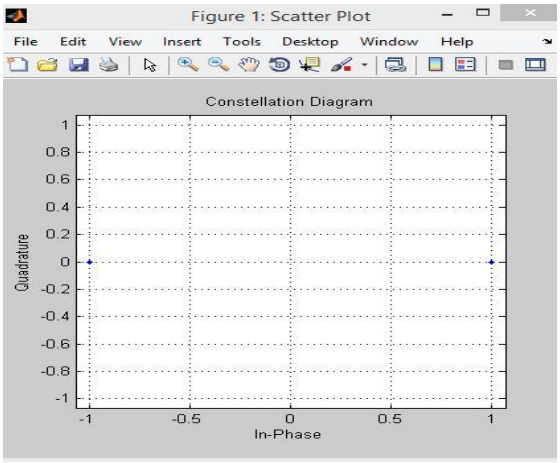


Figure 7.

(c) (ii) BPSK Constellation Diagram



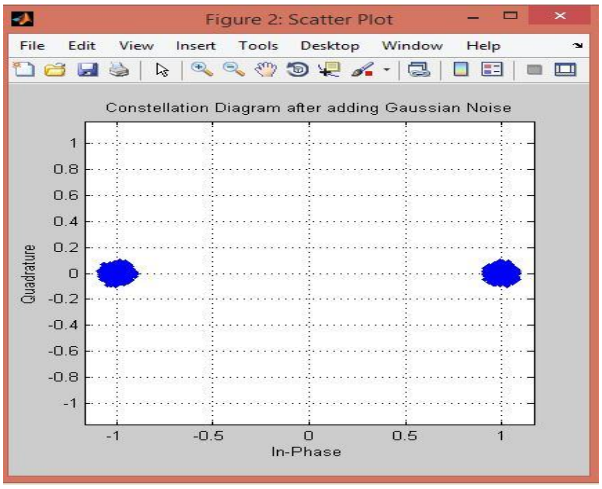


Figure 8.

(d) (i) QPSK

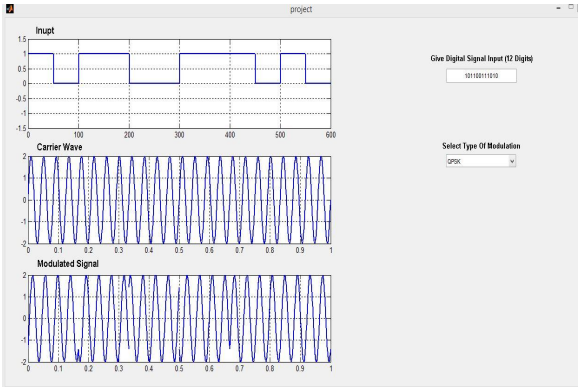


Figure 9.

(e) (ii) QPSK Constellation Diagram

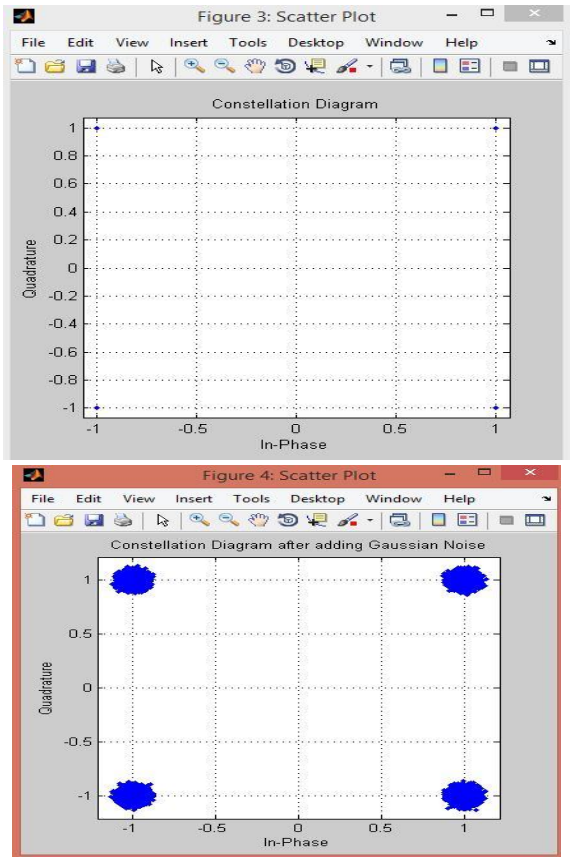


Figure 10.

Advantages of GUI:

A GUI is a rectangular window on a computer monitor that represents its contents independently of the rest of the system. A GUI provides an interface for human and computer. It is a window that has icons, menus, buttons and pointers that can be manipulated by the mouse or the keyboard. It is advance over command line arguments as in it only text can be displayed and only accessible by the keyboard. One can easily understand the work by the pictures and graphics and it becomes easier to learn.

VI. CONCLUSION

Thus GUI can be used to understand the modulation techniques and also it is easier to learn making GUIs in Matlab software. The carrier signals and waveforms are plotted on axes to briefly understand the modulation. Different waveforms are generated according to the different digital inputs.

REFERENCES

[1] Matlab Product Help

[2] Analog and digital modulation techniques: an overview by D.K.Sharma, A.Mishra, R.Saxena in Interantional Journal of computing science and communication technologies, volume 3, No. 1, July 2010.

[3] Analog and digital modulation techniques: an overview by Amritpal Kaur, International journal of engineering sciences and research technology, July 2014.

[4] http://www.mathworks.com/help/techdoc/creating_guis/bqzzla9-2.html

[5] http://www.mathworks.com/help/techdoc/creating_guis/bqz6p81.html

[6] http://en.wikipedia.org/wiki/Amplitude-shift_keying

[7] http://en.wikipedia.org/wiki/On-off_keying

[8] http://en.wikipedia.org/wiki/Frequency-shift_keying

[9] http://en.wikipedia.org/wiki/Phase-shift_keying