VOL. 14, NO. 8, APRIL 2019 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

FPGA BASED IMPLEMENTATION OF QAM MODULATOR USING 64 VALUES LOOK UP TABLE (LUT)

Amean S. Al-Safi and Liqaa D. Alhafadhi
Department of Electrical and Electronics Engineering, University of Thi-Qar, Thi-Qar, Iraq
E-Mail: ameansharea.ghazi@wmich.edu

ABSTRACT

This paper presents a new and an efficient method to implement quadrature amplitude modulator (QAM) using field programmable gate array (FPGA). Two QAM modulators were implemented i.e. 4QAM and 16 QAM. The targeted board for the implementation is ZYBO board from XILINX and the used software is XILINX Vivado. The main important point in the paper is the use of quarter wave instead of full wave sinusoidal signal to generate the carrier signal. This process has saved the utilization resources by more than 50%. Saving this many resources will make a great impact on the future implantation of digital modulators since the implementation depends heavily on how the carrier signal got generated because it consumes most of utilization recourses. The paper used very high speed integrated circuit hardware descriptive language (VHDL) without the help of high level simulation tools like DSP builder tools or XILINX system generator.

Keywords: FPGA, LUT, digital modulators, QAM, utilization resources, ZYBO.

1. INTRODUCTION

Software defined radio (SDR) has seen a lot of progress since the day it was presented by Joseph Mitola as a method of implementing radio transceivers in 1992[1]-[3]. It has become a promising and efficient technique to implement a variety of communication systems. Building any SDR based system depends heavily on the existence of the hardware terminals which represent the main part in these kinds of systems. Due to the high cost of the hardware terminals such as universal software radio peripherals (USRP), many researchers have been looking for alternative solutions. Almost all of the suggested solutions so far have focused on the use of Field Programmable Gate Array (FPGA) and Applied Specific Integrated Circuits (ASIC) devices as well as other electronic components to use instead of the USRPs. For the modeling and implementation purposes, Xilinx systems generators and DSP builders tools are the common methods to implement the systems after they got built using high level software such as MATLAB. Quadri and Tete [4] summarized the main work in this direction which also presented in our previous work [5]-[6].

The targeted communication systems that have been implemented have changed from simple modulation scheme such as binary amplitude shift keying (BASK), Binary Frequency Shift Keying (BFSK), and binary phase shift keying (BPSK) [7]-[9] up to the advanced ones such as quadrature phase shift keying QPSK and quadrature amplitude modulation (QAM) [5]-[6]. The main problem with the suggested models for the implementation is the usage of the Xilinx Systems Generators and DSP Builders Tools which increase the implementation cost since they use an available Intellectual Property (IP) blocks. These IP blocks as well as the used software will be an additional cost to the system.

Our main goal in this research area is to reduce the implementation cost by using FPGA instead of USRPs and avoid the use of unnecessary IP blocks. The current research represents an extension to the work that was presented before [5]-[6]. Avoiding the use of IP block, Xilinx systems generators and DSP builders tools, makes our work really unique since it will have the lowest cost as compared with others. In addition to the above mentioned point, we do have a direct control on the implemented models since we use hardware description language (HDL) instead of using other helping tools. Our previous work [5]-[6] and the current one have the same goal of reducing the implementation cost using a variety of technique to build different communication systems.

In the current research, we used a reduced size look up table (LUT) to generate a carrier signal which represent the most important signal in the modulation scheme. Using the new LUT we were able to reduce the used hardware recourses by more than 50%.

The rest of the paper is organized as follow: Section 2 presents the main research problem, section 3 will explain the suggested solution, sections 4 and 5 will show the implementation and results, and finally section 6 is the conclusion and future work.

2. RESEARCH PROBLEM

Implementation of communication systems using hardware components suffers from the higher cost and flexibility. Hence, SDR based systems have been used as an alternative solution since they give more flexibility and reduce the cost significantly. For certain carry-on and simple systems, even the small cost is still a problem. To overcome this challenge, several papers have been presented to reduce the implementation cost by reducing the used hardware recourses and even avoiding the use of unnecessary expensive software. In the current paper, we used a new method to generate the carrier wave signal. The next section will present the suggested method in detail.

3. PROPOSED IMPLEMENTATION METHOD

The suggested implementation method use the Random Access Memory (RAM) within the FPGA board to store 64 values of the fixed point representation of quarter period of the sinusoidal signal only to generate the

whole carrier wave. Let us assume that we generate a full period sine wave signal in a fixed point format using MATLAB. From these 256 samples we can use 64 samples only (quarter wave) to generate the whole wave carrier as shown in the code in Table-1.

Table-1. MATLAB code for sine wave generation.

```
LUT=zeros(128,1);
LUT2=zeros(256,1);
for i=1:64
LUT(i)=Rom(i);
End;
for i=1:64
LUT(64+i)=Rom(65-i);
end ;
LUT2(1:128)=LUT;
LUT2(129:256)=-1*LUT;
plot(Rom,'k');
hold on;
plot(LUT,'b');
plot(LUT2,'r');
```

If we use 256 samples to represent the whole wave then each quarter can be epresented by 64 samples as shown in Table-2.

Table-2. Quarter location based on the sample number.

Samples range	Quarter location
1-64	first quarter
65-128	second quarter
129- 192	third quarter
193-256	fourth quarter

But quarters 2-4 can be calculated based on the samples available in the first qurter as shown in the MATLAB code in Table-1. The main idea behind that can be explained by looking at Figures1-3 shown below. Figure-1 represents the 64 samples that we need to store in the LUT to generate half wave as in Figure-2 or the whole wave as in Figure-3 at any desired frequency.

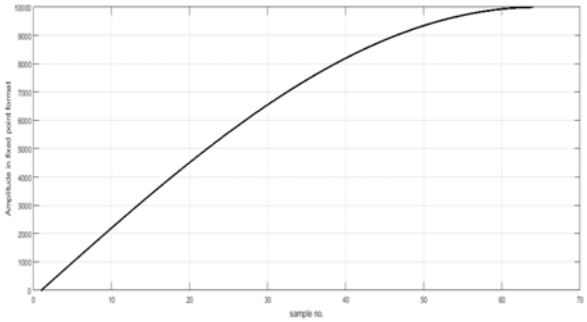

Figure-1. Quarter period of sinusoidal signal.

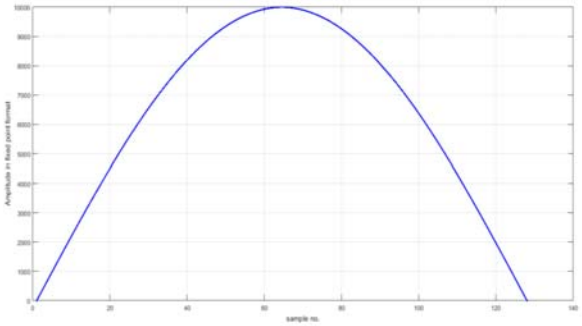

Figure-2. Half period of sinusoidal signal.

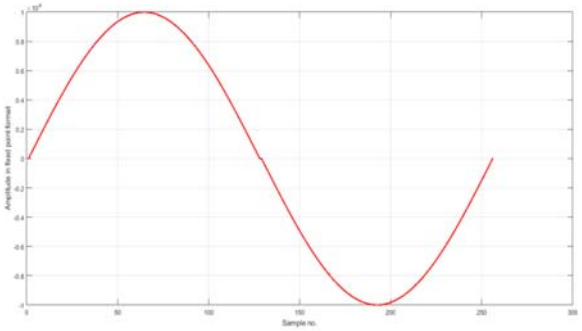

Figure-3. Full period of sinusoidal signal.

This process looks very simple in MATLAB but in VHDL it is a little more complex than that.

4. IMPLEMENTATION OF QAM MODULATORS

In order to test the suggested method for carrier wave generation, two QAM modulator systems were implemented. As it is shown in the block diagram in Figure-4, two carriers with 90 degree phase shift are needed to generate any QAM modulator system. The first carrier signal can be generated by using a accumulator register working on the rising edge of the clock signal of the board and the 64 value LUT within the FPGA RAM. The second carrier signal can be generated by the same method but the accumulator has to be working on the falling edge of the clock to get a 90 degree phase shift which represents the phase shift between the sine and cosine signal. The size of the accumulator is 8-bit which covers all the 256 values of the intended LUT even though we used a LUT with 64 values only. The two most significant bits will select the phase quarter while the rest 6-bit will select the phase values within the RAM as shown in Table-3.

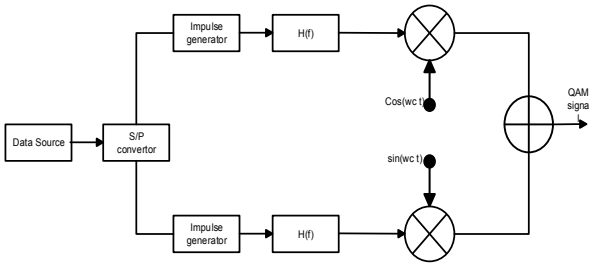

Figure-4. QAM generation.

Table-3. Quarter location based on the two most significant bits of the accumulator.

AC7 AC6	Quarter location
0 0	first quarter
0 1	second quarter
1 0	third quarter
1 1	fourth quarter

The block diagram of the 4QAM modulator system is shown in Figure-5 where we have two accumulators as described before, and there are two carrier signals (cosine and sine) which will be used to generate the QAM signal based on the incoming message signal as shown in Table-4.

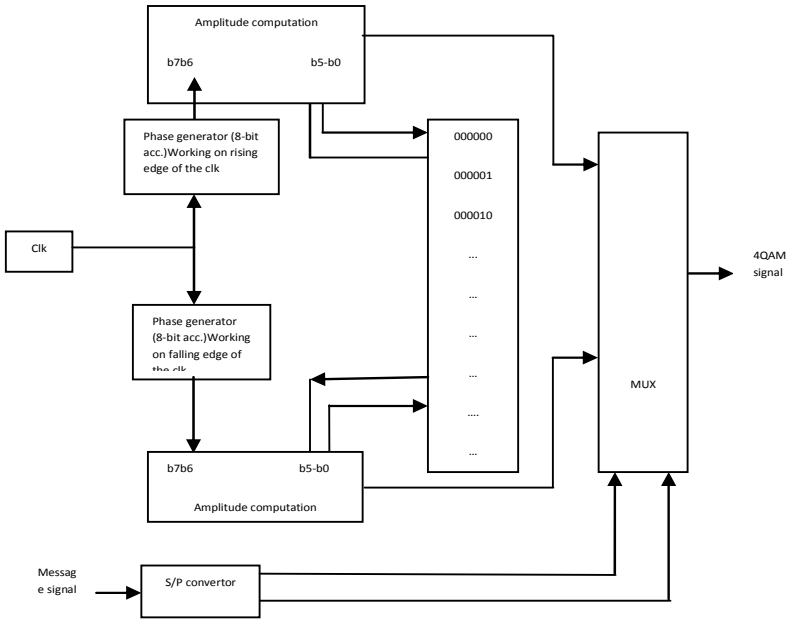

Figure-5. Block diagram of the 4QAM modulator in VHDL.

Table-4. 4QAM symbol representation.

Constellation symbol	Modulated signal
00	S_0+S_1
01	S_0-S_1
10	$S_1- S_0$
11	$-S_0-S_1$

Higher orders QAM which give us higher spectral efficiency can be implemented using the same concept. Figure-6 is the block diagram of a 16 QAM modulator system using FPGA and VHDL software. As it was described in the 4QAM modulator schem, the same two carrier waves were needed for the implementation. The two main differences that we can notice here as compared to the 4QAM implementation circuit are the changes in the multiplexer and serial to parallel convertor blocks. Now the serial to parallel circuit converts the incoming message data into four parallel signals instead of two as before and the multiplexer circuit generates sixteen different combinations of the carrier siganals based on the incoming symbols of data as shown in Table-5 which is taken from ref. [6].

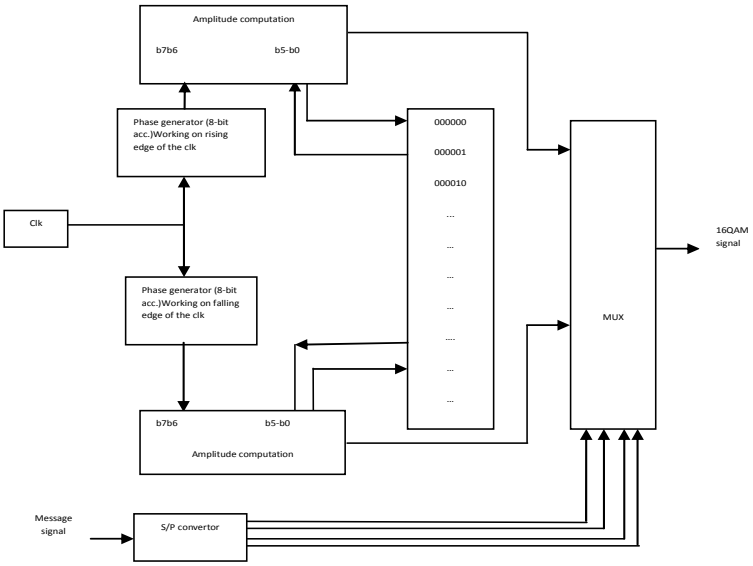

Figure-6. Block diagram of 16QAM modulator in VHDL.

Table-5. 16QAM symbol representation.

Constellation symbol	Modulated signal
0000	S_0+S_1
0001	$3S_0+S_1$
0010	$-S_0+S_1$
0011	$-3S_0+S_1$
0100	S_0+3S_1
0101	$3S_0+3S_1$
0110	$-S_0+3S_1$
0111	$-3S_0+3S_1$
1000	S_0-S_1
1001	$3S_0-S_1$
1010	$-S_0-S_1$
1011	$-3S_0-S_1$
1100	S_0-3S_1
1101	$3S_0-3S_1$
1110	$-S_0-3S_1$
1111	$-3S_0-3S_1$

5. IMPLEMENTATION RESULTS

The description of the QAM modulators systems (4QAM and 16QAM) were used to obtain the results. The 64 values quarter period sine wave LUT was implemented using the LUT within the ZYBO board shown in Figure-7. The complete systems implementations were implemented using Xilinx Vivado software. The input message signal was an arbitrary signal generated based on the main board clock which also can be replaced by any kinds of signals but it has to be read by the implementation board. The implementation results were exported into MATLAB for better visualization purposes. The modulated waveforms we got are shown in Figures 8-9 for the 4QAM and 16QAM respectively.

Figure-7. ZYBO board.

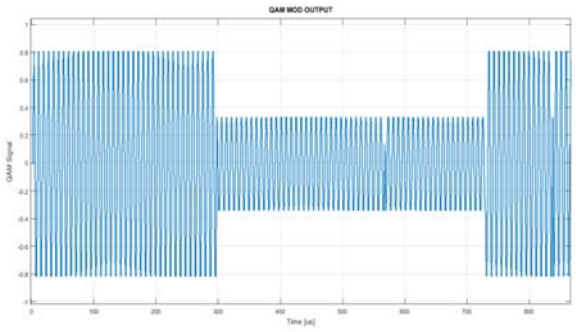

Figure-8. 4QAM modulation results in MATLAB.

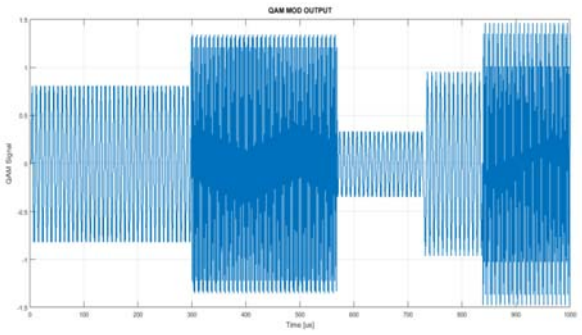

Figure-9. 16QAM modulator results in MATLAB.

To justify our new design schemes, we compared them with the work we did before [6] and the utilization recourses we got are less than those in [6] by more than 50%. The new resource utilization tables are shown in Tables 6-7 for the 4QAM and 16QAM respectively. We have to mention here that the utilization resources can be obtained after downloading the design on the board which has to be preceded by writing the .xdc file to select which inputs and/or output terminals we are using in the design.

Table-6. Device utilization summary of the 4QAM modulator.

Resource	Utilization	Available	Utilization %
LUT	31	17600	0.18
FF	6	35200	0.02
IO	9	100	9
BUFG	1	32	3.13

Table-7. Device utilization summary of the 16QAM modulator.

Resource	Utilization	Available	Utilization %
LUT	39	17600	0.22
FF	6	35200	0.2
IO	9	100	9
BUFG	1	32	3.13

To implement the same 4QAM and 16QAM modulators based on reference [6], we needed the recourses utilizations shown in Tables 8 -9 respectively. Based on the utilization recourses for the new method (Tables 6-7) and the old one (8-9), it can be seen easily that the reduction has occurred in the number of LUT and FF required for the implementation. The new method needs 31 LUT and 6 FF instead of 84 LUT and 14 LUT or the 4QAM modulator. The required LUT and FF for the 16QAM based on the new model are 39 LUT and 6 FF instead of 251 LUT and 17 FF in the old one. Hence, the reduction in the number LUT is 63% for the 4QAM and 84% for the 16QAM modulator while the reduction in the number of FF is 57% for the 4QAM and 65% for the 16QAM modulator system. It is expected also that as the modulator gets more complex the utilization reduction will be better based on the new method.

Table-8. Device utilization summary of the 4QAM modulator (old method [6]).

Resource	Utilization	Available	Utilization %
LUT	84	17600	0.48
FF	14	35200	0.04
IO	9	100	9.00
BUFG	1	32	3.13

Table-9. Device utilization summary of the 16QAM modulator (old method [6]).

Resource	Utilization	Available	Utilization %
LUT	251	17600	1.43
FF	17	35200	0.05
IO	9	100	9.00
BUFG	1	32	3.13

6. CONCLUSION AND FUTURE WORK

A reduced size look up table was used to generate carrier wave signal in the implementation of 4QAM and 16 QAM modulator. Instead of using a whole period signal to generate the carrier wave, quarter wave was used and the rest of the period was computed based on the nature of the sine wave and how it is changing as we move from quarter to quarter. This process has reduced the required number of samples that needed for carrier generation from 256 samples to 64 samples only. The LUT reduction has a direct impact on the required utilization resources. The new method has reduced the required number of LUT and FF to implement 4QAM by 63% and 57%, and the required number of LUT and FF to implement 16QAM by 84% and 65 % respectively. It is also expected that as the modulators schemes get more complex the utilization reduction will be better based on the new method.

As a future work, the authors will investigate the use of the new scheme for the carrier generation to implement other digital modulators such as 8PSK, 16PSK, and higher order QAM. If successful implementations can be reached, then a full range of modules can be available in a single FPGA or maybe in-circuit FPGA programmability can be used in building signal processing blocks for SDR.

REFERENCES

[1] J. Mitola. 1992. Software radios-survey, critical evaluation and future directions. In Proceedings of

IEEE National Tele systems Conference. pp. 13/15-13/23.

[2] J. Mitola. 1995. The software radio architecture. IEEE Communications Magazine. 33: 26-38.

[3] J. Mitola. 1999. Software radio architecture: a mathematical perspective. IEEE Journal on Selected Areas in Communications. 17(4): 514-538.

[4] F. Quadri and A. D. Tete. 2013. FPGA implementation of digital modulation techniques. in Proceedings of IEEE International Conference on Communications and Signal Processing (ICCSP), Melmaruvathur, India. pp. 913-917.

[5] A. Al-Safi and B. Bazuin. 2016. FPGA based implementation of BPSK and QPSK modulators using address reverse accumulators, in Proceedings of the 7th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference, Columbia University, New York City, USA.

[6] Al-Safi and B. Bazuin. 2017. Toward digital transmitters with ASK and QAM implementation examples. in Proceedings of the 7th IEEE Annual Computing and Communication Workshop and Conference (CCWC), Hotel Stratosphere, Las Vegas, USA, 9 - 11 January.

[7] Erdogan I., Myderrizi and S. Minaei. 2012. FPGA Implementation of BASK-BFSK-BPSK digital modulators. IEEE Antennas and Propagation Magazine. 54(2): 262-269.

[8] S. O. Popescu, A. S. Gontean and G. Budura. 2012. BPSK system on Spartan 3E FPGA. in Proceedings of IEEE 10th International Symposium on Applied Machine Intelligence and Informatics (SAMI). pp. 301-306.

[9] Y. H. Chye, M. F. Ain and N. M. Zawawi. 2009. Design of BPSK transmitter using FPGA with DAC. in Proceedings of IEEE 9th International Conference on Communications (MICC), Kuala Lumpur, Malaysia. pp. 451-456.