International Journal of Reconfigurable and Embedded Systems
Vol. 7, No. 3, November 2018, pp. 173~185
ISSN: 2089-4864, DOI: 10.11591/ijres.v7.i3.pp173-185

Design and Implementation of a New Architecture of a Real-time Reconfigurable Digital Modulator (DM) into QPSK, 8-PSK, and 16-PSK on FPGA

Walder Andre, Olivier Couillard
Department of Electrical and Computer Engineering, Royal Military College of Canada

Article Info

Article history:

Received Agu 8, 2018
Revised Oct 9, 2018
Accepted Oct 23, 2018

Keywords:

Digital Modulation
Encryption Key
FPGA
QPSK
8-PSK
16-PSK

ABSTRACT

One of the prerequisites of Electronic Warfare (EW) is to have the means to provide secure point-to-point wireless data and voice communications with other ground stations. New technologies are giving rise to bigger information security threats. This situation illustrates the best the urgency of reducing the development and upgrade time of EW systems. Previous works suggest that digital systems are the best candidates for this purpose and therefore form the backbone of modern Electronic Warfare. Indeed, Digital Modulation (DM) techniques are widely used in modern wireless communication systems. This is largely due to their high resistance to noise and their high transmission capacity that can be achieved through data multiplexing. In this article, a new reconfigurable architecture of a Phase Shift Keying (PSK) modulation is described. The latter can be configured in real time to produce the following modulation schemes: QPSK, 8-PSK, and 16-PSK without having to regenerate the FPGA configuration bits. This action can be done by software via programming or manually using a DIP switch. The proposed design is implemented on the Xilinx xc7k325tfbg900 FPGA using the Genesis 2 development board. The Vivado Physical Design Automation tool indicates a power consumption of 303 mW by the on-chip circuit. The experimental results are in agreement with the simulations.

Corresponding Author:

Walder Andre,
Departement of Electrical and Computer Engineering,
Royal Military College,
Box 17000, Station Forces,
Kingston, Ontario, K7K 7B4, Canada.
Email: walder.andre@rmc.ca

1. INTRODUCTION

DM techniques are mostly preferred over its analog counterpart for their greater transmission capacity and greater immunity to noise [1]-[3]. Among DM techniques we distinguish the Amplitude Shift Keying (ASK), the Frequency Shift Keying (FSK) and the Phase Shift Keying (PSK). These three types of DM techniques use a sinusoid and vary its amplitude, frequency or phase to produce the ASK, the FSK or the PSK modulation respectively.

With the great advances made in the manufacturing process of integrated circuits (ICs), we are seeing an explosion in the use of digital systems in the data processing for wireless communications systems.

This is due in part to the fact that digital circuits are much easier to manufacture being based on the CMOS fabrication process. Moreover, almost all transistors in CMOS digital ICs have minimum length to provide a high switching frequency. This feature ensures the design of very high frequency digital signal processors (DSP). In addition, digital ICs do not suffer from over-voltage nor from second order effects encountered in analog circuits such as noise, offset and mismatch [4]. Furthermore, with the arrival of the Field Programmable Gate Array (FPGA) technology, this phenomenon centered on digital processing techniques is accentuated even more. The advantages of the FPGA technology are based on the following five features: (1) It offers better computing power compared to digital signal processors (DSPs) benefiting from hardware parallelism, (2) enables rapid prototyping of circuits, (3) allows the reconfiguration of systems which reduces engineering costs, (4) does not use an operating system which minimizes reliability problems, and (5) unlike Applications Specific Integrated Circuits (ASICs) which requires remanufacturing in order to make an upgrade, FPGA technology allows us to achieve this by reprogramming only the necessary block or blocks in the design without respending money. As a result, the analog parts of modern communication systems have become increasingly absent. Their use is reduced to power amplification, ADC and DAC converters, and filters. Although the filters can be digital, we are still witnessing a strong use of off-chip analog components to realize the band-pass filter (BPF) found in some current communication systems. However, the FPGA technology remains the most used for data processing.

Through literature, we found several authors who chose the FPGA for their wireless communication system. For instance, Moubark et al. proposed a new architecture to realize a QPSK modulator [5]. The total power consumed by the proposed modulator is 40 mW. This result is very promising for the choice of the FPGA technology as a prototyping platform to build wireless communications systems. Over time, other architectures of the QPSK modulator have been proposed. Al Safi describes a new QPSK modulator that uses one LUT and two accumulators to generate the four phases of the QPSK [8]. Knowing that an accumulator is a register which role is to store LUT data temporarily, therefore the reading of these data will introduce delay into the phase generation process. However, the impact of this introduced delay on the system is negligible compared to the gain incurred with the reduction in the size of the circuit. In some other QPSK architectures a digitized carrier wave generator was used [6] and [7]. These subcarriers are generated from a Numerically Controlled Oscillator (NCO) to produce the I and Q modulation signals [7]. With this approach, the two multipliers, as well as the local oscillator, are no longer needed. In the quest to reduce the space occupied by the modulator in the FPGA several designs have been proposed, some of which are found in [8]-[10]. Overall, the use of the FPGA to implement digital communication systems is very well exploited in the literature [11]-[13]. This article is organized as follows. In section 1 we present the topic on the use of DM techniques in communications system design. In section 2 we will briefly review the theoretical foundations of the PSK modulation. In Section 3 a new architecture of a real-time reconfigurable PSK modulator is proposed. Sections 4 and 5 are dedicated to the presentation of simulation and experimental results respectively. In Section 6 we present an encryption key generator circuit. Finally, we conclude in Section 7.

2. PHASE SHIFT KEYING THEORY

In this study, the phases of developing an evaluation model of the suitability of anti-hypertensive drugs as presented in Figure 1. Development stage consists of the first Stage of making a knowledge base involving input data from the hospital, expert knowledge and knowledge derived from literature studies

The second stage is the process of evaluation of the fit between the conditions of the patient with the type of anti-hypertensive drugs. The method used is Profile Matching method.

$$s_n(t) = Acos\left(2\pi f_c t + \pi(1-n)\right) \tag{1}$$

For $n = 0, 1$ we have:

$$s_0(t) = -Acos(2\pi f_c t) \rightarrow for\ input\ 0 \tag{2}$$

$$s_1(t) = Acos(2\pi f_c t) \rightarrow for\ input\ 1 \tag{3}$$

While the BPSK modulation is more robust, Quadrature Phase Shift Keying (QPSK) modulation provides higher data rate, and therefore more suited for data transmission [14]. Figure 2 shows the conventional architecture of the QPSK modulator. This circuit is fully implementable in a FPGA, and can be described by using one or the other of the Hardware Description Language (HDL) Verilog or VHDL.

The 2-bit serial to parallel converter module combined with the two Non-Return Zero (NRZ) encoders occupy little FPGA space, and the two multipliers do not occupy a lot of FPGA depending on the multiplication algorithm used.

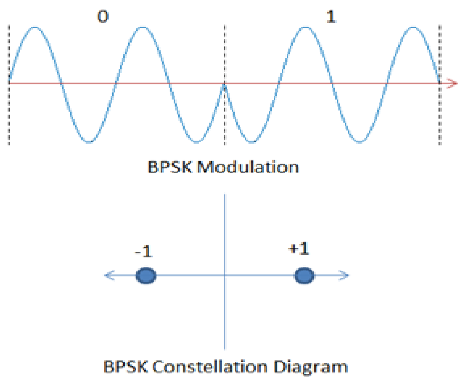


Figure 1. BPSK modulation and constellation diagram

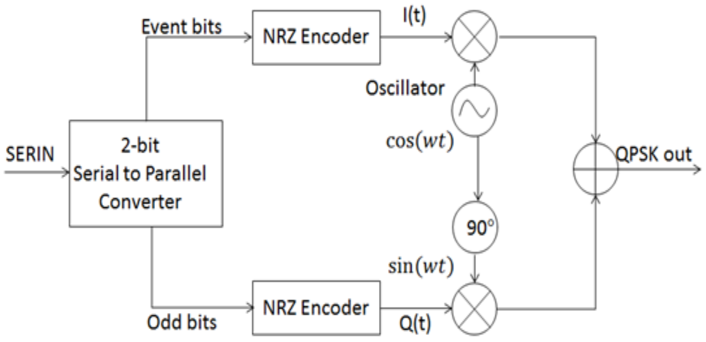


Figure 2. The conventional architecture of the QPSK modulator

Figure 3 illustrates the QPSK modulation and shows its constellation diagram. Here each symbol is associated with a phase of the signal. By going counter clockwise the symbols "11", "01", "00" and "10" are found at angles of $3\pi/4$, $5\pi/4$ $(-3\pi/4)$, and $7\pi/4$ $(-\pi/4)$ respectively. The constellation diagram of the QPSK can be drawn from Equation 4, where f_c is the frequency of the carrier:

$$s_n(t) = A\cos\left(2\pi f_c t + (2n-1)\frac{\pi}{4}\right), n = 1,2,3,4 \quad (4)$$

From Equation 4 we derive the following expressions of $f(I,Q)$:

$$f(0,0) = -\cos(wt) - \sin(wt) \quad (5)$$

$$f(0,1) = -\cos(wt) + \sin(wt) \quad (6)$$

$$f(1,0) = \cos(wt) - \sin(wt) \quad (7)$$

$$f(1,1) = \cos(wt) + \sin(wt) \quad (8)$$

The following rules to model the QPSK encoder are based on the IEEE Standard 802.16-2004:

$$\begin{cases} \frac{-1-j}{\sqrt{2}} \rightarrow for\ symbol\ \text{"00"} \\ \frac{-1+j}{\sqrt{2}} \rightarrow for\ symbol\ \text{"01"} \\ \frac{1-j}{\sqrt{2}} \rightarrow for\ symbol\ \text{"10"} \\ \frac{1+j}{\sqrt{2}} \rightarrow for\ symbol\ \text{"11"} \end{cases} \quad (9)$$

Where $1/\sqrt{2}$ is normalization factor of the the modulated waveform. Equation 9 dictates that the four symbols are separated by $\pi/2$, with the first symbol "11" located at the angle of $\pi/4$. Thus, the QPSK modulator having four symbols offers a higher data transmission rate than the BPSK modulator. Now if we look at the reception level, ie at the QPSK and BPSK demodulators, we note that the signal after passing through the channels paired with transmission noises, the BPSK demodulator has only two decisions to make while the QPSK demodulator has four. This tells us that the BPSK modulation is more robust than QPSK modulation using only two phases makes signal reception easier for the demodulator circuit. These two modulation schemes are widely used in today's communication systems. Depending on the design objectives, compromises will be made between speed and robustness. Here, we propose a new architecture that meets these two criteria and more.

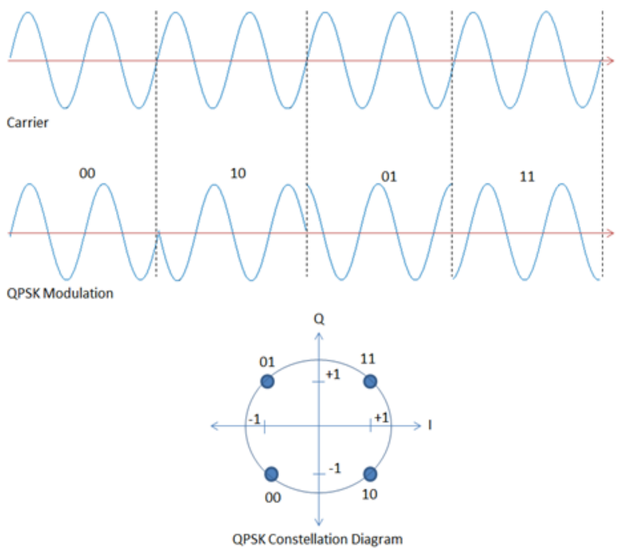


Figure 3. QPSK modulation and constellation diagram

3. NEW RECONFIGURABLE QPSK, 8-PSK, AND 16-PSK ARCHITECTURE

Here, a new architecture to implement a reconfigurable PSK modulator is presented in Figure 4. The proposed design consumes very little power and occupies a negligible space in the FPGA. This design is reconfigurable in real time (1) manually using two switches or (2) by software via programming. Although the three modulation schemes QPSK, 8-PSK, and 16-PSK are present in the FPGA bitstream, only one scheme can be selected at a time. Note that the circuit reconfiguration is done without the need of regenerating the FPGA configuration bits; therefore without any interruption of service. This has never been done in the past. Moreover, the proposed algorithm can be used to generate PSK modulation types up to N-PSK. Having in mind that the more we add phases ie 32-PSK and so on the more the reception of data will be difficult at the receiver end.

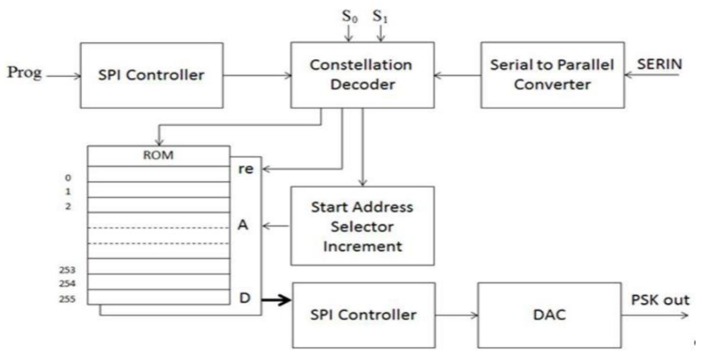


Figure 4. New archecture of a reconfigurable PSK modulator

This feature of real-time reconfiguration is very effective in situations of very strong jamming. Like the frequency hopping technique which consists of changing frequencies to counter jammings, here the scheme of the PSK modulation will be the parameter to change to accomplish the same task. Accordingly, the three modulation schemes implemented can be preprogrammed to be used in turn creating a phase hopping as a means of protection in the presence of jamming attacks. For this type of Electronic Protection Measure (EPM) to be effective, the modulator and the demodulator must be well syntonized. That is, the demodulator must be preprogrammed with the same phase hopping sequences used in the modulator. The Pseudocode for the algorithm is given in Table 1 to generate the different associated phases for the desired modulation.

Table 1. Pseudocode of the Algorithm Used

```
Algorithm
If reset then
Initialize the ROM addres counter
If modulator_selected=QPSK
Compute ROM start address counter for the reading and increment
    While QPSK
        Continue reading
If modulator_selected=8-PSK
Computer ROM start address counter from the reading and increment
    While 8-PSK
        Continue reading
If modulator_selected=16-QPSK
Compute Rom start address counter from the reading and increment
    While 16-QPSK
        Continue reading
If modulator_selected=others
        Do nothing
```

The constellation decoder module (phase encoder) depending on the type of modulation selected is responsible for calculating the ROM read start address. The state machine in this architecture will read the ROM continuously until a new symbol is received from the serial to parallel converter module. The latter generates the symbols for the three types of modulator being QPSK, 8-PSK, and 16-PSK. For each symbol, a unique start address for reading in the ROM is assigned, thus for each new symbol received the read counter is initialized to the new start address value calculated. That said, these start addresses indicate the phases of the modulator that are generated. The ROM contains the waveform to be used by the encoder phase module to generate the different phases of the selected modulation scheme. We used the 3-wire SPI controller core from Digilent™ to program the generic modulator and to communicate the transmission data to the SPI interface of the DAC used. The DAC used is the Digilent PMODDA2 which is a 12-bit DAC with 1Million samples / sec. The phase encoder module is a key part of the proposed modulator. It controls the access to the ROM and decodes the signals at the output of the serial to parallel converter. With $S_{n,p}$ being the constellation points on the constellation diagram we have:

$$S_{m,p} = sample\_size(2m+1)\left(\frac{\pi}{4(2^p)}\right)\frac{1}{2\pi} \tag{10}$$

$$S_{m,p} = sample\_size\,\frac{(2m+1)}{2^{p+3}} \tag{11}$$

With m=1, 2… 2^{p+3} and p=0, 1, 2 for QPSK, 8-PSK and 16-PSK respectively. Table 2 shows the pseudo-code used in the process of generating the phases for the three types of modulation described above. This pseudocode can be extended to 32, 64-PSK and so on. As shown in Figure 4, we used a single ROM of size 256x12 to contain the sinusoidal signal that we will be used to modulate our data signal. Initially, the system is reset to address zero by default regardless of the selected modulation type, QPSK, 8-PSK, and 16-PSK. By selecting the “start mode”, the modulator begins receiving the serial data signal and routes it to the serial to parallel converter. The serial to parallel converter is reconfigurable into 1 bit to 2-bit parallel converter, 1 bit to 3-bit parallel converter and finally into 1 bit to 4-bit converter to accommodate the QPSK, 8-PSK, and 16-PSK respectively. The converter will generate the symbols from the input serial signal. By selecting the type of modulation to be generated, the pseudocode for the HDL below is used to generate the phase differences. To generate the phases we start reading the ROM from a different address.

Table 2. Pseudocode for the HDL

qpsk signal (start addresses)
rom_size00_s <=divide (to_unsigned(1*rom_size_in, div_arg_width), to_unsigned(8, div_arg_width)) ;
rom_size01_s <=divide (to_unsigned(3*rom_size_in, div_arg_width), to_unsigned(8, div_arg_width));
rom_size11_s <=divide (to_unsigned(5*rom_size_in, div_arg_width), to_unsigned(8, div_arg_width));
rom_size10_s <=divide (to_unsigned(7*rom_size_in, div_arg_width), to_unsigned(8, div_arg_width));
8-psk signal (start addresses)
rom_size000_s <=divide (to_unsigned(1*rom_size_in, div_arg_width), to_unsigned(16, div_arg_width)) ;
rom_size001_s <=divide (to_unsigned(3*rom_size_in, div_arg_width), to_unsigned(16, div_arg_width)) ;
rom_size011_s <=divide (to_unsigned(5*rom_size_in, div_arg_width), to_unsigned(16, div_arg_width)) ;
rom_size010_s <=divide (to_unsigned(7*rom_size_in, div_arg_width), to_unsigned(16, div_arg_width)) ;
rom_size110_s <=divide (to_unsigned(9*rom_size_in, div_arg_width), to_unsigned(16, div_arg_width)) ;
rom_size111_s<=divide (to_unsigned(11*rom_size_in, div_arg_width), to_unsigned(16, div_arg_width)) ;

rom_size101_s <=divide(to_unsigned(13*rom_size_in, div_arg_width), to_unsigned(16, div_arg_width)) ;
rom_size100_s <=divide (to_unsigned(15*rom_size_in, div_arg_width), to_unsigned(16, div_arg_width)) ;
16-psk signal (start addresses)
rom_size0000_s <=divide (to_unsigned(1*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size0001_s <=divide (to_unsigned(3*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size0011_s <=divide (to_unsigned(5*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size0010_s <=divide (to_unsigned(7*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size0110_s <=divide (to_unsigned(9*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size0111_s <=divide (to_unsigned(11*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size0101_s <=divide (to_unsigned(13*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size0100_s <=divide (to_unsigned(15*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size1100_s <=divide (to_unsigned(17*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size1101_s <=divide (to_unsigned(19*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size1111_s <=divide (to_unsigned(21*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size1110_s <=divide (to_unsigned(23*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size1010_s <=divide (to_unsigned(25*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size1011_s <=divide (to_unsigned(27*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size1001_s <=divide (to_unsigned(29*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;
rom_size1000_s <=divide (to_unsigned(31*rom_size_in, div_arg_width), to_unsigned(32, div_arg_width)) ;

The indication of the start address is given by the symbols. To illustrate the phase encoder we give an example for each type of modulation. Let's start with the QPSK modulation which uses four symbols as illustrated in table 3. For this type of modulation, the constellation diagram is divided into four, so that each angle can be associated with a different start address. Referring to Table 2, rom_size_00 represents the start address that is associated with symbol “00” in the constellation. Recall Equation 11:

$$S_{n,p} = sample\_size\,\frac{(2n+1)}{2^{p+3}} \tag{12}$$

Where n=0, 1, 2, 3 up to 15 for the symbols and p=0, 1, 2 for the QPSK, 8-PSK and 16-PSK modulations respectively. Table 1 is completed as follows, for $S_{n,\ p}$=(0, 0) we have n=0 (symbol “00”) and p=0 (QPSK constellation). With a sample size of 256, the start address is given by:

$$ROM_{start\_address} = S_{0,0} = 256\,\frac{(2(0)+1)}{2^{0+3}} \tag{13}$$

$$ROM_{start\_address} = 32 \tag{14}$$

For $S_{n,\ p}$=(1, 0), we have n=1 (symbol “01”) and p=0 (QPSK constellation).

$$ROM_{start\_address} = S_{1,0} = 256\,\frac{(2(1)+1)}{2^{0+3}} \tag{15}$$

$$ROM_{start\_address} = 96 \tag{16}$$

So we get for symbols “10” and “11” $ROM_{start\_address}$=160 and 224 respectively. To define the 8-PSK, the symbols are from 0 to 7, see Table 4. The same equation is used with n=0, 1, 2... 7 and p=1 to indicate the 8-PSK constellation. An example, for Sn, p=(6.0), we have n=1 (symbol “110”) and p=1 (constellation 8-PSK).

$$ROM_{start\_address} = S_{6,0} = 256\,\frac{(2(6)+1)}{2^{1+3}} \tag{17}$$

$$ROM_{start\_address} = 208 \tag{18}$$

To define the 16-PSK, the symbols are from 0 to 15, see Table 5. The same equation is used with n=0, 1, 2, 3..., 15 and p=2 to indicate the 16-PSK constellation. An example, for $S_{n,\ p}$=(15.0), we have n=1 (symbol “1111”) and p=2 (constellation 16-PSK).

$$ROM_{start\_address} = S_{6,0} = 256\,\frac{(2(15)+1)}{2^{2+3}} \tag{19}$$

$$ROM_{start\_address} = 248 \tag{20}$$

Thus for M-PSK we have n=0, 1, 2, 3..., M-1 we have

$$ROM_{Address} = S_{M,p} = sample\_size\,\frac{(2M+1)}{2^{p+3}} \qquad (21)$$

For each symbol, the Phase encoder module reads the ROM from the start address and will increment the address continuously until the arrival of a new symbol. At this point, the ROM reading will be reset to the newly calculated address and the process will repeat. With this algorithm, our designed phase encoder circuit is capable of generating all the data phases associated with each symbol of the constellation diagram. From Equation 12 we fill in the ROM address column of Tables 3, 4, and 5.

Table 3. QPSK Modulator Specifications

Bit Code	Phase	ROM Address
00	45°	32
01	135°	96
10	225°	160
11	315°	224

Table 4. 8-PSK Modulator Specifications

Bit code	Phase	ROM Address	Bit code	Phase	ROM Address
001	45°	16	101	225°	144
011	90°	48	111	270°	176
010	135°	80	110	315°	208
100	180°	112	000	360°	204

Table 5. 16-PSK Modulator Specifications

Bit code	Phase	ROM Address	Bit code	Phase	ROM Address
0000	11.25°	8	1000	191.25°	136
0001	33.75°	24	1001	213.75°	152
0010	56.25°	40	1010	236.25°	168
0011	78.75°	56	1011	258.75°	184
0100	101.25°	72	1100	281.25°	200
0101	123.75°	88	1101	303.75°	216
0110	146.25°	104	1110	326.25°	232
0111	168.75°	120	1111	348.75	248

To realize this project, we used the design flow provided by Xilinx as illustrated in Figure 5. The Genesys 2 Kintex-7 FPGA Development Board from Digilent is used. This card contains 50,950 logic slices, fully populated 400-pin FPGA Mezzanine Card (FMC) HPC connector, 10 GTX lanes, 5 PMOD ports, and dedicated USB port for JTAG programming and data transfers to name only these. The FMC HPC connector is a very attractive feature as it allows the interface with the Analog Device AD-FMCDAQ2-EBZ development board. This card contains dual, AD9680 14-bit, 1.0 GSPS, JESD204B ADC, AD9144 quad, 16-bit, 2.8 GSPS, JESD204B DAC.

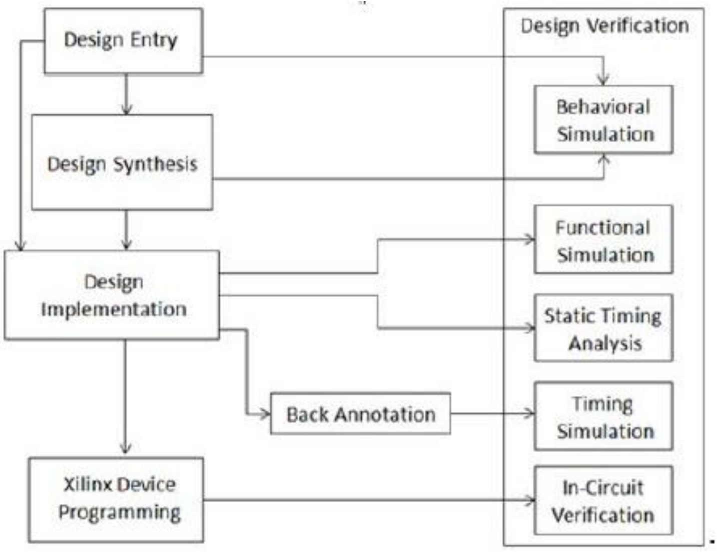


Figure 5. Xilinx basic design flow

4. FPGA IMPLEMENTATION RESULTS

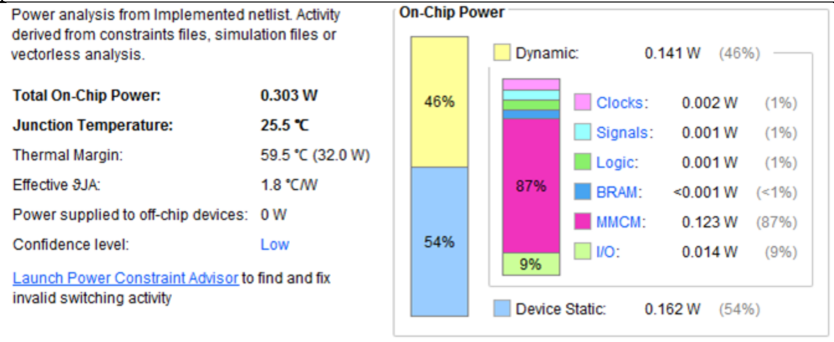
Table 6 shows FBGA area utilisation. Table 7 shows FBGA area utilisation continued. Figure 6 shows total on-chip power.

Table 6. FBGA Area Utilisation

Name	Slice LUTs (203800)	Slice Registers (407600)	F7 Muxes (101900)	Slice (50950)
QPSK Modulator	371	355	8	134
>clock_gen	0	0	0	0
Phase_enc	319	263	8	106
spi_master	42	59	0	25

Table 7. FBGA Area Utilisation Continued

Name	LUT as Logic (203800)	LUT Flip Flop Pairs (203800)	Bonded (500)	BUFGCTRL (32)
QPSK Modulator	371	221	25	4
>clock_gen	0	0	0	0
Phase_enc	319	193	0	0
spi_master	42	15	0	0


Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**0.303 W**
Junction Temperature:	**25.5 ℃**
Thermal Margin:	59.5 ℃ (32.0 W)
Effective ϑJA:	1.8 ℃/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

🟨 Dynamic:	0.141 W	(46%)	
🟪 Clocks:	0.002 W	(1%)	
🟦 Signals:	0.001 W	(1%)	
🟩 Logic:	0.001 W	(1%)	
🟦 BRAM:	<0.001 W	(<1%)	
🟥 MMCM:	0.123 W	(87%)	
🟩 I/O:	0.014 W	(9%)	
🟦 Device Static:	0.162 W	(54%)	

Figure 6. Total on-chip power

5. SIMULATION RESULTS

Figure 7 shows time domaine representation of the QPSK signal. Figure 8 shows time domaine representation of the 8-PSK signal. Figure 9 shows time domaine representation of the 16-PSK signal.

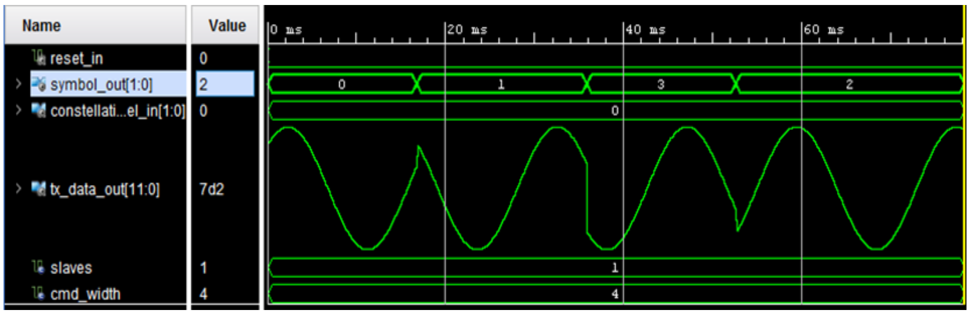


Figure 7. Time domaine representation of the QPSK signal

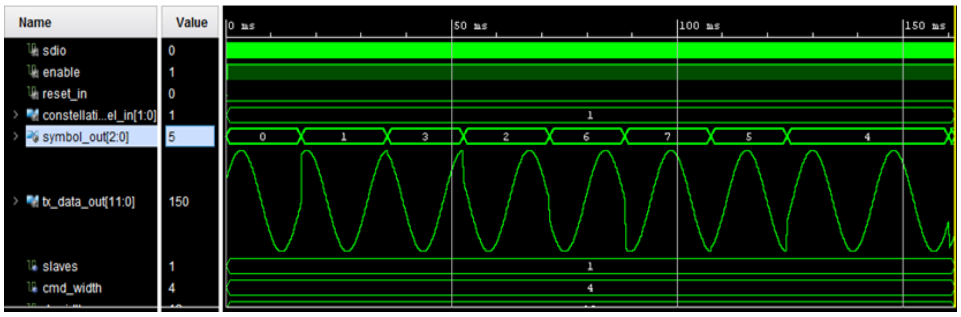


Figure 8. Time domaine representation of the 8-PSK signal

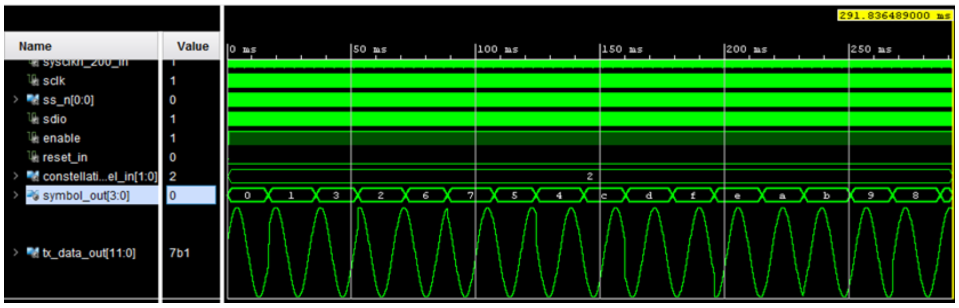
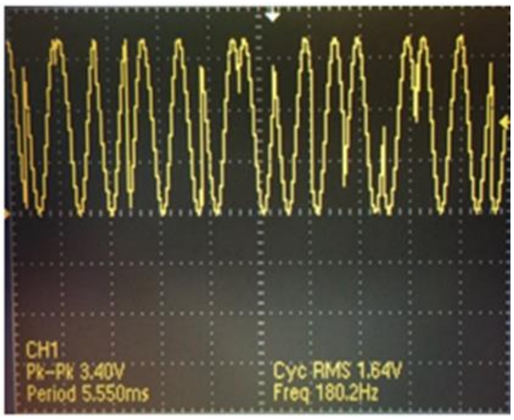


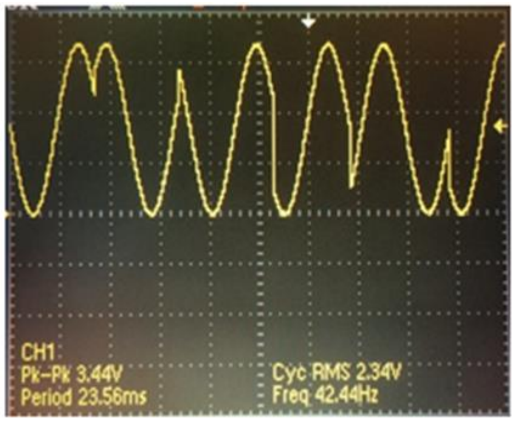
Figure 9. Time domaine representation of the 16-PSK signal

6. EXPERIMENTAL RESULTS

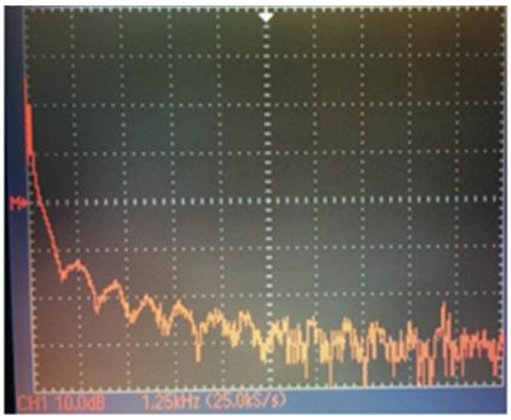
As shown in Figure 10(a) time domaine representation of the QPSK Signal, (b) QPSK result zoom in, (c) Frequency Domain Representation of QPSK signal, (d) time domain representation of the 8-PSK signal, (e) 8-PSK result zoom in, and (f) Frequency Domain Representation of the 8-PSK signal. As shown in Figure 11 (a) Time Domain Representation of the 16-PSK Signal, (h) 16-PSK result zoom in, (b) Frequency Domain Representation of the 16-PSK signal, and (c) experimental setup.



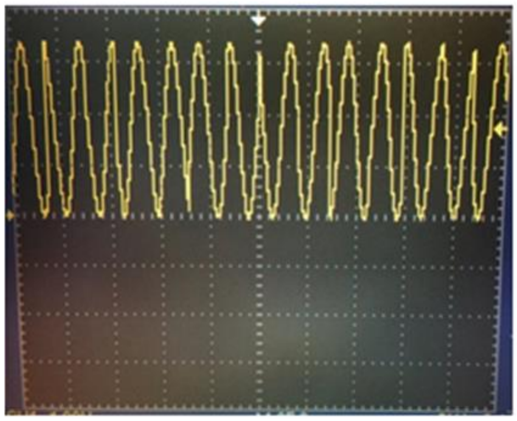
(a)



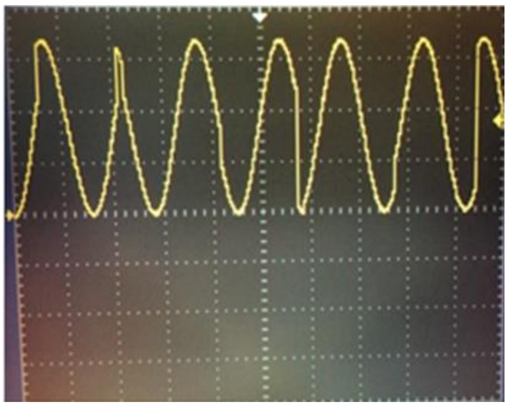
(b)



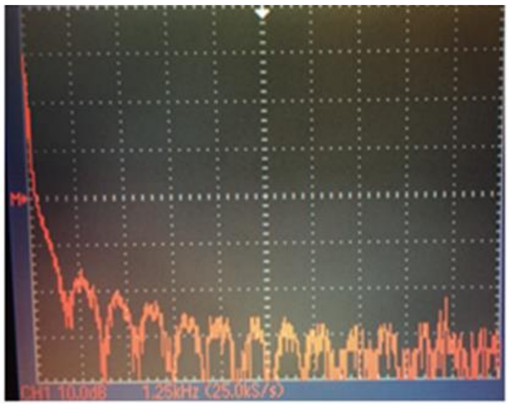
(c)



(d)

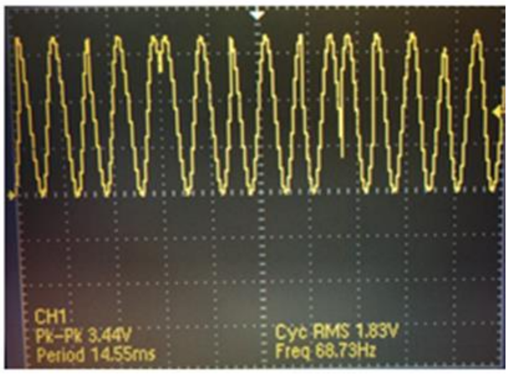


(e)

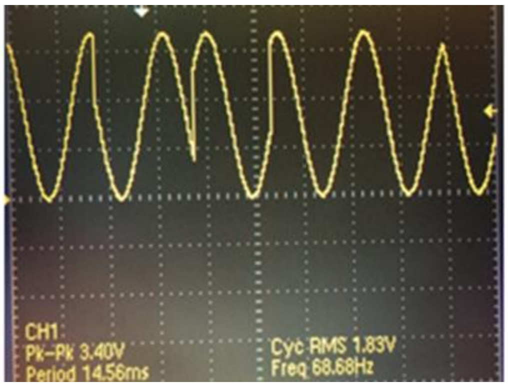


(f)

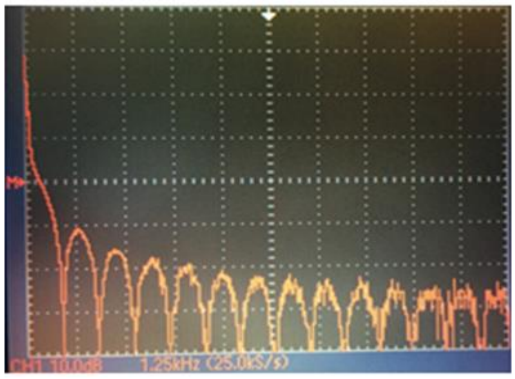
Figure 10. (a) Time Domaine Representation of the QPSK Signal, (b) QPSK Result Zoom in, (c) Frequency Domain Representation of QPSK signal, (d) Time Domain Representation of the 8-PSK Signal, (e) 8-PSK Result Zoom in, and (f) Frequency Domain Representation of the 8-PSK signal



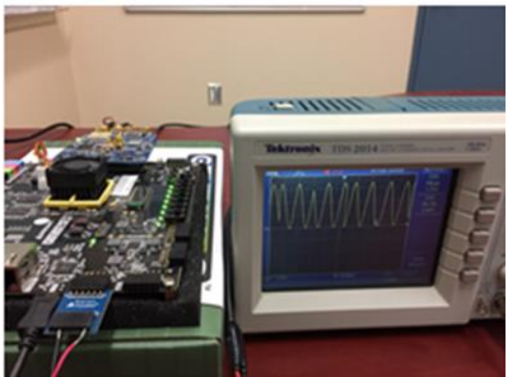
(a)



(b)



(c)



(d)

Figure 11. (a) Time Domain Representation of the 16-PSK Signal, (b) 16-PSK Result Zoom in, (c) Frequency Domain Representation of the 16-PSK signal, and (d) Experimental Setup

7. ENCRYPTION KEY GENERATOR CIRCUIT

Random numbers are essential to ensure the security of telecommunications, whether they are used in cryptographic algorithms such as Diffie-Hellman and RSA, or in frequency hopping mechanisms to name a few. It is important that randomness be part of the seed of a security algorithm to avoid reverse engineering. However, RNGs are often made in software through an algorithm and True RNGs (TRNGs), RNGs made in hardware, are often costly. This section of this article shows how to build a low cost TRNG that can reach very high entropy. Many such circuits have been reviewed. However, we used the circuit described in [15] as a starting point to create our noise generating circuit. In [15], bipolar junction transistors (BJT) have been used as a source of noise. In the modified architecture we replaced the two BJTs upstream by two Zener diodes as shown in Figure 12. The advantage of using two identical noise sources is that they will likely have the same statistical distribution in terms of voltages. Therefore, we will end up with close to the same amount of 1's and 0's at the output of the comparator. In Figure 12, the noise generated by the two Zener diodes will be amplified before being routed to the comparator circuit. The result of this amplification is shown in Figure 13 (a). Figure 13 (b) shows the frequency domain representation of the circuit which only measures noise as expected. Therefore, the output of the comparator will be a mixture of the high frequency and low frequency components. In order for this output signal to be usable, the high frequency components that can be considered as glitches must be removed using a low-pass filter.

Before applying the low-pass filter, the digital output as depicted in Figure 14 looks noisy. Therefore, it is necessary to do some statistical analysis to ensure that the bits generated are indeed random. Using NIST SP-800-90B (Second draft), we implemented several functions using Matlab to conduct this statistical analysis. The goal is to determine whether or not the bits are Independent and Identically Distributed (IID). Twenty tests were made using the sampled bits. These include calculating the amount of rising and falling edge, covariance tests, periodicity tests, etc. The results of every test were then compared to 10,000 binary lists shuffled using the Fisher-Yates algorithm. Two counters were used to compare the results: C0 was incremented when the output of a test was greater for the shuffled list and C1 was incremented when the output of the test was the same for the shuffled list and the sampled list. NIST determines that a sample is non-IID if (C0 + C1) ≤ 5 or if C0 ≥ 9995 for any test. Since the results obtained were far from those values, we concluded that the bits were IID according to NIST's standard. Then, we calculated a min-entropy of 0.96 using the Most Common Value Estimate (section 6.3.1 of NIST's SP-800-90B).

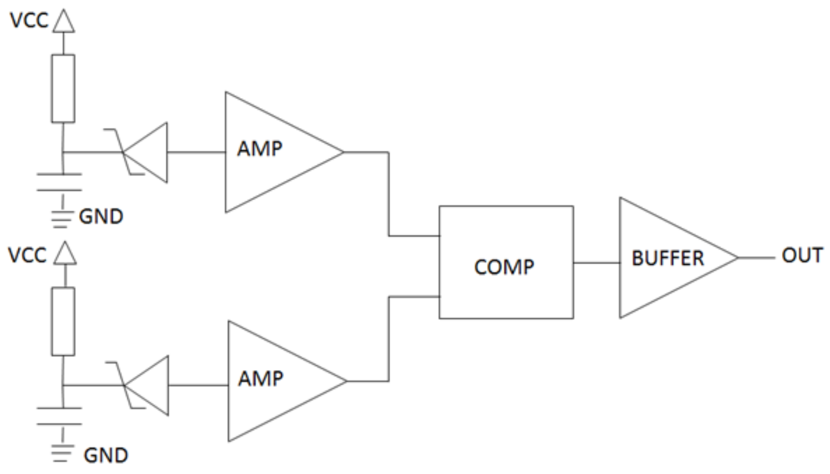
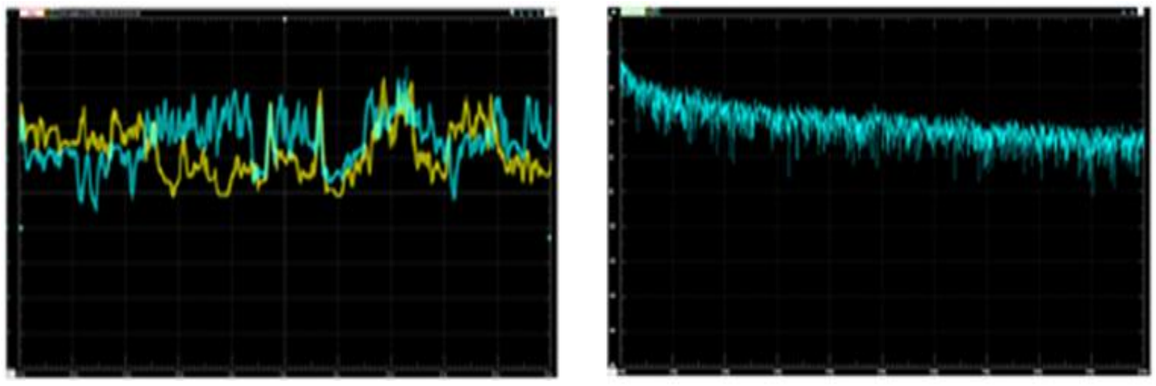


Figure 12. RGN circuit



(a) (b)

Figure 13. (a) Time domain representation of the circuit at the output of the two amplifiers depicted in blue and yellow respectively, and (b) frequency domain representation

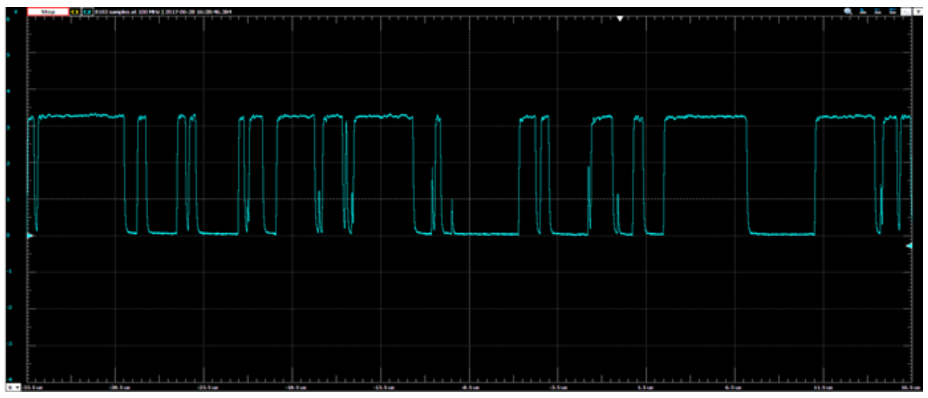


Figure 14. Time domain representation of the circuit at the output of the comparator

8. CONCLUSION

In the time of conflicts fast and accurate exchanging or gathering of intelligence between allies over a secured channel is an asset. Digital modulations techniques are widely used in wireless communication. Accordingly, in this article, we have proposed and described a new reconfigurable PSK modulator. The circuit is realized on the Digilent Genesis 2 development board. It consumes only 303 mW of power. The experimental results for the three types of modulations implemented (QPSK, 8-PSK, and 16-PSK) agree with the simulation data. The synthesis and implementation phases were carried out with a clock of 200 MHz. This clock has been reduced to meet the conversion rate of the PMODDA2 DAC. However, the FMC port of the Genesis 2 development board allows interfacing with the Analog Device AD-FMCDAQ2-EBZ evaluation board. The ADC (AD9680) and DAC (AD9144) mounted on the the AD-FMCDAQ2-EBZ support input data rate of 1.25 GSPS and 2.8 GSPS respectively. Therefore, by using the AD-FMCDAQ2-EBZ in conjunction with the Genesis 2 development board we will no longer need to do a down conversion of the current clock frequency. In addition of proposing this new design, we have also explored some avenues to generate random encryption keys. We arrived at a circuit capable of generating random encryption keys. This circuit has demonstrated a min-entropy of 0.96. This result is very promising to ensure data security.

REFERENCES

[1] P. R. Kolankar and S. V. Sakhare., "FPGA Implementation of QPSK Modulator by Using Hardware Co-simulatation," *International Journal of Engineering Research and Development,* vol. 10, pp. 86-93, Apr 2014.
[2] N. Tanawade, and S. Sudhansu, "FPGA Implementation of Digital Modulation Techniques BPSK and QPSK Using HDL Verilog," *International Journal of Computer Applications,* vol. 165, no. 12, May 2017.
[3] Q. Gu, RF System Design of Transceivers for Wireless Communications, New York City, Springer, 2006, pp. 66-67.
[4] B. Antao, Modeling and Simulation of Mixed Analog-Digital Systems, New York City, Springer, 1996, pp. 38-40.
[5] A. M. Moubark, *et al.*, "FPGA Implementation of a Low Power Digital QPSK Modulator Using Verilog HDL," *Journal of Applied Sciences,* vol. 3, pp. 385-392, 2013.
[6] H. He, *et al.*, "Design and Implementation of Signal Generator Based on DDS," *International Journal of Simulation, Systems, Science & Technology,* vol. 17, no. 32, 2013.
[7] K. A. Monpara, and S. B. Parmar, "Design and Implementation of QPSK Modulator Using Digital Subcarrier," *Journal of Information, Knowledge and Research in Electronics and Communication Engineering,* issue 01, pp. 394-398, Nov 2011.
[8] A. A. Safi and B. Bazuin, "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 (UEMCON), New York City, USA, Oct. 20-22,* 2016.
[9] A. Akbal and M. Sonmez, "FPGA Based, Low Cost Modulators of BPSK and BFSK, Design and Comparison of Bit Error Rate over AWGA," *Gazi University Journal of Science,* vol. 26, no. 32, 2013.
[10] S. Rajaram and R. Gayathre, "FPGA Implementation of Digital Modulation Schemes," *International Journal of Innovative Research in Science, Engineering and Technology,* vol. 3, special issue 3, 2014.
[11] S. O. Popescu, *et al.*, "QPSK Modulator on FPGA FPGA," *IEEE Explore, 9th International Symposium on Intelligent Systems and Informatics, Subotica, Serbia,* pp. 359-364, 2011.
[12] N. P. Shrirao and A. P. Thakare, "Design of Digital Modulators: BASK, BPSK, BFSK Using VHDL," *International Journal of Advance Research in Computer Science and Software Engineering,* vol. 3, issue 1, pp. 382-386, 2013.
[13] R. Garg and E. A. Mital, "Design and Implementation for Combination of QPSK & BPSK Modulation Techniques," *International Journal of Advanced Research in Electronics and Communication Engineering,* vol. 3, issue 8, 2014.
[14] M. Barnela, "Digital Modulation Schemes Employed in Wireless Communication: A literature Review," *International Journal of Wired and Wireless Communications,* vol. 2, issue 2, 2014.
[15] R. Marston, “Bipolar Transistor Cookbookx – Practical Oscillator And White Noise Waveform Generator Circuits”, *Nuts and Volts Magazine,* November 2013.