FPGA Implementation of QPSK Modulator Based on Matlab / Xilinx System Generator

Kaliprasanna Swain[1]
kaleep.swain@gmail.com

Manoj Kumar Sahoo[2]
mksahoo@hotmail.com

Gandhi Institute for
Technological Advancement
(GITA) /ECE, Bhubaneswar,
Odisha, India

***Abstract*: This paper presents an efficient approach for the implementation of typical communication structures studied in class. This scheme is beneficial where the objective is to implement the physical working of complex DSP or communication structures or algorithms without requiring detailed knowledge of hardware design and hardware description languages. The approach is based on the Xilinx System Generator for DSP tool, which integrates itself with the MATLAB, based Simulink Graphics environment and relieves the user of the textual HDL programming. The purposed design is the QPSK modulator which is then simulated using Matlab/ Simulink environment and System Generator, FPGA design as well as implemented on a Spartan 3 (xc3s-5ft256) Spartan3 starter kit board. The modulator algorithm has been implemented on FPGA using the VHDL language on Xilinx ISE Design suite 13.2.**

Index Terms— Xilinx, FPGA, Spartan3, System Generator, QPSK Modulator

I. Introduction

Digital modulation technology is an important content of modern communication. Modulation is essential in transmitting two or more signals in the same time because to avoid interference between the signals and also ensure that errors are avoided during transmission. The digital information is transmitted as a series of ones (1) and zeros (0) called bits. In order to transmit them over long distances, different modulation scheme have been used.

For digital modulation, instead of varying the amplitude or the frequency of the carrier signal, it is preferred to vary the phase because it offers better protection in transmitting signals [1]. To increase the bit rate without increasing the bandwidth, various modulation techniques have evolved. Straight forward extensions of the techniques considered in the previous section are QPSK, OQPSK, and MSK [1].

TABLE I
Voltages for Bit pattern

Bit Pattern	Mess-age	Signal Transmitted
00	m_1	$S_1(t) = V\cos(2\pi f_c t)$, $0 \le t \le T_s = 2T_b$
01	m_2	$S2(t) = V\sin(2\pi f_c t)$, $0 \le t \le T_s = 2T_b$
11	m_3	$S_3(t) = -V\cos(2\pi f_c t)$, $0 \le t \le T_s = 2T_b$
10	m_4	$S_4(t) = -\ V\sin(2\pi f_c t)$, $0 \le t \le T_s = 2T_b$

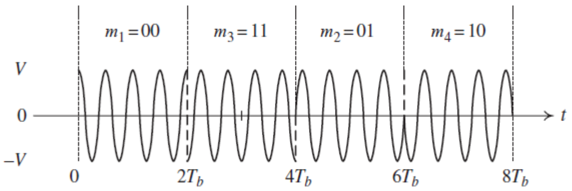


Figure.1 QPSK Signal mapping to message

The basic idea behind QPSK exploits the fact that cos ($2\pi f_c t$) and sin ($2\pi f_c t$) are orthogonal over the interval [0, *Tb*] when *fc* = *k*/*Tb*, *k* integer. Just as in analog modulation, this can be used to transmit two different messages over the same frequency band. To accomplish the bit stream is taken two bits at a time and mapped into signals as shown Figure 1.

II. GENERATION OF A QPSK SIGNAL

Figure below shows the block diagram of the mathematical implementation of QPSK.

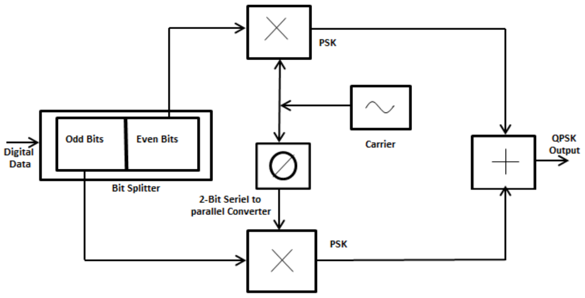


Figure.2 QPSK generation methods

At the input of the modulator, the digital data’s even bits (i.e., bits 0,2,4 and so on) are stripped from the data stream by a “bit-splitter” and are multiplied with a carrier to generate a BPSK signal (called PSK_I). At the same time, the data’s odd bits (i.e., bits 1, 3, 5 and so on) are stripped from the data stream and are multiplied with the same carrier to generate a second BPSK signal (called PSK_Q). However, the PSK_Q signal’s carrier is phase shifted by 90° before being modulated [2].

The two BPSK signals are then simply added together for transmission and, as they have the same carrier frequency, they occupy the same portion of the radio frequency spectrum. While this suggests that the two sets of signals would be irretrievably mixed, the required 90° of phase separation between the carriers allows the sidebands to be separated by the receiver using phase discrimination.

The constellation diagram of QPSK is as shown below

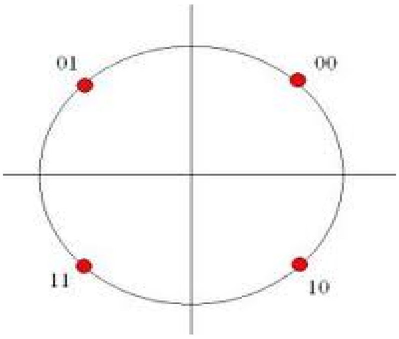


Figure.3 QPSK Signal Constellation

III. OVERVIEW OF THE XILINX SYSTEM GENERATOR

The Xilinx System Generator [3] [4] for DSP is a system level modeling and design tool that facilitates FPGA design and has the ability to work at a higher level of abstraction. It enables the use of the MathWorks graphical model based Simulink design environment for FPGA design. The System Generator integrates itself with Simulink and FPGA designs are captured by using the Xilinx specific Blockset. Thus, designing a hardware model in Simulink is as simple as designing any other Simulink model with the only difference being the use of Xilinx Blockset instead of those found in Simulink. The System Generator provides many DSP building blocks in the form the Xilinx DSP Blockset for the Simulink environment. The variety in this Blockset ranges from common DSP blocks such as adders, multipliers, registers etc to more complex blocks such as FFTs, filters, memories, forward error correction etc. Thus, previous experience with low level system design and HDLs is not required when using this tool. The System Generator uses the Xilinx ISE software and IP core generators to convert designed model into the equivalent HDL code. The remaining FPGA implementation steps including synthesis, place and route, etc. are automatically performed to generate bit file that is downloaded on to the FPGA.

IV.IPLEMENTATION OF THE QPSK SYSTEM IN SYSTEM GENERATOR

Fig.4 illustrates the implementation of a QPSK Modulator using System Generator [5][6] tools in Simulink whereas Figure 5 illustrates the output signal waveform generated by the scope.

The System Generator Blockset [7] contains:

The *gateway in* blocks: the inputs into the Xilinx portion of the Simulink design;

The *gateway out* blocks: the outputs from the Xilinx portion of the Simulink design;

The *Mux* block: implements a multiplexer. It has one select input and a configurable number of data inputs that can be defined by the user.

The *AddSub* block: The Xilinx AddSub block implements an adder/ subtractor. The operation can be fixed (Addition or Subtraction) or changed dynamically under control of the sub mode signal. When Addition/Subtraction is selected, the block operation is determined by the sub input port, which must be driven by a Boolean signal. When the sub input is 1, the block performs subtraction. Otherwise, it performs addition.

The *Mult* block: The Xilinx Mult block implements a multiplier. It computes the product of the data on its two input ports, producing the result on its output port.

The *Delay* block: The Delay block implements a fixed delay of L cycles. This block is used mainly for matching pipeline delays in other portions of the circuit.

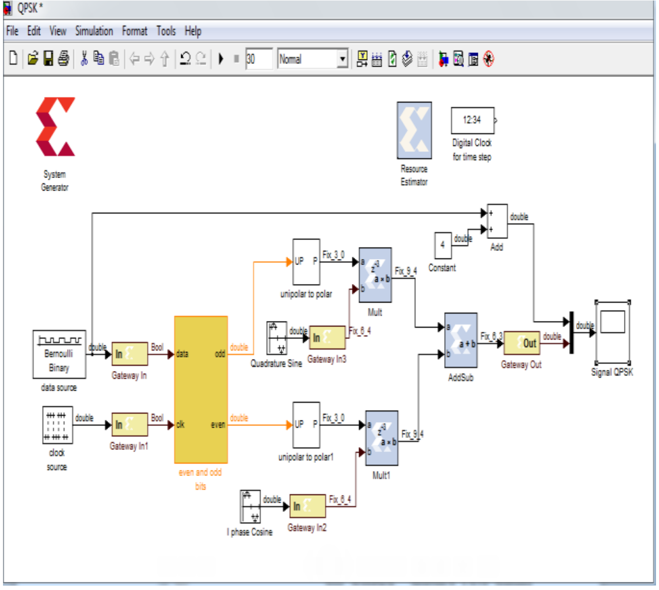


Figure.4 QPSK generation with Simulink / Xilinx System Generator

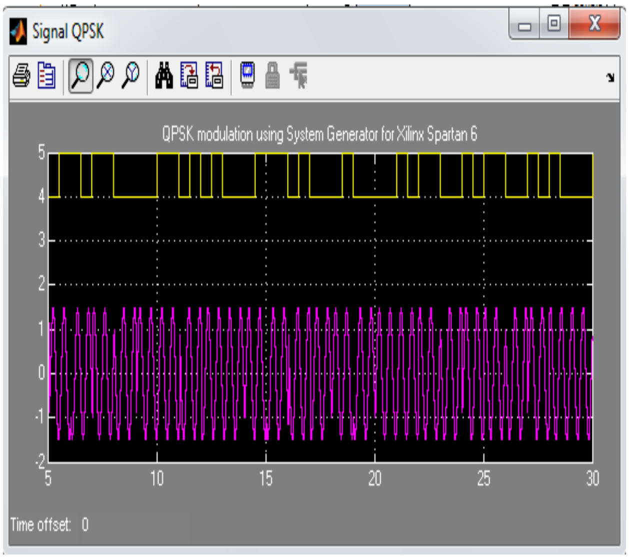


Figure.5 Simulation result of QPSK by Xilinx System Generator Method

V. DEVELOPMENT OF CO-HARDWARE SIMULATION OF QPSK MODULATION MODEL

Xilinx System Generator provides Co-Hardware simulation, making it possible to incorporate a design running in an FPGA directly into a Simulink simulation. "Hardware Co-Simulation" compilation targets automatically create a bitstream by using XFLOW and associate it to a block. When the design is simulated in Simulink, results for the compiled portion are calculated in hardware. This allows the compiled portion to be tested in actual hardware and can speed up simulation dramatically.

XSG also some of board definition for Co-Hardware simulation, for others the designer have to develop the board definition which is similar to device driver in embedded system.

A. The QPSK modulation HW library Generation

This is the process of generation of equivalent hardware of the above model, for a targeting device, here we consider Spartan3 (XC3S200-5FT256) with the help of XSG by using XFLOW the equivalent hardware Target device programmable bit file is generated as shown in the figure7.

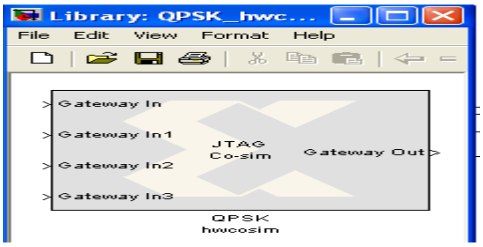


Figure.6 Generated QPSK Hard Ware Co-Simulation library

Then the library model is integrated with the basic model as shown below.

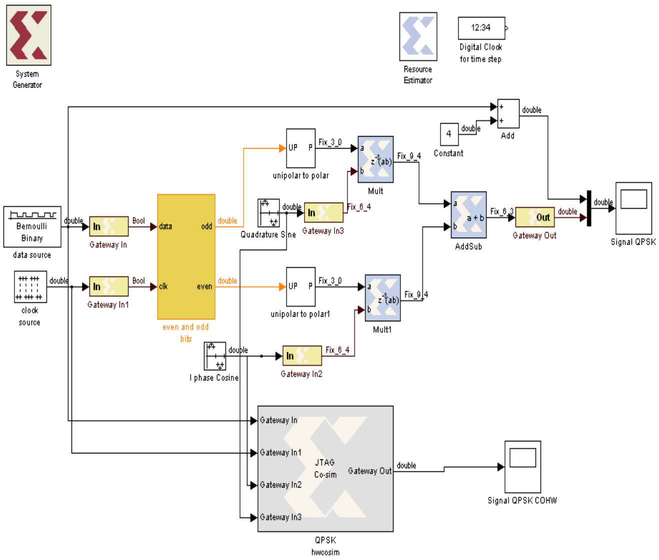
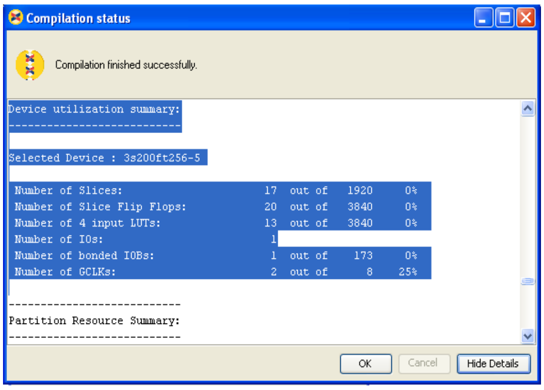


Figure.7 QPSK Co-Hardware simulation model

- The library file can be generated for standalone JTAG method, where the standalone process is considered.

- Then the device utility, timing parameter, and output are analysed for JTAG Co-sim block.



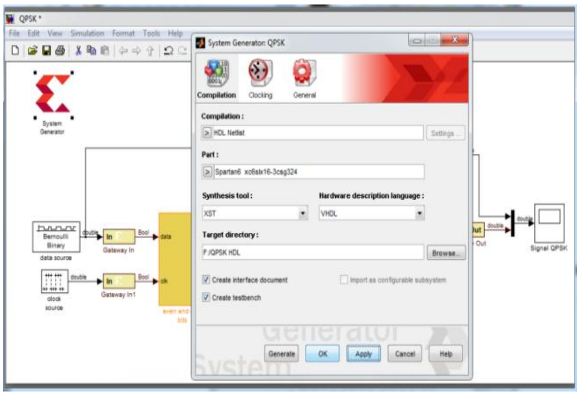
Compilation finished successfully.

```
Device utilization summary:
---------------------------

Selected Device : 3s200ft256-5

 Number of Slices:                      17  out of   1920     0%
 Number of Slice Flip Flops:            20  out of   3840     0%
 Number of 4 input LUTs:                13  out of   3840     0%
 Number of IOs:                          1
 Number of bonded IOBs:                  1  out of    173     0%
 Number of GCLKs:                        2  out of      8    25%


---------------------------
Partition Resource Summary:
---------------------------
```

Figure.8 Design utility in XFLOW process

VI. QPSK MODULATOR ON SPARTAN 3 (NEXYS2) KIT BOARD

The QPSK modulators are implemented on SPARTAN-3 starter kit board [8]. The SPARTAN-3 starter kit is a powerful digital system design platform built around a Xilinx Spartan 3 FPGA. With 16Mbytes of fast SDRAM and 16Mbytes of Flash ROM, the Nexys-2 is ideally suited to embedded processors like Xilinx's 32-bit RISC Microblaze™. The on-board high-speed USB2 port, together with a collection of I/O devices, data ports, and expansion connectors, allow a wide range of designs to be completed without the need for any additional components. Spartan 3E features include: 500K-gate Xilinx Spartan 3E FPGA, USB2-based FPGA configuration and high-speed data transfers (using the free Adept Suite Software), USB-powered (batteries and/or wall-plug can also be used), 16MB of Micron PSDRAM &16MB of Intel Strata Flash ROM, Xilinx Platform Flash for nonvolatile FPGA configurations, Efficient switch-mode power supplies (good for battery powered applications), 50MHz oscillator plus socket for second oscillator, 60 FPGA I/O's routed to expansion connectors (one high speed Hirose FX2 connector and four 6-pin headers), 8 LEDs, 4-digit 7-seg display, 4 buttons, 8 slide switches.



Compilation :
HDL Netlist

Part :
Spartan6 xc6slx16-3csg324

Synthesis tool : XST

Hardware description language : VHDL

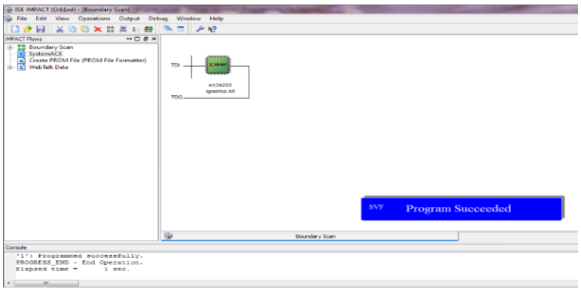
Target directory :
F:/QPSK HDL

☑ Create interface document

□ Import as configurable subsystem

☑ Create testbench

Figure.9 Netlister Configuration mode for QPSK on FPGA


ISE iMPACT (O.61xd) - [Boundary Scan]
File Edit View Operations Output Debug Window Help
iMPACT Flows
Boundary Scan
SystemACE
Create PROM File (PROM File Formatter)
WebTalk Data
TDI
TDO
xc3s200
qpsktop.bit
SVF Program Succeeded
Boundary Scan
Console

```
'1': Programmed successfully.
PROGRESS_END - End Operation.
Elapsed time =       1 sec.
```

Figure.10 Configuration mode for QPSK on FPGA

VII. RESULT

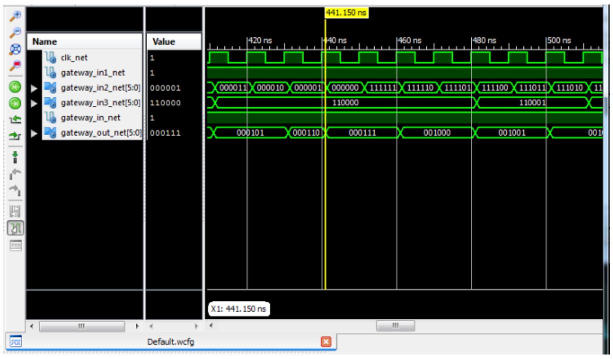
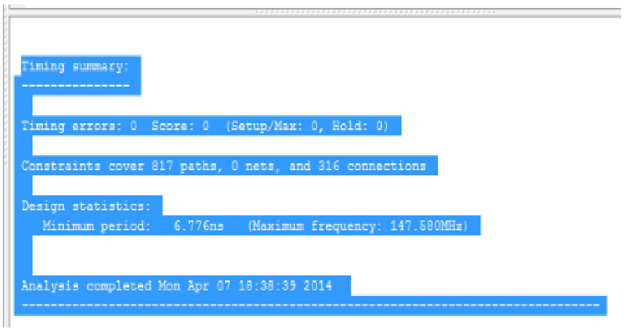


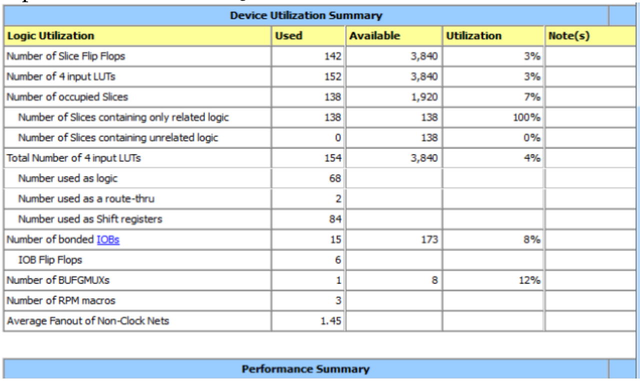
Figure.11 Behavioral simulation of QPSK with 2 step model



```
Timing summary:
---------------

Timing errors: 0  Score: 0  (Setup/Max: 0, Hold: 0)

Constraints cover 817 paths, 0 nets, and 316 connections

Design statistics:
   Minimum period:   6.776ns   (Maximum frequency: 147.580MHz)


Analysis completed Mon Apr 07 18:38:39 2014
--------------------------------------------------------------------------------
```

Figure.12 Timing Report of QPSK

Figure.13 represents the design summary which represents the utilization of flip-flops, LUTs, slices used from the capabilities of the FPGA from the Spartan 3E board. Figure.14 illustrates the design on the FPGA from the Spartan 3E board and figure.15 shows the schematic representation of the QPSK modulator.



Device Utilization Summary

Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	142	3,840	3%	
Number of 4 input LUTs	152	3,840	3%	
Number of occupied Slices	138	1,920	7%	
Number of Slices containing only related logic	138	138	100%	
Number of Slices containing unrelated logic	0	138	0%	
Total Number of 4 input LUTs	154	3,840	4%	
Number used as logic	68			
Number used as a route-thru	2			
Number used as Shift registers	84			
Number of bonded IOBs	15	173	8%	
IOB Flip Flops	6			
Number of BUFGMUXs	1	8	12%	
Number of RPM macros	3			
Average Fanout of Non-Clock Nets	1.45			

Performance Summary

Figure.13 Design Summery of Device Utilization for QPSK

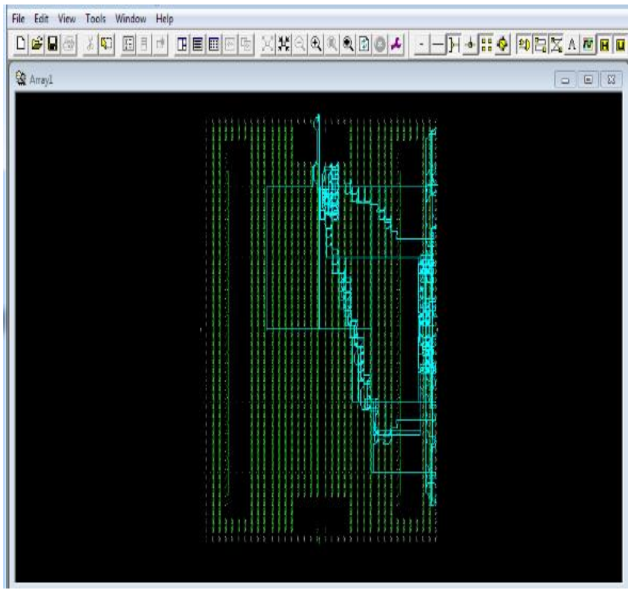


Figure.14 Route Path of the QPSK modulator on the Spartan 3E Board.

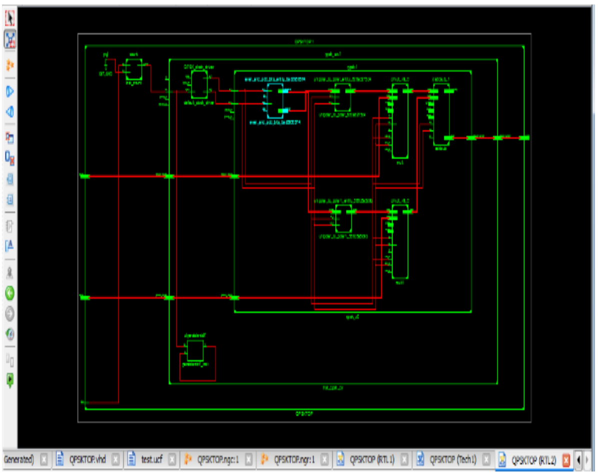


Figure.15 RTL Schematic for QPSK

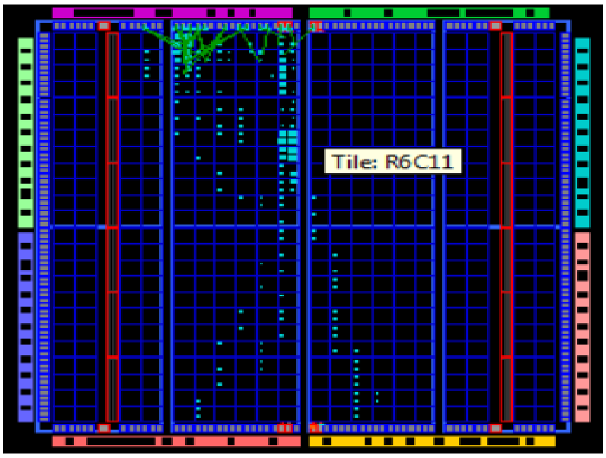
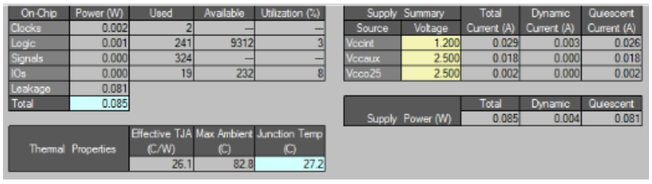


Figure.16 Utility of design on FPGA using Plan Ahead tool

The Xilinx power analyzer is analyzed the power consumption of the design which depend on the family of device is used and clock, logic, signal, I/Os and leakage power at a test condition.



On-Chip	Power (W)	Used	Available	Utilization (%)
Clocks	0.002	2	---	---
Logic	0.001	241	9312	3
Signals	0.000	324	---	---
IOs	0.000	19	232	8
Leakage	0.081			
Total	0.085			

Supply Summary		Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.200	0.029	0.003	0.026
Vccaux	2.500	0.018	0.000	0.018
Vcco25	2.500	0.002	0.000	0.002

	Total	Dynamic	Quiescent
Supply Power (W)	0.085	0.004	0.081

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	26.1	82.8	27.2

Figure.17 Power utility of QPSK Design

VIII. CONCLUSION

The use of the Xilinx System Generator tool for DSP education and research is presented. It is shown that this tool is ideal for developing FPGA based hardware without the requirement of learning HDLs and Hardware Design. The QPSK modulated signal is obtained by adding the two modulated signals. The odd-sequence was modulated with the cosine waveform and the even sequence with the sine waveform. Adding the two modulated signals, the QPSK signal was obtained. The design has been written in the VHDL programming code by Xilinx software.

REFERENCES

[1] H.Nguien, E.Shwedyk, “A First Course in Digital Communications”, Cambridge University Press, 2009.

[2] Sakshat Virtual Labs, QPSK Modulation (Real time experiment), URL: http://iitg.vlab.co.in/?sub=59&brch=163&sim=1065&cnt=1.

[3] S.O. Popescu, A.S. Gonteanand, D.Ianchis,“QPSK Modulator on FPGA”, SISY 2011, 2011 IEEE 9th International Symposium on Intelligent Systems and Informatics, September 8-10, 2011, Subotica, Serbia.

[4] S.O. Popescu, A.S.Gontean and G.Budura,“BPSK System on Spartan 3E FPGA”, SAMI 2012, 10th IEEE Jubilee International Symposium on Applied Machine Intelligence and Informatics, January 26-28, 2012, Herl’any, Slovakia.

[5] S.T.Karris, “Introduction to Simulink with Engineering Applications”, Orchard Publications, USA, 2006.

[6] ISE 13.2, Quick Start Tutorial, Xilinx

[7] Xilinx System Generator for DSP User Guide, r10.1.1, April 2008

[8] Spartan3 starter kit Reference Manual Revision: July 11, 2011, URL: www.digilentinc.com