FPGA Design Flow for SDR Transceiver using System Generator

Tahseen Flaih Hasan

Abstract- Software defined radio (SDR) is highly configurable hardware platform that provides technology for realizing the rapidly expanding third (even future) generation digital wireless communication infrastructure. While among the silicon alternatives available for implementing the various functions of SDR, field programmable gate array (FPGA) is an attractive option in terms of performance, power consumption, and flexibility. This paper examines 16-QAM (Quadrature Amplitude Modulation) SDR transmitter and receiver with an appropriate timing recovery system using FPGA. We provide a tutorial style overview of techniques and schemes for system (abstraction) level design of the 16-QAM SDR transmitter and receiver using Xilinx System Generator, ModelSim and Synplify Pro software, and FPGA implementation (realization) using Xilinx ISE software. Two design alternatives are presented to highlight the rich design space accessible using the FPGA configurable logic. At last, this new design technique would help in designing and realizing SDR to 3G wireless communication system and accelerate the transition to 4G wireless communication system.

Keywords- FPGA, 16QAM, SDR, System Generator

I. INTRODUCTION

In the late 1990s, Field Programmable Gate Array (FPGA) began to approach gate counts and speeds of Application Specific Integrated Circuit (ASIC), although ASIC remains superior for both parameters. However, when FPGA reached a semi-equivalent status, it became more attractive than ASIC for design and development work, due to its flexibility and reprogrammability [1]. With efficient massive parallel MAC (Multiply-Accumulate) structures, FPGA quickly became the method of choice for most of the Digital Signal Processing (DSP) algorithm implementations leading to high level of integration of various functionalities. However, the design flow for DSP algorithms and digital-logic design via FPGA.s did not develop in unison. The DSP design flow is much shorter. One of the most world-renowned software tools for DSP algorithm design is Matlab, produced by Mathworks Inc [2]. Many companies researching and developing DSP, controls, and communication algorithms currently realize all of their high level modeling in Matlab, including all of the necessary test harnesses. In some cases, especially for real time applications, these designers would utilize a subprogram of Matlab, called Simulink.

Manuscript published on 30 April 2014.
***** Correspondence Author (s)
Tahseen Flaih Hasan, Tahseen Flaih Hasan, Foundation of Technical Education, Baghdad, Iraq

Simulink is a block diagram approach (connect boxes together) to system modeling that allows for modeling of streaming data, multi rate systems, and other numerically application intensive areas [2]. The issue that arose was that there was a gap between the Matlab/Simulink design flow, and FPGA design flow, as is illustrated by Fig.1 (or elsewhere [3]). Designers needed the parallel processing advantages offered by FPGA implementation. However, there was no software suite of tools that would allow a DSP algorithm designer to interface directly with an FPGA system architect. Furthermore, there was no streamlined process to verify equivalency of a hardware implemented DSP algorithm against the original test harnesses used in Matlab and Simulink.

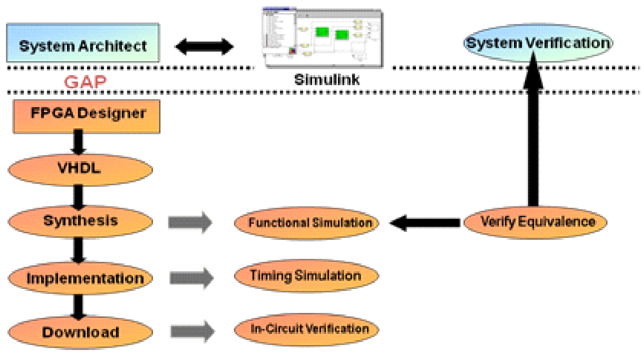


Fig.1. Separation between DSP and FPGA Design Flows [3].

To solve this separation in design flow issue, a new electrical engineering work force skill began to be requested by companies around the world that specialized in both DSP algorithm development and FPGA design. This skill was the ability to execute brute force translation of Matlab and C++ code into Verilog or HDL. This knowledge required not only abroad experience in software languages, but also an experience with converting software constructs into parallel hardware constructs. Therefore, the current DSP algorithm to FPGA design flow, illustrated in sections below, has pieced together Matlab Algorithm & Test Harnesses manual conversion to HDL (verilog / VHDL) Synthesis, Place & Route, & program Target FPGA Hardware verification from the entirely design flow.

II. SDR IN XILINX SYSTEM

This section focuses on the system level design using DSP Design Tools. Based on the available resources from [4] and [5], a simple 16-QAM (Quadrature Amplitude Modulation) SDR (Software Defined Radio) transmitter and receiver model is designed for Virtex-4 FPGA architecture, in Xilinx System Generator and MATLAB/Simulink environment. The GUI (Graphical User Interface) of the design is model-based structure using Xilinx specific blockset as shown in Fig. 2.

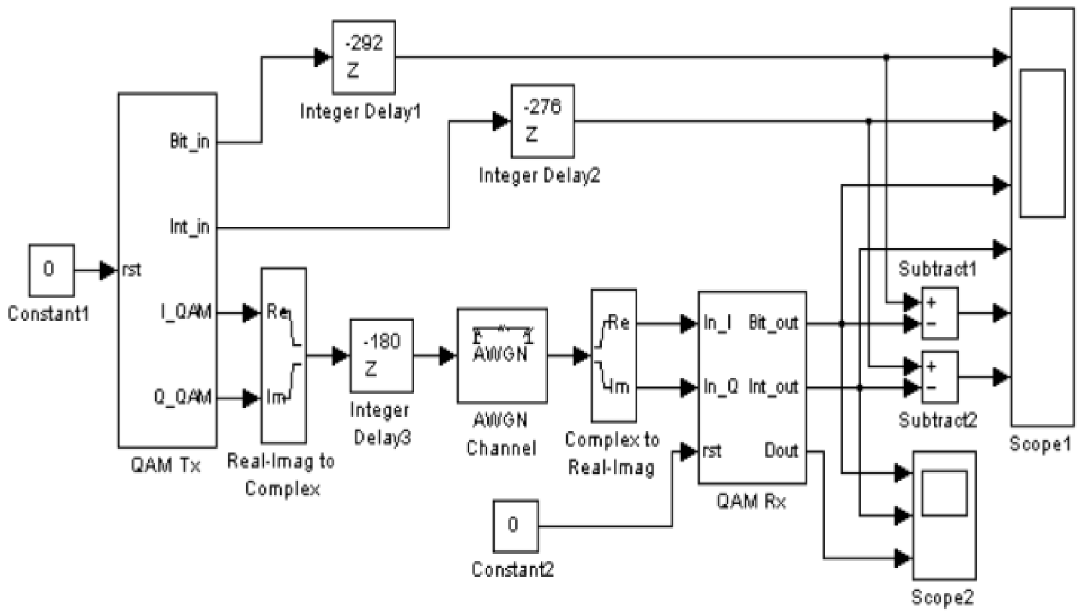


Fig.2. SDR Design using System Generator

2.1- Transmitter Link Modeling

The modeling of 16-QAM transmitter subsystem shown in Fig. 3 was modeled for Virtex-4 Xilinx FPGA. The Pseudorandom (PR) bit generator provides input bit to represent data received from the network layer. In 16-QAM modulator, each group of 4 successive bits is interpreted as 4-bit integer through serial-to-parallel conversion; then mapped to pairs of symbols in In-phase (I) and Quadrature (Q) channels using binary constellation ordering for normalized average power of 1 W. Finally, two identical root raised cosine (RRC) filters with fine gain perform pulse-shaping on the IQ symbol-pairs for band-limited transmission. The RRC filter design parameters are: order = 64, over-sampling rate = 16, roll-off factor = 0.35, scale passband with Kaiser window, and β = 0.5. The input and output signals of transmitter are shown in Fig. 4.

As shown from Fig. 4, the input bit (10 Mbps bit rate) is modulated and mapped to 16-QAM baseband modulated (IQ) signals (2.5 Msps sample rate). After pulse-shaping process, the resulting IQ signals are up-sampled by 16 to become 40 Msps IQ transmitted signals with fixed point precision of 16 bits and 13th binary point for the actual hardware modeling.

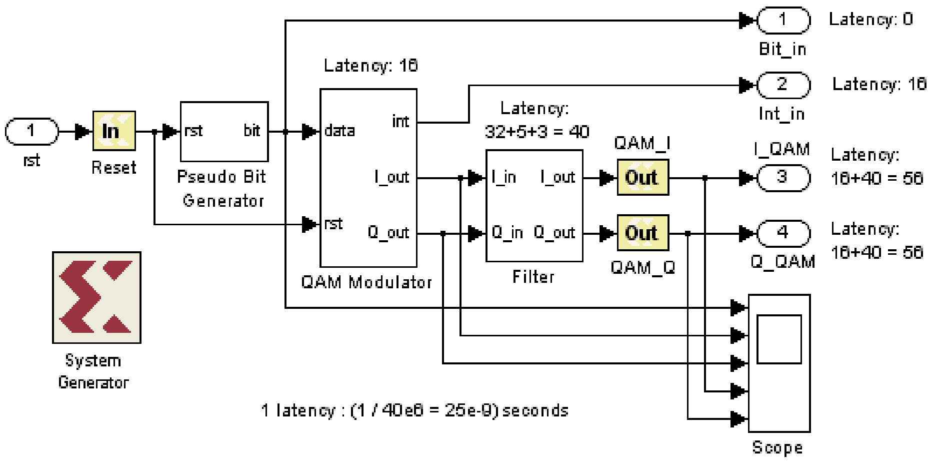


Fig.3. (16-QAM) Transmitter (sub-system) Link Modeling

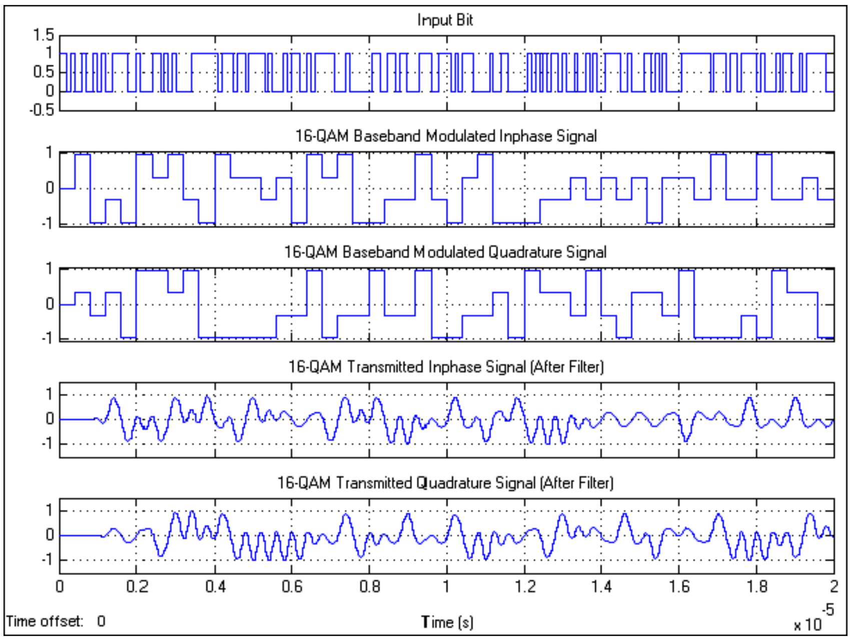


Fig.4.Transmitted Signal: Input bit = 10 Mbps ,(16-QAM Baseband Modulated (IQ) signal = 2.5Msps) , (16-QAM Transmitted (IQ) signal =16*2.5 = 40 Msps (16 bit, 13 fractional length).

2.2- Receiver Link Modeling

The receiver link accepts the IQ data streams from AWGN channel (20-dB SNR) and converts the data into fixed point format. Firstly, noises existing in the received IQ signals are reduced by two identical RRC filters (same design parameters as the RRC filters in transmitter); then the filtered signals are amplified to certain level. The filtered and fine-gained IQ signals are down-sampled to become IQ symbol-pairs before demodulation. In 16-QAM demodulation, decision for the IQ symbol-pairs are made using algorithms in *M-Code* blocks and are mapped back to 4-bit integer, subsequently translated back to bit stream through parallel-to-serial conversion. Additional process of pulse-shaping for the demodulated bit using normalized-gain raised cosine (RC) filter is included for better visualization of DAC output via oscilloscope. The filter design parameters are: order = 16, over-sampling rate = 8, roll-off factor = 0.35, scale passband with Kaiser window, and β = 0.5. The demodulation is by far the most complicated part in receiver because it must detect amplitude and phase of the signal correctly before decision making process. Thus, symbol synchronization is required before the demodulation. Fig. 5 shows the 16-QAM receiver subsystem modeling, whereas Figs. 6 and 7 show its input, intermediate, and output signals.

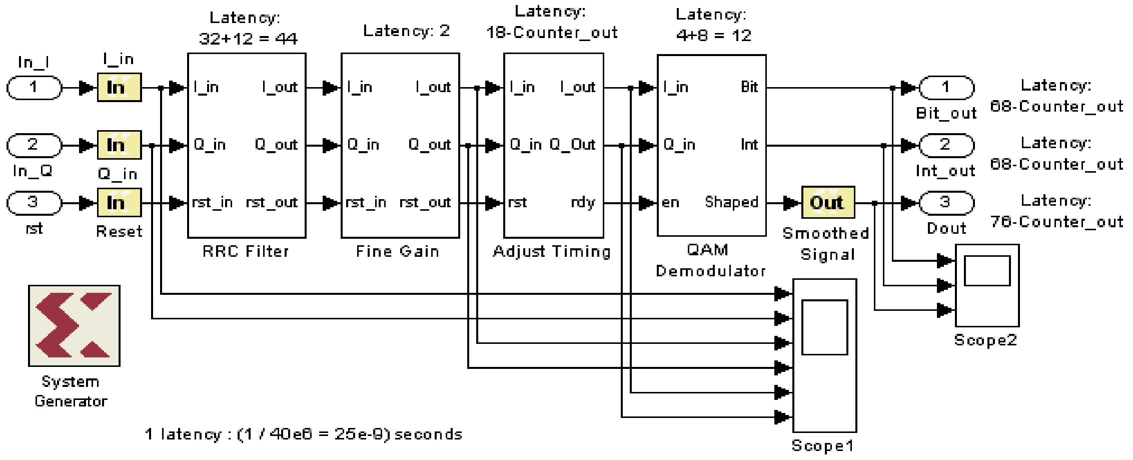


Fig. 5. 16-QAM Receiver (sub-system) Link Modeling.

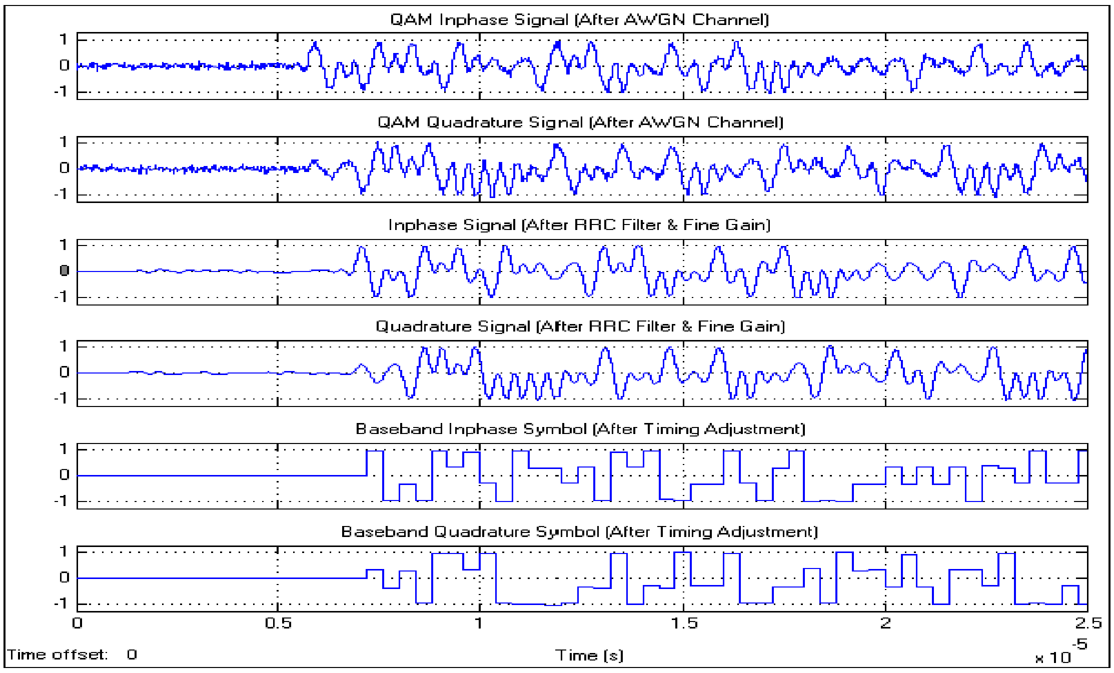


Fig.6. Receiver Signal: Received signal (After Channel): 40 Msps (14 bit, 9 fractional length) , Received signal (before Adjust Timing subsystem):40 Msps (16 bit, 14 fractional length), Timing-adjusted signal (After Adjust Timing subsystem): 40/16 = 2.5 Msps after Down sampling.

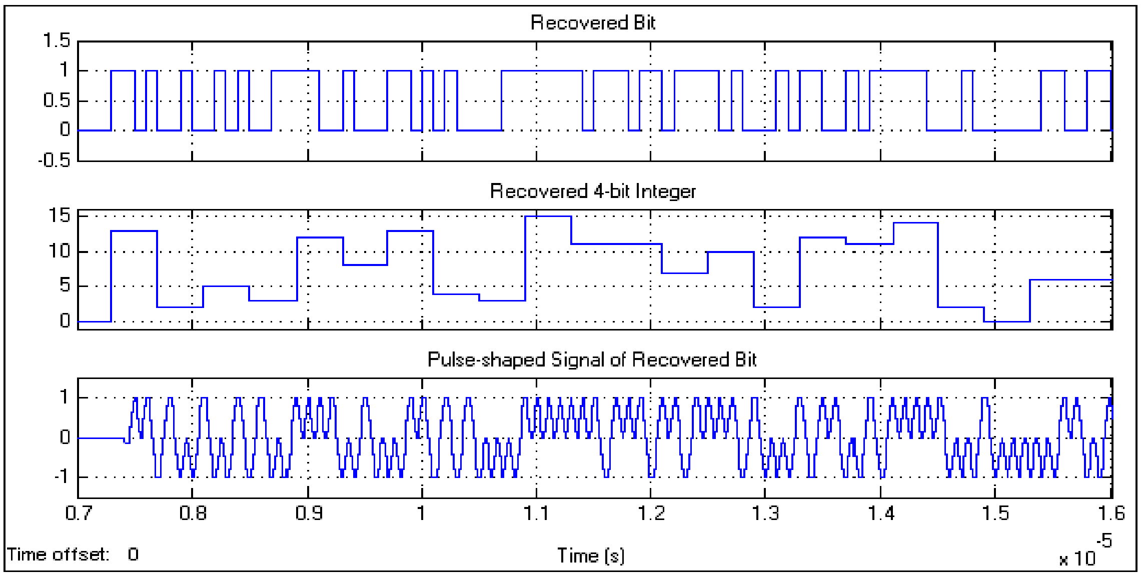


Fig.7. Receiver Output Signal :Recovered 4-bit Integer =5 Msps , Recovered bit: 2.5*4 = 10 Mbps, Pulse shaped signal of Recovered bit: 10*8 = 80 Msps (16 bit, 14 fractional length).

2.3- Timing Recovery

In any QAM scheme timing recovery (synchronization) poses a significant problem for the recovery of a quality signal. A wide variety of methods are available for timing recovery, however, their usefulness actually depends on the type of transmitted data.

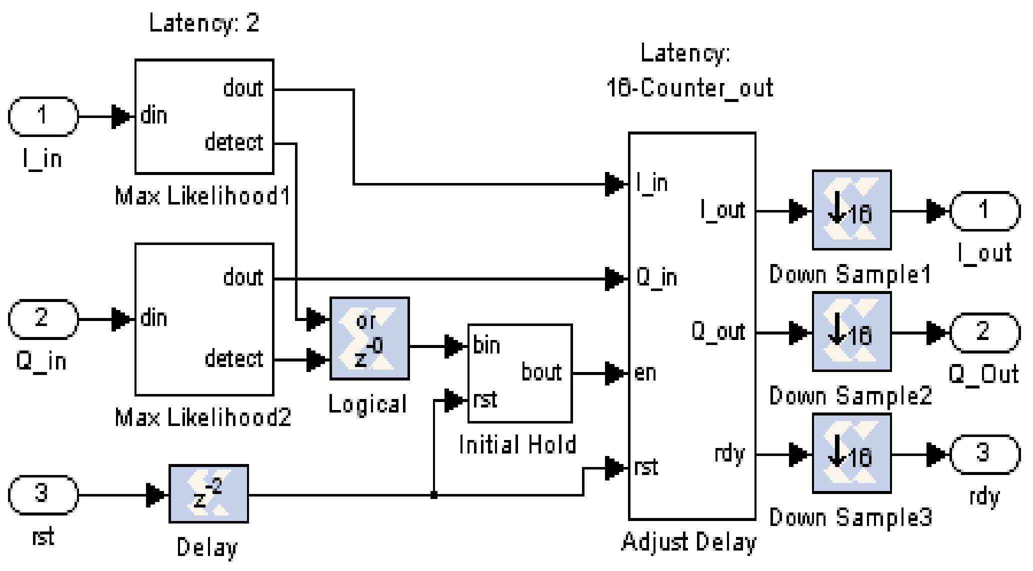


Fig.8. Timing Recovery (Adjust Timing subsystem in QAM Rx7)

The simulation software will be using a pseudo-noise (PN) random bit sequence so none of the data-aided (DA) based schemes would have an advantage. In this case the best choice would be a synchronizer timing recovery circuit. To reduce complexity, a simple timing recovery system was implemented. After a significant amount of self-tuning, the timing recovery checks the maximum likelihood of filtered signal to obtain optimum sampling time [6]. Maximum likelihood timing recovery is the best recovery scheme for this 16-QAM receiver design as shown in Fig. 8.

3- Xilinx FPGA Design Flow and Software Tools

The basic flow for designing with most every FPGA vendor is illustrated in Fig. 9. This flow also illustrates the software tools used specifically for the Xilinx FPGA design. Xilinx owns and maintains a complete tool set for the entire FPGA design flow, some of which is in collaboration with individual companies. Essentially, all of its tools are integrated under one umbrella called the Integrated Software Environment (ISE) package. Simulation and testing of the SDR transmitter and receiver design were done using System Generator, a system level modeling tool from Xilinx [3]. This tool can be used for designing and testing DSP systems for FPGAs in a visual data flow environments such as MATLAB Simulink. This diagram shows that we can use Xilinx System Generator's blocks in the design and generate a synthesizable design which can be implemented using Xilinx ISE's Project Navigator [7]. It also uses ModelSim block which is a helper block to invoke ModelSim simulator and actually simulate the design. The simulator's output is fed back to Simulink for verification and the results can be displayed using Simulink's sinks. The techniques have been incorporated in the HDL Simulation and ModelSim behavioral synthesis tool that reads in high-level descriptions of DSP applications written in MATLAB, and automatically generates synthesizable RTL models in VHDL or Verilog. Experimental results are reported and mapped onto the Xilinx Virtex-4 FPGAs.

4- FPGA Implementation.

The synthesis output files that are required in Xilinx ISE

The input file types to a synthesizer are either .V (Verilog) or .VHD (VHDL), with the output file type of Synplify being an EDIF (Electronic Data Interchange Format) file, and the output file of XST being an NGD (Native Generic Database) file. Since the netlist has not been mapped into Xilinx specific building blocks at this stage, synthesis tools cannot give accurate timing results in its timing and area log files, only estimation. The output of the Synthesis tool is then fed into the next stage of the design flow, which is called Implementation in the Xilinx flow, and is the core utility of the ISE software suite. Before this step is executed, the user constraints file (UCF) is typically filled out. The most critical information in the constraints file is the pin locations for each I/O specified in the HDL design file, and the timing information, such as the system clock frequency [8].

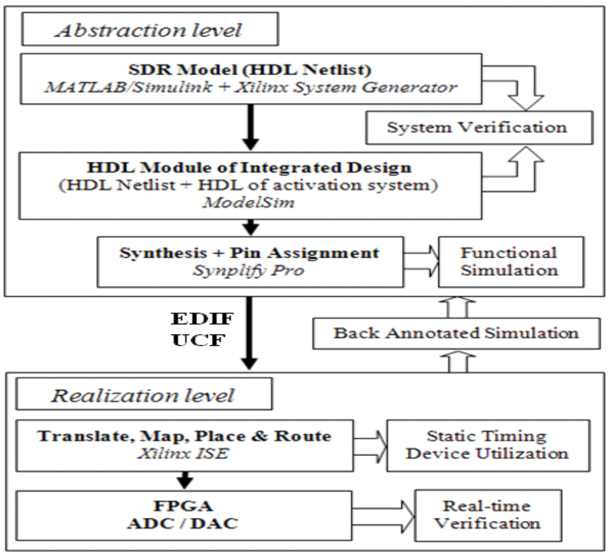


Fig.9. System Generator based design flow

software are in EDIF (Electronic Design Interface File) and UCF (User Constraints File) formats, which represent optimized netlist of integrated design, and timing constraints and FPGA pin assignment respectively. To implement the synthesized design into Virtex-4 FPGA development board, Xilinx ISE performs steps as described in Fig. 9 under Section 3, i.e. Translate, Map, Place & Route (PAR), Bit Generation, and Program Download to FPGA for the SDR model. Make sure no errors for each step described above. The timing requirement is satisfied as shown in the post-PAR (final) static timing report in Tables 1 and 2.

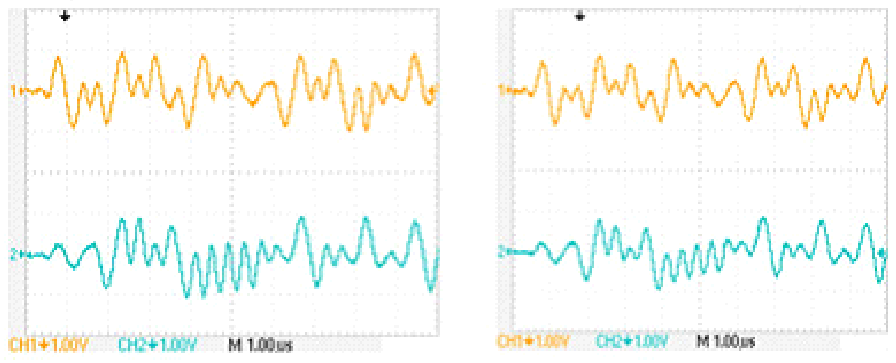
Due to no error and warning for DRC (Design Rules Checker) and bit-stream generation process, the configuration bit-stream file is downloaded to Virtex-4 FPGA board. The ADC input and DAC output signals in P240 Analog Module for the SDR transmitter and receiver are connected to oscilloscope in order to display real-time result as shown in Figs. 10. Both the transmitted and received signal should be similar, although noise effect and distortion may occur (real-world application issue). Notice that the empirical (real-time) result is similar to the simulated result.

Table 1.Post-PAR Static Timing Report for 16-QAM Transmitter

```
------------------------------------------------------------------------------------------
  Constraint                                | Check    | Worst Case |  Best Case | Timing |
                                            |          |    Slack   | Achievable | Errors |
------------------------------------------------------------------------------------------
  TS_CLK_100 = PERIOD TIMEGRP "CLK_100" 10  | SETUP    |     4.412ns|     5.588ns|       0|
  ns HIGH 50%                               | HOLD     |     0.542ns|            |       0|
------------------------------------------------------------------------------------------
  TS_LIO_CLKIN_1 = PERIOD TIMEGRP "LIO_CLKI | SETUP    |     5.607ns|     6.893ns|       0|
  N_1" 12.5 ns HIGH 50%                     | HOLD     |     0.257ns|            |       0|
------------------------------------------------------------------------------------------
```

Table 2.Post-PAR Static Timing Report for 16-QAM Receiver

```
------------------------------------------------------------------------------------------
  Constraint                                | Check    | Worst Case |  Best Case | Timing |
                                            |          |    Slack   | Achievable | Errors |
------------------------------------------------------------------------------------------
  TS_LIO_CLKIN_1 = PERIOD TIMEGRP "LIO_CLKI | SETUP    |     1.465ns|    11.035ns|       0|
  N_1" 12.5 ns HIGH 50%                     | HOLD     |     0.358ns|            |       0|
------------------------------------------------------------------------------------------
  TS_CLK_100 = PERIOD TIMEGRP "CLK_100" 10  | SETUP    |     5.021ns|     4.979ns|       0|
  ns HIGH 50%                               | HOLD     |     0.512ns|            |       0|
------------------------------------------------------------------------------------------
```



(a)Transmitted (IQ) (b) Received (IQ)

Fig.10. Real-time Transmitted/Received (IQ) 16-QAM Signals

The paper proposes larger algorithms which should provide better performance. It also demonstrates means of using this design as the basis of 16-QAM transceivers. The transceiver is capable of communicating (10-20) M-baud of data at the provided SNR vs. BER ratio. Finally, the selected algorithms provide an adequate means of solving for frequency, phase, and bit-time errors. Algorithms were chosen such that errors are reduced while maintaining resource efficiency. The resource consumption by the receiver and transmitter is shown in Table 3.

Table 3: Resource consumption

Total	Receiver	Transmitter	Resource
10,442	10,000 33%	442 1%	Slices
7,3224	7,229 23%	295 1%	LUTs
8	-	8	No. of RAMs
208 MHz	208 MHz	145 MHz	Maximum frequency

5- Conclusion

This paper has presented design of SDR transmitter and receiver model in both the abstraction and realization levels. The equivalence of the real-time (empirical) and simulated results for transmitted signal and receiver output in time domain proves the bit and cycle accurate of FPGA processing which is important in high speed DSP for further processing of pre-coding, decoding and equalization in more advanced SDR model for wireless communication system. FPGAs offer mass parallelism for implementing DSP algorithms, specifically for implementing MAC functions. In any application that requires real time processing, such as electronic warfare applications, parallel MAC implementations are needed to speed up the hardware. The Xilinx Virtex-4 FPGA architecture provides many system resources for automatic or implicit implementation of DSP MAC functions. Today DSP and FPGA designers have an increasing need to begin and end their design flow in MATLAB. The current approach of manually converting software to HDL, implementing it on FPGA, and then attempting to compare the hardware results to the software results, must cross many error prone boundaries. In order to assure equivalency between a DSP algorithm designed in MATLAB and the one implemented in hardware, it is useful to have a high level, single flow, software tool. The Xilinx System Generator for DSP, which is essentially hardware added on to Simulink in MATLAB, offers a unique solution. System Generator allows for DSP algorithm design and hardware implementation to be executed in essentially the same step.

REFERENCES

[1] A. Pal, "FPGA-based (Xilinx) Embedded System Design," *Workshop on Microcontroller and FPGA-based Embedded System Design*, July 2007.
[2] The MathWorks, Inc.. DSP – Digital Signal Processing & Communications. , 2009, [Online]. Available: http://www.mathworks.com/applications/dsp_comm
[3] Xilinx, Inc., *DSP Design Flows in FPGA Tutorial Slides*, 2003.
[4] J. G. Prokis, *Digital Communications*, 4th ed., New York: McGraw-Hill,2001
[5] D. Haessig, J. Hwang, S. Gallagher, and M. Uhm, "Case-Study of a Xilinx System Generator Design Flow for Rapid Development of SDR Waveforms," in *Proc. SDR 05 Forum Tech. Conf. and Product Exposition*, Orange County, California, 2005.
[6] C. Dick, F. Harris, and M. Rice, "Synchronization in Software Radios Carrier and Timing Recovery Using FPGAs," , pp. 195- 204. IEEE, 2000.
[7] *System Generator for DSP User Guide*, Release 9.2.01, Xilinx, Inc., 2007.
[8] *Xilinx ISE 9.2i Software Manuals: Constraints Guide, and Development System Reference Guide*, Xilinx, Inc., 2007.
[9] *P240 Analog Module User Guide*, Rev 1.0, Avnet, Inc., May 2006. [Online]. Available: http://www.files.em.avnet.com/files/177/p240_analog-ug.pdf
[10] *Virtex-4 MB Development Board User's Guide,* Ver. 3.0, Avnet Memec, Dec. 2005. [Online]. Available: http://www.files.em.avnet.com/files/177/v4mb_user_guide_3_0.pdf
[11] "ADS5500: 14-bit, 125 Msps, Analog-to-Digital Converter Data Sheet," Texas Instrument, Inc., Feb. 2007. [Online]. Available: http://focus.ti.com/lit/ds/symlink/ads5500.pdf
[12] "DAC5687: 16-bit, 500 Msps, 2x-8x, Interpolation Dual-Channel Digital-to-Analog Converter (DAC) Data Sheet," Texas Instrument, Inc., June 2005. [Online]. Available: http://focus.ti.com/lit/ds/symlink/dac5687.pdf
[13]*Synplicity FPGA Synthesis Reference Manual*, Synplicity, Inc., 2007.