ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

Optimizing Digital Filter for Effective Signal Processing

Okhaifoh Joseph, James Agajo, Idigo V.E
jokhaifoh@yahoo.com, agajojul@yahoo.com, vicugoo@yahoo.com

Abstract - Implementing hardware design in Field Programmable Gate Arrays (FPGAs) is a formidable task. There is more than one way to implement the digital FIR filter. Based on the design specification, careful choice of implementation method and tools can save a lot of time and work. MatLab is an excellent tool to design filters. There are toolboxes available to generate VHDL descriptions of the filters which reduce dramatically the time required to generate a solution. Time can be spent evaluating different implementation alternatives. Proper choice of the computation algorithms can help the FPGA architecture to make it efficient in terms of speed and/or area.

I. BACKGROUND

A. Why Implement The Digital Filter?

Digital filters are used extensively in all areas of electronic industry. This is because Digital filters have the potential to attain much better signal to noise ratios than analog filters and at each intermediate stage the analog filter adds more noise to the signal, the digital filter performs noiseless mathematical operations at each intermediate step in the transform, as the digital filters have emerged as a strong option for removing noise, shaping spectrum, and minimizing inter-symbol interference in communication architectures. These filters have become popular because their precise reproducibility allows design engineers to achieve performance levels that are difficult to obtain with analog filters. [1]

FIR and IIR filters are the two common filter forms. A drawback of IIR filters is that the closed-form IIR designs are preliminary limited to low pass, band pass, and high pass filters, etc. Furthermore, these designs generally disregard the phase response of the filter. For example, with a relatively simple computational procedure we may obtain excellent amplitude response characteristics with an elliptic low pass filter while the phase response will be nonlinear. In designing filters and other signal-processing system that pass some portion of the frequency band undistorted, it is desirable to have approximately constant frequency response magnitude and zero phases in that band. For casual systems, zero phases are not attainable, and consequently, some phase distortion must be allowed. As the Effect of linear phase with integer slope is a simple time shift. A nonlinear phase, on the other hand, can have a major [2] effect on the shape of a signal, even when the frequency-response magnitude is constant. Thus, in many situations it is particularly desirable to design systems to have exactly or approximately linear phase.

Compare to IIR filers, FIR filters can have precise linear phase. Also, in the case of FIR filters, closed-form design equations do not exist. While the window Method can be applied in a straightforward manner, some iteration may be necessary to meet a prescribed specification. the window method and most algorithmic methods afford the possibility of approximating more arbitrary frequency response Characteristics with little more difficulty than is encountered in the design of low pass filters. Also, it appears that the design problem for FIR filters is much more under control than the IIR design problem because there is an optimality theorem for FIR filters that is meaningful in a wide range of practical situations. [3]

The magnitude and phase plots provide an estimate of how the filter will perform; however, to determine the true response, the filter must be simulated in a system model using either calculated or recorded input data. The creation and analysis of representative data can be a complex task. Most of the filter algorithms require multiplication and addition in real-time. The unit carrying out this function is called MAC (multiply accumulate). Depends on how good the MAC is, the better MAC the better performance can be obtained. Once a correct filter response has been determined and a coefficient table has been generated, the second step is to design the hardware architecture. The hardware designer must choose between area, performance, quantization, architecture, and response.[4]

II. DESIGN AND IMPLEMENTATION OF DIGITAL FIR FILTER

MATLAB combines the high-level, mathematical language with an extensive set of pre-defined functions to assist in the creation and analysis of filter data. Toolbox are available for designing filter response and generating coefficient tables, each with varying levels of sophistication. Graphical filter design tools provide selections for specifying pass band, filter order, and design methods, as well as provide plots of the response of the filter to various standard forms of inputs. FDA tool from the Math Works, which can generate a behavioral model and coefficient tables are commonly used in Digital Filter design.

Once a correct filter response has been determined and hardware architecture has been defined, the implementation can be carried out. Three choices of technology exist for the

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

implementation of filter algorithms. These are: Programmable DSP chips, ASICs and FPGAs.

At the heart of the filter algorithm is the multiply-accumulate operation; Programmable DSP chips typically have only one MAC unit that can perform one MAC in less than a clock cycle. DSP processors or programmable DSP chips are flexible, but they might not be fast enough. The reason is that the DSP processor is general purpose and has architecture that constantly requires instructions to be fetched, decoded and executed. ASICs can have multiple dedicated MACs that perform DSP functions in parallel. But, they have high cost for low volume production and the inability to make design modifications after production makes them less attractive. [5]

FPGAs have been praised for their ability to implement filters since the introduction of DSP savvy architectures, which can be efficiently, realized using dedicated DSP resources on these devices. More than 500 dedicated multiply-accumulate blocks are now available, making them exceptionally well suited for high-performance, high-order filtering applications that benefit from a parallel, non-resource shared hardware architecture. In this particular project, FPGA has been chosen as the implementation tool. To program FPGA, hardware description language is needed. VHDL synthesis offers an easy way to target a model towards different implementation fig 1.

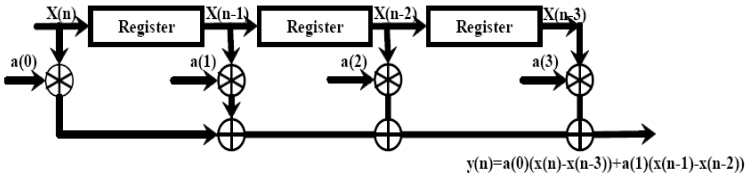


Fig.1 "Unrolled" Direct Form FIR Filter Architecture [2]

III .AIMS AND OBJECTIVE

The objective of the project is to meet the demand for the development of courseware in learning and research purposes in industries, universities and colleges. Digital filters like these can be used to do research in fields like Radio Astronomy, Digital Signal Processing; Software defined radio, Aerospace and Defense Systems, Asic Prototyping, Medical Imaging, computer vision, speech recognition, cryptography, bioinformatics, computer hardware emulation and a growing range of other areas. There are several results has been achieved through the project life which are: examination of design procedures or design cycle from different approaches. There is more than one way to implement the digital FIR filter as in fig 2. The purpose of the project is to identify the different approaches compare and contrast the different methods, so based on the design specification; careful choice of implementation method can save designer a lot of time and work. To investigate relationship among filter order, hardware architecture and FPGA resources usage; to identify the relationship between the performance and filter hardware architecture; Using MatLab as major design tool to design filters then compare and contrast the outcome with outcome from other software package.

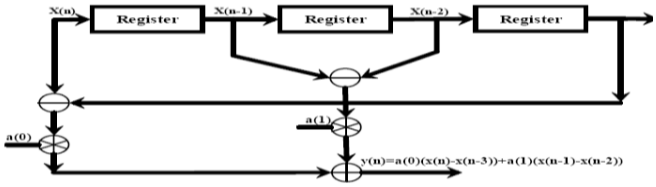


Fig. 2 Direct Form FIR Filter Architecture/Symmetric form

IV. HARDWARE ARCHITECTURE

Deciding on an optimal architecture for a particular application involves tradeoffs between area, performance, and response. The abundance of dedicated MAC (Multiply-Accumulate) blocks in the FPGA devices present the option for implementing either an area efficient "serial" or a high-performance "parallel" FIR filter or something in between. Serial architectures share a single MAC resource, making it very area efficient but at the expense of performance, because one clock-cycle is required for each tap delay register. This architecture is an excellent choice for area sensitive, low-performance applications. (For any details please refer to the chapter 7)

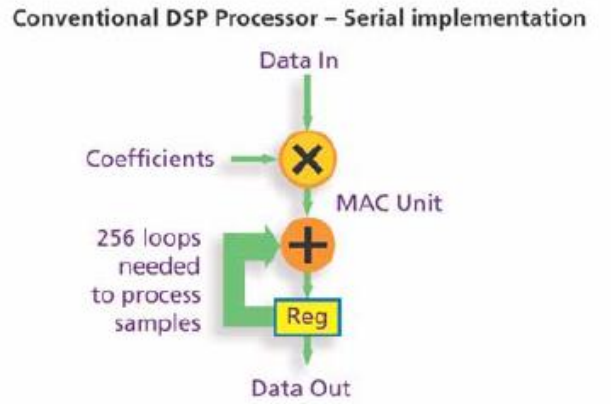


Fig 3 FPGAs Parallel Approach to DSP Enables Higher Computational Through (a)

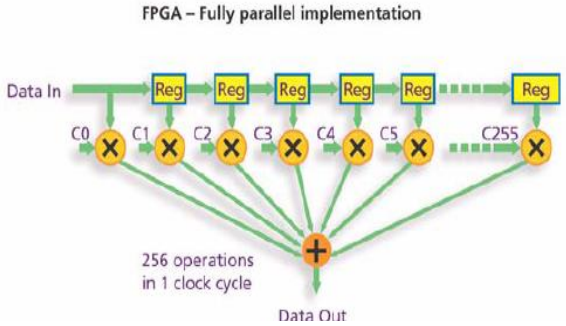


Fig 4 FPGAs Parallel Approach to DSP Enables Higher Computational Through (b) [6]

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

Although commonly implemented on FPGAs, a serial MAC FIR filter does not exploit the large number of dedicated hardware MAC units available on the newer devices. High-performance, high-order filtering applications, that are able to exploit dedicated multiplier or DSP blocks, often turns to FPGAs for a solution. By replicating the multiply-adder logic once for each tap delay register, the entire filtering calculation can be performed in a single clock cycle. With more than 500 DSP blocks available, even very large order filters can sustain input sampling rates over 400 MHz. This type of performance far exceeds even the fastest DSP processors by orders of magnitude. Fig 1, 2, 3 and 5 are the block diagram for an "unrolled" implementation of a direct form FIR filter and its symmetric and anti-symmetric form When targeting an FPGA device with dedicated DSP blocks capable of supporting cascaded "multiply-add" operations such as the Xilinx Vertex 4, highest performance is achieved using a "transposed" architecture. Utilizing the same resources as a "direct" form FIR filter, data samples are applied in parallel to all tap multipliers through pipeline registers. The products are then applied to a cascaded chain of registered adders, combining the effect of accumulators and registers.[8]

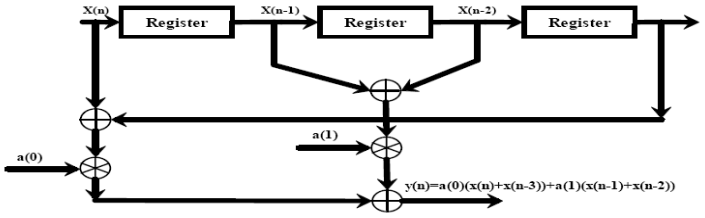


Fig. 5 Direct Form FIR Filter Architecture/Anti-Symmetric form [2]

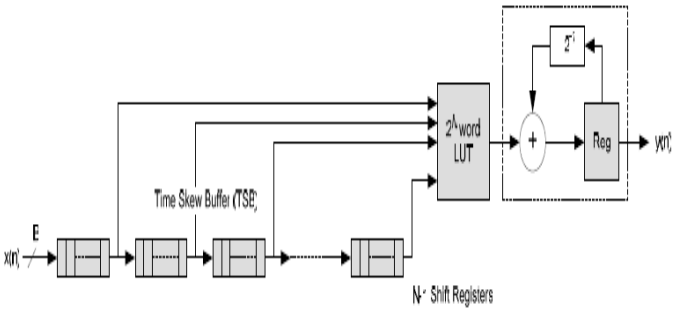


Fig 6.Serial Distributed Arithmetic FIR Filter Architecture [3]

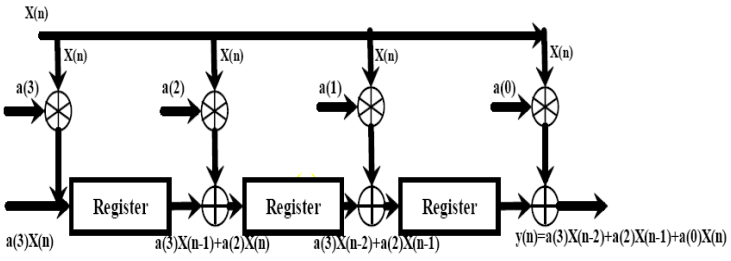


Fig 7 Transposed Direct form FIR Filter Architecture

A fourth FIR filter architecture worth considering for FIR implementation in FPGAs is called "distributed arithmetic" (DA). A simplified view of a DA FIR is shown in Figure 2-8. In its most basic form, DA-based computations are bit-serial in nature – serial distributed arithmetic (SDA) FIR. These computations can be parallelized using multiple sets of Look-up table to achieve concurrency. This architecture is referred to as parallel distributed arithmetic (PDA).

The advantage of the distributed arithmetic approach is its efficiency. The basic operations required are a sequence of table look-ups, additions, subtractions, and shifts of the input data sequence. These functions map efficiently into LUT-based FPGA, such as Xilinx Virtex II, Virtex 4™ and Altera Stratix™ and Stratix II. Distributed arithmetic architectures are an excellent choice for high-performance filter applications targeting FPGA devices that do not include dedicated multipliers or DSP48 blocks or when these resources are not available. A unique characteristic of the DA architecture in Fig IV is that the sample rate is decoupled from the filter length, making the DA architecture appealing for high-order filters. The trade off introduced [7] here is one of silicon area (FPGA slices) for performance. As the filter length is increased in a DA FIR filter, more logic resources are consumed, but throughput and performance are maintained. The table below summarizes the benefits of each architecture.

Although this discussion on filter hardware architectures has been limited to constant coefficient, single-rate, single-channel FIR filters, the concepts will apply for programmable coefficient, multi-rate, and multi-channel variations. The adaptive filters referenced in the overview, however, have distinct architectural characteristics that depart significantly from FIR filters.

FIR FILTER IMPLEMENT OPTIONS There is more than one way to reach the solution, proper choice of the implementation tools and techniques can save the designer a lot of work and time.

V. HDL CODER

HDL Coder is integrated with the graphical user interface and command line of the Filter Design Toolbox to provide a unified design and implementation environment. Filter Design and Analysis Tool (FDATool) or the MATLAB command line can be used to design a filter and generate VHDL or Verilog code. The design specification input to the Filter Design HDL Coder is quantized filters that can be create in one of two ways: Design and quantize the filter with the Filter Design Toolbox, Design the filter with the Signal Processing Toolbox and then quantize it with the Filter Design Toolbox. The Filter Design HDL Coder supports several important filter structures, including: Direct Form Finite Impulse Response (FIR), Symmetric FIR, Anti-symmetric FIR Transposed FIR, Direct Form I SOS, infinite Impulse Response (IIR), Direct Form I Transposed

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

SOS IIR, Direct Form II SOS IIR, and Direct Form II Transposed SOS II Each of these IIR and FIR structures supports fixed-point and floating-point (double precision) realizations. In addition, the FIR structures also support unsigned fixed-point coefficients. [9]

Table I AccelChip Synthesis and Effect

AccelChip Synthesis Directive	Effect on Results
Rolling/unrolling of For loops	Improves sustainable data throughput rate by reducing the number of cycles for processing per input sample
Expansion of vector and matrix additions and multiplications	Improves sustainable data throughput by reducing the number of cycles for processing per input sample
RAM/ROM memory mapping of 1D and 2D arrays	Improves FPGA utilization by mapping 1D and 2D arrays into dedicated FPGA RAM resources
insertion	Improves sustainable data throughput rate by improving clock frequency performance
Shift register mapping	Improves FPGA utilization by mapping data buffers to shift register logic

VI. GENERATORS

When implementing the actual filter on an FPGA, the designer is presented with a fundamental choice of whether to use an IP core or design a custom implementation. Both options have merits and limitations. FIR filter IP cores are readily available from multiple sources with the most common forms being technology-specific enlists, synthesizable RTL, and synthesizable MATLAB. All these generators can construct a filter using a pre-defined coefficient table.

Xilinx and Altera both offer high-performance but device-specific filter IP in the form of technology mapped enlists. This form of IP is typically the most cost effective and offers the best results but provides the fewest options. Often each core is hand-crafted to leverage device specific hardware resources, such as Xilinx Block RAM, and includes placement information which helps maintain performance as filter size grows. The format of the IP, however, precludes user modification of the source or retargeting to a different device. The Math Works offers device independent filter IP; however, they don't provide the ability to exploit the high-performing resources of the FPGA. This form of IP tends to be a bit more expensive but offers the user the additional options of retarget ability and modification of the RTL source. The quality of the results will vary with the quality of the RTL model and the RTL synthesis tool used for implementation. In general, however, some area and performance is sacrificed to obtain general purpose, retarget able RTL models. AccelChip offers both high-performance and device-independent IP available in AccelWare® IP Toolkits. AccelWare IP cores are provided as synthesizable MATLAB models that offer several advantages. First, the abstraction level of the IP allows for the greatest range of options. Describing a filter in MATLAB is much easier than creating a technology-specific placed netlist or RTL.

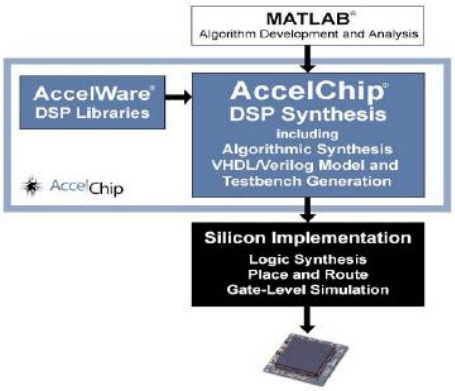


Fig. 8 True-**down DSP Design Flow**

This allows AccelChip to build a wider range of hardware architecture options into the model in fig VI, providing the user the greatest number of choices. Secondly, the architecture of the final hardware can be further modified during the DSP synthesis process with AccelChip through the use of "directives." This provides an efficient means of leveraging the architectural features of the specific FPGA device, such as dedicated DSP blocks, RAMs, ROMs, and shift registers, to achieve high quality of results in the target device. The table below summarizes the synthesis directives available in AccelChip.

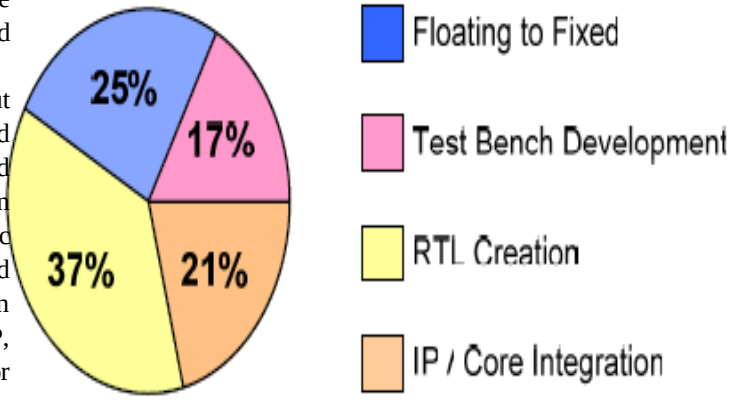


Fig.9 Bottlenecks in the DSP Design Process

IP generators provide an excellent choice when time-to-market or ease-of-use demands prevail. The general purpose nature of these programmable IP generators, however, seldom yields the optimal hardware for a specific application. For this, a user-defined architecture should be considered.[10]

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

VII. USER DEFINED

FIR filter implementations with aggressive area or performance targets may require the use of a hand-crafted implementation over a pre-defined IP block. Doing so allows the designer to exploit application-specific characteristics, such as unique impulse response, coefficient symmetry, ones in the coefficient table, negative coefficients, coefficient length, or knowledge about the dynamic range of the input data to the filter. Most filter IP generators include some simple coefficient symmetry optimization but do not account for every possible situation.

User-defined implementations have traditionally been achieved using hand-coded RTL. This approach offers the designer a high degree of control over the implementation process but is decoupled from the MATLAB simulation environment and can be time consuming. A typical 32 tap FIR filter Verilog model can range between 300 – 500 lines of hand-created VHDL or Verilog code and will often require the instantiation of FPGA-specific RAMs or DSP blocks to achieve optimal results.

The quantization must be determined for each signal and register in the design, which typically requires a re-write of the original floating-point model in language that supports fixed-point modeling and filter response analysis. Once complete, the final RTL code must be verified back to the original filter model to insure functional correctness. A recent survey conducted by AccelChip found that designers were evenly split between these major DSP design tasks when asked to identify their most significant bottlenecks.

VIII .SOLUTION TO THE CURRENT PRACTICE

Implementing hardware design in Field Programmable Gate Arrays (FPGAs) is a formidable task. There is more than one way to implement the digital FIR filter. Based on the design specification, careful choice of implementation method and tools can save a lot of time and work. MatLab is an excellent tool to design filters. There are toolboxes available to generate VHDL descriptions of the filters which reduce dramatically the time required to generate a solution. Time can be spent evaluating different implementation alternatives. Proper choice o f computation algorithms can improve the FPGA architecture to make it efficient in terms of speed and/or area.

Investigation through the first phase of the project has found that, in the current market there is so many design, implementation tools, each of them has the advantage and disadvantage in some aspects. Conclusion can be draw from the investigation is that to implement the linear phase filter MATLAB or MATCAD is the best choice as there are several toolboxes available which remove the complexity for the design and implementation. To design and implement the Multi-rate FIR filter, Filter visualization tool from MATLAB and Accelware generations software package is the best choice. For the distributed arithmetic hardware architecture the best way to implement it is to use the look-up table technique also distribute arithmetic algorithm is the best choice for implementing FIR filters with Xilinx FPGAs. One of these implementations, namely, Serial Distributed Arithmetic (SDA) exhibits efficiency in terms of area. The other implementation, named Parallel Distributed Arithmetic (PDA) shows a very high sample rate. Linear-phase response FIR filters were also developed and implemented in an FPGA using the DA technique. [11]

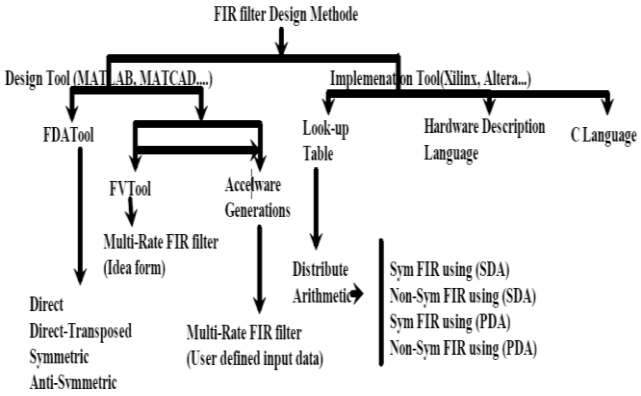


Fig 10 Proper Implement Techniques Choices for the Different Filter Hardware Architecture

IX. DESIGN SPECIFICATIONS

EXPECTED FEATURES OF NEW SYSTEM

The objective of this system is to provide a hardware platform to test FIR filters that have been generated using MATLab. The system performs the following functions:

1. Communicate with a PC using a standard RS-232 serial interface.
2. Receive a data file from the PC. Data will be received in binary form so they can be applied to the FIR filter without further transformation. For this exercise consider that the FIR filter receives 8-bit samples.
3. The outputs of the FIR filter must be returned to the PC where they will be stored in a file. Again assume data from the FIR filter have to be sent the outputs in binary form (without transformation).
4. To send and receive the files from/to the PC a program like "HyperTerminal" can be used. The serial port must be configured with the following settings: Bauds rate is 57600 which is 54 times slower than the 50MHz crystal clock onboard;1 start bit, 1 stop bit ,8 data bits, No parity ,No flow control.[12]

X. OVERVIEW OF THE SYSTEM

In above figure, there are two blocks that do not have to be designed: the RS-232 interface, it is provided as part of the development board, and the FIR filter, it is implemented using MATLab. The blocks that have to be developed are the Universal Asynchronous Receiver-Transmitter (UART), the

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

clock divider. A brief description of each one of these blocks provided next.

UART must receive and transmit data in a serial format. Every frame that is transmitted contains 1 start bit, 8 bits of information and 1 stop bit. The function of the UART can be divided two: the transmission and the reception circuits. In this particular application, the FIR filter receives 8-bit numbers at its input. Hence, it is possible to feed the filter directly from the UART. The receiver must perform the following functions: Detect the start of a frame by sensing the start bit. Sample the 8 bits of information at the frequency corresponding to the baud rate, least significant bit first. Present the 8 bits of information as one byte to the FIR filter.

Generate a signal **DRdy** to latch the new data into the filter. This signal must work as a "write" signal to the filter. The transmitter must perform the following functions: Receive 8-bit data from the FIR filter. Data will be latched in the transmitter using the **WrD** signal generated by the data manager. Transmit the data in serial format at the baud rate, least significant bit first. 1 start bit at the beginning and 1 stop bit at the end of the frame must be inserted Clock divider in this circuit must divide the 50MHz clock available on the board to deliver a frequency related to the baud rate. This new clock has to be used to receive and send data over the serial channel at the correct speed.

FIR filter with its function is to receive 8 bit data from the UART and apply the filtering process for the incoming data. This circuit must provide an interface compatible with the UART. In figure above, this interface is represented by the signal EOF (End of Filtering) that indicates when the filter has a new data available on its output.

XI .DESIGN SPECIFICATION OF FIR FILTERS

Low-pass FIR filter has been implemented with

- Response type: Low pass;
- Design method: Equi ripple;
- Density factor: 20;
- Filter order: order 5;
- Hardware architecture: Direct form;
- Sampling frequency: 48000Hz;
- Pass band frequency: 9600Hz;
- Stop band frequency: 12000Hz;
- Input data length: 8 bits;
- Output data length: 8 bits;

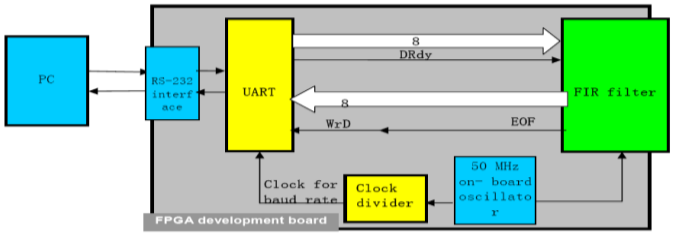


Fig 11 Magnitude response of the 8 bit-input and 8 b

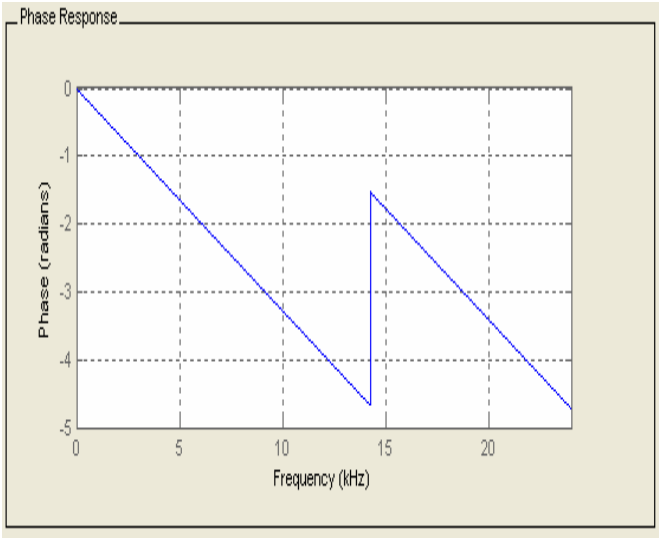


Fig.12 Magnitude response of the 8 bit-input and 8 bit-output FIR filter

XII. OVERVIEW OF UART

Function Diagram of the UART

The UART is for interfacing computers or microprocessors to an asynchronous serial data channel. The receiver converts serial start, data, parity and stop bits. The transmitter converts parallel data into serial form and automatically adds start, parity and stop bits. The data word length can be 5, 6, 7 or 8 bits. Parity may be odd or even. Parity checking and generation can be inhibited. The stop bits may be one or two or one and one-half when transmitting 5-bit code. It can be used in a wide range of applications including modems, printers, peripherals and remote data acquisition systems.

XIII. DESIGN SPECIFICATIONS OF UART

In the UART (Universal Asynchronous Receive and Transmission) all data transmit or received in unit of bit, at the beginning and the end of the data frame there are the start bit and the stop bit respectively.

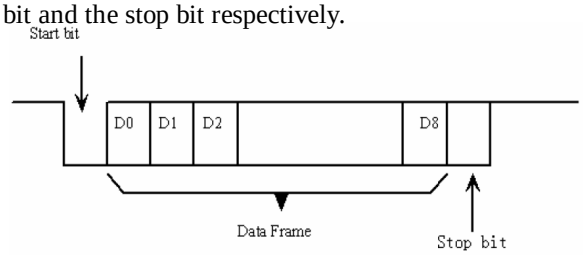


Fig 13 Flow Chart Representation of the System

Start bit is logic 0, it always stays in the front of the data frame in order to remain the receiver to receive the data, and in the process of the receiving the start bit will be removed. According to the TCP/IP, it allows to have the data frame within the range of the 5 to 8. Usually, the data frame is7 or 8 bits, if it required to transmitting ASCII data (Assume all data in binary number), in THIS design the data frame is in the length of the 8 bits. When the data was transmitted start

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

from the LSB and the MSB is at the end of the data frame. Such as the letter C in the ASCII code it can be represented in 67 in decimal number and 01000011 in the binary number, when it started to transmitted it will become 11000010. There is something wrong with the data frame. Top bit is logic 1, it always stays at the end of the data frame, and it can be 1 bit, 1.5bit or 2 bits. In the project, it is 1 bit. And there is no parity bit and framing error bit. Baud Rate is the number of the bits can be received and transmitted within a second. If the time need for transmitting and receiving one bit is t then the baud rate will be 1 over t, the corresponding transmit and receive clock is 1/t Hz. The baud rate for the transmitter and receiver should be consistent otherwise the error can be occurred, in this particular design the baud rate has been assumed as 54 times slower than the clock signal on the board.

XIV. DESIGN MODEL FOR TRANSMITTER

To simplify the design for the project, there is a start bit, 8 bits data in length and a stop bit and there is no parity bit, assume the baud rate is 54 times slower than the clock signal from the board.

XV.DESIGN FOR TRANSMITTER

According to design specification, it needs to send 10bits data (1 start bit, 8 bits data, and 1 stop bit), after the stop bits, the transmitter will stop and the logic level will stay back to logic1, and then it will stay in idle state to wait for the next transmission. The transmitter section accepts parallel data, formats the data and transmits the data in serial form on the serial data output (sdo) terminal. Data is loaded from the inputs din 7 –din0 into the transmitter buffer register by applying logic high on the write input port.

1) Design for Finite State Machine and Pseudo Code for Finite State Machine

The finite state machine is triggered from state S0 to state S1 when there is a rising edge of the clock signal; it stays at state S0 when the reset signal is at the higher level.

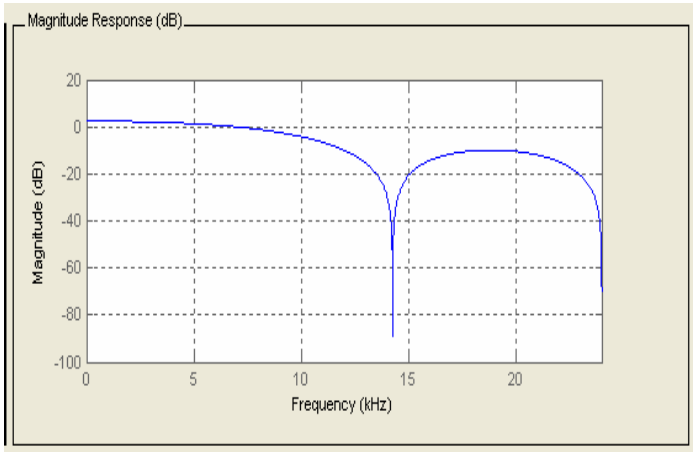


Fig 14 Phase Response of the 8 Bit-Input and 8 Bit-Output FIR Filter

If the signal at the input port reset has been scanned as logic 1 then
The current state is state S0
If there is a rising edge of the clock signal then
The current change to the next state S1
Valid data must be present at least tset prior to and thold following the rising edge of write signal. If words less than 8 bits are used, only the least significant bits are transmitted. The character is right justified, so the least significant bit corresponds to txbr (0). The rising edge of write signal clears transmitter buffer register. 0 to 1 Clock cycles later, data is transferred to the transmitter register, the tx_rd pin goes to a low state, tx_rd is set high and serial data information is transmitted. The output data is clocked at a clock rate 16 times the data rate. A second high level pulse on write loads data into the transmitter buffer register. Data transfer to the transmitter register is delayed until transmission of the current data is complete. Data is automatically transferred to the transmitter register and transmission of that character begins one clock cycle later.

XVI. DESIGN AND PSEUDO CODE FOR 8 BITS COUNTER

The 8 bits counter signals at the predefined state; the 8 bit counter uses this signal as its clock to tap data. It is counting 11 such signals from 8 bit counter.
If the logic level at the reset input port has been scanned as the logic 1 or the current state is in the state S0
Then counter initialize itself to the “00000000”
Elsif there is the rising edge of the clock signal the

2) Counter increment by 1

DESIGN FOR 16 TIMES BAUD RATE CLOCK AND IT’S PSEUDO CODE

To generate baud rate of 16 times faster than the baud rate of 57600 frequency of onboard clock needs to be divided by (16*57600 = 921600) which is 54.2534. After rounding it becomes integer number 54. If the signal 1 has been scanned at the input port en if there is rising edge of the 50MHz clock Then count incremented by 1 .

XVII. IMPLEMENTATION ON BOARD

OVERVIEW The Virtex-4™ FX12 LC Development kit provides a complete development platform for designing and verifying applications based on the Xilinx Virtex-4 FPGA family. This kit enables designers to implement DSP and embedded processor based applications with extreme flexibility using IP cores and customized modules. The Virtex-4 FPGA along with its integrated PowerPC processor core makes it possible to prototype processor based applications, enabling software designer early access to a hardware platform prior to working with the final product/target board. The Virtex-4 FX12 LC system board utilizes the Xilinx XC4VFX12-10FF668C FPGA. The board

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

includes 64MB of DDR SDRAM, 4MB of Flash, USB-RS232 Bridge, a 10/100/1000 Ethernet PHY, 100 MHz clock source, RS-232 port, and additional user support circuitry to develop a complete system. The board also supports the P160 expansion module standard, allowing application specific expansion modules to be easily added.

XVIII. DDR SDRAM, FLASH

The Virtex-4™ FX12 LC development board provides 64MB of DDR SDRAM memory (x16). A high-level block diagram of the DDR SDRAM interface is shown below followed by a table describing the SDRAM memory interface signals. The Virtex-4™ FX12 LC development board provides 4MB of flash memory (x16). A high-level block diagram of the flash interface is shown below followed by a table describing the flash memory interface signals.

XIX. CLOCK SOURCES

The Clock Generation section of the Virtex-4 FX12 LC board provides all the necessary clocks for the PowerPC processor, the I/O devices located on the board, as well as the DDR SDRAM memory. An on-board 100MHz oscillator provides the system clock input to the processor section. This 100 MHz clock will be used by the Virtex-4 Digital Clock .Fig 15 16 and 17 are shown in Appendix.

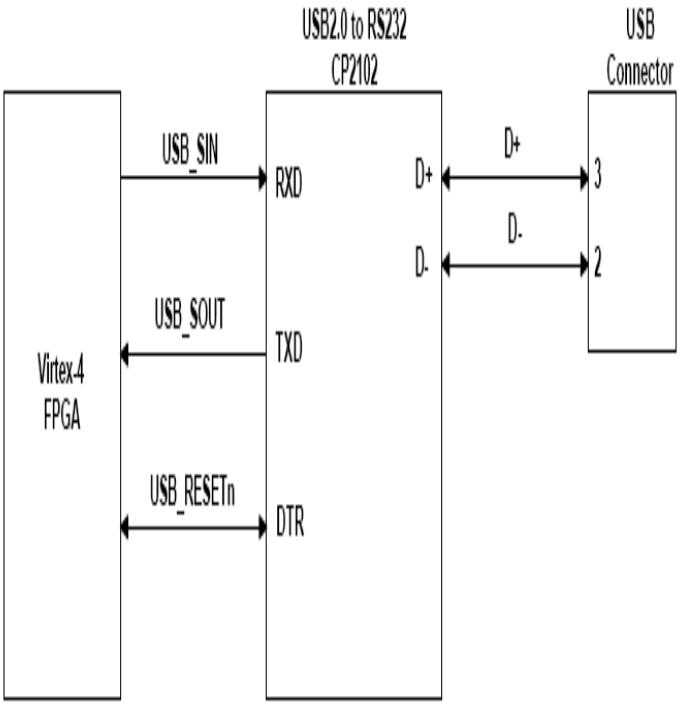


Fig. 18 USB2.0 to RS-232 port

Managers (DCMs) to generate various processor clocks. In addition to the above clock inputs, a socket is provided on the board that can be used to provide single ended LVTTL clock input to the FPGA via an 8 or 4-pin oscillator.

XX. 10/100/1000 ETHERNET PHY, LCD PANEL

The Virtex-4 FX12 LC development board provides a 10/100/1000 Ethernet port for network connection. This interface uses the Virtex-4 embedded 10/100/1000 MAC. A high-level block diagram of the 10/100/1000 Ethernet interface is shown in the following figure followed by FPGA pin assignments for this interface. The Virtex-4 FX12 LC development board provides an 8-bit interface to a 2x16 LCD panel (MYTECH MOC-16216B-B). The following table shows the LCD interface signals.

USB 2.0 TO RS232 PORT

The Virtex-4 FX12 LC development board implements a USB 2.0 port. This is accomplished using the Cygnal CP2101 USB-to-UART Bridge Controller. The FPGA interfaces to the CP2102 as a simple UART. The UART interface to the CP2102 can run at speeds ranging from 300 to 921,600 baud. The CP2102 is a highly integrated USB-to-UART Bridge Controller, providing a simple solution for USB serial communications using a minimum of components and PCB space. The CP2102 includes a USB 2.0 full-speed function controller, USB transceiver, oscillator, EEPROM, and asynchronous serial data bus (UART) with full modem control signals in a compact 5mm X 5mm MLP-28 package. No other external USB components are required. The on-chip EEPROM may be used to customize the USB Vendor ID, Product ID, Product Description String, Power Descriptor, Device Release Number, and Device Serial Number as desired. The EEPROM is programmed on-board via the USB allowing the programming step to be easily integrated into the product manufacturing and testing process. Royalty-free Virtual COM Port (VCP) device drivers provided by Cygnal allow the Virtex-4 FX12 LC development board to appear as a COM port to PC applications. The CP2102 UART interface implements all RS232 signals, including control and handshaking signals. These signals are interfaced to the Virtex-4 FPGA as follows:

XXI. RS232, CONFIGURATION AND DEBUG PORTS

The Virtex-4 FX12 LC development board provides an RS232 interface with RX and TX signals and jumpers for connecting the RTS and CTS signals. The following figure shows the RS232 interface to the Virtex-4 FX12 FPGA. Various methods of configuration and debug support are provided on the Virtex-4 FX12 development board to assist designers during the testing and debugging of their applications. The following sections provide brief descriptions of each of these interfaces.

XXII. CONFIGURATION SETUP

The configuration for implementation on the board has been set up as the following. For the 50MHz clock onboard Pin 184 has been assigned For the USB switch P16 has been assigned. For the reset button Pin 22 has been assigned. For

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

the serial data input into the receiver Pin11 has been assigned. For the serial data output from the transmitter Pin 3 has been assigned. (For any details please refer to the chapter 7 for the reference reading)

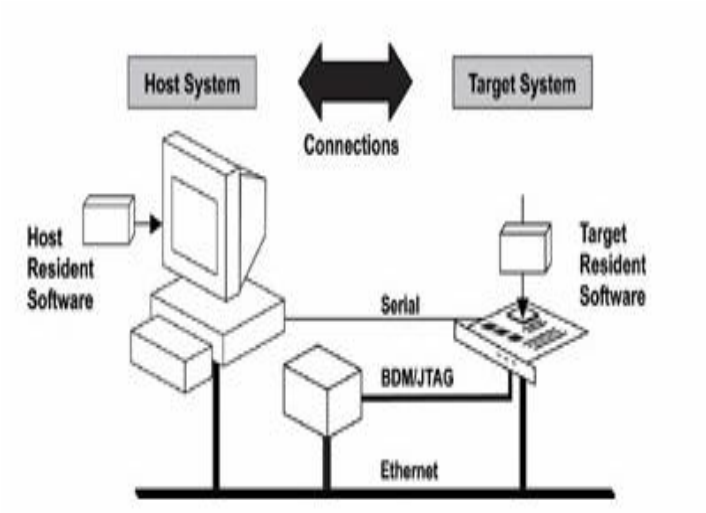


Fig. 19 Connection between the Host System and the Target System

4.4 Connection with PC

Design and implementation needs to be done at the host computer system before it downloads the configuration file into the FPGA. Connection needs to be made between the host computer system and target system by using connection cable. A terminal program called hype terminal is an application that will enable a PC to communicate directly with a modem. This can be useful to test the overall system design and diagnose problems. For this particular design the rate for the HyperTerminal has been setup as 57600 baud rate. (For any details please refer to the chapter 7 for the reference reading) From the observation simulation result the conclusions can be drawn is that for the same filter order, direct hardware architecture takes larger resources than the transposed architecture. Anti-Symmetric hardware architecture takes larger resources than the symmetric hardware architecture as symmetric hardware architecture shares some component in common so the result of the simulation is reasonable.

XXIII. RESOURCES USAGE FOR DIRECT FIR FILTER

Observation from the graph has found that for direct form FIR filter architecture as the filter order increases the amount of the resources usage is increasing correspond.

XXIV. RESOURCES USAGE FOR TRANSPOSED FORM FIR FILTER

Observation from the graph has found that for transposed form of FIR filter architecture as the filter order increases the amount of the resources usage is increasing correspond.

XXV. RESOURCES USAGE FOR SYMMETRIC FIR FILTER 0 50

Observation from the graph has found that for Symmetric form FIR filter architecture as the filter order increases the amount of the resources usage is increasing correspond.

XXVI. RESOURCES USAGE FORANTI-SYMMETRIC FIR FILTER

Observation from the graph has found that for Anti-symmetric form FIR filter architecture as the filter order increases the amount of the resources usage is increasing correspond.

XXVII. SIMULATION OF THE BEHAVIOR MODEL

The simulation result for the UART transmitter is correct and the behavior model has met the design requirement as the result shown that the most significant bit received from the parallel port of the 8 bits register now is becoming the least significant bit and the least significant bit received form the parallel port of the 8 bits register now is becoming the most significant bit and so on. There is no parity check bit and frame error bit.

XXVIII. SIMULATION RESULTS

Device utilization summary As one of the objectives of the project is to investigate the resources usage inside of FPGA, simulation.

Table II Direct FIR Order 5

Selected Device : 3s400pq208-4	
Number of Slices:	**143 out of 3584 3%**
Number of Slice Flip Flops:	**131 out of 7168 1%**
Number of 4 input LUTs:	**172 out of 7168 2%**
Number of bonded IOBs:	**54 out of 141 38%**
Number of MULT18X18s:	**6 out of 16 37%**
Number of GCLKs:	**1 out of 8 12%**

Table III Transposed FIR Order 10

Selected Device : 3s400pq208-4	
Number of Slices:	213 out of 3584 5%
Number of Slice Flip Flops:	398 out of 7168 5%
Number of 4 input LUTs:	213 out of 7168 2%

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

Number of bonded IOBs:	55 out of 141 39%
Number of MULT18X18s:	4 out of 16 25%
Number of GCLKs:	1 out of 8 12%

Table IV Transposed FIR Order 5

Selected Device : 3s400pq208-4	
Number of Slices:	117 out of 3584 3%
Number of Slice Flip Flops:	221 out of 7168 3%
Number of 4 input LUTs:	173 out of 7168 2%
Number of bonded IOBs:	54 out of 141 38%
Number of MULT18X18s:	3 out of 16 18%
Number of GCLKs:	1 out of 8 12%

Table V Anti-Symmetric FIR Order 10

Selected Device : 3s400pq208-4	
Number of Slices:	188 out of 3584 5%
Number of Slice Flip Flops:	212 out of 7168 2%
Number of 4 input LUTs:	174 out of 7168 2%
Number of bonded IOBs:	55 out of 141 39%
Number of MULT18X18s:	6 out of 16 37%
Number of GCLKs:	1 out of 8 12%

Table VI Symmetric FIR Order 5

Selected Device : 3s400pq208-4	
Number of Slices:	117 out of 3584 3%
Number of Slice Flip Flops:	131 out of 7168 1%
Number of 4 input LUTs:	120 out of 7168 1%
Number of bonded IOBs:	55 out of 141 39%
Number of MULT18X18s:	3 out of 16 18%
Number of GCLKs:	1 out of 8 12%

Table VII. Direct FIR Order 15

Selected Device : 3s400pq208-4	
Number of Slices:	413 out of 3584 11%
Number of Slice Flip Flops:	292 out of 7168 4%
Number of 4 input LUTs:	534 out of 7168 7%
Number of bonded IOBs:	55 out of 141 39%
Number of MULT18X18s:	16 out of 16 100%
Number of GCLKs:	1 out of 8 12%

Table VIII Direct FIR order 10

Selected Device : 3s400pq208-4	
Number of Slices:	206 out of 3584 5%
Number of Slice Flip Flops:	212 out of 7168 2%
Number of 4 input LUTs:	210 out of 7168 2%
Number of bonded IOBs:	55 out of 141 39%
Number of MULT18X18s:	7 out of 16 43%
Number of GCLKs:	1 out of 8 12%

Table IX Symmetric FIR Order 15

Selected Device : 3s400pq208-4	
Number of Slices:	346 out of 3584 9%
Number of Slice Flip Flops:	293 out of 7168 4%
Number of 4 input LUTs:	389 out of 7168 5%
Number of bonded IOBs:	56 out of 141 39%
Number of MULT18X18s:	8 out of 16 50%
Number of GCLKs:	1 out of 8 12%

result shown that the amount of the component has been used in the FPGA as shown in the following diagram for the linear phase FIR filter for the order 5, order 10 and order 15 becoming the least significant bit and the least significant bit received form the parallel port of the 8 bits register now is becoming the most significant bit and so on. There is no parity check bit and frame error bit. The simulation result for the 16 times baud rate clock is correct as it can be seen that there is only one pulse signal for the clk16x only when 54 50MHz pulses passed so it is correct. It is really hard to judge the behavior of the FIR low pass filter form this simulation data however the simulation behavior is generated based on the design specification in section 3.3 so the behavior model should be correct.

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

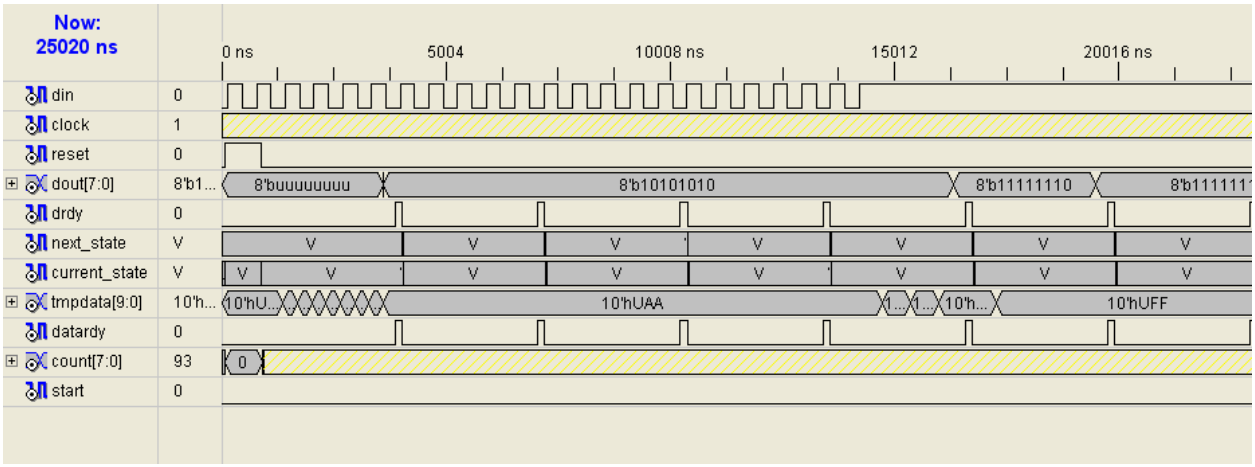


Fig. 20 Simulation behavior of the UART Transmitter

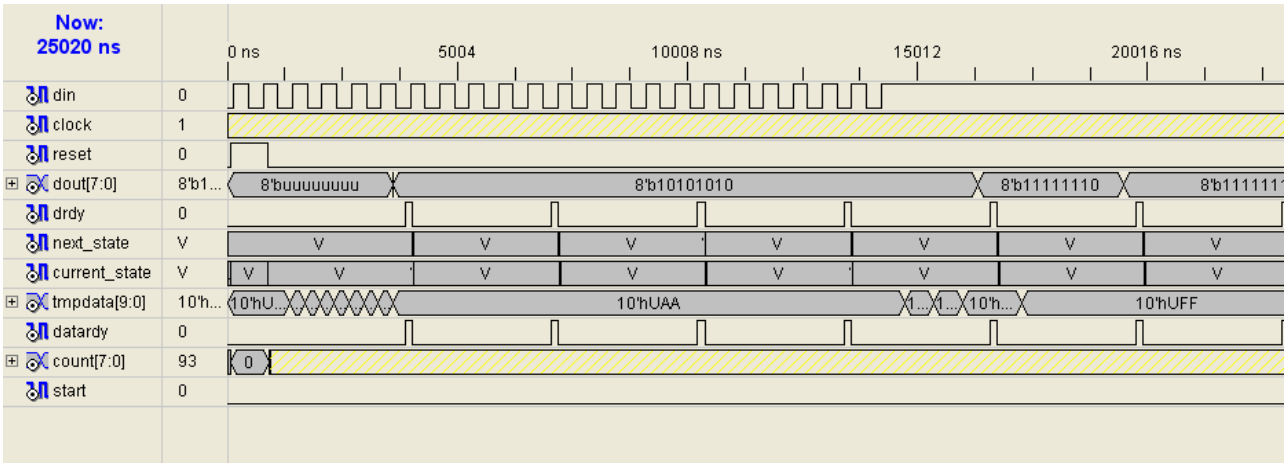


Fig. 21 Simulation behavior of the FIR filter

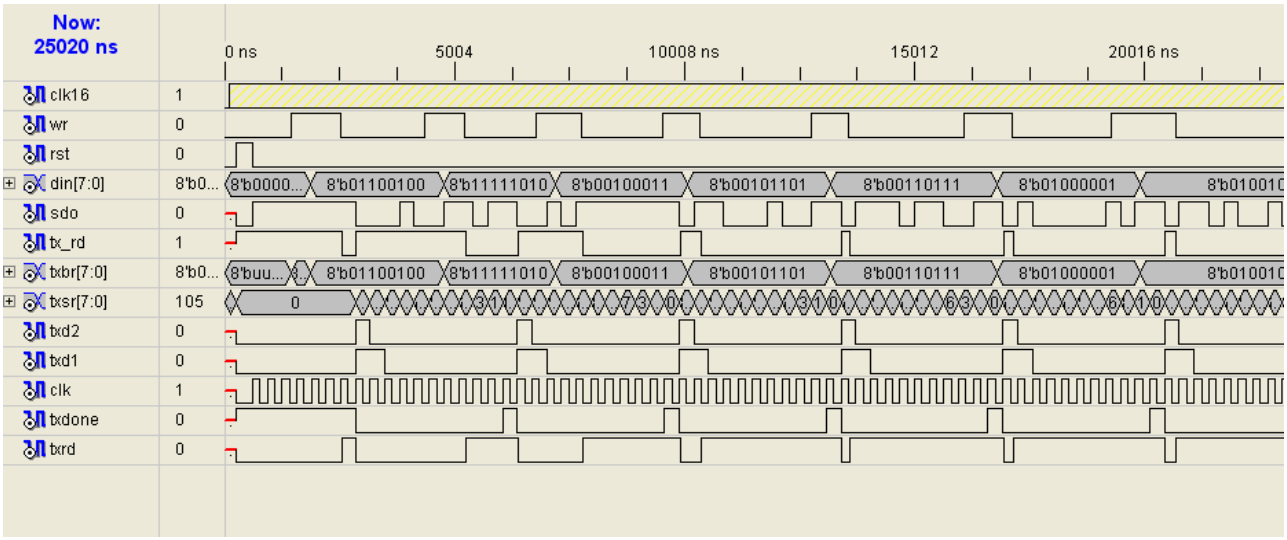


Fig. 22 Simulation behavior of the UART receiver

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

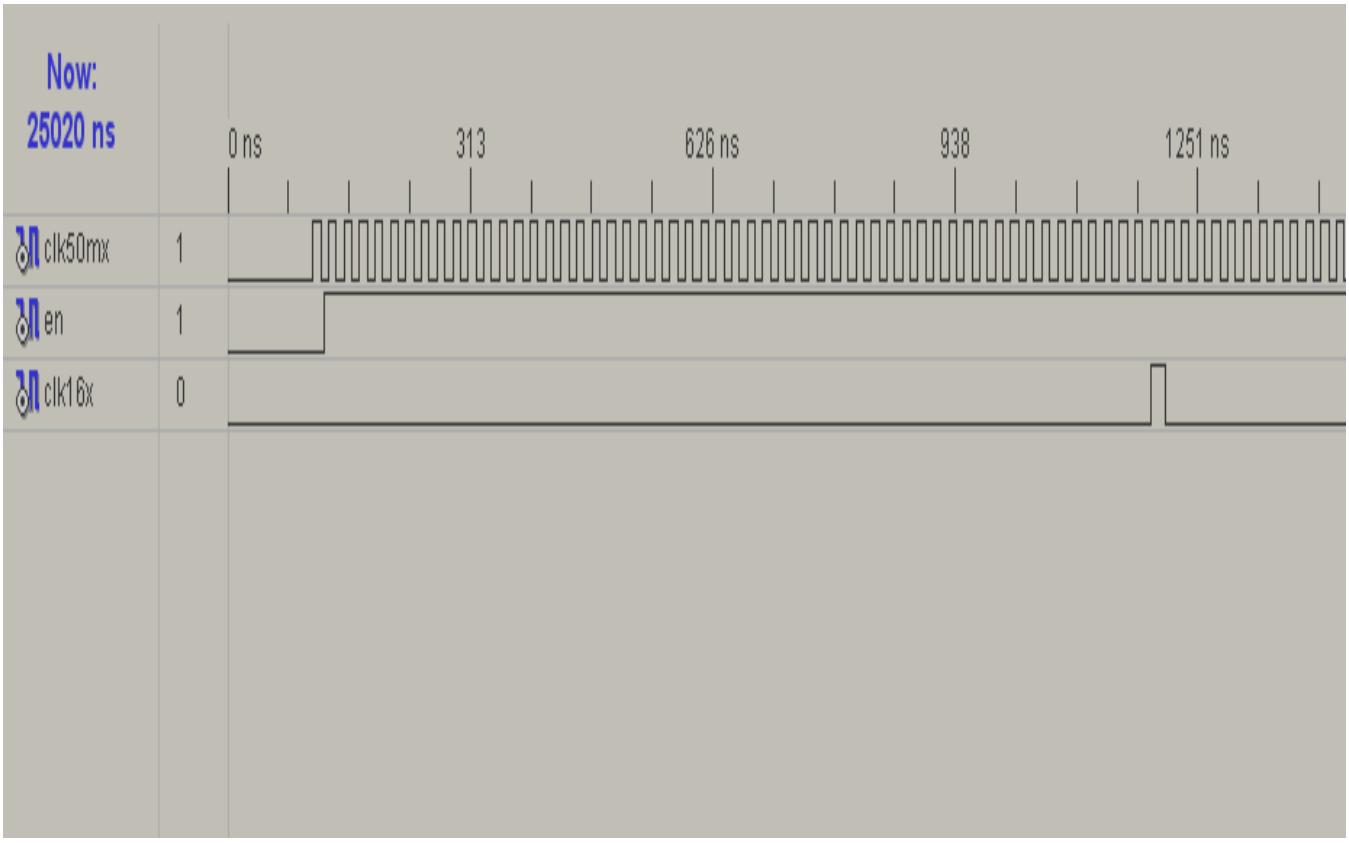


Fig. 23 Simulation behavior of the16 times baud rate clock

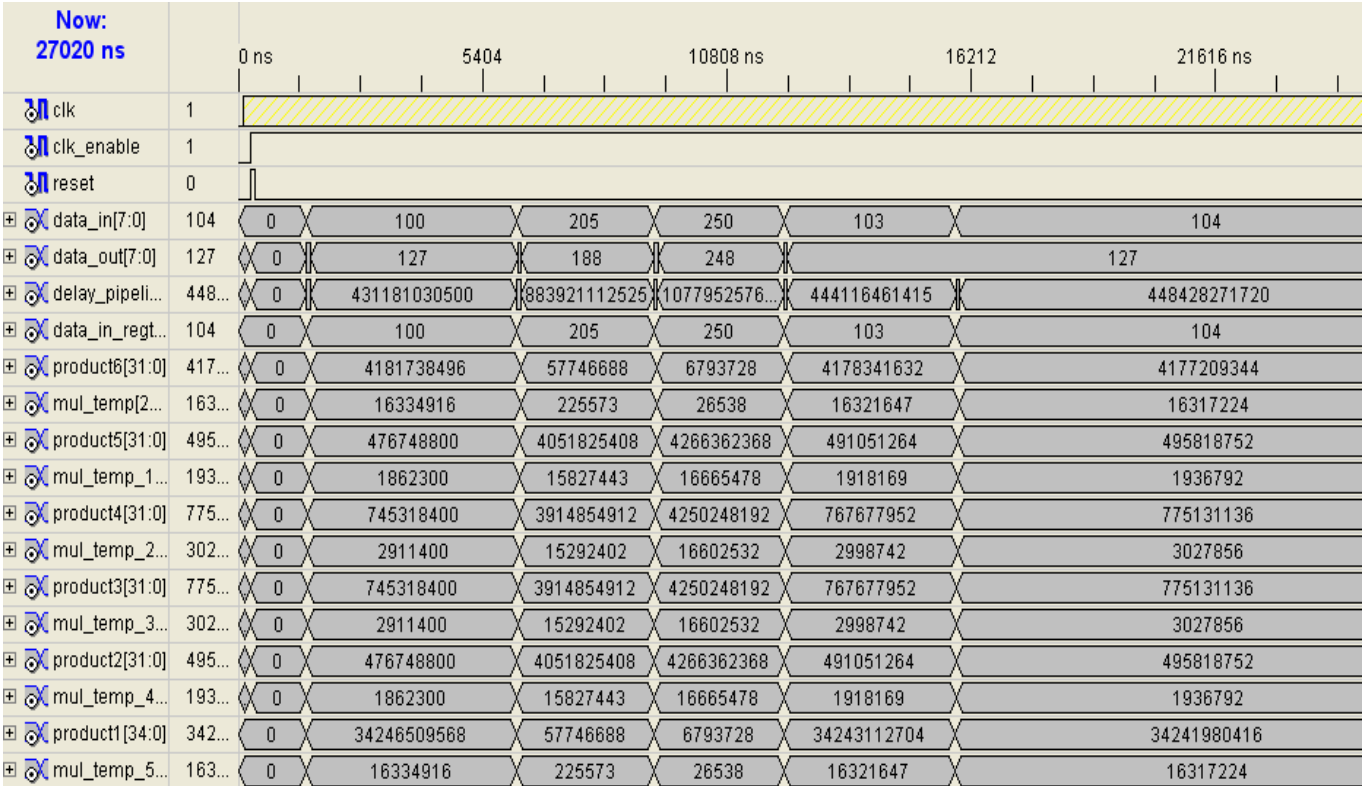


Fig. 24 Simulation behavior of the UART receiver

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

XIX. CONCLUSION

This work addressed the conception of Digital filter capable of measuring, processing signals effectively as they are made efficient based on their mode of production using FPGA. Speed of processing was one major aspect this project achieves with proper choice of the computation using the FPGA algorithms.

REFERENCES

[1] Holloway, J.L, 1958; Smoothing and Filtering of Time Series and Space Fields, in Landsberg, H.E, and J. Van Mieghem, editors, Advances in Geophysics Vol. 4, pp. 351 - 389, Academic Press, New York.

[2] Jacobson, L.A., 1990; A fast data smoothing algorithm, Computers in Physics, V. 4 No. 4, pp. 400 – 402.

[3] Mueller, M.S., 1981; Least-Squares Algorithms for Adaptive Equalizers, Bell System Technical Journal, V. 60 No. 8, pp. 1905 – 1925.

[4] Openheim, A.V, and R.W. Schafer, 1975; Digital Signal Processing, Prentice-Hall, Englewood Cliffs NJ, 585 pages, ISBN 0-13-214635-5.

[5] Press, W.H., and S.A. Teukolsky, 1990; Savitzky-Golay Smoothing Filters Computers in Physics, V. 4, No. 6, pp. 669-672.

[6] Rabiner, L.R. and B. Gold, 1975; Theory and Application of Digital Signal Processing, Prentice-Hall, Englewood Cliffs NJ, 762 pages, ISBN 0-13-914101-4.

[7] Roberts, D.A., 1993; An algorithm for finding spurious points in turbulent signals, Computers in Physics, V. 7 No. 5, pp. 599 – 607.

[8] Roberts, J. and T.D. Roberts, 1978; Use of the Butterworth Low-Pass Filter for Oceanographic Data, Jour of Geophysical Research, V. 83 No. C11, pp. 5510-5514.

[9] Robinson, E.A., 1967; Multichannel Time Series Analysis with Digital Computer Programs, Holden Day, San Francisco, 298 pages, ISBN 0-8162-7254-1.

[10] Robinson, E.A. and M.T. Silvia, 1978; Digital Signal Processing and Time Series Analysis, Holden Day, San Francisco, 411 pages, ISBN 0-8162-7264-6.

[11] Robinson, E.A. and S. Treitel, 1980; Geophysical Signal Analysis, Prentice-Hall, Englewood Cliffs NJ, 466 pages, ISBN 0-13-352658-5.

[12] Stearns, S.D., 1975; Digital Signal Analysis, Hayden Book Company, Rochelle Park NJ, 280 pages.

[13] Vaccaro, R.J. and F. Li, 1987; A State-Space Approach to Positive Sequences, IEEE ICASSP 1987 Proceedings, pp. 1304 – 1307.

[14] http://www.bukisa.com/articles/215991_design-of-Digital filters James Agajo.

AUTHORS BIOGRAPHY

Engr. James Agajo is into a Ph.D Programme in the field of Telecommunication and Computer Engineering; He has a Master's Degree in Electronic and telecommunication engineering from Nnamdi Azikiwe University Awka Anambra State, and also possesses a Bachelor degree in Electronics and Computer Engineering from the Federal University of Technology Minna Nigeria. His interest is in Wireless Sensor Sytems network and intelligent system development with a high flare for Engineering and Scientific research. He has Designed and implemented the most resent computer controlled robotic arm with a working grip mechanism which was aired on a national television , he has carried out work on using blue tooth technology to communicate with microcontroller. Has also worked on thumb print technology to develop high tech security systems with many more He is presently on seconded with UNESCO TVE as a supervisor and a resource person. James is presently a member of the following association with the Nigeria Society of Engineers (NSE), International Association of Engineers (IAENG) UK, REAGON, MIRDA, MIJICTE.Engr. James agajo has presented Papers in well over 3 continents of the World (England, China, Japan, Kenya, Uganda, Ghana e.t.c)

Dr Victor Eze Idogo is an associate Professor with the Nnamdi Azikiwe University Awka Anambra State Nigeria. He's area Of interest is Wireless Sensor Sytems and Communication, with over 20 years of academic experience. He is currently the Head of Electronics and Computer Engineering, he is a member of various Professional bodies amongst which are NSE, IAENG, COREN, He has much Publication to his credit both locally and internationally and he is a recipient of many awards.

Engr. Joseph Okhaifo is into a Ph.D Programme in the field of Telecommunication Engineering, He has a Master's Degree in Electronic and Telecommunication Engineering from University of Benin Edo State, and also possesses a Bachelor degree in Electrical and Electronics Engineering, and he is a member of many international Bodies amongst which are IAENG, NSE, COREN e.t.c. He has a lot of publication to his credits both locally an international.

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

APPENDIX

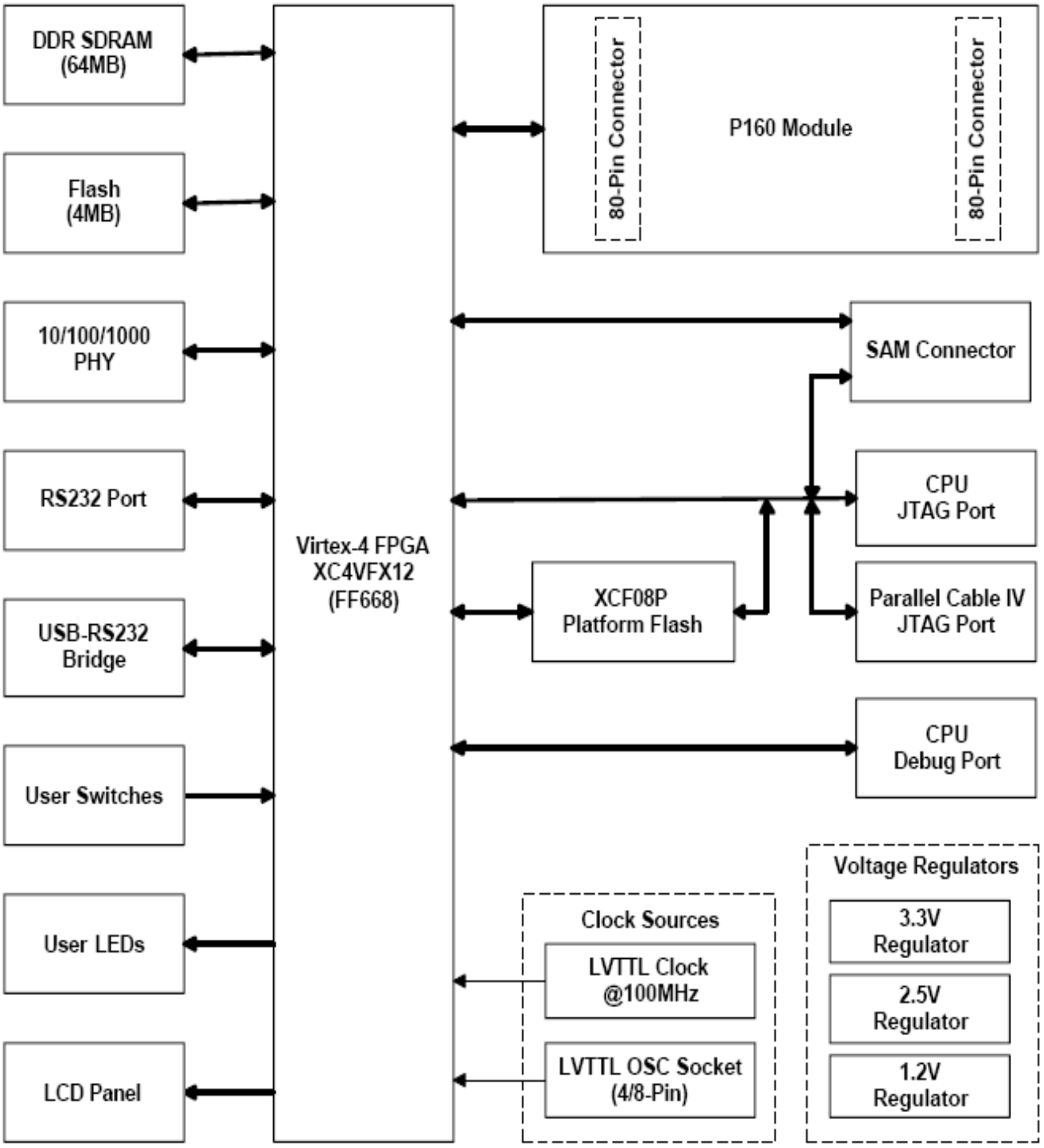


Fig 15 Onboard peripheral [6]

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012

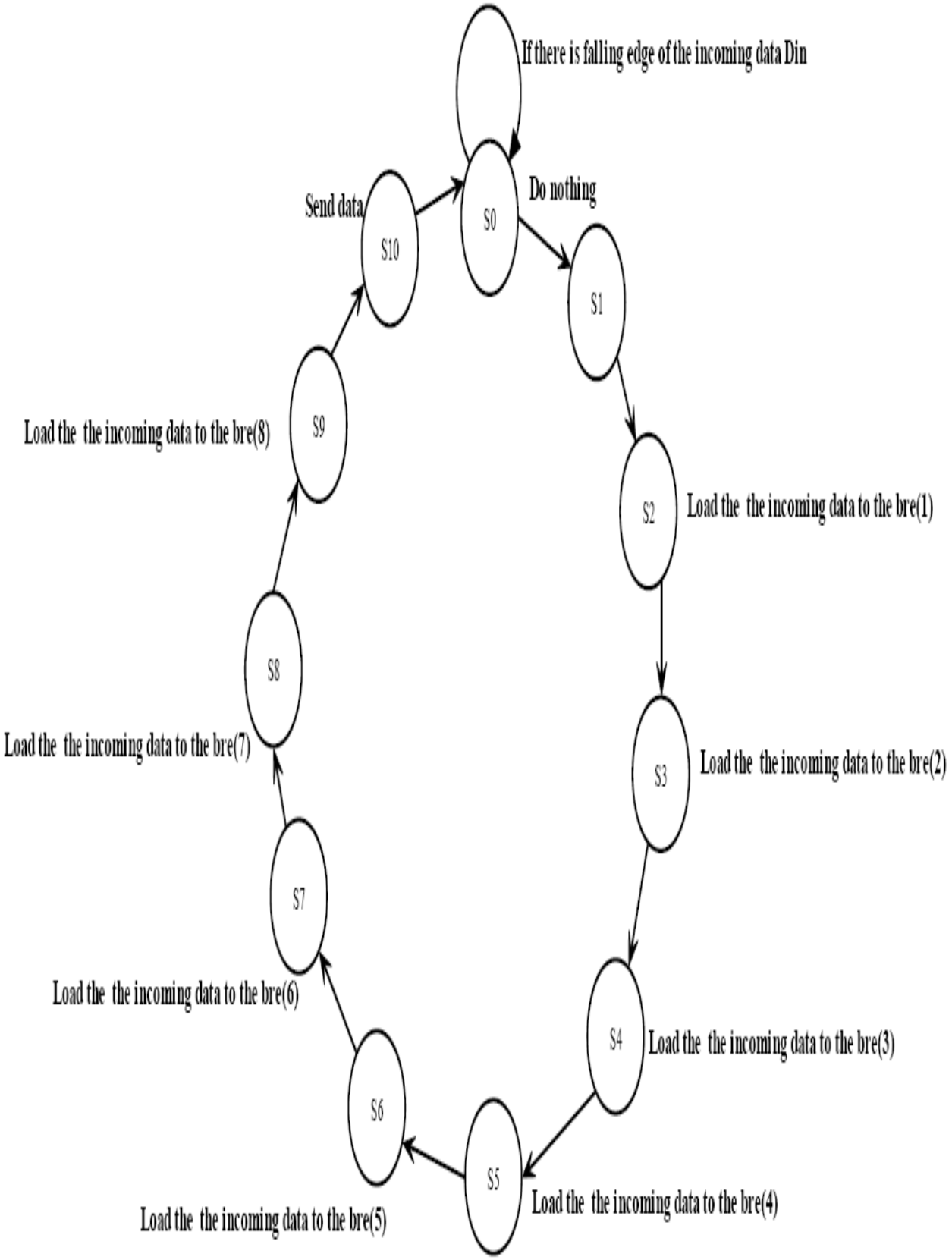
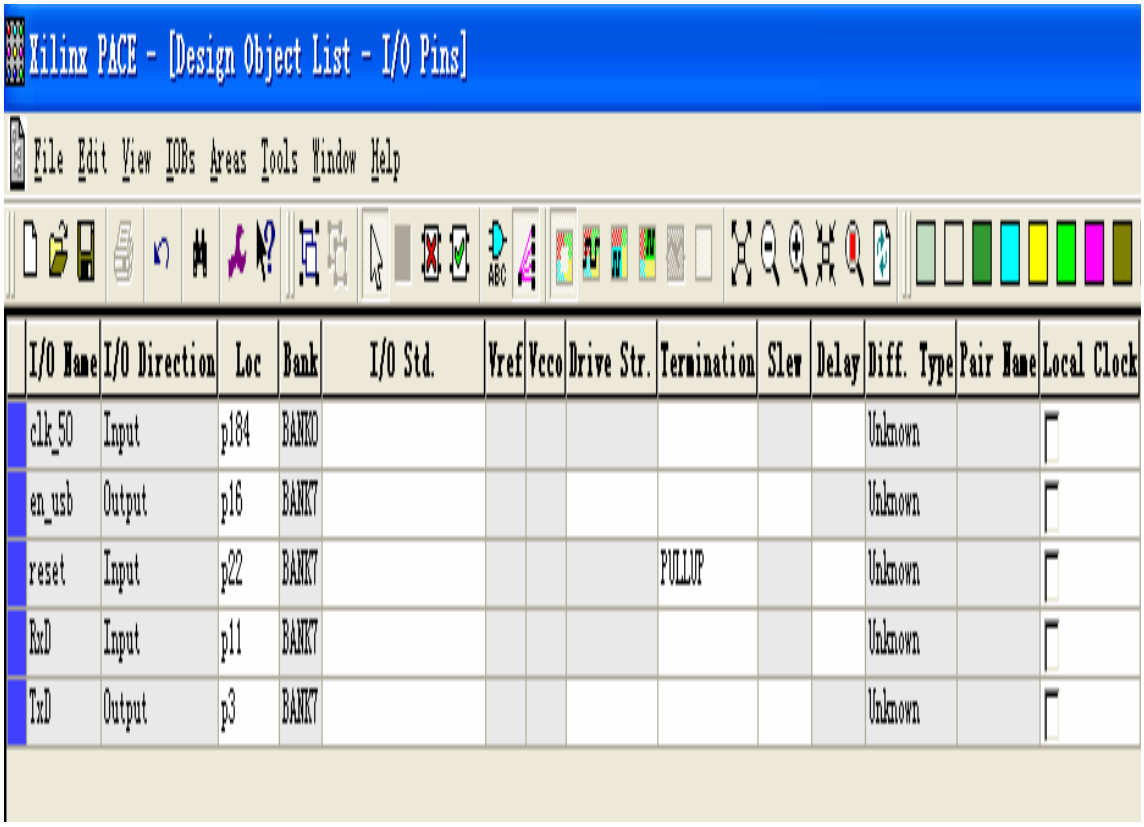


Fig 16 Finite State Machine of the Receiver [9]

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 3, September 2012



I/O Name	I/O Direction	Loc	Bank	I/O Std.	Vref	Vcco	Drive Str.	Termination	Slew	Delay	Diff. Type	Pair Name	Local Clock
clk_50	Input	p184	BANK0								Unknown		
en_usb	Output	p16	BANK7								Unknown		
reset	Input	p22	BANK7					PULLUP			Unknown		
RxD	Input	p11	BANK7								Unknown		
TxD	Output	p3	BANK7								Unknown		

Fig 17 Assign Pins for the Designed System