International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 02, February-2015

Synthesizable IP Cores of Low Cost Peripherals for Embedded SoC Applications

B. S. S. V. Ramesh Babu[1], A. Sai Ramya[2], Y. Ramesh Babu[3], B. Alekhya[4], E. Srikala[5]
[1,2,3,4] Department of ECE, Raghu Institute of Technology.
[5]Xinthe Technologies

***Abstract*— In general the IP Cores are classified into two types. They are Soft cores and hardcore's. FPGA, Altera and Altec etc comes under soft cores where as ASIC comes under hard cores.Soft cores have more flexibility when compared to the hardcore because verilog code is used to realize any logical system and further synthesized and can be modified. This is not possible in hardcore. This paper deals with the advantage of the soft cores over the hard cores that make a significant impact for their implementation. The methodology involves in writing verilog code and dumping in FPGA, then verifying for the desired operation and reconfiguration. This enables a closed loop monitoring to achieve the end result without much time consumption. By considering the above features and constraints of IP cores, the main objective of the project is to develop low-cost peripherals. In this due course UART, Programmable Timer, Programmable peripheral Interface are developed using verilog programming and test bench waveforms are generated to test these modules. Xilinx ISE 13.1 is used to develop and test the above mentioned IP Cores. The developed soft cores of embedded peripherals are interfaced with the RISC processor to enhance the advantages of being low cost, reliable and compatible with most of the systems. This confirms the compatible characteristics of the developed softcores which enable to integrate with any other processor of interest.**

Keywords—Softcore,UART,Programmable Timer,Programmable Peripheral Interface,Verilog

I. INTRODUCTION

In the last few years the growing complexity of algorithms in every area of electronics and the accelerated shrinking of Integrated circuits have begun to change the scenario on which hardware and IC designers devise their implementations. At that pace, with the current and still coming complexity level, and with a non-declining consumer market asking for cheaper but also more sophisticated products, designers and companies cannot afford to develop any design from scratch. It is at this point that reusability and specifically designing at the "macro" level gain great importance.Hence, the use of Intellectual property cores (IP) as basic building blocks in these designs will be the best choice.[1].IP core is a macro structure with specific function module which is pre-designed, reusable and flexible to adopt several applications such as development of SoC products. It is also with independent intellectual property rights and certain features.IP core has three different design domains as soft IP cores, Solid IP cores, and Hard IP cores [1].As system on chip is integrated with different types of IP cores, it plays a vital role in designing SoC product.

A Soft-core processor is used for customizing a given application and synthesized for a target. Soft core processor provides several advantages than hard-core like reduced cost, flexibility, platform independence and great immunity to obsolescence [2].hence, soft-cores are preferred. Commercial soft-core processors such as Altera NiosII and Xilinx micro blaze are to be widely used [3].However, soft-core HDL source code is not freely available. This restricts the reusability of soft-core[4].

`This paper presents synthesizable IP cores of low cost peripherals like Universal Asynchronous Receiver and Transmitter(UART) ,Programmable Timer, Programmable Peripheral Interface(PPI) are used for embedded SoC applications. UART is a microchip with programming that controls a computer interface to its attached serial devices. As part of this interface, the UART also converts the bytes it receives. The Interval Timer core with Avalon® interface is an interval timer for Avalon-based processor systems, such as a Nios® II processor system. The core provides the key features like 32-bit and 64-bit counters, powered with controls to start, stop, and reset the timer. There are two count modes namely, count down once and continuous count-down. It has the option to enable or disable the interrupt request (IRQ) when timer reaches zero. The optional watchdog timer feature that resets the system if timer ever reaches zero. The optional periodic pulse generator feature that outputs a pulse when timer reaches zero. The Intel 82C55A is a high-performance, CHMOS version of the industry standard 8255A general purpose programmable I/O device which is designed for use with all Intel and most other Microprocessors. It provides 24 I/O pins which may be individually programmed in 2 groups of 12 and used in 3 major modes of operation. The 82C55A is pin compatible with the NMOS 8255A and 8255A-5. In MODE 0, each group of 12 I/O pins may be programmed in sets of 4 and 8 to be inputs or outputs. In MODE 1, each group may be programmed to have 8 lines of input or output. 3 of the remaining 4 pins are used for handshaking and interrupt control signals. MODE 2 is a strobed bi-directional bus configuration.

The main objective of this paper is to develop low cost peripherals like UATR, PPI, Programmable timer and are designed and simulated using Xilinx13.1.This paper uses verilog code to implement above mentioned peripherals and integrate them into a FPGA chip to achieve reliable transmission of data.

The rest of the paper structured as follows. In section II we are going to discuss about design and implementation of UART. Section III is regarding

Programmable timer and PPI. Results are discussed in section IV. Finally conclusion is mentioned in section V.

II. Design and Implementation of UART

The transmission and reception data using UART is dependent on Asynchronous serial communication .This type of communication is preferred when two or more devices are to be communicated. UART is a serial communication protocol which is extensively used for data communication as it uses full duplex communication in serial link[5]. It gives us advantages like use of less no. of transmission lines, increased transmission distance, and more reliability and also it reduces the distortions in a signal. Basically UART can be described with its three sub-modules as local clock generator(Baud-rate generator),Transmitter module and receiver module as shown in the figure.

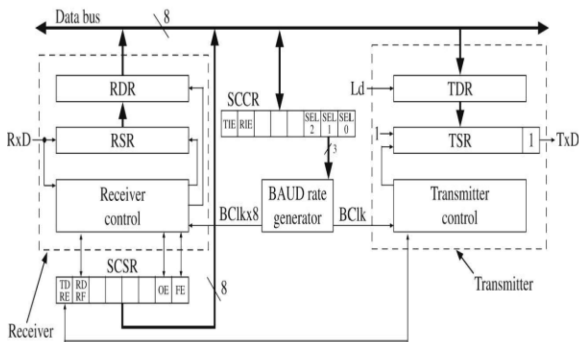

Figure 1.Architecture of UART

A) **BAUD rate generator:** BAUD rate generator is here used as a frequency divider to provide the system clock for transmitting and receiving data .Here in this regard the obtained frequency clock is 16 times the baud rate clock.

B) **Receiver Module:** The Receiver block detects the start bit of an incoming serial data and samples the data bits, bit by bit, according to the baud clock of the baud rate generator[6]. It completes the receive process of a single symbol of 6, 7 or 8 bits with the detection of the stop bit. Parity check of the received symbol ensures that the data has been received correctly. In the case of invalid stop bit or parity check error, the status signals parity error or frame error will be set. Finally the receiver writes the received symbol data onto the local data bus, which connected to the CPU Bus controller, and sets a signal to indicate "Receiver data Write". This signal initiates that the CPU Bus controller informs the CPU via an interrupt about the data arrival. The operation of the UART receiver can be described as When the UART detects a start bit, it reads in the remaining bits serially and shifts them into the RSR. When all the data bits and the stop bit have been received, the RSR is loaded into the RDR, and the RDRF flag in the SCSR is set. The microcontroller checks the RDRF flag, and if it is set, the RDR is read and the flag is cleared.

Transmitter module: The transmitter block is responsible for the serial transmitting of the data. First the transmitter detects whether the UART transmitter buffer (FIFO or TxD Hold Register) contains data for transmission. If it does, it loads the data onto the transmit register at the transmitter circuit via the local data bus (which connects the CPU Bus controller and the transmitter) and sets a signal to indicate "Transmit data Read". This signal initiates a sequence where the CPU Bus controller informs the CPU via an interrupt about the transmitted data, so that the CPU can load a new value. Synchronous to the baud clock which is generated by the baud rate generator, the transmitter sets the start bit on the TxD signal line to initiate the start of a frame and then bit by bit the symbol data. It finally completes the transmission by sending a parity bit that represents the parity of transmitted data and completes the frame with the final stop bit. The procedure will be repeated for another symbol, if the transmitter buffer contains another symbol, else the transmitter goes to an idle mode, transmitting "Marks"[7].

III .Design and implementation of programmable timer and PPI

A)Programmable Timer: The programmable timer is an eight-bit timer that works with three different modes, a one-shot timer, a pulse generator, and a 50% duty cycle waveform generator. For each and every mode, a particular data value can be loaded into the timer before the timer begins working. The timer can easily recognize which mode to operate in by using an internal register. This internal register is a "control word register" which is a three-bit register with the Most Significant Bit representing "timer enable" and bit 1 and bit 0 represents the mode of operation.

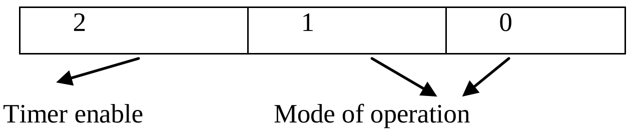

Figure 2.Control word register

Mode of operation: **Mode "00"** (one-shot timer) In this mode, the timer is loaded with an eight-bit binary value. A three-bit value of "100" (timer enable and operate in mode 0) is written into the ***control word register*** of the timer. The timer then begins and when it a hex value of "FF", the output of the timer generates a one-clock-width pulse. The timer overwrites its MSB of the ***control word register*** to a "0" and the timer stops. To continue it the value must be 1.
Mode "01"(pulse generator) In this mode, the timer is loaded with an eight-bit binary value. A three-bit value of "101" (timer enable and operation in mode 1) is written into the ***control word register*** of the timer. The timer then begins counting and when it reaches a hex value of "FF", the output is one-clock-width pulse. The timer is then automatically reloaded again with the initial value .The timer in this mode does not stop unless the user writes a "0" to the MSB of the ***control word register*** or if ***ceb*** is pulled to a logic "1"[8].
Mode "10"(50% duty cycle waveform generator) In this waveform generator mode, The timer is loaded with an eight-bit binary value. A three-bit value of "110" (timer enable and operation in mode 2) is written into the ***control word register*** of the timer. Unlike mode 0 or 1, the timer would count down and not count up. When the timer reaches a hex value that is half of the loaded value,the output of the timer would generate a logic "1". The timer continues to count down and when it reaches a hexadecimal value of "00", the timer would be reloaded automatically

with the initial loaded value and count down again. Similar to mode 1, the timer in this mode does not stop unless the user writes a "0" to the MSB of the ***control word register*** or if ***ceb*** is pulled to a logic "1".

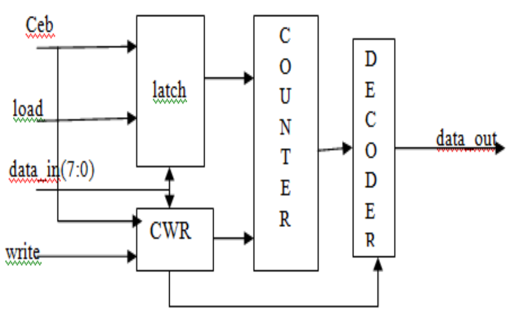

Figure 3.Micro architecture of programmable timer

B)PPI(Programmable peripheral interface): A peripheral interface is used to communicate between several devices.

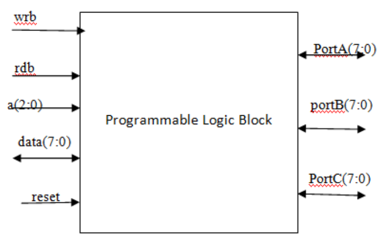

Figure 4.Interfacing signals of PLB

The PLB interconnect device has 8 bit wide 24 I/O pins, separated into three groups: portA, portB, and portC. PortC can be further separated into two subgroups, portCupper and port-Clower, each four bits wide. The PLB can also operate in three different modes: mode 0, mode 1, and mode 2. Each mode of operation allows for different configurations on the I/O ports[10].

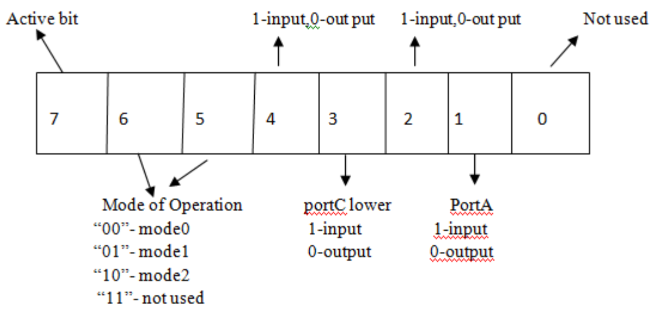

Figure 5.Diagram showing functionality of each bit in CWR register

A)Description of bits 4 to 1 of CWR register in MODE-0:

Bit	Logic Value	Functionality of the ports in MODE-0 Operation
4	0	PORTC[7:4] operates as output bus
	1	PORTC[7:4] operates as input bus
3	0	PORTC[3:0] operates as output bus
	1	PORTC[3:0] operates as input bus
2	0	PORTB operates as output bus
	1	PORTB operates as input bus
1	0	PORTA operates as output bus
	1	PORTA operates as input bus

Table 1: Functionality of bits in CWR register in MODE-0

B)Description of bits 4 to 1 of CWR register MODE-1:

Bit	Logic Value	Functionality of the ports in MODE-0 Operation
2	0	PORTB operates as Strobed output bus
	1	PORTB operates as Strobed input bus
1	0	PORTA operates as Strobed output bus
	1	PORTA operates as Strobed input bus

Table 2:Functionality of bits in CWR register in MODE-1.

IV IMPLEMENTATION RESULTS

A) UART

We used Verilog hard ware description language for implementation. And also test bench wave forms are generated for the above mentioned modules. Xilinx ISE 13.1 is used for synthesis of above mentioned IP cores.

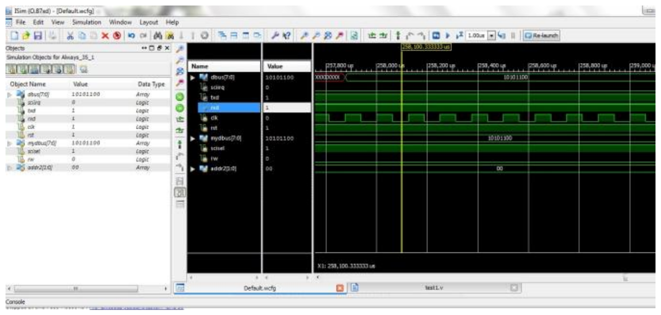

Figure 6.Diagram showing output of test bench of UART

Select bits	Baudrate(bits per second)
000	38,462
001	19,213
010	9615
011	4808
100	2404
101	1202
110	601
111	300.5

Table 3:Generated frequencies

Figure 6 shows the output waveform of UART.On the transmitter side we have applied a input bitstream of 1010110.The corresponding output at the receiver output is also the same which can be read from the above diagram. With the inclusion of the Baud Rate Generator with the delay of clock pulse ,the input appears at the receiver.

B) PROGRAMMABLE TIMER:

I)MODE-0

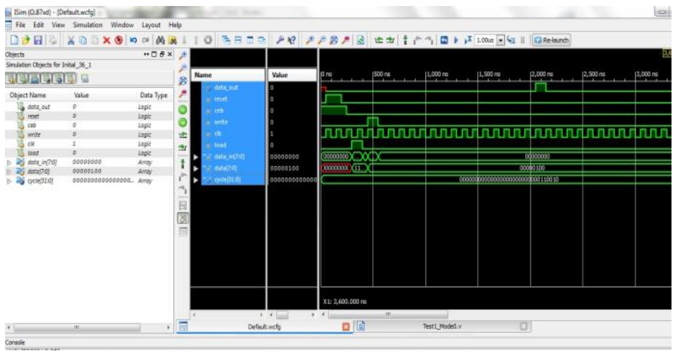

Figure 7.Diagram showing output of test bench of Programmable timer mode-0

MODE-1

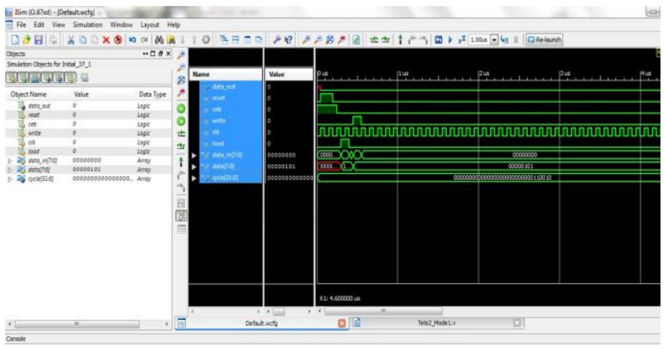

Figure 8.Diagram showing output of test bench of Programmable timer mode-1

MODE-2

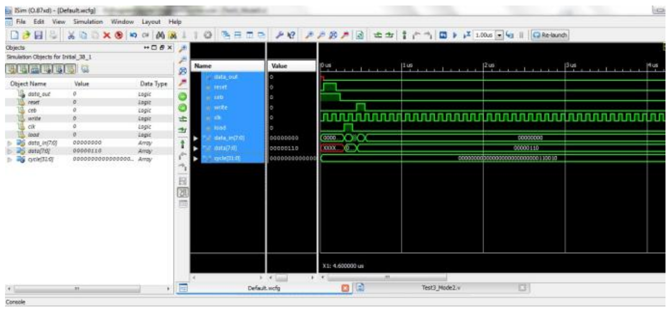

Figure 9.Diagram showing output of test bench of Programmable timer mode-2

The above figures show the simulation results of Programmable Timer in Mode-0,Mode-1,Mode-2 operations. With the input bit stream of 00000100 the configuration acts as an One-Shot time,pulse generator,50%dutycycle waveform generator respectively. The same is depicted in the above figures 7,8&9.

C) PROGRAMMABLE PERIPHERIAL INTERFACE:

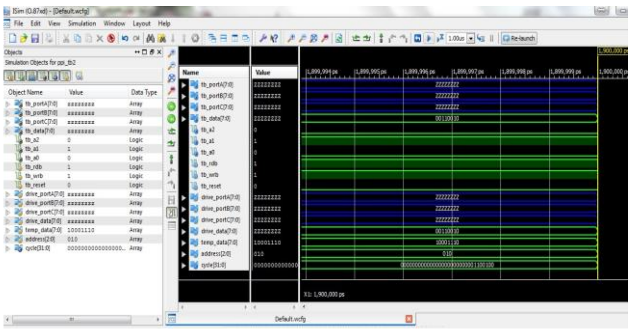

Figure 10.Diagram showing output of test bench of Programmable peripherial interface

Figure 10 shows the simulation results of Programmable Peripheral Interface in Mode 0 operation. In this mode all three ports portA, portB, portC can act as either input or output ports. In the above simulation the portA, portB, portC lower is configured as Input and portC upper as output port. An input bit stream 10001110 is given.

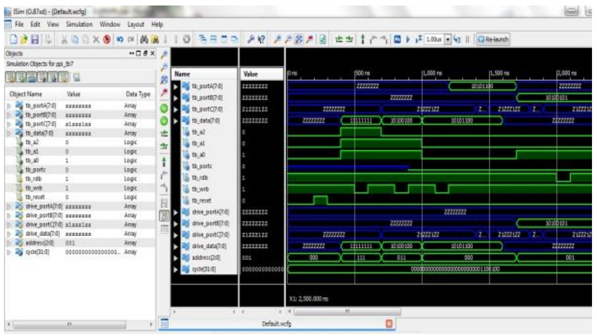

Figure 11.Diagram showing output of test bench of Programmable peripherial interface

Figure 11shows the simulation results of Programmable Peripheral Interface in Mode 1operation. In this simulation portA, portB are configured as strobed input ports in the mode 1 operation.In this mode, portC is used as control signals for the strobed input ports.

V CONCLUSION AND FUTURE SCOPE

In depth study of soft-core processors is done. A Verilog code for low cost peripherals is developed. UART serial communication controller is designed and tested using synthesizable Verilog codes carried in Xilinx 13.1 and synthesis report is generated. This will be an initial point to integrate the synthesized Soft cores of embedded peripherals to the RISC processor and to study their compatibility characteristics. This paper can be extended to other processors like I2C, SPI, and Firewall etc. This can be explored using embedded soft cores for other processor mentioned above.

REFERENCES

1) Xuemin Pang Daojie Yu, Jianbing Li and Yuhua Guo, " Design and Application of IP Core in SoC Technology," Information Science and Engineering (ISISE), 2010 International Symposium , Shanghai, 24-26 Dec. 2010,pp. 71 – 74.

2) Anderson, I.D.L.,Khalid, M.A.S. and Tong, J.G, "Soft-Core Processors for Embedded Systems," Microelectronics, 2006. ICM '06. International Conference, Dhahran, Dec. 2006,pp.170 – 173.

3) Al Rayahi, O.A. and Khalid, M.A.S., "UWindsor Nios II: A soft-core processor for design space exploration," Electro/Information Technology, 2009. eit '09. IEEE International Conference, Windsor, ON, 7-9 June 2009,pp. 451 – 457**.**

4) Le Gal,B and Jego,c., "Softcore Processor Optimization According to Real-Application Requirements," Embedded Systems Letters, IEEE Vol:5, Issue: 1 ,pp.4 – 7, March 2013.

5) Chun-zhi, Xia Yin-shui and Wang Lun-yao, " A universal asynchronous receiver transmitter design, " in Electronics, Communications and Control (ICECC), 2011 International Conference , Ningbo, Sept. 2011,pp. 691 – 694

6) Chun-zhi, Xia Yin-shui and Wang Lun-yao, " A universal asynchronous receiver transmitter design, " in Electronics, Communications and Control (ICECC), 2011 International Conference , Ningbo, Sept. 2011,pp. 691 – 694.

7) Norhuzaimin, J. and Maimun, H.H. "The design of high speed UART," in Applied Electromagnetics, 2005. APACE 2005. Asia-Pacific Conference, Johor, Dec. 2005

8) David A. Paatterson, John L. Hennessy, Computer Organization & Design, Morgan Kaufmann Publishers, 1999

10) Zargari, A. ; Combs, M.S." Construction, interfacing, and application of an 8255-based programmable peripheral interface card",in Electrical Insulation Conference and Electrical Manufacturing &amp; Coil Winding Conference, 2001. Proceedings,Cincinnati Oct.2001,pp.119-123.