FPGA IMPLEMENTATION OF USB TRANSCEIVER MACROCELL INTERFACE WITH USB2.0 SPECIFICATIONS

K. Babulu[1], K. Soundara Rajan[2]

[1]*Assoc.Professor, Dept of E.C.E, JNTU College of Engineering, Kakinada, kapbbl@gmail.com*
[2]*Professor, Dept of E.C.E, JNTU College of Engineering, Anantapur*

ABSTRACT: - *The Universal Serial Bus(USB) Transceiver Macro cell Interface (UTMI) is a two wire, bi-directional serial bus interface. The USB2.0 specifications define three types of UTMI implementations depends on data transmission rates, those are Low Speed (1.5MHz) only (LS), Full Speed (12MHz) only (FS) and High Speed (480MHz)/Full speed (12MHz) (HS). UTMI consists of Transmitting and Receiving sections, in which the Transmitter of the UTMI sends data to different USB devices through D+ and D- lines whereas the Receiver gets data on the same lines. This presentation reveals the FPGA implementation of UTMI with HS/FS transmission rate providing with USB 2.0 specifications. Further UTMI has been designed by using VHDL code and simulated, synthesized and programmed to the targeted Spartan2 family of FPGA in the Xilinx environment.*

1. INTRODUCTION

The Universal Serial Bus (USB) Transceiver Macrocell Interface (UTMI) is a two wire, bi-directional serial bus interface between USB devices through D+ and D- lines. This is one of the important functional blocks of USB controller, which can transmit and receive data to or from USB devices. There are three functional blocks present in USB controller; those are Serial Interface Engine (SIE), UTMI and Device Specific Logic (DSL). Figure 1 shows the block diagram of UTMI. The parallel data from SIE is taken into the transmit hold register and this data is sent to transmit shift register from where the data is converted serially. This serial data is bit stuffed to perform data transitions for clock recovery and NRZI (1) encoding. Then the encoded data is sent on to the serial bus. When the data is received on the serial bus, it is decoded, bit unstuffed and is sent to receive shift register. After the shift register is full, the data is sent to receive hold register.

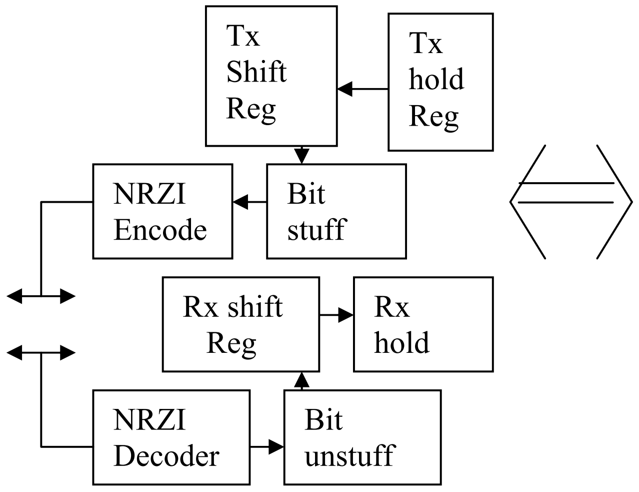


Figure 1: Block diagram of UTMI.

This data will be presented on the parallel interface where it is sampled by the SIE. The intent of the UTMI is to accelerate USB 2.0 peripheral development.

Features of the UTMI:

Supports 480 Mbit/s High Speed (HS)/ 12 Mbit/s Full Speed (FS), FS Only data transmission rates.

- Utilizes 8-bit parallel interface to transmit and receive USB 2.0 cable data.
- SYNC/EOP generation and checking.
- Data and clock recovery from serial stream on the USB.
- Bit-stuffing/unstuffing and bit stuff error detection.
- Holding registers to stage transmit and receive data.
- Ability to switch between FS and HS terminations/signaling.

978-0-7695-3267-7/08 $25.00 © 2008 IEEE
DOI 10.1109/ICETET.2008.77

2. DESIGN ASPECTS

The present UTMI has been designed according to the following specifications provided by the USB 2.0 protocol.

- SYNC and End of Packet (EOP) generation by the transmitter.
- SYNC and EOP detection by the receiver.
- Receive error reporting.
- Enabling or disabling the bit stuffer and NRZI encoder depends on the operational mode.
- Suspension of the transceiver by the SIE.

Further the UTMI is divided into two important modules which are the Transmitter module and the Receiver module. In this section the design Considerations of these modules have been explained separately and integrated to get top level Transceiver (UTMI) module.

2.1 The Transmitter Module

The block diagram of the UTMI transmitter is shown in Figure2. The transmitter module has been implemented by considering the following specifications.

- The SYNC pattern “01111110” has to be transmitted immediately after the transmitter is initiated by the SIE.
- After six consecutive ‘1’s occur in the data stream a zero to be inserted.
- The data should be encoded using Non Return to Zero Invert on 1 (NRZI -1) encoding technique.
- The EOP pattern two single ended zeroes(D+ and D- lines are carrying zero for two clock cycles) and a bit one have to be transmitted after each packet or after SIE suspends the transmitter

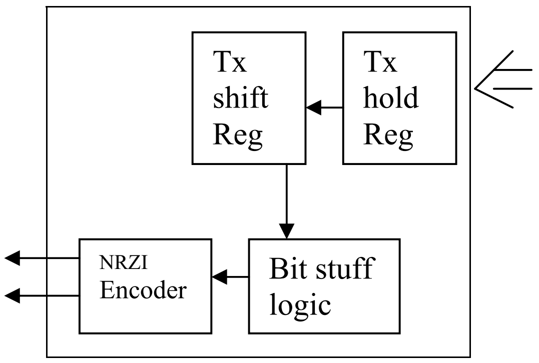


Figure 2: Block diagram of UTMI Transmitter.

The transmitter logic facilitates SYNC transmission, holding parallel 8- bit data from SIE, parallel to serial conversion of data, bit stuffing, NRZI encoding , transmission of the data and EOP transmission on to the serial bus.

2.2 The Receiver Module

The block diagram of the UTMI receiver is shown in Figure 3. The receiver module has been implemented by considering the following specifications.

- When SYNC pattern is detected that should be intimated to the SIE.
- If a zero is not detected after six consecutive ‘1’s an error should be reported to the SIE.
- When EOP pattern is detected that should be intimated to the SIE.

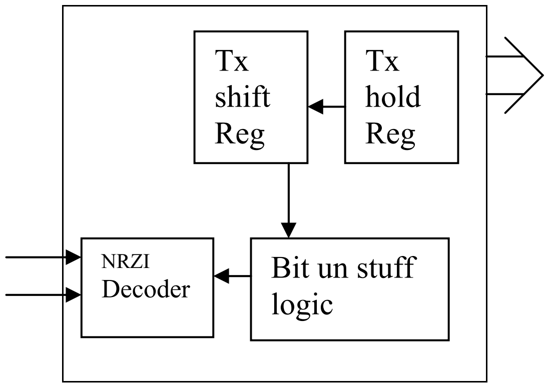


Figure 3: Block diagram of UTMI Receiver.

The receiver logic facilitates SYNC detection, NRZI decoding, bit unstuffing, serial to parallel conversion of data, receive error reporting and EOP detection.

2.3 The Transceiver Module

The transmitter and the receiver modules are combined together to design the transceiver (UTMI) module. This transceiver met all the USB2.0 specifications considered above. The transceiver logic facilitates the output of the transmitter to feed to the input of the receiver for functional verification. The Transceiver module has been designed with the considerations of individual modules of the transmitter and the receiver Specifications. Further the required Transceiver module logic has been verified with the functional simulation followed by necessary Synthesis and perform Programming to the targeted FPGA Device.

3. SIMULATON RESULTS

The individual modules of the UTMI are designed using VHDL as stated above and they are simulated within the Xilinx based Model Sim 6.0 environment.

The Transmitter Module

The Figure 4 shows the Simulation results of UTMI transmitter. When TX valid signal goes high, encoded SYNC pattern “01010100”is transmitted and the signal txready is asserted. The data “10110100” present on the dataln bus is NRZI encoded and transmitted on to the txdp, txdm lines. The signal txready goes low when the data is sampled by the TX hold register.

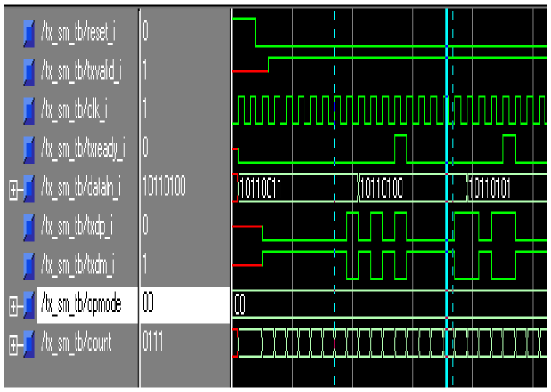


Figure 4: UTMI Transmitter Module

The Receiver Module

The Figure 5 shows the Simulation results of UTMI receiver. When SYNC is detected rxactive is asserted. The data present on rxdp, rxdm lines is decoded, serial to parallel converted and sent to the SIE through data out bus by asserting rxvalid signal.

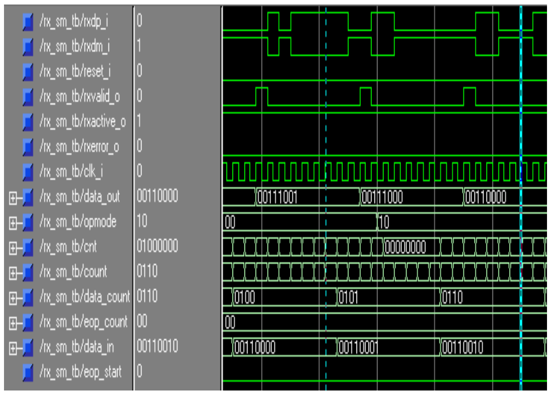


Figure 5: UTMI Receiver Module

3.3 The Transceiver Module

The Figure 6 shows the Simulation results of the Transceiver module which transmits and receives data. When txvalid goes high, SYNC is transmitted. The data “00000000” present on the data_bustx is NRZI encoded and transmitted on to the dp, dm lines.

When SYNC is detected by the receiver rxactive is asserted by the UTMI. The data present on the dp, dm lines is NRZI decoded and sent to the SIE through rxdata_bus by asserting rxvalid signal. Rxdata_bus contains “00000000” since the transmitted data is fed back to the receiver.

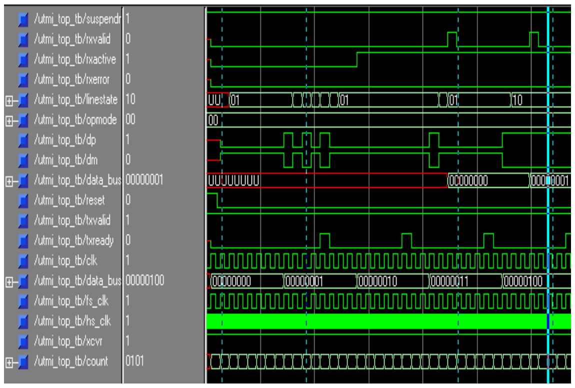


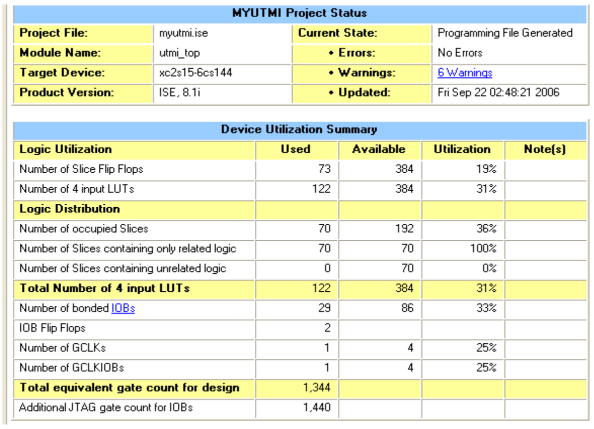
Figure 6: Transceiver (UTMI) Module.

4. FPGA IMPLEMENTATION

The top order module, UTMI is synthesized within the Xilinx 8.1 ISE software tool and it is programmed to the targeted SPARTAN 2 family of FPGA Device. The various levels pf implementation such as Synthesis report, RTL View, Place and Route Report and Device Programming has been explained and visualized in the following sub sections.

4.1 Synthesis Report

The Table 1 shows the synthesis Summary of top order module, UTMI. It is observed from the Table 1 that the total equivalent gate count required for this design is 1344 gates. From the same table we can get the information about the target FPGA device utilization.



MYUTMI Project Status			
Project File:	myutmi.ise	**Current State:**	Programming File Generated
Module Name:	utmi_top	**• Errors:**	No Errors
Target Device:	xc2s15-6cs144	**• Warnings:**	6 Warnings
Product Version:	ISE, 8.1i	**• Updated:**	Fri Sep 22 02:48:21 2006

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	73	384	19%	
Number of 4 input LUTs	122	384	31%	
Logic Distribution				
Number of occupied Slices	70	192	36%	
Number of Slices containing only related logic	70	70	100%	
Number of Slices containing unrelated logic	0	70	0%	
Total Number of 4 input LUTs	122	384	31%	
Number of bonded IOBs	29	86	33%	
IOB Flip Flops	2			
Number of GCLKs	1	4	25%	
Number of GCLKIOBs	1	4	25%	
Total equivalent gate count for design	1,344			
Additional JTAG gate count for IOBs	1,440			

Table 1: Synthesis Summary

4.2 RTL View

This section gives the visualization of Resister Transistor Logic (RTL) views in the form of schematic and Netlist diagrams which are shown in Figure 7 and Figure 8 respectively. Figure 7 which gives RTL schematic diagram reveals the pin diagram of Top order module with the required specified notes whereas Figure 8 reveals the Gate level logic diagram of Top order module with the required input and output ports(Netlist view).

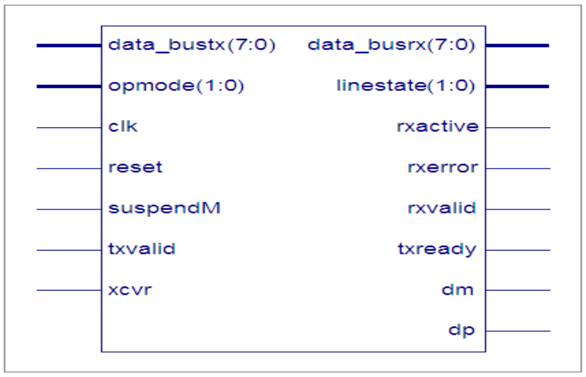


Figure 7: RTL Schematic diagram

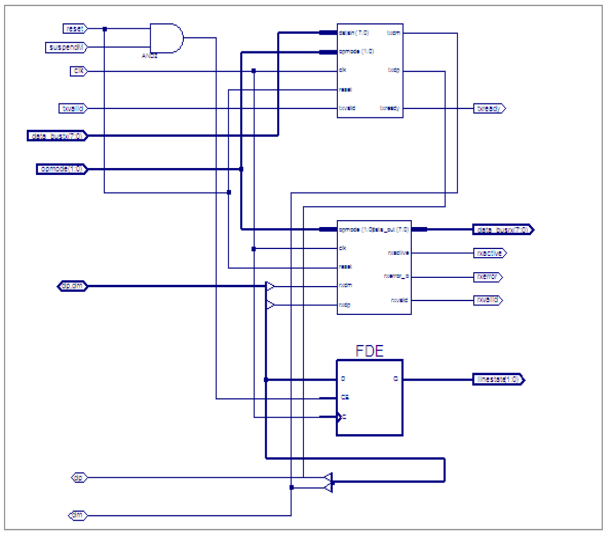


Figure 8: RTL netlist view

4.3 Place and Route Report

This section concentrates on target FPGA device utilization summary which reveals the information required for proper layout at the level of manufacturing in the form of Place and Route report. Further it gives the timing synchronization of CPU with the REAL time environment.

Device Utilization Summary:

Number of GCLKs	1 out of 4	25%
Number of External GCLKIOBs	1 out of 4	25%
Number of LOCed GCLKIOBs	0 out of 1	0%
Number of External IOBs	29 out of 86	33%
Number of LOCed IOBs	0 out of 29	0%
Number of SLICEs	70 out of 192	36%

Total REAL time to Placer completion: 2 secs
Total CPU time to Placer completion: 2 secs

4.4 Device Programming

After successful process of synthesis the Target device xc2s15 of Spartan2 is connected to the system through printer port. The pin assignment is specified in the User Constraint File (UCF). The functional verification is carried out by using a pattern generator.

5. CONCLUSION

The individual modules of UTMI have been designed using VHDL and verified functionally with the Model Sim 6.0.

The UTMI Transmitter is capable of converting parallel data into serial bits, performing bit stuffing and NRZI encoding.

The UTMI Receiver is capable of performing NRZI decoding bitunstuffing and converting serial bits into parallel data.

The functional simulation has been successfully carried out. The design has been synthesized using FPGA technology from Xilinx. This design is targeted to the device family→spartan2, device→xc2s15, and package→cs144 and speed grade→ 6. The device belongs to the Vertex-E group of FPGAs from Xilinx. The UTMI is designed to support HS/FS, FS Only and LS Only UTM implementations. The three options allow a single SIE implementation to be used with any speed USB transceiver. A vendor can choose the transceiver performance that best meets their needs.

6. FUTURE SCOPE

The UTMI has been implemented for 8-bit, but it can also be extended to 16- bit UTMI. It can also be designed to generate CRCs for control and data packets. If an SIE and Device specific logic are designed, the combination of UTMI, SIE and Device specific logic can be used as a controller of any USB device.

7. APPLICATIONS

The UTMI has been developed into a common code (Generalized USB Transceiver) which can be used for developing the complete USB device stack. Some of the Low speed and High speed USB devices, which are presently available in the market are:

1. Optical Mouse
2. Key Board
3. Printer
4. Scanner
5. Joy Stick
6. Memory Stick
7. Flash Memory
8. Mobiles
9. Video cameras.

8. REFERENCES

[1] Charles H Roth "Digital system using VHDL".2nd edition, Thomson publication

[2] Jayaram Bhasker "A VHDL Primer" 2nd edition, Prentice Hall publications

[3]Stephen Brown, Zvonko Vranesic "Fundamentals Digital logic with VHDL design". 2nd edition, McGraw-Hill, Hardcover, Published July 2004

[4] Zainalabedin Navabi "Vhdl Analysis and Modeling of Digital Systems" 2nd edition, McGraw- Hill, Hardcover, Published January 1998

[5] William Stallings, "Data and Computer Communications" ,McGraw-Hill Publications.

[6] Andrew S.Tannenbaum, "Computer Networks", Pearson publications.

[7] Z.Kohavi, "Switching and finite Automata Theory", Tata Mcgraw-Hill Publications

[8] N.N.Biswas, "Logic design theory", Printice Hall of India Publications.

[9] Morris Mano, "digital design", Tata McGraw-Hill Publications.

[10] Lala, "Digital system Design Using PLDs", BSP Publications.

Websites:

www.usb.org

www.opencore.org

www.digitalcoredesign.org

www.deeps.org