ISSN 2278-3091
International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 47-51 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

Development of USB using Manchester Encoding

Ch.Satish[1], P.Srisuresh[2], T.L.Lavanya[3]
[1]Asst.Prof,ECE Department,Pragati Engineering College,AP,India,chembrolusatish@gmail.com
[2]Asst.Prof,ECE Department,Pragati Engineering College,AP,India,srisuresh.penke@gmail.com
[3]Asst.Prof,ECE Department,Pragati Engineering College,AP,India,tummalapalli.lavanya@gmail.com

Abstract *:* The universal serial bus (USB) is a Full duplex serial bus interface. USB devices receive the data from Transmitter Macro cell interface : a transmitter and a receiver section both of which uses 2 lines D+ and D- for transmission and reception. There are three functional blocks present in USB controller: Serial Interface Engine (SIE), UTMI and Device Specific Logic (DSL).The transmit hold register holds the data received from SIE and is converted to serial data and is send to transmit shift register. This serial data is added to some extra bits known as bit stuffing in order for clock synchronization and Manchester 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. The same can be designed using VHDL code and synthesized using spartan**.**

Key words :USB,Macrocell, Stuffing,SIE,UTMI

INTRODUCTION

Introduction

The USB controller consists of

- The USB 2.0 Transceiver Macrocell Interface (UTMI),
- The Serial Interface Engine (SIE),
- The device specific logic.

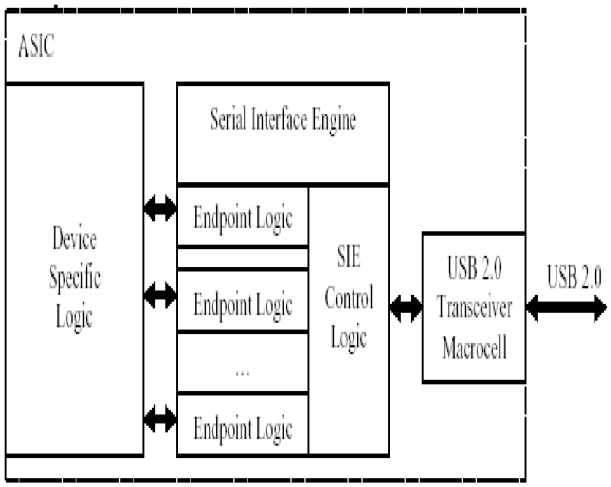


Fig 1: Block diagram of USB controller

Applications

[5]Some of the Low speed and High speed USB devices, which are presently in the market, are:Optical Mouse, Key Board, Printer, Scanner, Joy Stick,Memory Stick ,Flash Memory, Mobiles.

UTMI

This block functions for data serialization and de serialization, bitstuffing and clock synchronization. It is also used to shift the clock rate of the data from the USB 2.0 rate to one that is compatible with the general logic in the ASIC.[1]The UTMI is designed to support HS/FS, FS Only and LS Only UTM implementations.
A HS/FS implementation of the transceiver can operate at either a 480 Mb/s or a 12 Mb/s rate.

Serial Interface Engine

[4]This block is divided into 2 sub-blocks; the SIE Control Logic and the Endpoint logic.
The SIE Control Logic contains the USB PID and address recognition logic, and other sequencing and state machine logic to handle USB packets and transactions. The Endpoint Logic contains the endpoint number recognition, FIFOs and FIFO control, etc.

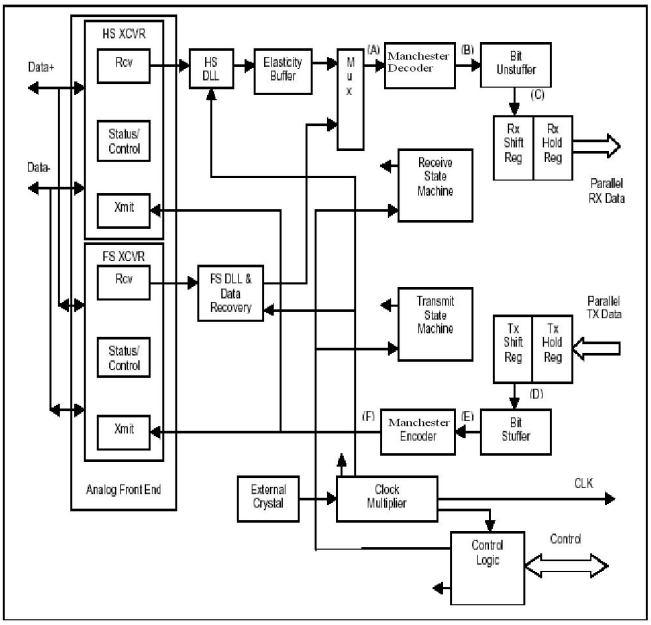
Device Specific Logic

This is the stage where USB connects to the application device.

USB TRANSCEIVER MACROCELL INTERFACE

Key features of the USB 2.0 Transceiver

[2]It Supports 480 Mbit/s "High Speed" (HS)/ 12 Mbit/s,Parallelinterface of 8 bits Bit-stuffing/unstuffing,bit stuff error detection Holding registers to stage transmit and receive data Logic to facilitate Wake Up and Suspend detection.



Functional Block Diagram

Fig 2: UTMI Functional Block Diagram

ISSN 2278-3091
International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 47-51 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

USB Interface Signals

Table1: Signals of Interfacing

Name	Direction	Active level	Description
DP	Bidirectional	N/A	USB datapin data+
DN	Bidirectional	N/A	USB datapin data-

[3]The data transmission and reception occurs from D+ and D- lines. If bit 1 is present on DP line, bit 0 will present on DM line in the same clock duration in order to avoid magnetic field interference and hence noise gets eliminated.

Operational Modes

There are 3 test modes:

- Normal Operation (0)
- Non-Driving (1)
- Disable Bit Stuffing and Manchester encoding (2)

In Mode 0 USB operates in normal encoding and decoding mode. Mode 1 allows the transceiver logic to tri-states both the HS and FS transmitters that they have been disconnected from the bus. Mode 2 disables Bit Stuff and Manchester encoding logic so 1's loaded from the Data In bus becomes 'J's on the DP/DM lines and 0's become 'K's..

Table 2: Mode of Operation

Operation mode	i/p	N/A	Operational Mode.		
[0-1]			[1]	[0]	Description
			0	0	0:Normal Operation
			0	1	1:Non-driving
			1	0	2:Disable bit stuffing and Manchester encoding
			1	1	3:Reserved

Bi-directional 8-bit Interface

[3]8 data lines will be presented by the transceiver, where Data 0-7 is a bi- directional data bus.If TX Valid is asserted (1), then the signals Data0-7 accepts transmit data from the SIE. If TX Valid is negated (0) then the signals.Data 0-7 present received data to the SIE.

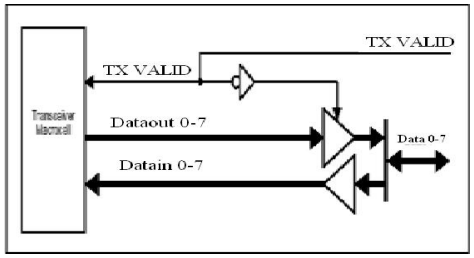


Fig 3: Bit Bi-directional Data Bus Interface

DESIGN OF HARDWARE MODEL

The UTMI is divided into two modules,which are the transmitter module and the receiver module.
The SYNC pattern 01111110 has to be transmitted immediately after the transmitter is initiated by the SIE.

The Transmitter module Specifications

After six consecutive 1's occur in the data stream, a zero to be inserted. The data should be encoded using Manchester encoding technique.A logical "1" is defined as a mid-point transition from low to high and a "0" is a mid-point transition from high to low.The EOP pattern (D+ and D- lines are carrying zero for two clock cycles) and a bit 1 have to be transmitted after each packet or after SIE suspends the transmitter. The Reset signal forces the state machine into the Reset state which negates TX Ready.When Reset is negated, the transmit state machine will enter the TX Wait state.

[7]In the TX Wait state, the transmit state machine looks for the assertion of TX Valid. When TX Valid is detected, the state machine will enter the Send SYNC state and begin transmission of the SYNC pattern.When the transmitter is ready for the first byte of the packet (PID), it will enter the TX Data Load state, assert TX Ready and load the TX Holding Register. The state machine may enter the TX Data Wait state while the SYNC pattern transmission is completed.

TXReady signal is used to transmit data. The state machine will remain in the TX Data Wait state until the TX Data Holding register is available for more data. In the TX Data Load state, the state machine loads the Transmit Holding register. [8]The state machine will remain in the TX Data Load state as long as the transmit state machine can empty the TX Holding Register before the next rising edge of CLK.

When TXValid is negated the transmit state machine enters the Send EOP state where it sends the EOP. While the EOP is being transmitted TXReady is negated and the state machine will remain in the Send EOP state.

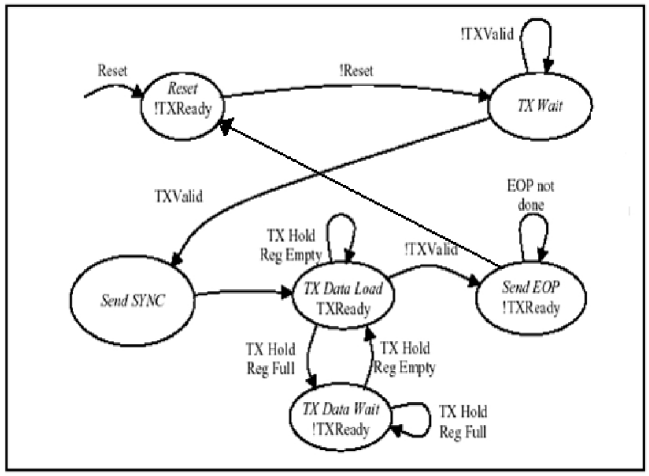


Fig 4: Transmit state machine

The Transmitter Module Design

[5]VHDL is used to design the transmitter module. A transmit state machine is developed by considering all the states given by USB 2.0 transmit state machine. Initially

ISSN 2278-3091
International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 47-51 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

the transmitter is at Reset state where the reset signal is high. If reset signal goes low state the state of the transmitter is changed to TX wait state where it is waiting for assertion of TX valid signal by the SIE.

SYNC generator

When Tx valid signal is enabled, sync enable is asserted. This signal is checked at every rising edge, a sync pattern "01111110" is send to the Manchester encoder.

Transmit hold and shift register

[5]The data byte placed on the data lines by the SIE sampled by the UTMI at the rising edge of the clock. For this purpose, an 8-bit vector is declared in the entity declaration of the transmitter module. This 8-bit vector is considered as transmit hold and shift register.The transmit hold and shift register is loaded with 8-bit parallel data from SIE at the rising edge of the clock. At this movement the transmit state machine is in Data load state. After the register is loaded, the data is sent to the other modules serially. Each bit of the register is sent to the Bit stuff module.After all the bits are serially sent to the Bit stuff module, Tx ready signal is asserted by the transmit state machine. During parallel to serial conversion data, the transmit state machine is in Data wait state.

EOP Generator

When TX valid signal is negated by the SIE, the transmit state machine enters into send EOP state where it enables a signal called eop_enable. This signal is checked out side the state machine for every clock. If this signal is high then the EOP pattern: two single ended zeros (i.e, DP, DM lines contain zeros) and a J (i.e, a 1 on DP line and a 0 on DM line) is transmitted on to DP, DM lines.

The Receiver Module Specification

The receiver module has been implemented by considering the fallowing 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.

The assertion of Reset will force the Receive State Machine into the Reset state. The Reset state negates RXActive and RXValid. When the Reset signal is negated the Receive State Machine enters the RX Wait state and starts looking for a SYNC pattern on the USB. When a SYNC pattern is detected the state machine will enter the Strip SYNC state and assert RXActive.

After 8 bits of valid serial data is received the state machine enters the RX Data state, where the data is loaded into the RX Holding Register on the rising edge of CLK and RXValid is asserted.
Stuffed bits are stripped from the data stream. Each time 8 stuffed bits are accumulated the state machine will enter the RX Data Wait state, negating RXValid thus skipping a byte time.When the EOP is detected, the state machine will enter the Strip EOP state and negate RXActive and RXValid. After the EOP has been stripped the Receive State Machine will reenter the RX Wait state and begin looking for the next packet.

If a Receive Error is detected, the Error State is entered and RXError is asserted.

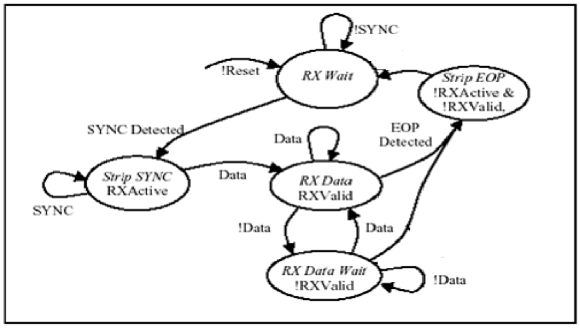


Fig 5: Receive state machine

The Receiver Design

The receiver module is designed by considering all the above specifications. VHDL is used to design the receiver module. The receiver module of the UTMI consists of various blocks such as SYNC detector, NRZI decoder, bit unstuffer, receive shift and hold Register and EOP detector.

A receive state machine is developed by considering all the states given by USB 2.0 receive state machine. Initially the receiver is at Reset state where the reset signal is high and RX active and RX valid signals are low. If reset signal goes low the state of the receiver is changed to RX wait state where it is waiting for SYNC pattern.

SYNC Detector

[6]When the receiver detects encoded SYNC pattern "01010100", the Receive state machine will enter into strip sync state where the SYNC pattern is stripped off. To detect the SYNC pattern a state machine is developed. It checks every bit for every rising edge of the clock. If the pattern is detected, a signal called sync detected is enabled. This signal is checked by the Receive state machine. If the signal is high, the Receive state machine will enter into strip sync state where RX active signal is asserted and the state machine will enter into RX data state.

Manchester Decoder

The received data on DP, DM lines are decoded. The Decoder simply XOR the present bit with the provisionally received bit. During Manchester decoding, the receive state machine is in RX wait state.

Receive Shift and Hold Register

[6]The serial data received from the bit Unstuffed is shifted into the receive shift register. After the shift Register is full, it is held there for one clock duration and then the data is placed on to the data out bus. This 8-bit data is sampled by the SIE at the next rising edge of the clock during shifting, the receive state machine is in RX data wait state. During holding, the receive state machine is in

ISSN 2278-3091
International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 47-51 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

RX data state where it asserts RX valid signal.

EOP Detector

[6]A state machine is developed for EOP detection, which is invoked at every rising edge of the clock. When two single ended zeroes followed by a „J state is detected, it asserts a signal called eop_detect which is checked by the Receive state machine at every rising edge of the clock. When this signal is high, the receive state machine will enter in to Strip eop state where the EOP pattern is stripped off and RX active, RX valid signals are negated. At the next rising edge of the clock. The Receive state machine will enter into the RX wait state.

RESULTS

The results obtained when executed ob FPGA Spartan kit, hat the bits first when converted from parallel to serial such that data transmission is easier.

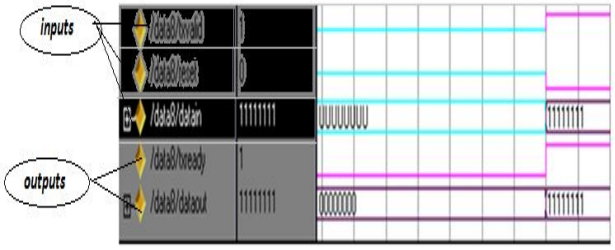


Fig 6: Parallel to Serial converter

After transmission of data, bit by bit in order to encode the data, using Manchester encoding technique the bits are stuffed with some extra bits and then transmitted**.**

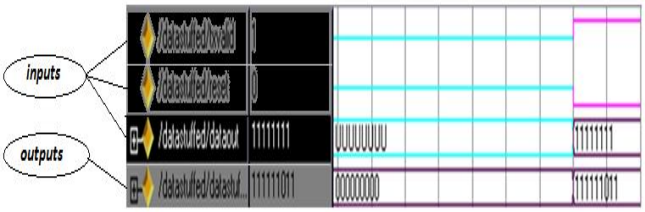


Fig 7: Bit stuffing

During decoding state ,the extra bits added should be removed and then to be received by the destination**.**

Fig 8: Bit UnStuffing

Then the serial bits are converted into parallel bits and received so that bits are transmitted in a short interval.

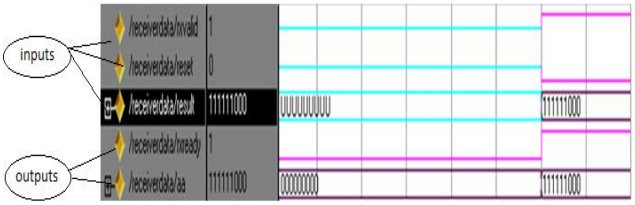


Fig 9: Serial to Parallel converter

UTMI can be used in 2 modes ,they are HS and FS modes.The results obtained when used in 2 different modes is as follows.

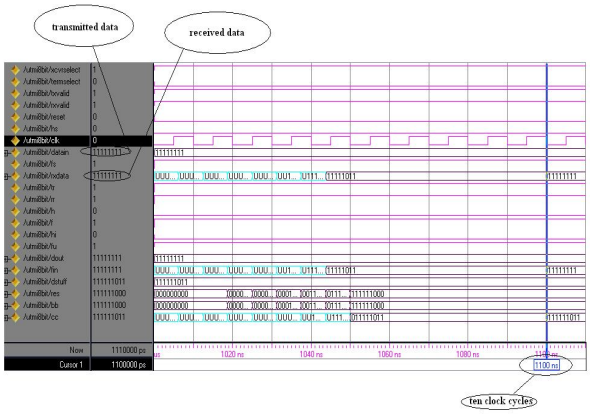


Fig 10: 8 bit UTMI in FS MODE

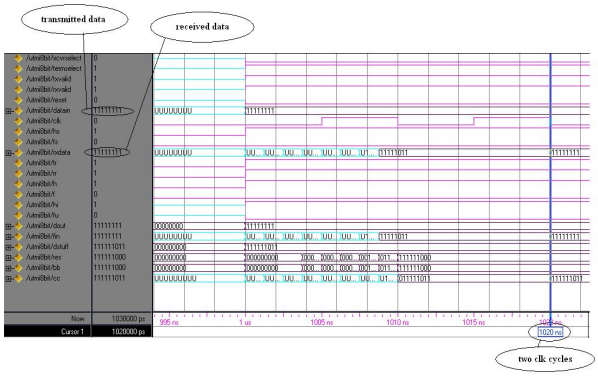


Fig 11: 8-bit UTMI in HS Mode

CONCLUSION

The individual modules of UTMI have been designed, verified functionally using VHDL simulator.The UTMI Transmitter is capable of converting parallel data into serial bits, performing bit stuffing and Manchester encoding.

The UTMI Receiver is capable of performing Manchester 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→xc2s30, package→cs144 and speed grade→ - 5.Thedevice belongs to the Vertex-E group of FPGAs from Xilinx.

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 47-51 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

REFERENCES

[1] USB 2.0 Specification, April 27, 2000
[2] USB 2.0 Transceiver Macrocell Interface (UTMI) Specification, version 1.05
[3] On-The-Go Supplement to the USB 2.0 Specification, revision 1.0, Dec 18, 2001
[4] UTMI+ Specification, revision 0.9, February 21, 2003
[5] Data and Computer Communications by William Stallings
[6] Computer Networks by Andre S.Tannenbaum
[7] www.opencore.org
[8] www.digitalcoredesign.org