PROJECT REPORT

DIGITAL SYSTEM ARCHITECTURE-2009

Submitted By:

Dorav Kumar      200601206
Jayesh Gumber  200601164
INDEX

<table>
<thead>
<tr>
<th>Section</th>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Acknowledgement</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Project Problem Statement</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Introduction</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>• RS 485</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>• Max 3483</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>• Single Twisted Pair</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>• Spartan 3E</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>Working and Concept</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Protocol</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>• Frame Description</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>• Frame Table</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>• System Diagram</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>• Slave</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>• Master</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>• Start Bit Detector</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>• Receiver</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>• Transmitter</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>Verification Strategy</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>RTL Schematics</td>
<td>14</td>
</tr>
<tr>
<td>7</td>
<td>Post Place and Route Simulation</td>
<td>16</td>
</tr>
<tr>
<td>8</td>
<td>Synthesis Report and Timing Analysis</td>
<td>17</td>
</tr>
<tr>
<td>9</td>
<td>References</td>
<td>23</td>
</tr>
</tbody>
</table>
ACKNOWLEDGEMENT

This Project could not have been completed without Dr. Rahul Dubey who not only is our lecturer in this course but also our mentor for the Project, he guided and encouraged us throughout the Project. He and TAs of the course, Subhash Roy and Umesh Lad, patiently helped us through the project, never accepting less than our best efforts. We thank them all.

19\textsuperscript{th} April, 2009
PROJECT PROBLEM STATEMENT

Create a master-slave RS-485 based 2 node multi-drop network using HDL and Maxim chip MAX 3483. The UART port connects to the MAX 3483 chip for each node. Please see. One node is programmed to become a master on the network and broadcasts messages to the two slaves. The addressed slave either consumes data and displays it on its LED or transmits back requested information, which the master displays on its LEDs.

INTRODUCTION

The project involves implementation of RS-485 network link, communication between Spartan – 3E FPGA kits one acting as Master and other as slaves is being carried based on RS 485 link. A protocol is developed for transmission of data which can be closely related to MODBUS protocol. Data is carried over a Twisted Wire connected to MAX 3483E chips which in turn are connected to the Spartan - 3E FPGAs.

RS – 485 :

RS-485 is a variation on RS-422 used for low-cost networking and is commonly used in many industrial applications. It is one of the simplest and easiest networks to implement[1]. It allows multiple systems (nodes) to exchange data over a single twisted pair. RS-485 is based on a master/slave architecture. All transactions are initiated by the master, and a slave will transmit only when specifically instructed to do so.

The interface to the RS-485 network is provided by a transceiver, such as a Maxim MAX3483
MAX 3483E:

MAX 3483 is a low-power transceiver for RS-485 and RS-422 communications. The device contains one driver and one receiver. The MAX348 feature slew-rate-limited drivers that minimize EMI and reduce reflections caused by improperly terminated cables, allowing error-free data transmission at data rates up to 250kbps[2].

![MAX 3483E Schematic Diagram](image)

**Single Twisted Pair:**

Twisted pair cabling is a form of wiring in which two conductors (the forward and return conductors of a single circuit) are twisted together for the purpose of canceling out electromagnetic interference (EMI) from external sources.

![Twisted Pair Image](image)

**Spartan 3E:**

Spartan 3E is University Trainer kit which is designed to execute many electronic applications based on Field-Programmable Gate Arrays (FPGA technology)[3].

![Spartan 3E Image](image)
WORKING AND CONCEPT

PROTOCOL

In the project Master and Slaves have to communicate with each other over RS 485, so there has to be a protocol that they have to follow in order to detect when to start receiving the signal, to recognize from where the signal is coming from, what function a device has to perform on receiving the signal from a particular device and on what data the device has to perform the function, as well as to recognize any error in the signal received and finally when to stop receiving the signal sent by the device. To solve all these issues and to attain proper synchronization a protocol needs to be defined. In this project, a protocol is made which is like the MODBUS protocol which helps the Master and Slaves to communicate over RS 485.

A 24 bit Frame is defined with the following configuration.

![Frame Diagram](image)

FRAME DESCRIPTION[4]

- **Start bits[0:3]** On detecting these 4 bits the device will come to know a valid frame has arrived and device has to accept and work on the frame. Value for a proper start bit are 4'b1111.
- **Error Alert bit[4]** If there was a error in parity or error due to any other reason than this bit can be set to 1 and sent to master or slave informing about the error.
- **Recognition bit[5]** This bit is to tell the master if the bit is to read by master or not.
  - If 1 is set: Master has to accept the bit.
  - If 0 is set: Master can ignore the bit.
- **Address bits[6:7]** These bits are used for addressing the frame to a particular device.
- **Function bits[8:11]** These bits are used to tell the device what function to perform with the data.
- **Data bits[11:18]** These bits are used to send the data on which a function needs to be performed or the data which has been generated by slave.
- **Parity bit[19]** This is the parity check bit. Even parity has been followed in the project.
- **Stop bits[20:23]** On detecting these bits the device detects the completion of frame.
<table>
<thead>
<tr>
<th>Bit Number</th>
<th>Protocol/function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Start Bit&lt;3&gt;</td>
</tr>
<tr>
<td>1</td>
<td>Start Bit&lt;2&gt;</td>
</tr>
<tr>
<td>2</td>
<td>Start Bit&lt;1&gt;</td>
</tr>
<tr>
<td>3</td>
<td>Start Bit&lt;0&gt;</td>
</tr>
<tr>
<td>4</td>
<td>Error Alert Bit&lt;0&gt;</td>
</tr>
<tr>
<td>5</td>
<td>Master/Slave Detection Bit&lt;0&gt;</td>
</tr>
<tr>
<td>6</td>
<td>Slave Address Bit&lt;1&gt;</td>
</tr>
<tr>
<td>7</td>
<td>Slave Address Bit&lt;0&gt;</td>
</tr>
<tr>
<td>8</td>
<td>Function Bit&lt;2&gt;</td>
</tr>
<tr>
<td>9</td>
<td>Function Bit&lt;1&gt;</td>
</tr>
<tr>
<td>10</td>
<td>Function Bit&lt;0&gt;</td>
</tr>
<tr>
<td>11</td>
<td>Data Bit&lt;7&gt;</td>
</tr>
<tr>
<td>12</td>
<td>Data Bit&lt;6&gt;</td>
</tr>
<tr>
<td>13</td>
<td>Data Bit&lt;5&gt;</td>
</tr>
<tr>
<td>14</td>
<td>Data Bit&lt;4&gt;</td>
</tr>
<tr>
<td>15</td>
<td>Data Bit&lt;3&gt;</td>
</tr>
<tr>
<td>16</td>
<td>Data Bit&lt;2&gt;</td>
</tr>
<tr>
<td>17</td>
<td>Data Bit&lt;1&gt;</td>
</tr>
<tr>
<td>18</td>
<td>Data Bit&lt;0&gt;</td>
</tr>
<tr>
<td>19</td>
<td>Parity Bit&lt;0&gt;</td>
</tr>
<tr>
<td>20</td>
<td>Stop Bit&lt;3&gt;</td>
</tr>
<tr>
<td>21</td>
<td>Stop Bit&lt;2&gt;</td>
</tr>
<tr>
<td>22</td>
<td>Stop Bit&lt;1&gt;</td>
</tr>
<tr>
<td>23</td>
<td>Stop Bit&lt;0&gt;</td>
</tr>
</tbody>
</table>
SYSTEM DIAGRAM

- RS 485 IMPLEMENTATION USING MAX 3843E

The above two diagram show the working of the project overall frame is sent from MASTER GPIOs where they are transmitted to slave over RS 485 to the SLAVE GPIOs and vice versa.
SLAVE

- STATE DIAGRAM

EXPLANATION: Following is the explanation of working of the Slave Device i.e. how it receives the frame sent by the master, validates it and corresponds accordingly. Firstly the Slave is always in receiving mode and is always detecting the start bits, once start bits are detected through Start Bit Detector, the Slave then receives the data frame through Receiver, now it checks the address bits if the address sent in the frame is the address of this Slave then it moves to next state, and if the address is not matching then Slave again moves to its Start Bit Detection State, if the parity of the frame is not matched then the receiver goes to the Frame Error State where it tells the transmitter to transmit the same data frame which it received in its buffer but just changing the value of the Recognition bit to 1 as well as Error Alert bit is also set to 1 so that master could send the correct frame again. If there is no parity error than device moves to function state where it checks the function that has to performed on the 8 bit data sent. After performing the function, Slave tells the Transmitter to transmit the new calculated data value along with the function address and the device address to the master. After the transmission is complete it again moves to the Start Detected State.

MASTER

- STATE DIAGRAM
- **EXPLANATION:** Following is the explanation of the working of the Master Device i.e. how it sends commands to the slaves and how it reciprocates to the commands received from the slaves. The initial state of the Master Device is always receive state and its Start Bit Detector continuously checks for any start bits coming, if the master wishes to send Frame to the slave (here achieved through switches) then it goes into Command state where it tells the Transmitter to send a pre-stored frame onto the Tx port, and thus comes back to receive state. Now if the Start Bit Detector detects 4 start bits i.e. 1111 then the Master device goes into receive state where its Receiver stores the data frame, master then goes into the master recognition state where it checks the Master Detection Bit, if it is 0 then the master discards the frame and goes back into receive state, but the bit is 1 then the master checks for the error alert bit, if this bit is high then the master resends the data frame for which it received this particular frame if the error bit is not high then master just goes into responds state here which is defined as glowing of LEDs on Master.

**START BIT DETECTOR**

- **STATE DIAGRAM**

- **EXPLANATION:** Every Frame will only be received by the devices if the first four bits of the frame sent are 1 1 1 1. A mealy machine is implemented to detect these four bits. When the first high bit is detected, the machine moves from state S0 to state S1, only on detecting another 1 will the machine move from state S1 to state S2, if state S1 detects a zero next then it moves to state S0. Similarly if state S2 receives a bit 1 then only machine will move to state S3 if it zero is received at state S2 machine will move to state S0. Now if machine at state S3 receives another one than its output 1 (which implies that the device can go to next state of Receiving) and it goes to S0 and if it receives a zero than then the output is 0 and machine goes back to state 1.
**RECEIVER**

- STATE DIAGRAM

  On Rx Enable
  
  20 Bit stored in Data Buffer
  
  Parity Check
  
  error
  
  Error Response State
  
  No error
  
  Address Check State

  EXPLANATION: On receiving an output of 1 from the mealy state machine of Start Detection, receiving State is enabled. In this state rest of the twenty bits are stored in a data buffer of 20 bits wide and then for the first 15 bits parity is calculated. If the calculated parity matches the parity bit sent in the frame the device can move to the next state of address detection and if the parity bit is not checked then device can move to error response state accordingly.

**TRANSMITTER**

- STATE DIAGRAM

  On Tx Enable
  
  15 Bit Data Buffer is accepted
  
  Parity calculated & added to frame
  
  Start and Stop bits added
  
  24 Bits serially transmitted
  
  Transmission signal sent to device

  EXPLANATION: On getting a transmission enable signal from the device state machine this particular part of code makes the buffer of 24 bits the inputs of the buffer are provided by the device state machine only for example the address, function and data bit are provided by the device state machine and the transmitter calculates the parity of the 15 bit frame and adds 4 start bits and 4 stop bits to the frame and sends the data frame serially to the data transmission pin. After sending the data correctly it tells the device machine to go back to its initial state.
VERIFICATION STRATEGY

The Project and various modules were verified using the following two methodology:

- **Software**: Main verification of master and slave module were done in the Xilinx Simulator.
- **Hardware**: The waveforms were analyzed on the Tectronix Digital CRO after the Project was burned into the FPGA Spartan – 3E

Following are the results based on the above two mentioned methodologies:

Software based Master Simulation: Xilinx Simulator
IMPLEMENTATION OF RS 485

Software based Slave Simulation: Xilinx Simulator

Hardware based: Digital CRO
RTL SCHEMATIC

MASTER

```
switch_input(2:0)       out_LED(7:0)
clk
data_rx
master_enable
rst
data_tx
rx_enable_chip
tx_enable
```
SLAVE
POST PLACE AND ROUTE SIMULATION

The floor Plan of our design is as follows:
### HDL Synthesis Report

**Macro Statistics**

<table>
<thead>
<tr>
<th>Component</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td># Adders/Subtractors</td>
<td>2</td>
</tr>
<tr>
<td>32-bit subtractor</td>
<td>2</td>
</tr>
<tr>
<td># Counters</td>
<td>2</td>
</tr>
<tr>
<td>32-bit up counter</td>
<td>2</td>
</tr>
<tr>
<td># Registers</td>
<td>36</td>
</tr>
<tr>
<td>1-bit register</td>
<td>31</td>
</tr>
<tr>
<td>15-bit register</td>
<td>2</td>
</tr>
<tr>
<td>32-bit register</td>
<td>2</td>
</tr>
<tr>
<td>8-bit register</td>
<td>1</td>
</tr>
<tr>
<td># Comparators</td>
<td>3</td>
</tr>
<tr>
<td>33-bit comparator greataeval</td>
<td>1</td>
</tr>
<tr>
<td>33-bit comparator less</td>
<td>2</td>
</tr>
<tr>
<td># Multiplexers</td>
<td>1</td>
</tr>
<tr>
<td>1-bit 24-to-1 multiplexer</td>
<td>1</td>
</tr>
<tr>
<td># Xors</td>
<td>2</td>
</tr>
<tr>
<td>1-bit xor15</td>
<td>1</td>
</tr>
<tr>
<td>1-bit xor16</td>
<td>1</td>
</tr>
</tbody>
</table>

**Final Report**

**Final Results**

- **RTL Top Level Output File Name**: master_top.ngr
- **Top Level Output File Name**: master_to
- **Output Format**: NGC
- **Optimization Goal**: Speed
- **Keep Hierarchy**: NO

**Design Statistics**

<table>
<thead>
<tr>
<th>Component</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td># IOs</td>
<td>18</td>
</tr>
</tbody>
</table>

**Cell Usage**

<table>
<thead>
<tr>
<th>Component</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td># BELS</td>
<td>633</td>
</tr>
<tr>
<td># GND</td>
<td>1</td>
</tr>
<tr>
<td># INV</td>
<td>67</td>
</tr>
<tr>
<td># LUT1</td>
<td>64</td>
</tr>
<tr>
<td># LUT2</td>
<td>41</td>
</tr>
<tr>
<td># LUT2_D</td>
<td>1</td>
</tr>
<tr>
<td># LUT2_L</td>
<td>2</td>
</tr>
<tr>
<td># LUT3</td>
<td>55</td>
</tr>
</tbody>
</table>
Device utilization summary:

Selected Device: 3s500efg320-4

Number of Slices: 183 out of 4656 3%
Number of Slice Flip Flops: 194 out of 9312 2%
Number of 4 input LUTs: 342 out of 9312 3%
Number of IOs: 18
Number of bonded IOBs: 18 out of 232 7%
Number of GCLKs: 2 out of 24 8%

Timing Report

TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT GENERATED AFTER PLACE-and-ROUTE.

Clock Information:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock Signal</td>
<td>Clock buffer(FF name)</td>
<td>Load</td>
</tr>
<tr>
<td>clk</td>
<td>BUFGP</td>
<td>58</td>
</tr>
<tr>
<td>baud_clk1</td>
<td>BUFG</td>
<td>136</td>
</tr>
</tbody>
</table>

Timing Summary:

Speed Grade: -4

Minimum period: 6.795ns (Maximum Frequency: 147.167MHz)
Minimum input arrival time before clock: 6.619ns
Maximum output required time after clock: 5.407ns
Maximum combinational path delay: No path found

Timing Detail:

(All values displayed in nanoseconds (ns))

Timing constraint: Default period analysis for Clock 'clk'
Clock period: 5.240ns (frequency: 190.857MHz)
Total number of paths / destination ports: 1689 / 103

Delay: 5.240ns (Levels of Logic = 9)
Source: baud_delay_8 (FF)
Destination: baud_delay_0 (FF)
Source Clock: clk rising
Destination Clock: clk rising
Data Path: baud_delay_8 to baud_delay_0

Total 4.283ns (3.863ns logic, 0.420ns route)

(90.2% logic, 9.8% route)
HDL Synthesis Report

Macro Statistics
# Adders/Subtractors : 3
32-bit subtractor : 2
4-bit adder carry out : 1
# Counters : 2
32-bit up counter : 2
# Registers : 73
1-bit register : 68
15-bit register : 1
32-bit register : 2
8-bit register : 2
# Comparators : 3
33-bit comparator greatequal : 1
33-bit comparator less : 2
# Multiplexers : 1
1-bit 24-to-1 multiplexer : 1
# Xors : 2
1-bit xor15 : 1
1-bit xor16 : 1

Final Results
RTL Top Level Output File Name : slave_top.ngr
Top Level Output File Name : slave_top
Output Format : NGC
Optimization Goal : Speed
Keep Hierarchy : NO

Design Statistics
# IOs : 15

Cell Usage :
# BELS : 696
# GND : 1
# INV : 70
# LUT1 : 64
# LUT2 : 67
# LUT2_D : 1
# LUT2_L : 1
# LUT3 : 47
# LUT3_D : 3
# LUT3_L : 2
# LUT4 : 129
IMPLEMENTATION OF RS 485

# LUT4_D : 5
# LUT4_L : 10
# MUXCY : 156
# MUXF5 : 12
# MUXF6 : 1
# VCC : 1
# XORCY : 126
# FlipFlops/Latches : 228
# FDE : 48
# FDR : 32
# FDRE : 132
# FDSE : 16
# Clock Buffers : 2
# BUFG : 1
# BUFGP : 1
# IO Buffers : 14
# IBUF : 3
# OBUF : 11

Device utilization summary:

Selected Device : 3s500efg320-4

Number of Slices: 213 out of 4656 4%
Number of Slice Flip Flops: 228 out of 9312 2%
Number of 4 input LUTs: 399 out of 9312 4%
Number of IOs: 15
Number of bonded IOBs: 15 out of 232 6%
Number of GCLKs: 2 out of 24 8%

Timing report

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
GENERATED AFTER PLACE-and-ROUTE.

Clock Information:

Clock Signal | Clock buffer(FF name) | Load |
-------------|-----------------------|------|
clk          | BUFGP                | 123  |
baud_clk1    | BUFG                 | 105  |

Asynchronous Control Signals Information:

No asynchronous control signals found in this design
Timing Summary:

Speed Grade: -4

Minimum period: 7.762ns (Maximum Frequency: 128.833MHz)
Minimum input arrival time before clock: 5.965ns
Maximum output required time after clock: 5.407ns
Maximum combinational path delay: No path found

Timing Detail:

All values displayed in nanoseconds (ns)

---

Timing constraint: Default period analysis for Clock 'clk'
Clock period: 7.762ns (frequency: 128.833MHz)
Total number of paths / destination ports: 4519 / 251

---

Delay: 7.762ns (Levels of Logic = 12)
Source: func_delay_8 (FF)
Destination: next_state_FFd5 (FF)
Source Clock: clk rising
Destination Clock: clk rising

REFERENCES
[2]. Max3483E – Data Sheet by Maxim.
[3]. Spartan 3E starter kit board user guide.