Design and Development of JTAG for Trace and Debug of Controller and Implementation on FPGA

Mr. Manjunath T.N, Sunil T.D, Dr. M. Z Kurian, Imran Rasheed

Abstract— **The paper aims about the trace and debug of any N-bit controller using a JTAG, here controller core is traced using a interfacing device known as JTAG (Joint Test Action Group). JTAG is an advanced DFT Technique for the purpose of testing an ASIC, as such there are various technique for this purpose, but JTAG is chosen for its unique feature of in built state machine which can used for the purpose of both interfacing and display unit with a device as well as a testing device.**

Keyword: JTAG ,DFT, Testing

I. Introduction

The integrated circuits become more complex with more functions and higher performances, but it also becomes more difficult to test the complex chips, so DFT (Design For Testability) circuits must be added to the chips during the design stage to reduce test costs.

This architecture defines the hardware and software for debugging and testing . It's a milestone event for DFT technology, and this architecture is approved as IEEE 1149.1 protocol by IEEE organization.

One of the most popular enhancement is on-chip debug by adding some JTAG instructions and special logics by analyzing the design architecture and DFT schemes, one functional enhancement methodology to standard IEEE 1149.1 JTAG controller will be proposed in this paper. Besides traditional boundary scan tests, this enhanced JTAG controller can also control internal scan tests especially provide the on-chip real time debug feature to facilitate software development.

While going through this process the time taken to test and verify the design shall be reduced at least by 20% thus reducing the design cost when compared to the traditional methods of testing and debugging

This paper is organized as follows. Section II presents the Related work. Section III Methodology. Section IV JTAG/DEBUG Interface. Section V System Design VI System Implementation VII Advantages VIII Application X conclude/future work of the paper.

II. RELATED WORK

In this paper designing and developing the architecture for debug of controller. Then finally the JTAG architecture shall be developed which follows IEEE standard 1149.1, once these architecture are developed, interfacing will done between the JTAG and the controller. This entire setup shall be implemented and prototyped on FPGA. In the current paper the enhancement on the JTAG feature is included with optimal speed and frequency. Designing of JTAG interface with the core and matching the baud frequency is Partially a challenging task. Hence we try to increase the speed and performance compared to the previous version of JTAG interface. Solution Building the right interface with the JTAG controller and prototype on FPGA.This all flow and setup is built in the system on FPGA.

.

III. METHODOLOGY

The architecture is designed based on the requirements and functionality of the system. For the architecture designed, code was written using Verilog HDL. Then this code was simulated using Xilinx Modelsim 6.3c Simulator and verified its functionality. After that synthesis, placement, routing and bit stream (FPGA physical programming information) generation was done using Xilinx ISE 13.1 design tool suite. Then the generated bit stream file is ported on to XC5VLX110T device of Xilinx Virtex-5 family prototyping board.

IV. JTAG/DEBUG INTERFACE

FIELD OF THE JTAG/DEBUG INTERFACE:

This interface is an improved method for using the JTAG port as a means of communicating with a CPU which includes a debug mode of operation .Debug mode is a means by which a development tool can control the operation of and enhance the debugging of a computer based system.

Manuscript received April, 2014.

***MANJUNATH T.N** is currently pursuing M.Tech VLSI & Embedded Systems in dept of E&C from Sri Siddhartha Institute of Technology, Tumkur, India.*

***SUNIL TD** is currently working as a assistant professor in dept of E&C at Sri Siddhartha Institute of Technology,. Tumkur, India.*

***IMRAN RASHEED** is currently working as a. Asst. Professor, Dept. of EEE, M.S.Ramaiah School of Advanced Studies, Bangalore, Karnataka, India*

***Dr.M.Z.Kurian**, is currently working as a Dean & HOD in dept of E&C at Sri Siddhartha Institute of Technology, Tumkur. India.*

BACKGROUND OF THE JTAG/DEBUG INTERFACE:

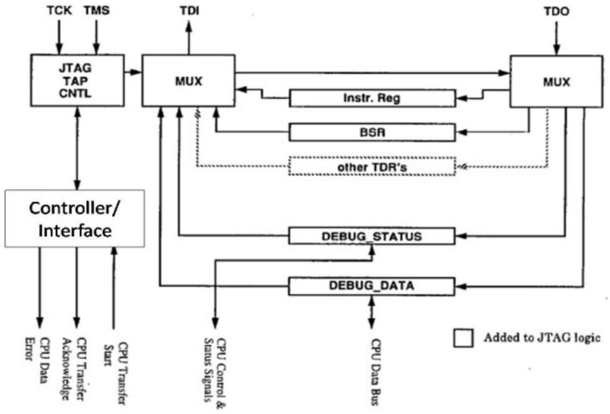
In order to debug increasingly complex microprocessors and microcomputers many of them have been designed with special debug modes of operation. This allows a development tool to control the operation of the processor without degrading the properties of signals going into and out of the processor. These debug modes have the following goals:

1. Provide non-intrusive debugging.
2. Minimal silicon area to implement a debug interface with the processor.
3. Minimal pin count for debug mode interface.
4. Fast data transfer rate for upload/download memory.
5. Allow the development tool to read and write memory and registers.
6. Start and stop execution from appropriate debug command

In the prior art, a debug mode was implemented on the MC68300, MC68HC16, MC56000 and the MC88304families. The debug mode interface on the 300 and HC16families has the disadvantage that it requires three additional pins on the package. The debug mode interface on the 56000 and 88304 families has the disadvantages that it requires five additional registers, uses 4K bytes of the processor memory map and provides only a half duplex serial interface.

V. SYSTEM DESIGN

Systems design is the process of defining the architecture, interfaces, components, modules data for the system to be satisfy specified requirements.



]

prior art by using the JTAG port to provide a means of communicating with a CPU in debug mode. Two additional test data registers are required in the JTAG logic together with CPU interface control logic. The first added JTAG test data register is called the DEBUG_DATA register. It receives serial data from the JTAG test data input and transfers it to the CPU following the JTAG Update-DR state. It is also loaded with data from the CPU at the JTAG Capture-DR states that data can be shifted out serially when the next data is shied in. This register is also part of the CPU register set but is available only when the CPU is in a special debug mode of operation. The second JTAG test data register provides the additional status and control needed with this implementation. This register is called the DEBUG STATUS register ,and will have the following bits:

1. Breakpoint request- A control bit which requests an unconditional breakpoint.
2. Ready-A status bit which indicates that the CPU is trying to read an instruction or data from the DEBUG_DATA test data register.
3. Freeze-A status bit which indicates that the CPU is in debug mode.
4. Debug mode enable- A control bit which enables the CPU's debug mode.

The CPU interface control logic and a communications protocol allow full duplex operation of the JTAG serial interface. The CPU interface decodes CPU accesses to the DEBUG_DATA register for instructions or data and sets the ready bit in the DEBUG-STATUS register .When the JTAG TAP controller enters the Update-DR state with the DEBUG_DATA register selected the interface provides an acknowledge handshake back to the CPU to indicate that data is valid and the read cycle can be terminated. It also provides an acknowledge to the CPU when the CPU writes to the DEBUG_DATA register.

The protocol requires that the DEBUG_DATA register not be accessed by the JTAG test access port (TAP)controller until the CPU is trying to read an instruction or data from the DEBUG_DATA register. This is defined by both the FREEZE and READY bits being asserted in the DEBUG_STATUS register. The protocol also requires that the CPU not perform more than one write to the DEBUG-DNA register between reads of the debug data register.

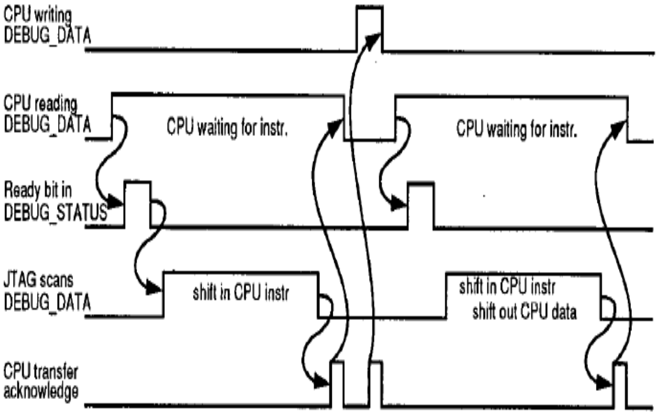


Fig 2: Timing diagram

The attached block diagram and timing diagram shows the interface between the JTAG and Debug logic. The timing diagram shows the sequence of activity in the debug logic. It does not show JTAG pin activity, Normal JTAG operation is assumed .First, the DEBUG_STATUS register is selected for scan by loading the appropriate instruction in the JTAG instruction register. Next, the DEBUG_MODE ENABLE bit and the BREAKPOINT_REQUEST bi tare set in the DEBUG_STATUS register by shifting in data from the JTAG test data in (TDI) pin. The DEBUG_STATUS register is then scanned out on the JTAG test data out (TDO) pin until the READY and FREEZE its are both set indicating that the CPU has entered debug mode and the CPU is trying to read an instruction from the DEBUG_DATA register.

Next, the DEBUG_DATA register is selected for scan by loading the appropriate instruction in the JTAG instruction register. A CPU instruction is then shift into the DEBUG_DATA register during the JTAG Shift-DR state. Upon entering the JTAG Update-DR state, all the bits in the DEBUG_DATA register are driven onto the CPU Data Bus. The CPU interface control logic sends a transfer acknowledge signal to the CPU .The CPU completes its read of the instruction on the bus, terminates the bus cycle and begins execution of the instruction. This instruction may result in a read of data horn or a write of data back to the DEBUG_DATA register. If the CPU writes data to the DEBUG_DATA register the READY bit is not set in the DEBUG_STATUS register. The READY bit is only set when the CPU starts the next read from the DEBUG_DATA register. If data was written into the DEBUG_DATA register by the CPU it is shied out through the TDO pin while the next instruction is being shifted in through the TDI pin during the next JTAG Shift-DR state.

The DEBUG-STATUS register is again selected for scan by loading the appropriate instruction in the JTAG instruction register. It is then scanned out on the JTAG test data out (TDO) pin until the READY bit is set indicating that the CPU is trying to read the data value or the next instruction from the DEBUG-DATA register .When the DEBUG-DATA register is next selected for scan the appropriate data or the next CPU instruction is shied into the DEBUG-DATA register during the JTAG Shift-Dr state. This sequence of accessing first the DEBUG-STATUS register to determine if the, READY bit is set and then the DEBUG-DATA register to transfer data or instructions to the processor continues until the CPU is given an instruction which causes it to exit debug mode. It then proceeds to fetch instructions from memory at the address indicated in its instruction register. When the CPU exists debug mode the FREEZE bit in the DEBUG-STATUS register is negated.

VI. SYSTEM IMPLEMENTATION

This chapter describes the implementation of the architecture system. For implementation Bottom-up methodology is followed where the basic building blocks required in implementing the design like Tap Controller, Instruction Register and the Boundary Scan register, Debug Status, Debug Data modules are implemented.

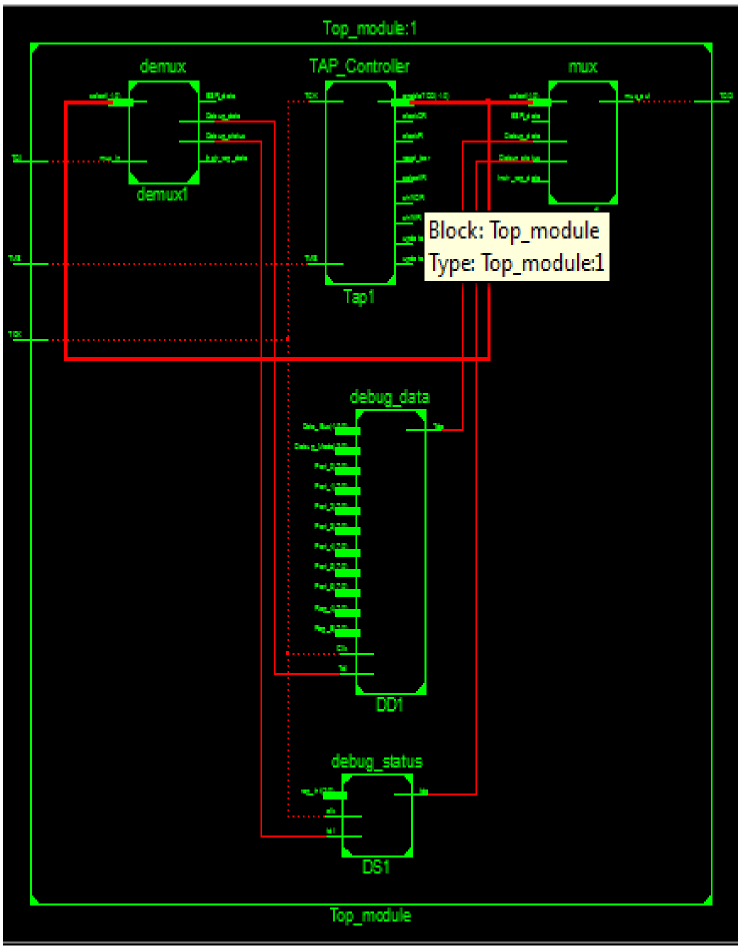
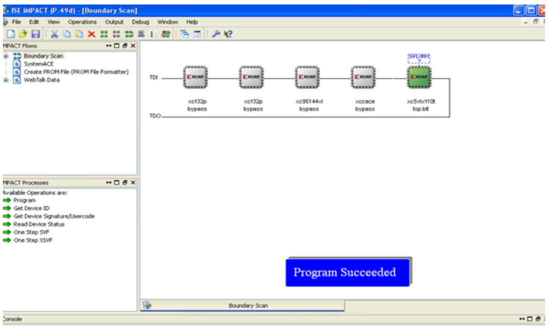
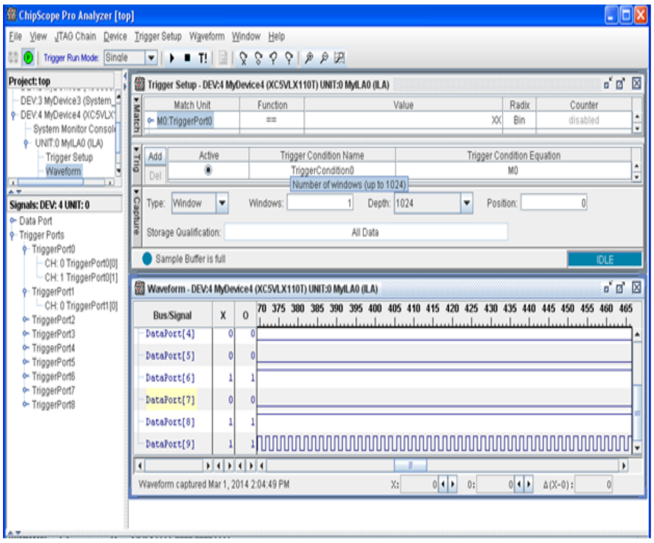


Fig 1: Top module of RTL Schematic

VII. RESULTS



Program Succeeded



Trigger Run Mode: Single

Trigger Setup - DEV:4 MyDevice4 (XC5VLX110T) UNIT:0 MyILA0 (ILA)

Match Unit	Function	Value	Radix	Counter
M0:TriggerPort0	==	XX	Bin	disabled

Active	Trigger Condition Name	Trigger Condition Equation
●	TriggerCondition0	M0

Number of windows (up to 1024)

Type: Window Windows: 1 Depth: 1024 Position: 0

Storage Qualification: All Data

Sample Buffer is full

Waveform - DEV:4 MyDevice4 (XC5VLX110T) UNIT:0 MyILA0 (ILA)

Bus/Signal	X	O
DataPort[4]	0	0
DataPort[5]	0	0
DataPort[6]	1	1
DataPort[7]	0	0
DataPort[8]	1	1
DataPort[9]	1	1

Waveform captured Mar 1, 2014 2:04:49 PM X: 0 O: 0 Δ(X-O): 0

VII. ADVANTAGES

- Interfacing is easier using daisy chain architecture for parallel interfacing of multiple block.
- Testing and debugging is easier due to inbuilt JTAG TAP controller.

VIII. APPLICATIONS

- Debugging ports in microprocessor
- Debugging ports in microcontroller
- Debugging ports in ARM
- Debugging ports for any N-bit controller

X. CONCLUSION

In Interface between the JTAG and Debug logic the standard IEEE 1149.1 JTAG controller is proposed in this project. With the enhanced features, all test functions including stuck-at scan, at-speed scan, memory BIST and high-speed physical layer tests can be control by the JTAG controller besides traditional boundary scan tests, and further on-chip debug features are also integrated in this enhanced JTAG controller. Therefore, the chip costs can be reduced, and the software development and debug can be facilitated with the enhanced JTAG controller. The proposed approach will have implemented on FPGA and optimized for area, speed and power.

ACKNOWLEDGMENT

We would like to express our sincere thanks to the anonymous reviewers for their critical suggestions which helped in improving the manuscript and also we express our gratitude to our respective faculty and our parents for supporting this work.

REFERENCES

[1]. Online design bug detection: RTL analysis, flexible mechanisms, and evaluation -Kypros Constantinidesz Onur Mutlux Todd Austinz .

[2]. Internal structure of software application for controlling devices via JTAG 1149 interface -Igor Ilyin , Rostislav Grushvitsky .

[3]. A vertically integrated test methodology based on JTAG IEEE 1149.1 standard interface .

[4]. Design and Implementation of a JTAG Boundary-Scan Interface Controller - Shen XuBang , LiangSongHai.

[5]. Design of IEEE1149.1 Testing Bus Controller IP Core- Zhu Min, Yang Chunling, PengliZhang.

[6]. A High Speed Reduced Pin Count JTAG Interface - Lee Whetsel .

[7] A overview of JTAG Technology: Development and Application, Microelectronics.

[8] Implementation of debug interface of DSP processor based on JTAG architecture, Computer Engineering.

[9] IEEE Std Pl14911 -2001 IEEE standard test access port and boundary2scan architecture [S], 2001

[10] Microprocessor Test and Test Tool Methodology for the 500MHz IBM S/390 G5 Chip.