PROJECT REPORT

COEN 6521: DESIGN FOR TESTABILITY

IMPLEMENTATION AND INCORPORATION OF JTAG WITH 8 - BIT ARRAY MULTIPLIER

Group: 12

Authors:
Avaneesh Gaddam
Student Id: 40127009

Prabhu Guna Deepak Pallikonda
Student Id: 40166765

Instructor:
Dr. Mahmoud Masadeh

Date:
05-08-2022

Abstract:
Technological trends have changed a lot during the course of years, which led to multiple adaptations in the VLSI technologies. People today are interested in using compact and efficient electronic devices such as laptops, electronic watches, etc. In today's design methodologies, we may find multiple compressed PCB layers with numerous components in Ball-Grid-Array packaging to create portable electronic devices. The complexity and density of circuit cards, which are widely used nowadays, present major problems in the field of test and inspection as complexity leads to assembly faults.

In the past, the In-Circuit Test (ICT) was used during electronics assembly to identify any defects that might have been created. However, with modern techniques that have concealed contacts and thousands of electrical nodes, boundary scan has emerged as a new option since ICT is not suitable for many electrical probes. Today, for testing during assemblies, we use JTAG or the IEEE 1149.1 boundary scan test standard. In this project, using VHDL and Modelsim, we have designed JTAG and performed interconnect testing on an 8-bit array multiplier. The area details of stand-alone 8-bit multiplier, stand-alone JTAG and JTAG with 8-bit multiplier are captured using 'Mentor Graphics Precision Software'.

Table of Contents:

List of Figures:

List of Tables :

1. INTRODUCTION

1.1 Motivation

Circuit cards with tens of thousands of devices, more than ten thousand electrical nodes, and 20 or more PCB layers are common in today's design methodologies. Unfortunately, as complexity rises, the likelihood of defects also rises. On the other hand, as the density has increased, the capacity to offer a sufficient number of test points to enable conventional test procedures including the In-Circuit Test (ICT), to validate the structural integrity of complex circuit cards. The widespread use of Ball-Grid-Array (BGA) packing makes the access problem more challenging. The IEEE 1149.1 boundary-scan standard has been widely used as a result of this lack of physical access.

This test method, also known as JTAG (Joint Test Action Group), which started work on the subsequent standard, resolves the physical access issue by offering extremely high test coverage of complex digital circuit cards over a simple 4-wire (5-wire if the optional asynchronous reset TRST signal is included) interface called the Test Access Port (TAP) [2]. This testing strategy has primarily been used for manufacturing tests as well as board-level in-system programming and configuration tests. However, a range of scan path management tools from different silicon vendors enables the extension of boundary-scan capabilities to the system environment. The system-level boundary-scan architecture permits field-based embedded boundary-scan testing as well as remote test, diagnostic, and setup [3].

1.2 Objective

This project was carried out to gain a better understanding of post-manufacturing circuit testing. Here, an 8 - bit array multiplier is considered and a boundary scan is performed by designing a JTAG combined logic circuit. A boundary-scan or IEEE 1149.1, compliant device consists of a number of fundamental parts, including a TAP, an instruction register, and at least two data registers, among other mandatory and optional structures. The boundary register, which is made up of boundary-scan cells that wrap around the edge of the device, is accessible from either end through the Test Data Input (TDI) and Test Data Output (TDO) pins of the TAP. This register provides serial access to each device pin for the purposes of driving signals, accumulating responses, or both in the case of bi-directional pins [1].

In the Boundary scan register, many boundary scan cells are arranged to form a chain by joining the TDO signal of one cell to the TDI signal of the following. The finite state machine, a component of the TAP Controller in charge of controlling the boundary-scan test logic, is managed by the Test Mode Select (TMS) and Test Clock (TCK) pins of the TAP. The boundary-scan logic is put into predetermined modes using the instruction register, and the data register is inserted between the TDI and TDO pins. This entire setup is designed in VHDL and is connected to an 8 - bit Array Multiplier.

1.3 Overview

In this project, our main goal is to incorporate JTAG IEEE 1149.1 (Boundary scan) within an 8- bit Array Multiplier to perform interconnect testing. First, we verified the functionality of the 8-bit Array Multiplier circuit using a testbench in the Multisim Tool. Next, we synthesized this stand-alone circuit using the Mentor Graphics precision tool to evaluate the area. Then, using Multisim software we designed the JTAG interface to test the 8-bit Array Multiplier circuit and synthesize the Multiplier circuit combined with JTAG to find the overhead in terms of area. Finally, the report concludes by stating the advantages and drawbacks of JTAG inclusion with the given 8-bit multiplier.

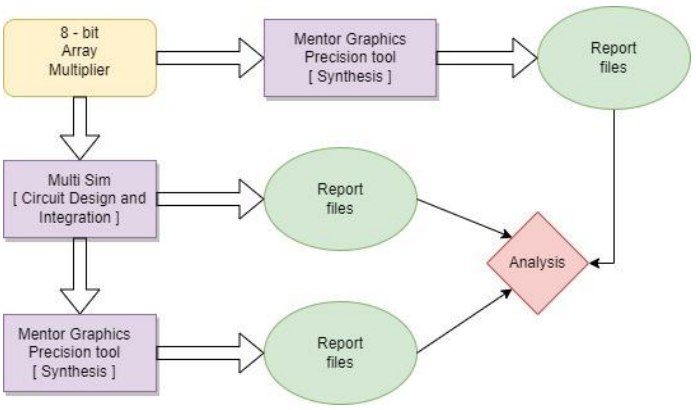

Fig 1.1 *Design flow*

2. DESCRIPTION OF 8-BIT ARRAY MULTIPLIER

An array multiplier is a combinational circuit that performs the multiplication of two binary numbers. It is a critical piece of hardware in the vast majority of digital and high-performance systems. A multiplier is necessary for numerous applications, including digital communication, digital filtering, and digital signal processing. It uses the add and shift method and is considered the most popular and straightforward multiplier. [4]

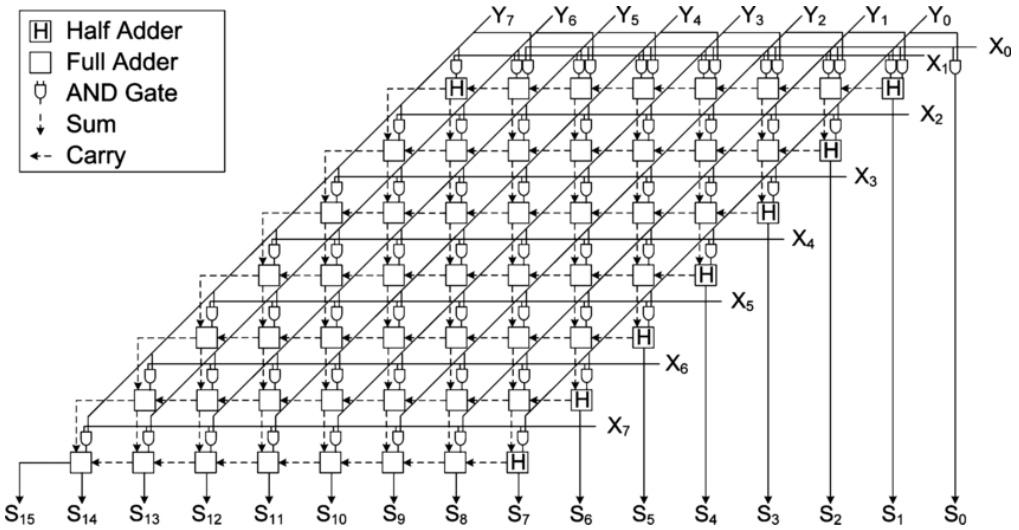

Fig 2.1 *Block diagram of 8-bit multiplier*

As shown in the figure, binary multiplication is performed by successive multiplying each element of one number with another binary number with their successive addition. This is performed with a number of half and full adder circuits employing add and shift mechanisms. In the case of our circuit, instead of taking half adders, the total circuit is designed by converting full adders to half adders in the case where necessary. The multiplier generates a partial product by multiplying the multiplicand by the multiplier, and the partial products are added to produce the final result. [5]

2.1. Multiplication Algorithm

The algorithm for the multiplication of two binary numbers is shown below:

A = Am-1 Am-2 .....A2 A1 A0
B = Bn-1 Bn-2 ..... B2 B1 B0

Here one binary number consists of m bits, while the other binary number has n bits. The way the partial products are generated differs between multiplier architectures. Multiplication of binary numbers can be broken down into additions. Consider multiplying two 8-bit numbers, A and B, to get the 16-bit product P. [6]

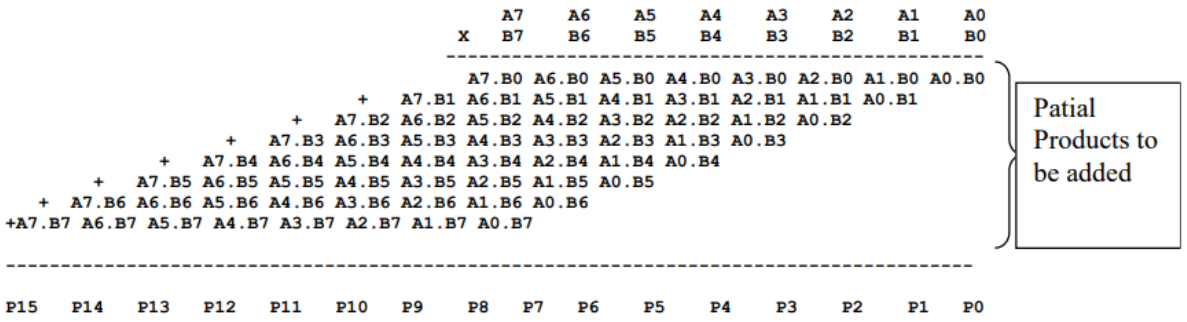

Fig 2.2. *Explanation of Multiplication process*

The equation for the partial products is: $P(m+n) = A(m)B(n) = \sum_{i=0}^{m-1}\sum_{i=0}^{n-1} a_i b_j 2^{i+j}$

3. TESTING THE VLSI SYSTEM (STRUCTURAL VS FUNCTIONAL)

Functional Testing:

- A functional test ensures that a design will perform its intended function. It determines whether or not the circuit is functionally correct, but it does not test the operation of its internal components. It is far more subjective than structural testing in that verifying the complete functionality of a complex logic circuit under all conceivable operating conditions is extremely difficult.

- It exercises the functions according to the specification and often requires designers' inputs.

- It requires a large number of test patterns with low fault coverage.

- It is difficult to be optimized for production tests concerning micro fabrications. [7]

Structural Testing: In this project, we have used structural testing

- It uses the information of interconnected components, which are combinational gates to perform a test regardless of the functions.

- Fault modeling is the key for structural testing.

- Some of its techniques include BIST, Boundary scan, AOI, AXI, DMM, etc.

- It is the base of present used testing framework in the industry such as ATPG, Fault simulator, DFT tools, etc. [7]

- A test strategy which is based on a structural approach, involving scan and BIST, clearly has reduced requirements on ATE. Scan pins require a large amount of memory behind them, but do not have high-speed requirements. Also, scan-based tests typically do not require accurate edge placement on all pins, which means the number of high-performance tester channels is reduced.

4. DESCRIPTION OF JTAG INSERTED INTO THE VLSI SYSTEM

For a better understanding of the working of JTAG, it can be divided into four components namely

1. Test Access Port (TAP)
2. TAP Controller
3. JTAG Registers
4. Instruction decoder

The process of performing boundary scan can be understood from the below figure

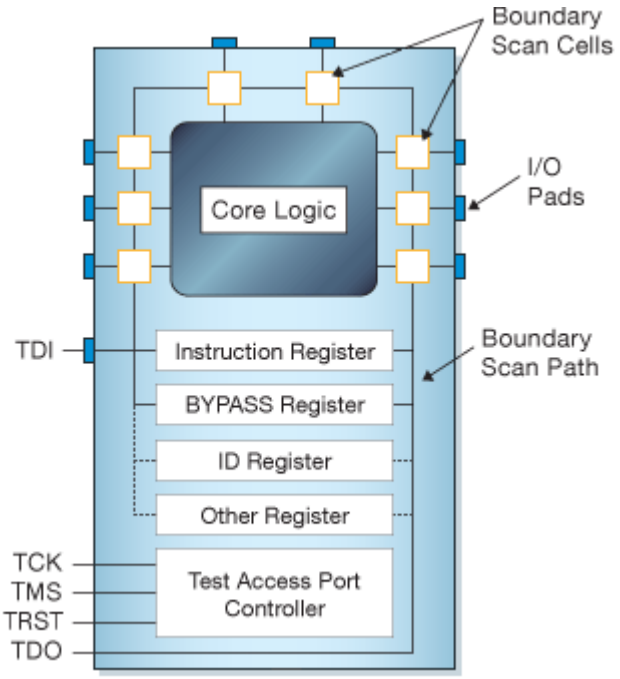

Fig 4.1 *Schematic diagram of JTAG system*

4.1. ***Test Access Port (TAP)*** - It provides interface signals to JTAG for performing boundary scan operations. The Interface signals can be classified as

- Test Clock (TCK) - This pin provides the clock signal for JTAG components

- Test Mode Select (TMS) - This pin is used for generating a signal to select various modes to control the JTAG operation

- Test Data Input (TDI) - This pin is used to transfer the data into the device for testing.

- Test Data Output (TDO) - When the internal state machine is in the correct state, this signal represents data shifted out of the device after device testing and is valid on the falling edge of TCK, while TDI is taken on the rising edge of TCK.

- Test Reset (TRST) - This pin is used to reset the TAP controller operation.

4.2. *Test Access Port Controller*

The TAP controller is used for controlling various JTAG operations, where it is represented by a state machine whose state transitions are controlled by the TMS and TRST signals. It has a total of 16 states, where TCK is the clock and TMS is taken as input. The TAP controller 16-state machine diagram is shown in the below figure [8]

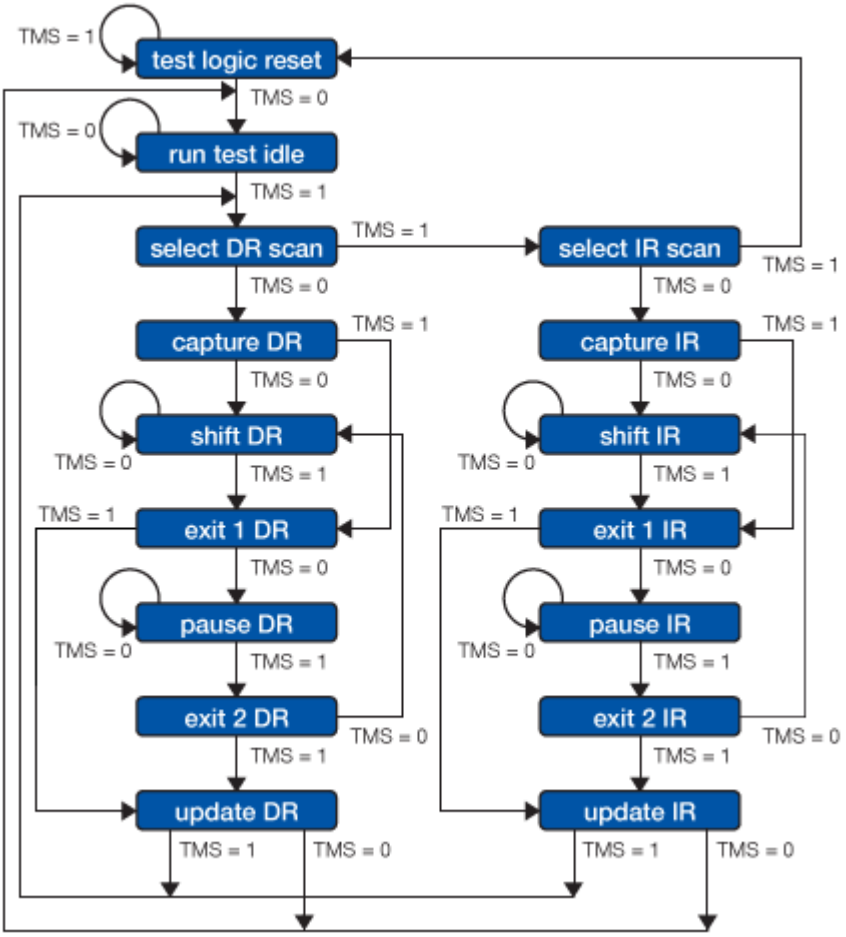

Fig 4.2 *State transition diagram of TAP controller*

By giving TMS = 1 / 0 signals periodically, we can move from one state to another state in the state diagram. Since every state has two possible outcomes, every transition can be managed by the lone TMS signal sampled on TCK. The device's data register or instruction register may be set or retrieved using one of the two primary methods. For example, in order to reset the TAP controller we need to give TMS=1→1→1→1→1 or TRST = 0.

4.3. *JTAG Registers*

JTAG has two types of registers involved with it. One instruction register and two or more data registers are present on every compliant device which share the same TDI and TDO signals.

1. *Instruction Registers* - The current instruction is kept in the instruction register. The TAP controller uses its contents to determine what to do with incoming signals. Most frequently, the instruction register's contents will specify which data register signals should be routed. This is decoded through the Instruction decoder. The below figure shows the block diagram of the Instruction Register.

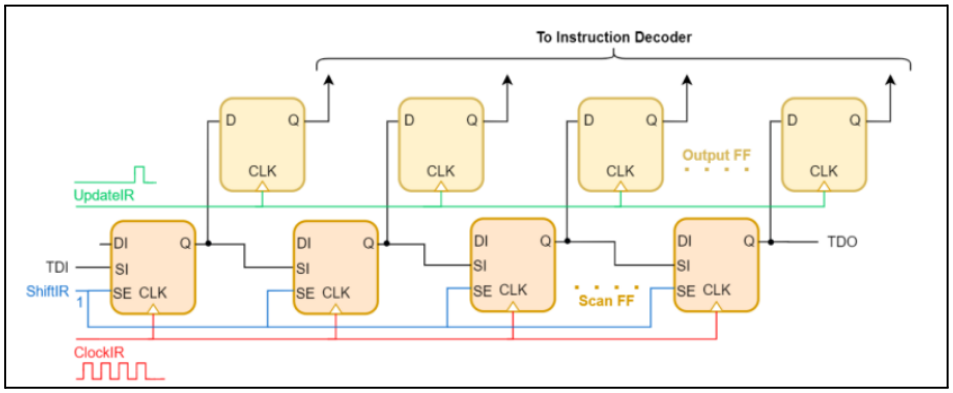

Fig 4.3. *Instruction Register block diagram*

2. *Data Registers*

There are two primary data registers used for data storage and transfer in the JTAG circuit i.e, The Boundary Scan Register (BSR) and the Bypass register. There are also other data registers such as the Device ID register and the Device specific register which may be present, but they are not mandatory according to the IEEE 1149.1 standard. They can be included based on designer necessity.

Boundary Scan Register (BSR) - This is the primary data register for testing. It consists of two types of BSR cells, one is used for taking input data, while the other is for output data. Each BSR cell has 2 flip flops. The 1st Flipflop will be triggered by the Scan/Capture clock and the 2nd Flipflop will be triggered by an update clock. Input to these Flipflops is controlled by Shift_dr and capturing of output is controlled by mode signal. Based on the control and clock signals either system pin input or test input are fed to the system logic. In this way, BSR cells decide which data to be fed into system logic based on the signals generated by the Instruction decoder. The Block diagram of the Input Boundary Scan Cell is shown in the figure below.

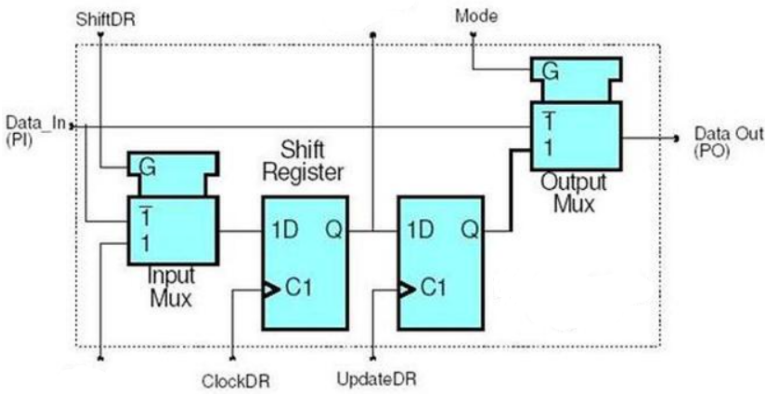

Fig 4.4 *Input Boundary Scan Cell diagram*

There are four BSC operations that are performed namely,

- Normal: In this, the system pin is directly connected to system logic.
- Scan: move data from the last cell to the next so that we can load the test pattern
- Update: make changes to the output Flip flop as well as apply our test patterns to system logic.
- Capture: It captures the system input and sends it to the Flipflops.

These four operations are selected by signals as shown in the table.

Operation	Mode	ShiftDR	Clock
Normal	0	X	System
Scan	X	1	ClockDR
Update	1	X	UpdateDR
Capture	X	0	ClockDR

Table 1.1 *BSR Cell operations*

BYPASS Register - It is a single-bit register that transfers data from TDI to TDO when ShiftDR = 1. It is used to transfer data from a device input to output without the need to pass through a Boundary scan register, thus reducing the delay time. It enables the testing of other devices in a circuit with minimal overhead.

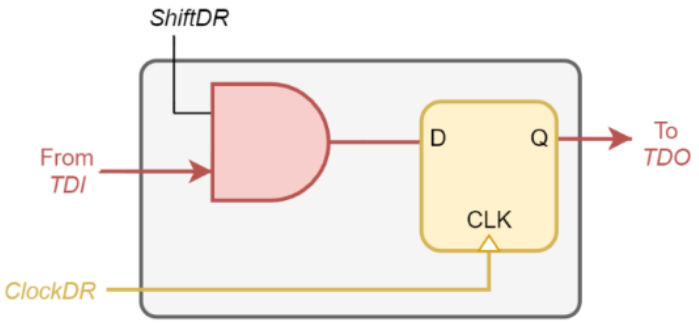

Fig 4.5. *Bypass Register Block Diagram*

4.4 Instruction Decoder

It performs the decoding of instructions stored in the instruction register. It generates two signals Mode and Select (DR). Mode signal is passed to BSR cells to select which input to pass to circuit logic, while select (DR) is used as a select signal for the MUX to decide which register data needs to be captured.

4.5 JTAG Instructions

1. *BYPASS* - This instruction connects the TDI and TDO lines using the BYPASS register. This instruction enables the testing of other devices in the JTAG chain with minimal overhead.

2. *EXTEST* - It connects the TDI and TDO to the Boundary Scan Register (BSR). With the 'CaptureDR' JTAG state, the device's pin states are sampled, and new values are shifted into the BSR with the 'ShiftDR' state; these values are then applied to the device's pins with the 'update dr' state.

3. *SAMPLE/PRELOAD* - The TDI and TDO are connected to the BSR as a result of this instruction. The device, however, remains in its normal operational mode. During this instruction, the BSR can be accessed by performing a data scan operation to obtain a sample of the functional data entering and leaving the device. Prior to loading an EXTEST instruction, the instruction is also used to preload test data into the BSR. [8] [9]

There are other JTAG instructions that are not mandatory and can be incorporated based on the necessity of the designer such as INTEST, RUNBIST, CLAMP, IDCODE, USERCODE and HIGHZ.

5. RTL SYNTHESIS OF VARIOUS BUILT COMPONENTS

The individual components of JTAG and the multiplier are synthesized using the Mentor Graphics Precision Tool. Below are the RTL schematics obtained from it.

1. **Full Adder**

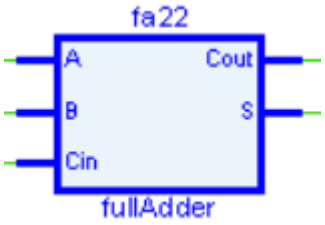

Fig 5.1 *Top view of Full Adder circuit*

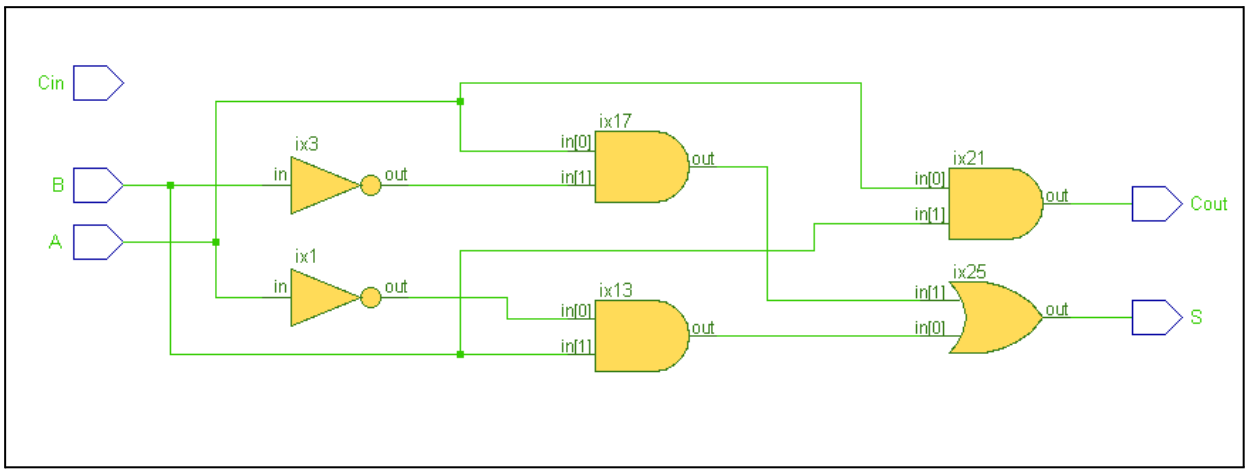

Fig 5.2. *RTL schematic of full adder circuit*

2. 8 - Bit Array Multiplier

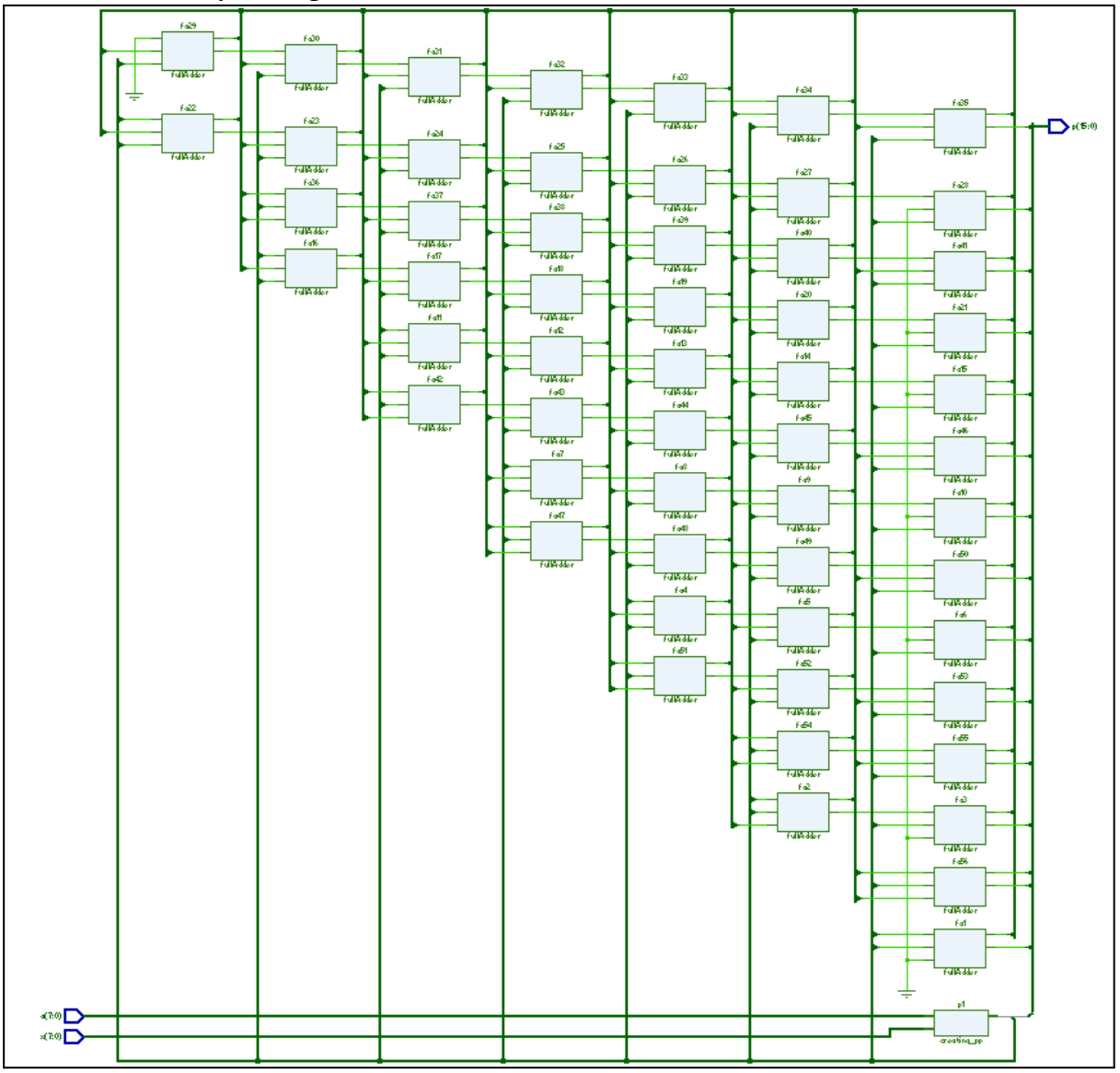

Fig 5.3. *RTL schematic of 8-bit Multiplier*

3. Instruction Register

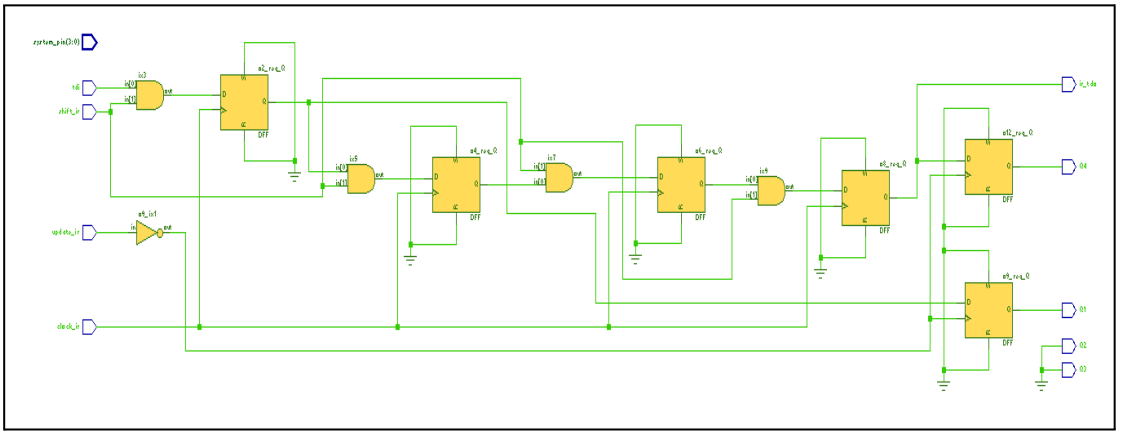

Fig 5.4 *RTL schematic of Instruction Register*

4. TAP Controller

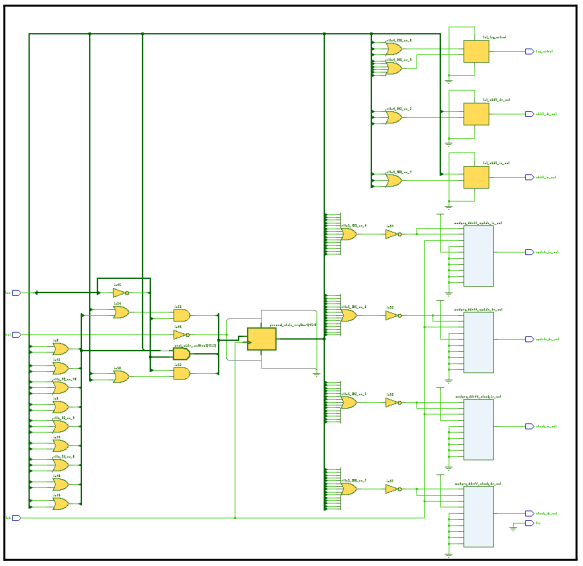

Fig 5.5 *RTL schematic of TAP controller*

5. Boundary scan cell circuit and Bypass Register

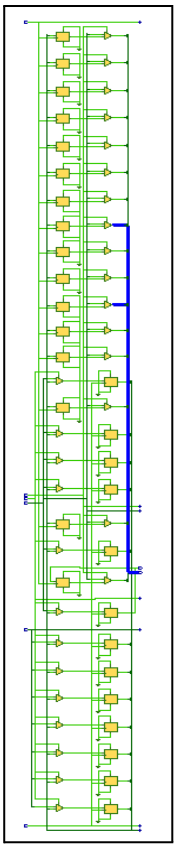

Fig 5.6 *RTL schematic of BSC circuit*

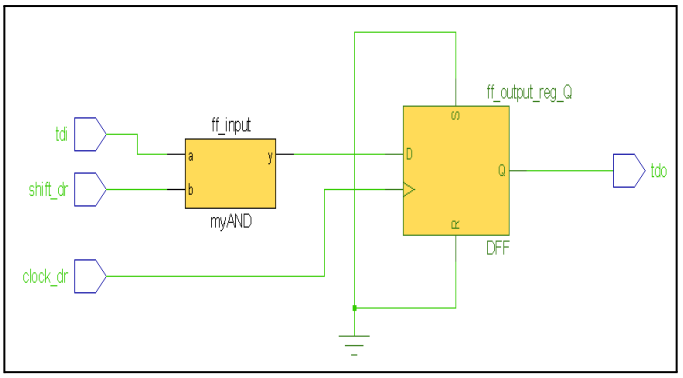

Fig 5.7 *RTL schematics of Bypass Register*

6. Stand-alone JTAG circuit

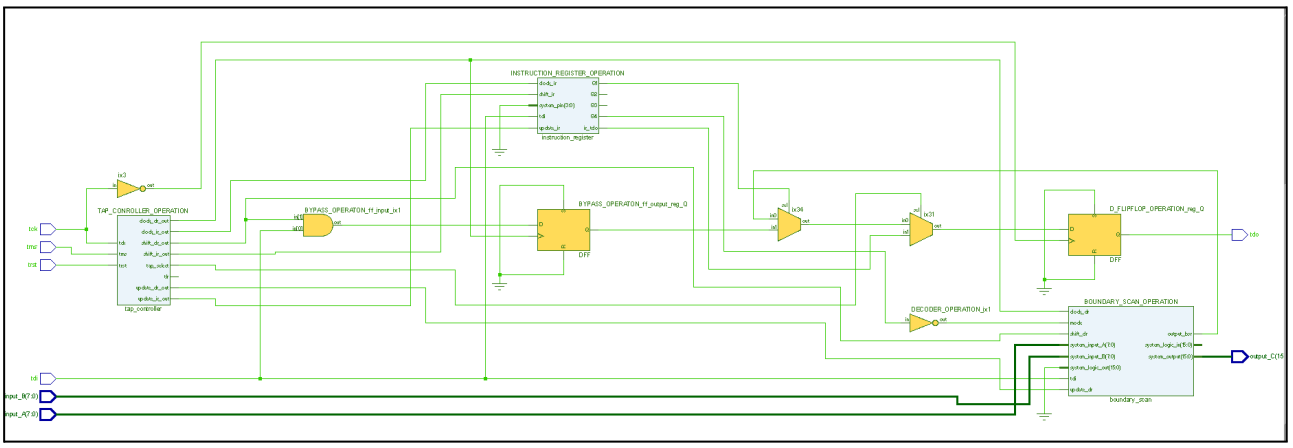

Fig 5.8 *RTL schematics of Stand-alone JTAG circuit*

7. JTAG circuit with 8 - bit Array Multiplier

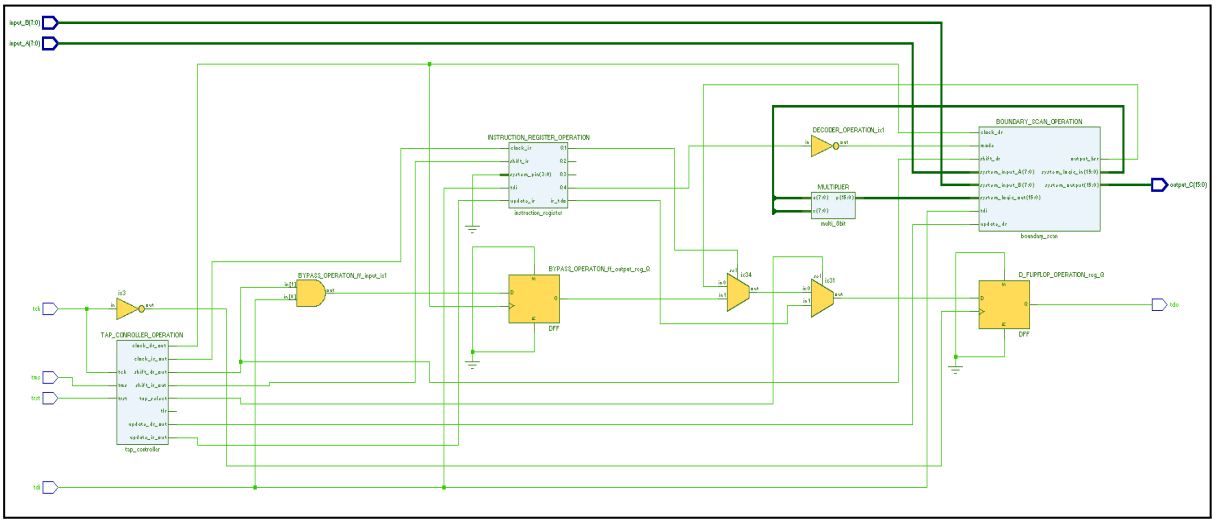

Fig 5.9 *RTL schematics of 8 - bit Array Multiplier with JTAG circuit*

6. DESCRIPTION OF OVERHEAD AND ADDED BENEFITS OF JTAG

Benefits of JTAG:

- Including the JTAG testing methodology at the circuit level and board level can reduce the time to distribute the product in the market, as the time to do In-circuit takes a lot of time. By avoiding In-circuit, the time to introduce the product in the market greatly reduces which changes a lot as there is high competition in the electronics market.

- Inclusion of JTAG in the circuit is way cheaper than buying a new In-circuit testing equipment which may cost millions of dollars. Hence this technology can be adopted by microelectronic producers.
- The inclusion of the JTAG circuit can reduce the cost of test generation. At the board level, boundary scan testing simulates in-circuit testing without the use of numerous probes. This adoption may enable the reuse of test patterns at various levels of hierarchy, including chip, board, and system.

- Ordinary manufacturers should avoid investing in In-circuit testing if their circuit designs are not fully developed. In such cases, JTAG inclusion is the best solution for producing circuit boards quickly, with a high yield, and at a low cost. [8] [9]

Overheads of JTAG

- One of the main overheads of JTAG is an increase in area size. As we include new components the area of the synthesized circuit increases. This area size is not constant but depends on the input and output pins of the logic circuit. In our designed JTAG circuit the Area overhead is mentioned in table 6.1.

- One more overhead of boundary scan is the increase in the number of pins. The addition of JTAG can cause an increase in 5 pins in the circuit which is used for testing purposes.

- There is a requirement for more effort during the design phase concerning the JTAG circuit as for every circuit that the JTAG is included in, the JTAG design should be modified accordingly and some data registers and operating modes should be added according to the circuit.

- As new components are added to the circuit for boundary scan, the power consumption of the total circuit increases. In order to reduce the overall power consumption, during normal operation of the circuit, the JTAG components can be kept in sleep mode and can be made active during test operations thus regulating power usage.

- And the last overhead is performance degradation. As the new components are added there might be some delays generated in the circuit which may reduce the performance of the circuit. As the logic circuit is bigger than the JTAG circuit, this overhead can be neglected.

Overheads	8-bit Array Multiplier	Standalone JTAG circuit	JTAG + Multiplier
LUT's	158	88	275
Accumulated Instances	185	98	299
Gates	190	206	408

Table 6.1. Overview of JTAG Area Overhead

Using the Mentor graphics tool, 8-bit multiplier with and without JTAG and standalone JTAG circuits are synthesized and their respective captured area report is shown in the below figures

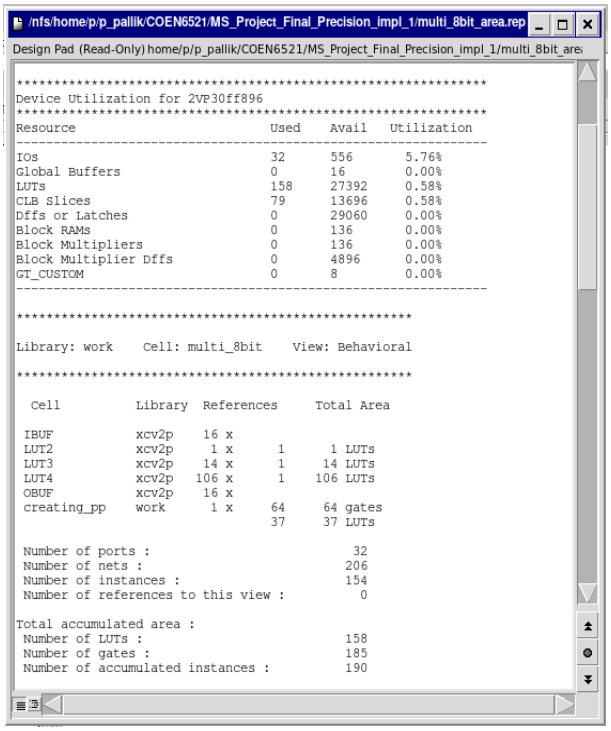

Fig 6.1 *Area report for 8-bit Array Multiplier*

Fig 6.2 *Area report for Stand-alone JTAG Circuit*

Fig 6.3 *Area report for JTAG included circuit*

7. DESCRIPTION OF TEST PROCESS USING JTAG

The process of implementing and testing the JTAG in different modes is mentioned below. In this process, the individual entities of JTAG and the entire JTAG setup with the 8-bit multiplier are tested.

Step-1: Generate Instruction code

Step-1.1: TAP Controller

It has two modes i.e., Instruction mode and Data Mode. In this project TAP controller is used to generate signals (shift_dr_out, shift_ir_out) and clocks (clock_dr_out, clock_ir_out, update_ir_out, update_dr_out) and are driven to other modules such as Instruction register, Boundary Scan Register and Bypass Register.

Instruction Mode:

Before loading the instruction code that needs to be executed, the TAP controller needs to be initialized by applying the TMS sequence of 1→1→1→1→1. This particular sequence will force the TAP into reset mode. Now by applying a sequence of 0→1→1→0→0 the state will be changed to the Shift state. A sequence of 0→0……..0 is applied till the complete instruction code is shifted via TDI. Now the state will be changed to Update-IR. Tap controller in instruction mode is tested by a separate testbench and the overview of the testbench is shown below.

TMS	1→1→1→1→1→0→1→1→0→0→0……..0→1→1→0
Initial State	Test Logic Reset
Final State	Run Test Idle
Signals Generated	Shift_IR_out, Shift_DR_out
Clocks Generated	Clock_IR_out, Clock_DR_out, Update_IR_out,Update_DR_out

Table 7.1. *TestBench overview of TAP controller*

Since TAP is operated in instruction mode, clock_ir_out will be generated as long as TAP is in shift state and update_ir_out is generated in update state. Since it's not in data mode, clock_dr_out and update_dr_out will be undefined as expected.

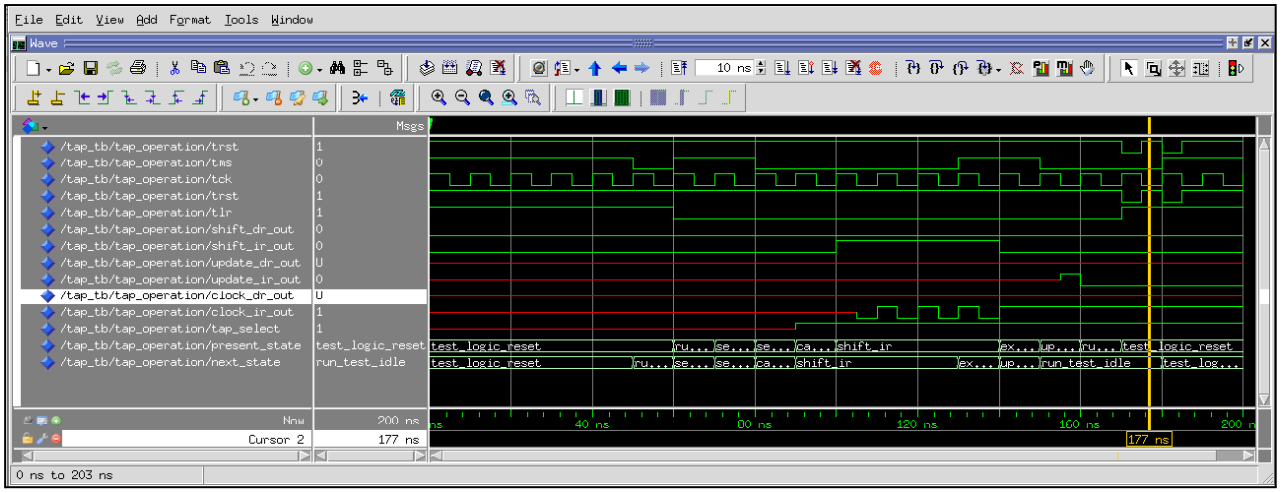

Fig 7.1. *Simulation result of TAP controller*

Step-2: Generate Control Signals

Step-2.1: Testing of Instruction Register

The instruction opcode is driven to it from the TAP controller. Since the opcode is loaded bit-by-bit in the shift state, an entity needs to be in place to hold the entire instruction opcode bits before driving it to the instruction decoder. So the Instruction register will hold the bits and the complete opcode will be given to the decoder when a update_ir_out clock is generated by the TAP controller. An overview of Instruction decoder functionality in this project is shown in figure

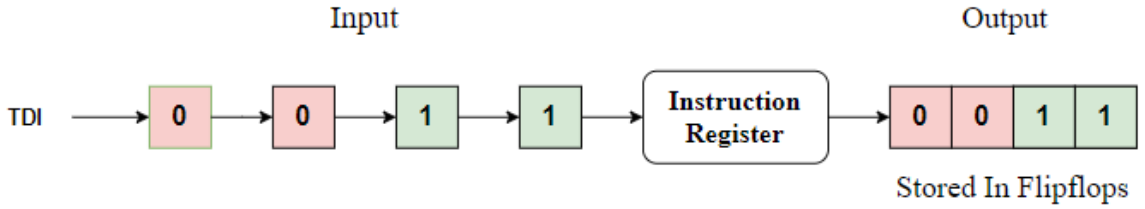

Fig 7.2. *TestBench overview of Instruction Register*

In the testbench, the outputs from TAP are forced as 0-0-1-1 which should store in the flip flops(q1, q2, q3, q4). Verified this functionality using the testbench and the complete simulation results are shown in the figure below.

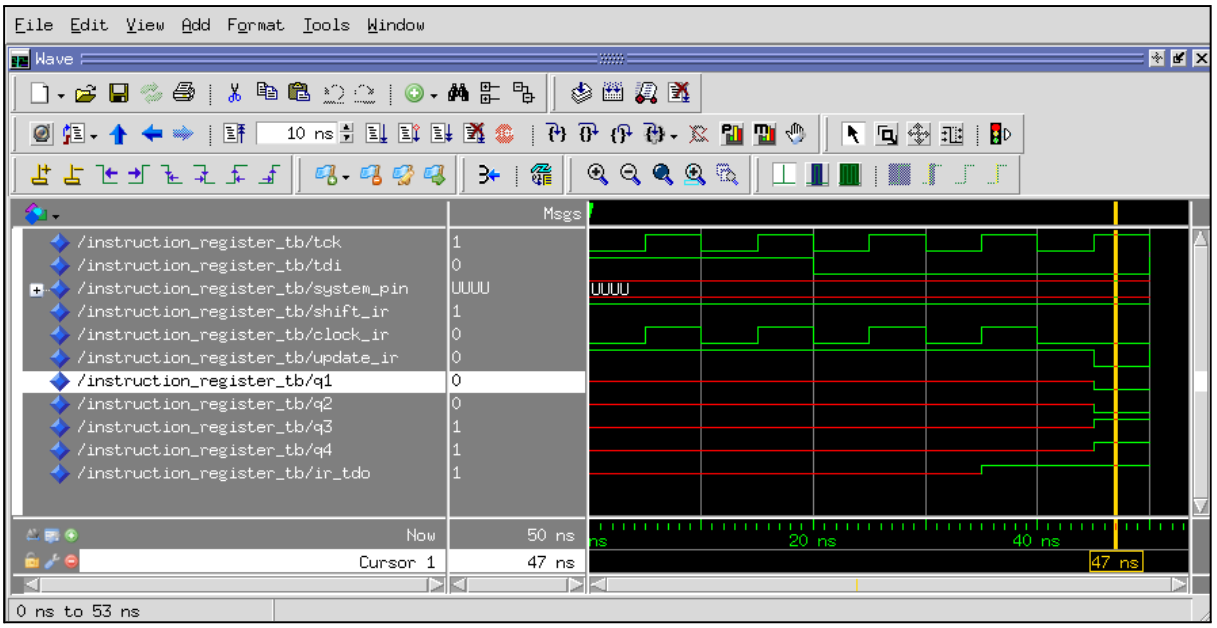

Fig 7.3. *Simulation result of Instruction Register*

Step-2.2: Testing of Instruction Decoder

The instruction opcode bits from the Instruction Register are given to the decoder, based on which the mode and select signals will be generated. The select signal is used to choose from which register i.e, Boundary-scan or Bypass register the output data needs to be captured. The mode signal is used for the controllability of the system input and output pins. The functionality of the decoder involving all the modes and signals that it can generate is verified using a testbench.

Cell-0	Cell-2	Cell-3	Cell-4	Mode	Select
0	0	0	0	1	0
1	1	1	1	U	1
0	0	0	1	0	0
0	0	1	0	1	0
Other Combinations				U	U

Table 7.2. *TestBench overview of Instruction Decoder*

All the above cases have been verified successfully and the complete simulation results of the decoder are shown in the figure below.

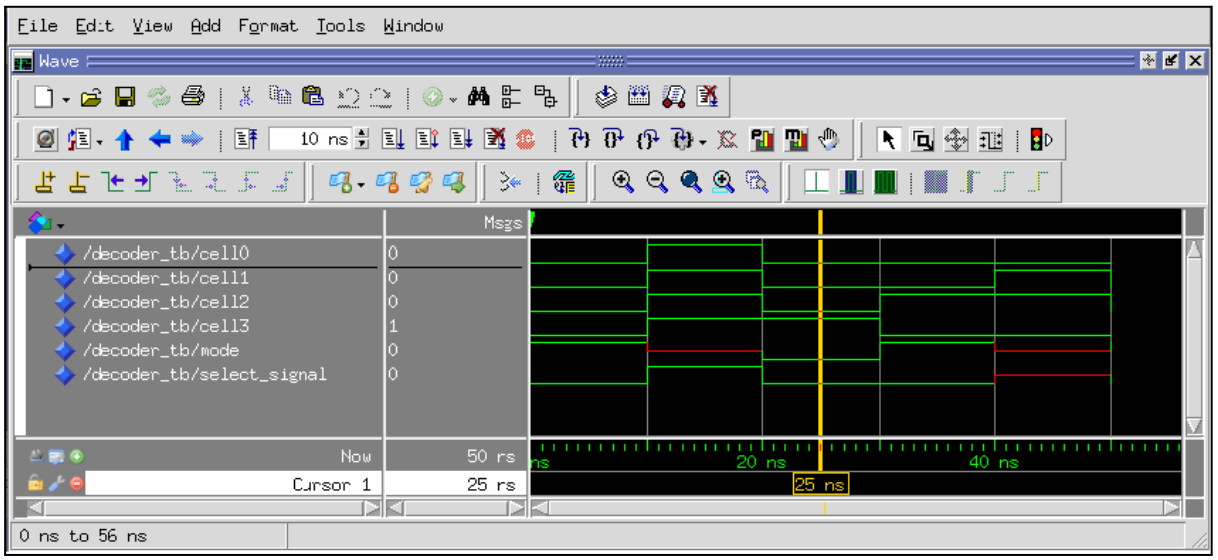

Fig 7.4. *Simulation result of Instruction Decoder*

Step-3: Testing control signals in JTAG

Step-3.1: Bypass Register

In order to capture data from the bypass register, the selected signal needs to be '1' so that the data will be driven through Bypass using TDI to TDO. For this, first, the TAP controller should be in data mode. Now by applying the TMS sequence 0→1→0→0 from the test idle state, it is forced to shift_DR state and the intended data is. The TAP needs to be in a shift state till the data that needs to be shifted is completed.

TestBench: In order to verify the functionality of the bypass register, a testbench is created and different inputs are sent using TDI and the same output is captured at the Output. Testbench overview is shown and the entire simulation results are shown in the below figure.

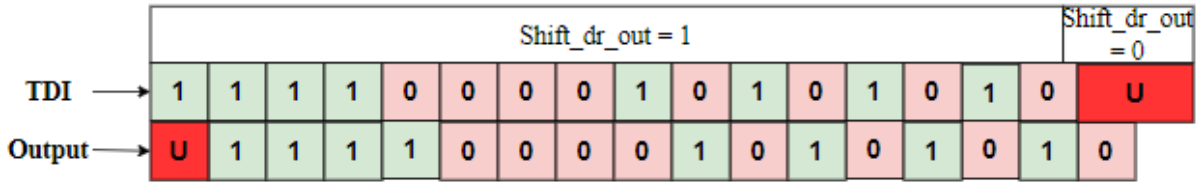

Fig 7.5 *TestBench overview of Bypass Register*

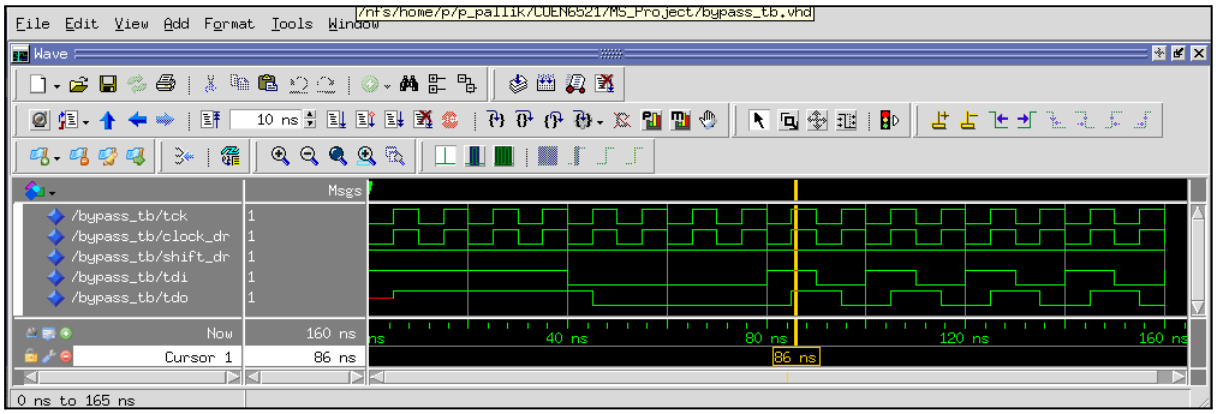

Fig 7.6. *Simulation result of Bypass Register*

Step-3.2: Boundary Scan Register

This component makes use of both the input boundary scan cells and output boundary scan cells. All the four operations mentioned in Table 1.1 are performed and the input boundary scan cell functionality is verified by the testbench. In this, when mode=0, system input is observed at the system_logic_in, and when mode=1, shift_dr_out =0/1, the input is captured from system input/last_cell passed through both the capture and update stages and got captured in the system_logic_in. Captured simulation results for the input boundary scan cell are shown below.

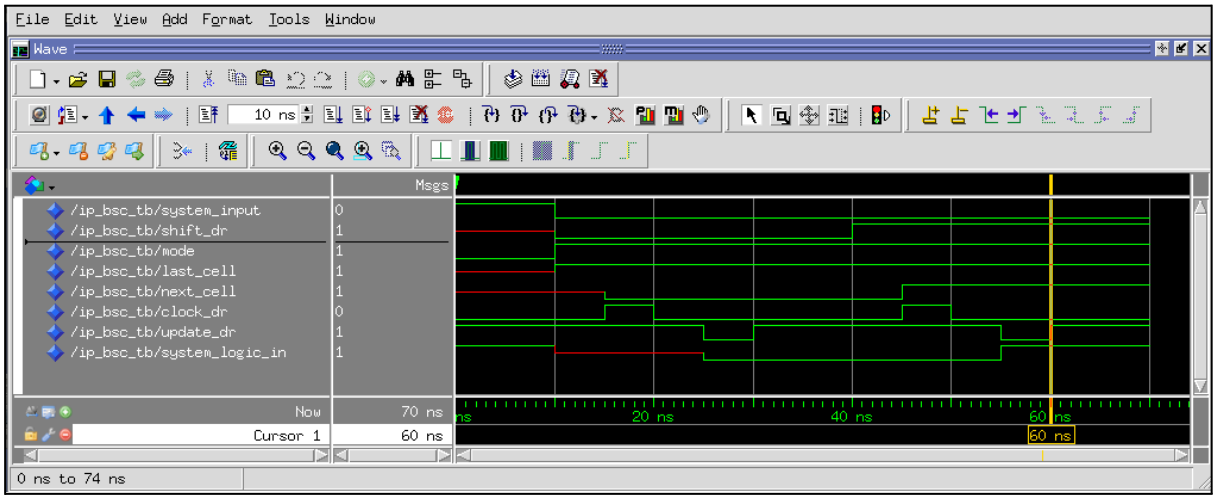

Fig 7.7. *Simulation result of Input Boundary Scan Cell*

Also the Boundary Scan register is tested using a separate testbench. In this when shift_dr_out=1, the data is pushed by shifting the TDI(16 times) and it gets updated in system_logic_in when the update_dr_out clock is generated. Also in one test case, TDI is shifted 32 times, so that the pushed data can be seen at the system_output level, which ensured the successful interconnection between input and output boundary scan cells. Verification of this module is done successfully and complete simulation results of the Boundary Scan Register are shown in the waveform below.

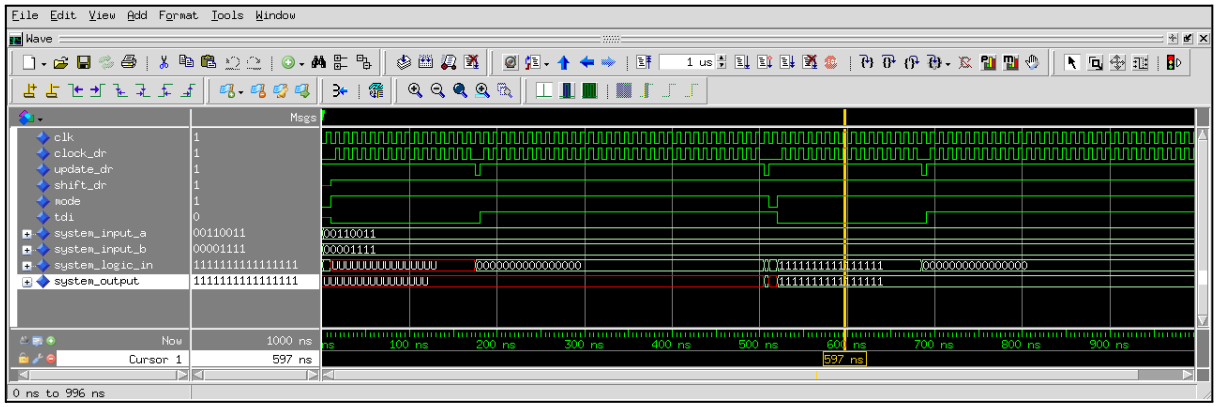

Fig 7.8. *Simulation result of Boundary Scan Register*

Step-4: Testing JTAG functionality when integrated with 8-bit multiplier

Entire JTAG functionality is tested by creating 3 test benches that verify JTAG with multiplier in three different scenarios.

Test-1: JTAG-Normal Mode

In this testbench, Input is given at the system pin level and the instruction code is given as "0001" which generates mode =0 and select_signal=0. The expectation of this test is that system input should be driven directly to the multiplier input ports, so that multiplier will perform multiplication on the input data and the result should be driven to the system output pins directly.

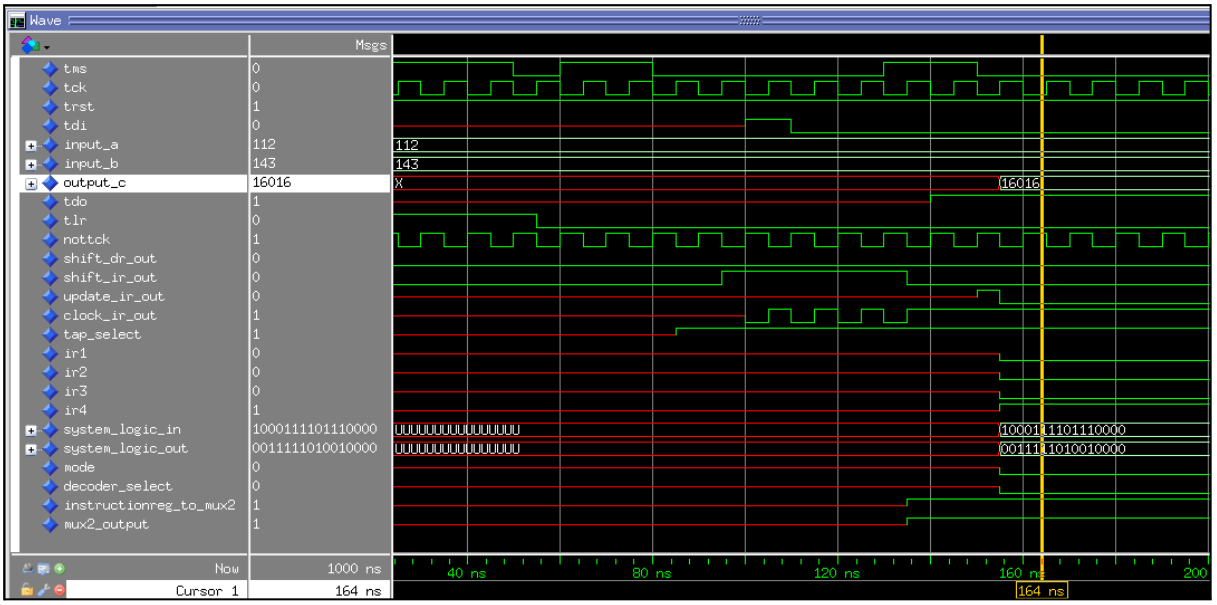

Fig 7.9. *Simulation result of JTAG multiplier circuit under normal operation*

In the above testbench, the instruction opcode from the instruction register is expected as “0001” and it is verified (ir1, ir2, ir3, ir4) at the time instance t=164 ns. Also, the decoder outputs of mode=0 and decoder_select=0 are verified at the same time. The system inputs given in this scenario are a=112 and b=143. Since mode = 0, the input from the system pins got driven directly to the multiplier, which performed the multiplication operation and produced a result of 16016 (112*143). This output can be observed on the system output pin(output_c). This normal operation of JTAG is successfully verified and the complete simulation results are shown in above figure 7.9.

Test-2: JTAG-Bypass Mode

For this test, the instruction code is given as “1111” which generates select_signal=1, and mode is not considered in this(Since the mode is only involved in boundary scan registers). Expectation of this test is that the data given in TDI is shifted to TDO through the bypass register.

The simulation complete results are shown in figure 7.10. From the results, the expected instruction code “1111” can be observed (refer - ir1, ir2, ir3, ir4). The Select signal to the mux component, for deciding from which register the data needs to be captured is shown at t= 150ns. Since decoder_select=1, data will get tapped from the bypass register. From the waveform, it is observed that data that is being sent via TDI is shifted to TDO through the Bypass register. So the bypass functionality of the JTAG is successfully verified.

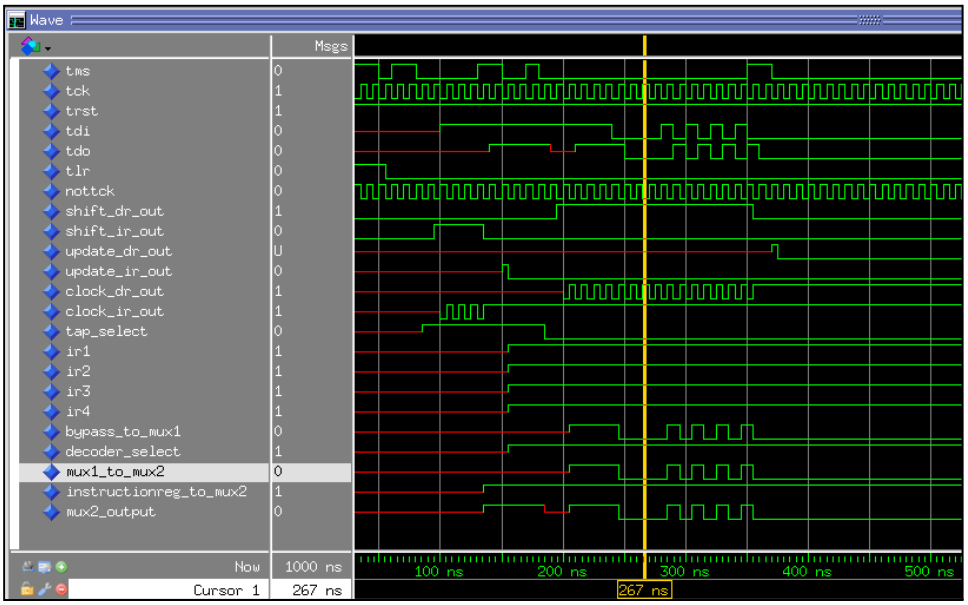

Fig 7.10. *Simulation result of JTAG multiplier circuit under Bypass operation*

Test-3: JTAG-Sample/Preload

For this test, the instruction code is given as “0010” which generates select_signal=0 and mode=1.

In the SAMPLE part, the input boundary scan cells are loaded with the data that is available on the system input pins on the rising edge of TCK. In the Preload part, the data should be loaded into the input boundary scan cells during Shift-DR. The data is sent in both ways to the multiplier and the output of the multiplier with the input data is captured at the System_logic_out.

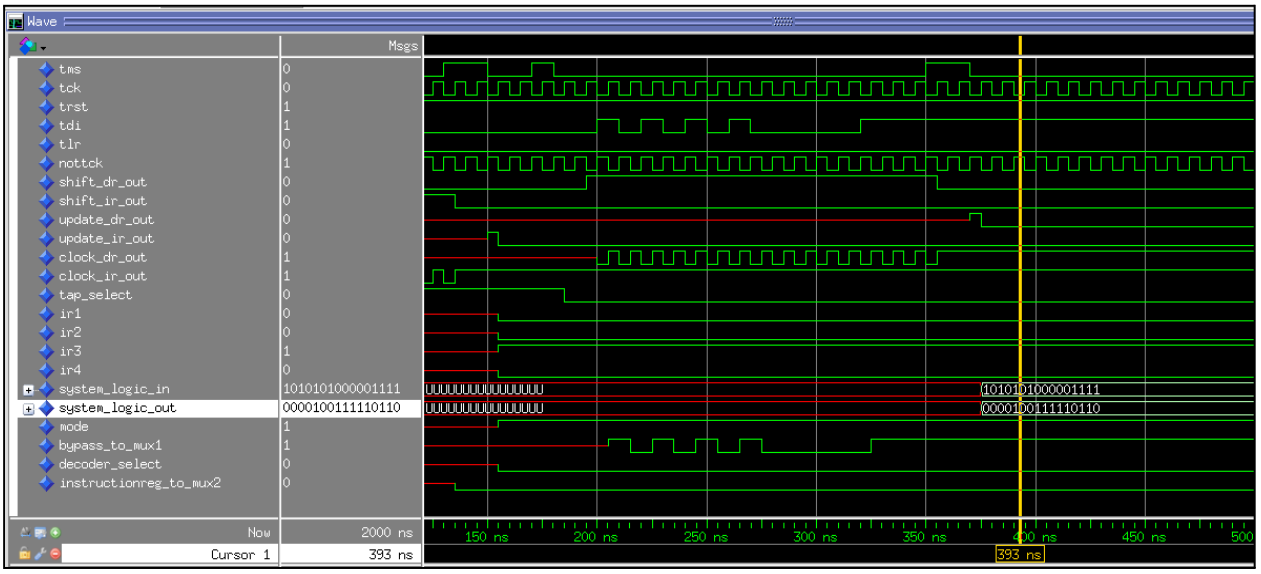

Fig 7.11. *Simulation result of JTAG multiplier circuit under SAMPLE/PRELOAD operation*

Along with the JTAG components, both the given Full Adder and the multiplier are verified with the test benches and the simulation results are shown in the figures respectively.

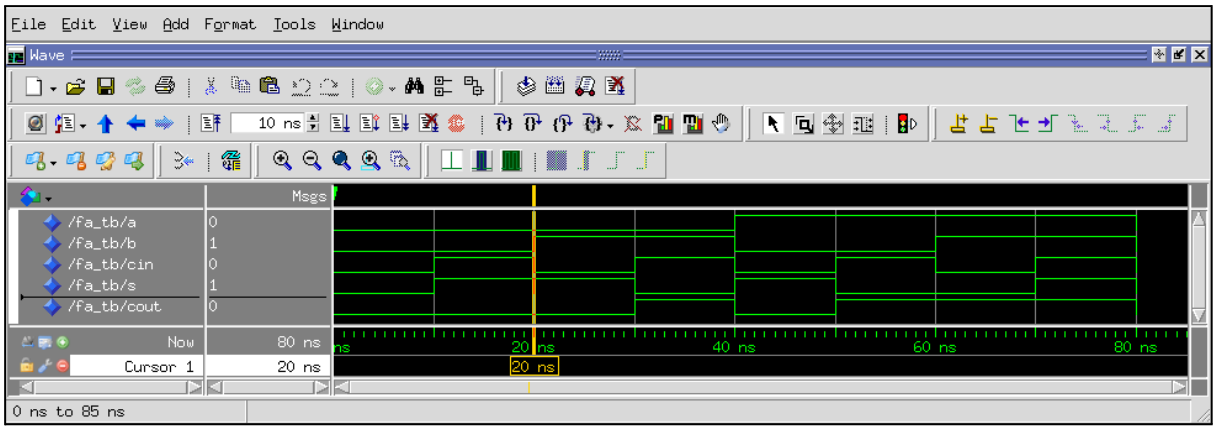

Fig 7.12. *Simulation result of full adder circuit*

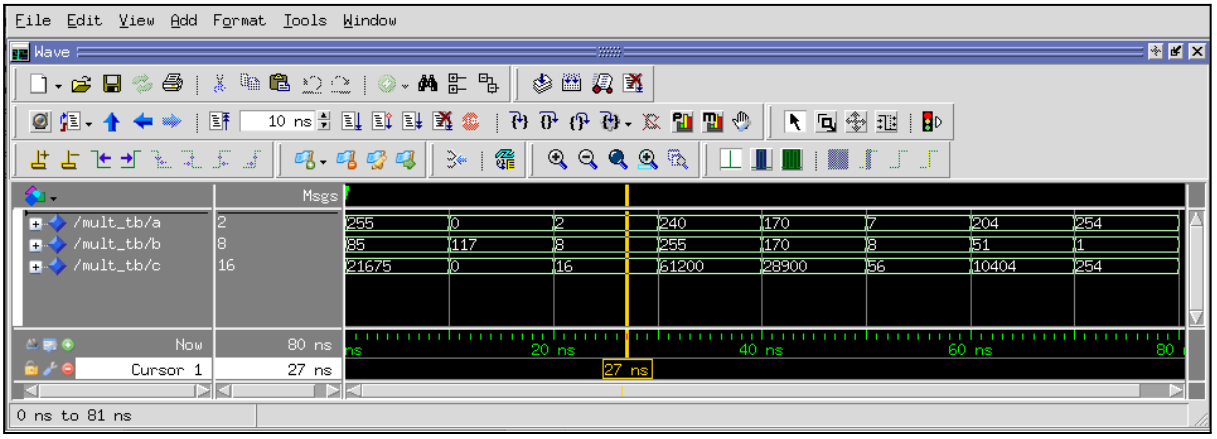

Fig 7.13. *Simulation result of Standalone 8 - bit Array Multiplier*

8. CONCLUSION

8.1. Summary

In conclusion, the boundary scan technique affects the board and system level test in both favourable and unfavourable ways. A boundary scan is cost-effective because it expedites the testing procedure, ensures greater manufacturing yields, and includes numerous other cost-saving steps. By diversifying into other fields like functional testing, it can produce more solutions for the manufacturers. From our project, it can be inferred that, even though JTAG has significant advantages, it also comes with major challenges such as area overhead and additional design work. When comparing the requirements for test points and fixtures in an in-circuit test system, JTAG inclusion drawbacks such as an increase in Area and Design work can be overlooked.

8.2. Challenges

In the present, we use various devices in day to day life which are equipped with JTAG circuits. These JTAG circuits can be used as a medium to hack these devices, which can be very dangerous as we are living in an IOT era, where a device can store users personal information. Some of such exploitations can be done by Physical Pin Modification, JTAGulator Tool ,etc. [10]

8.3. Discussion

As we know there are quite security challenges that can be caused by the JTAG circuit, there are also some methods which can prevent such a situation from happening. One of them is fusing off the TAP port by the chip supplier. Normally this is left open for further upgradation, but by fusing it off we can prevent chip hacking at board level. It is to be further discussed on how to prevent JTAG from being used in a wrong way and new methodologies should be developed.

9. References

[1] Lecture Notes by Dr. Mahmoud Masadeh, Design for Testability, Summer 2022, Available at: https://moodle.concordia.ca/moodle/course/view.php?id=145715 Accessed on: 13th July 2022

[2] Be Van Ngo, P. Law and A. Sparks, "Use of JTAG boundary-scan for testing electronic circuit boards and systems," 2008 IEEE AUTOTESTCON, 2008, pp. 17-22, doi: 10.1109/AUTEST.2008.4662576.

[3] H. B. Shashidhara, S. Yellampalii and V. Goudanavar, "Board level JTAG/boundary scan test solution," International Conference on Circuits, Communication, Control and Computing, 2014, pp. 73-76, doi: 10.1109/CIMCA.2014.7057760.

[4] M. Sjalander and P. Larsson-Edefors, "Multiplication Acceleration Through Twin Precision," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 17, no. 9, pp. 1233-1246, Sept. 2009, doi: 10.1109/TVLSI.2008.2002107.

[5] Lecture Notes for COEN 6501, *Fall 2021, Available at:* https://users.encs.concordia.ca/~asim/COEN_6501/Lecture_Notes *Accessed on: 20th July 2022*

[6] Shakya, Mr. Rahul and Jindal, Poonam, Comparative analysis of 8-bit and 16-bit Array multipliers, modified Booth Multiplier-A Study (February 25, 2022). Proceedings of the 3rd International Conference on Contents, Computing & Communication (ICCCC-2022), Available at SSRN: https://ssrn.com/abstract=4043967 or http://dx.doi.org/10.2139/ssrn.4043967

[7] P. Maxwell, I. Hartanto and L. Bentz, "Comparing functional and structural tests," Proceedings International Test Conference 2000 (IEEE Cat. No.00CH37159), 2000, pp. 400-407, doi: 10.1109/TEST.2000.894231.

[8] JTAG, https://www.jtag.com/ , Accessed on : 22 July, 2022.

[9] JTAGlive, https://www.jtaglive.com/product/jtag-maps-for-altium/, Accessed on : 24 July, 2022.

[10]Vishwakarma G, Lee W. Exploiting JTAG and Its Mitigation in IOT: A Survey. *Future Internet*. 2018; 10(12):121. https://doi.org/10.3390/fi10120121