SUMMER INTERNSHIP REPORT

Submitted by

SHRINIDHI MM

Roll number: 108121117

In fulfilment of Summer Internship for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY
Tiruchirappalli-600015
Tamil Nadu, India

Table of Contents

About the Institute

The National Institute of Technology Tiruchirappalli (NIT Trichy) is a distinguished research university near Tiruchirappalli, Tamil Nadu, India. Originating as the Regional Engineering College Tiruchirappalli in 1964, it attained university status in 2003 and was rebranded as NIT Trichy. Acknowledged as an Institute of National Importance under the NITSER Act of 2007, the institution focuses on engineering, management, science, technology, and architecture. It offers an array of programs through its multiple academic departments, catering to bachelor's, master's, and doctoral levels. The institute offers 10 bachelor's, 42 master's, and 17 doctoral programmes through its 17 academic departments and awards more than 2000 degrees annually.

The National Institutional Ranking Framework (NIRF) ranked NIT Trichy first among the NITs for seven years in a row from 2016 to 2022.

Research at NIT Trichy is sponsored by various Indian government agencies. The departments of the institute also undertake consulting projects with government agencies. The institute's scientific output averages 700 papers and 10,000 citations per year. In addition, the institute's research community is actively involved in transforming unique concepts into products or processes, and it has several published and issued patents to its name.

The Institute has great infrastructure, with well equipped labs with high level licensed engineering software which provides students with great access to learning from them and improving their knowledge in wide areas of their interest.

ACKNOWLEDGEMENT

The first and foremost person I would like to express my boundless gratitude is my project guide, Dr. G. Lakshmi Narayanan, Professor, Department of Electronics and Communication Engineering, National Institute of Technology, Tiruchirappalli for his wholehearted support, valuable guidance and constant motivation throughout the project work.

I wish to thank PhD scholars and my team members for their numerous ideas and useful suggestion. They provided me a warm and friendly environment for completing this work.

Finally, to my family and friends, I extend my heartfelt thanks for their unwavering belief, encouragement, and endless motivation. Their unflagging support has propelled me forward, even during moments of challenge.

With gratitude,

SHRINIDHI MM

108121117

Department of Electronics and Communication Engineering

National Institute of Technology, Tiruchirappalli

Department of Electronics & Communication Engineering
NATIONAL INSTITUTE OF TECHNOLOGY
(formerly known as Regional Engineering College)
TIRUCHIRAPPALLI - 620 015.TAMIL NADU , INDIA
Tel : +91(431) 2503307
Fax: +91(431) 2500133
Email: laksh@nitt.edu

Dr. G. Lakshminarayanan
Professor

DATE: 30-08-2024

Certificate

This is to certify that **Ms. SHRINIDHI.M.M**, Roll No. 108121117, Department of Electronics and Communication Engineering, National Institute of Technology, Tiruchirappalli has worked on a project titled **"Modification of RTL design of a FPGA based 16-bit fixed point DSP Processor"** under my guidance in the Wireless System Design Laboratory of Electronics and Communication Engineering department, NIT, Tiruchirappalli from 26.05.2024 to 18.07.2024. The work carried out by him was good.

30/8/24
(Dr.G.Lakshminarayanan)

DEPT.OF ECE
N.I.T. TRICHY-620 015

ABSTRACT

In this Project, we have modified and implemented the Verilog design of an existing 16-bit fixed point DSP processor. The Existing 16-bit DSP processor is based on Harvard architecture and it is composed of a CPU with optimized hardware logic, 64kx16 on-chip data memory, and 64kx16 program memory. The CPU contains an Arithmetic and Logical Unit block, a multiply and Accumulate block, and a shift/Rotate block to perform mathematical operations. The DSP can handle 28 types of interrupts and it can perform various operations like timer, counter, interpolation, and decimation. This DSP processor is now modified for even better applications. Works done are instructions are stored in Program memory itself, Pipelining has been implemented, Interrupt controller module has been updated with instructions for ISR address, Interrupt Priority has been implemented and Synthesized using Xilinx ISE and various reports have been generated and studied.

Project Work

Modification of Verilog design of a FPGA based 16 bit fixed point DSP processor

1) Store Instructions in Program memory:

The first work in our project is to store instructions directly in Program memory instead of giving the instructions one by one in the test bench by the user. This is done by writing the set of instructions that needs to be executed are written inside the initial block in Program memory.

```
initial
begin
ROM[16'h1]<=16'h0201;
ROM[16'h2]<=16'h0101;
ROM[16'h3]<=16'h0102;
ROM[16'h4]<=16'h0302;
ROM[16'h5]<=16'h6A01;
ROM[16'h6]<=16'h2002;
ROM[16'h7]<=16'h6002;
ROM[16'h8]<=16'h7301;
ROM[16'h9]<=16'h5402;
ROM[16'hA]<=16'h2002;
ROM[16'hbe63]<=16'b1111111000000000;  //interpolator and decimator
ROM[16'hbe64]<=16'b1111111000000010;  //mac
ROM[16'hbe65]<=16'b1111111000000001;  // timer
ROM[16'hbe68]<=16'b1111110100000000;  //return
isr_address_previous <= 16'b0000000000000000;
end
```

Fig.1

Since these instructions are inside an initial block, they will be stored immediately in the Program memory just when the code starts running. These instructions are stored in ROM(Read Only Memory) at a specific address.

Table for the above set of instructions is given below:

ROM[16’h1]<=16’h0201	Load status register
ROM[16’h2]<=16’h0101	STORE DATA
ROM[16’h3]<=16’h0102	STORE DATA
ROM[16’h4]<=16’h0302	LOAD DATA
ROM[16’h5]<=16’h6A01	LOAD ACC
ROM[16’h6]<=16’h2002	ADD
ROM[16’h7]<=16’h6002	SUB
ROM[16’h8]<=16’h7301	LOAD TREG
ROM[16’h9]<=16’h5402	MULTIPLY
ROM[16’hA]<=16’h2002	ADD
ROM[16’hbe63]<=16’b1111111000000000	ISR address for I&D = 16’hbe63
ROM[16’hbe64]<= 16’b1111111000000010	ISR address for Mac = 16’hbe64
ROM[16’hbe65]<= 16’b1111111000000001	ISR address for Timer = 16’hbe65
ROM[16’hbe68]<= 16’b1111110100000000	RET

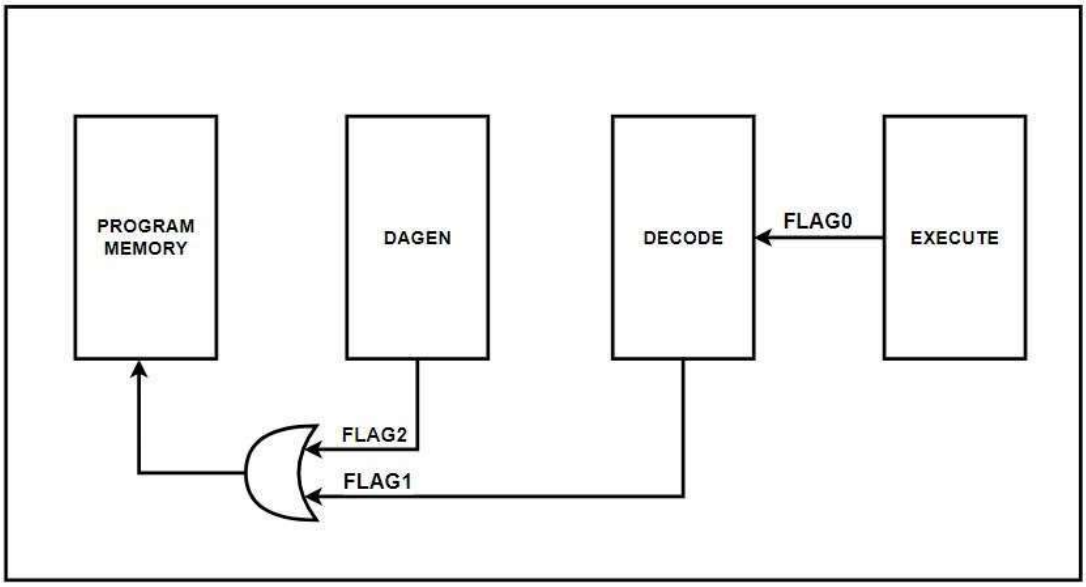
2) Optimization of code:

To efficiently run the above set of instructions stored in minimal time, we have implemented the concept of Pipelining using flags. This will reduce the total time taken for all instructions to be executed. In general, if the program memory starts executing an instruction, the next instruction has to wait until the previous instruction is completely executed but with the implementation of pipelining, the fetch, decode, and execute operations occur parallelly.

We have defined FLAG0, FLAG1, and FLAG2 in EXECUTE, DECODE, and DAGEN blocks respectively.

- The FLAG2 is set high when the instruction in the DAGEN block is executed.
- The FLAG1 is set high when the instruction in the DECODE block is executed.
- The FLAG0 is set high when the instruction in the EXECUTE block is executed.

Concept of Pipelining



Some of the instructions like STORE don't go into the DECODE block. So, we defined separate flags for DAGEN and DECODE. So the instructions in either of them if it is executed, it is will be informed to the program memory to send the next instruction immediately. Similarly, if the instruction in the EXECUTE block is executed, it will be informed to the DECODE block through FLAG0 to send the next instruction waiting in the DECODE block.

In this way, we can execute the instructions in parallel without waiting for the entire single instruction to get executed.

3) Interrupt controller:

In the existing DSP processor, the interrupt controller program was till the generation of Interrupt Service Routine(ISR) address.

In our project work, we have written the set of instructions in Verilog for a specific interrupt function to execute with the help of ISR address. These interrupts are Hardware interrupts.

Verilog code for executing the Interrupt function

```
always@(posedge CLK)
begin
    if(en)
    begin
    int_en_previous <= int_en;

    if(int_en)
        PC <= isr_address;
    else if(int_en_previous==1'b1 && int_en==1'b0)
        PC <= 16'hbe68;
    else if(PRDB[15:8] == RET)
        PC <= dataOut;
    else if ((flag2||flag1)&&(int_en==1'b0))
        PC<=PC+1;
    end
end

assign  PRDB=RST?16'h0000:ROM[PC];
assign fifo_rd_en=RST?1'b0:((ROM[PC][15:8]==STORE) || (ROM[PC][15:8]==load_DP ));
assign stack_en= (int_en_previous==1'b0 && int_en==1'b1)||(PRDB[15:8] == RET);
assign RW=(PRDB[15:8] == RET);
assign dataIn=(int_en_previous==1'b0 && int_en==1'b1)?PC+1:16'hxxxx;
```

These are set of instructions by which the currently running instruction is stopped and stored in a stack for the Program Counter(PC) to jump to ISR address to execute the interrupt function stored in that address.

After the interrupt function is executed, the PC value will be updated with the instruction stored in the stack and now the processor will continue the execution of instructions stored until a new interrupt comes.

From Fig.1, we have implemented three operations as hardware interrupts to execute the following functions,

- ✓ Interpolator and Decimator
- ✓ Mac
- ✓ Timer

We have assigned ISR addresses to these functions and stored them in Program memory. So, when an interrupt with these ISR addresses is generated, that function will get executed.

4) Priority Interrupts:

After finishing the Interrupt Controller module, we implemented the concept of Priority Interrupts.

This Interrupt Priority is very much important in DSP processors because we have so many priorities for interrupts with hardware and software interrupts, maskable and non-maskable interrupts.

Priority Order

- The non-maskable interrupts have the highest priority.
- Next priority is for external hardware-initiated maskable interrupts.
- Next priority is for internal hardware-initiated maskable interrupts.
- Software interrupt has the lowest priority.

Verilog code for Priority Interrupts

```
case(IFR)
16'bxxxxxxxxxxxxxxx1:IMR<=16'b0000000000000001;
16'bxxxxxxxxxxxxxx10:IMR<=16'b0000000000000010;
16'bxxxxxxxxxxxxx100:IMR<=16'b0000000000000100;
16'bxxxxxxxxxxxx1000:IMR<=16'b0000000000001000;
16'bxxxxxxxxxxx10000:IMR<=16'b0000000000010000;
16'bxxxxxxxxxx100000:IMR<=16'b0000000000100000;
16'bxxxxxxxxx1000000:IMR<=16'b0000000001000000;
16'bxxxxxxxx10000000:IMR<=16'b0000000010000000;
16'bxxxxxxx100000000:IMR<=16'b0000000100000000;
16'bxxxxxx1000000000:IMR<=16'b0000001000000000;
16'bxxxxx10000000000:IMR<=16'b0000010000000000;
16'bxxxx100000000000:IMR<=16'b0000100000000000;
16'bxxx1000000000000:IMR<=16'b0001000000000000;
16'bxx10000000000000:IMR<=16'b0010000000000000;
16'bx100000000000000:IMR<=16'b0100000000000000;
16'b1000000000000000:IMR<=16'b1000000000000000;
default:IMR<=16'h0000;
endcase
```

Synthesis in ISE:

This modified Verilog code of DSP processor has been simulated and synthesized in Xilinx ISE software.

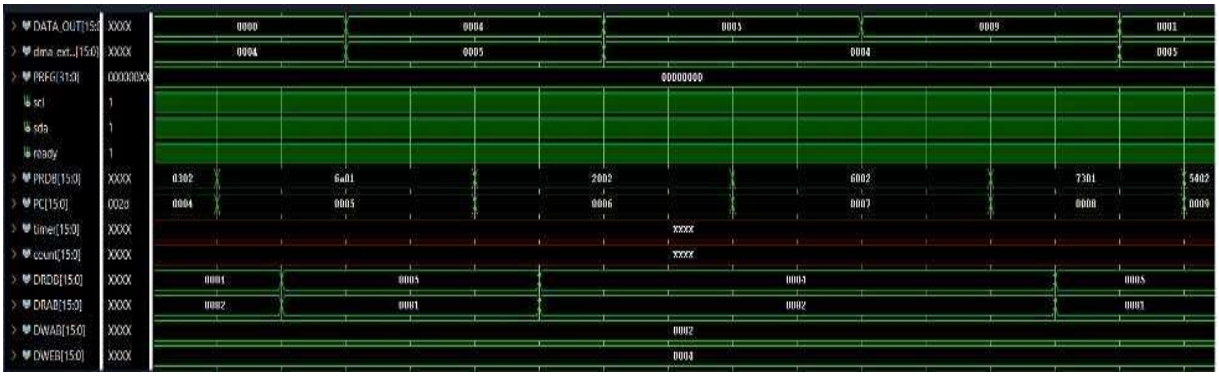
Simulation results, synthesis summary and reports have been generated and studied.

Simulation and Results:

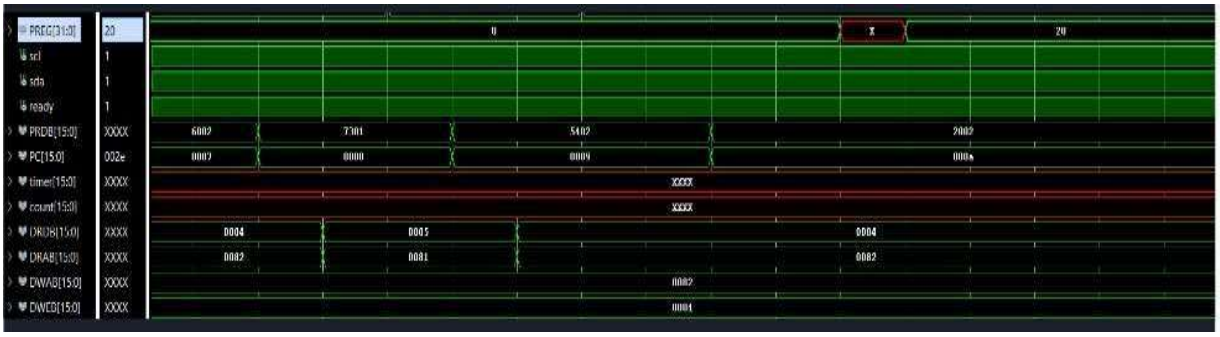
PC and Respective Instruction

LOAD DATA POINTER, STORE, LOAD

LOAD ACCUMULATOR, ADD, SUB



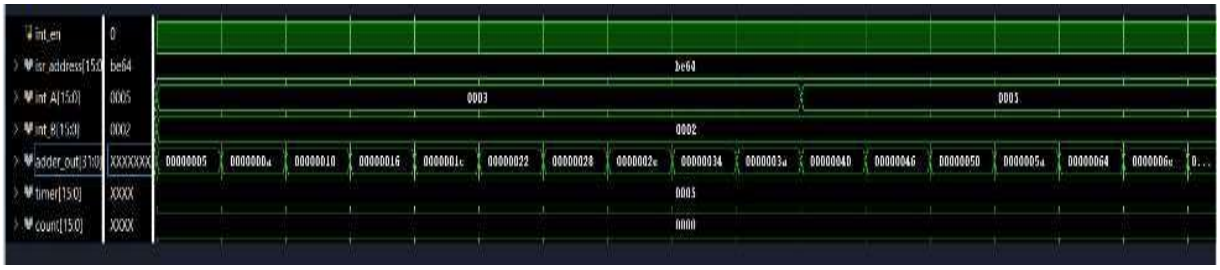
LOAD TREG, MULTIPLY



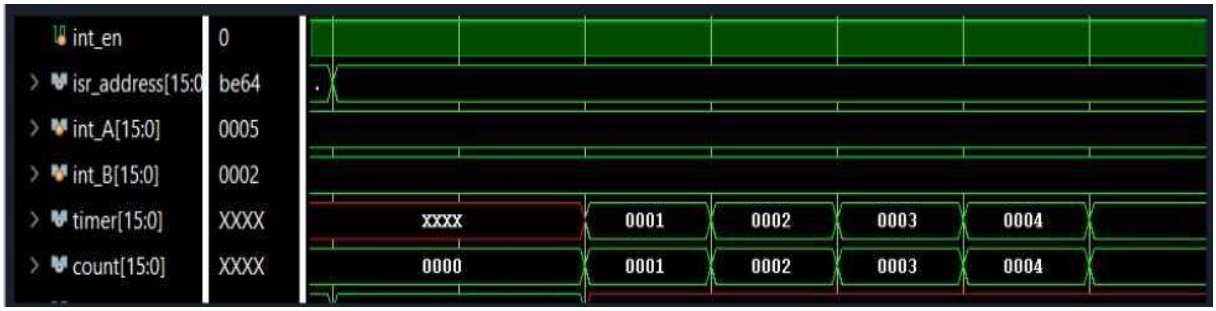
INTERRUPT INTERPOLATOR

INTERRUPT DECIMATOR

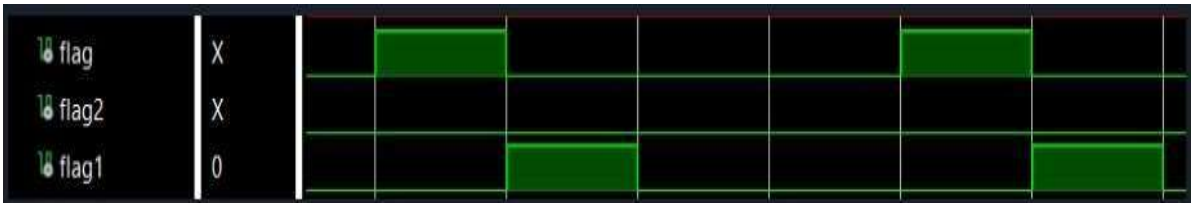
INTERRUPT MAC



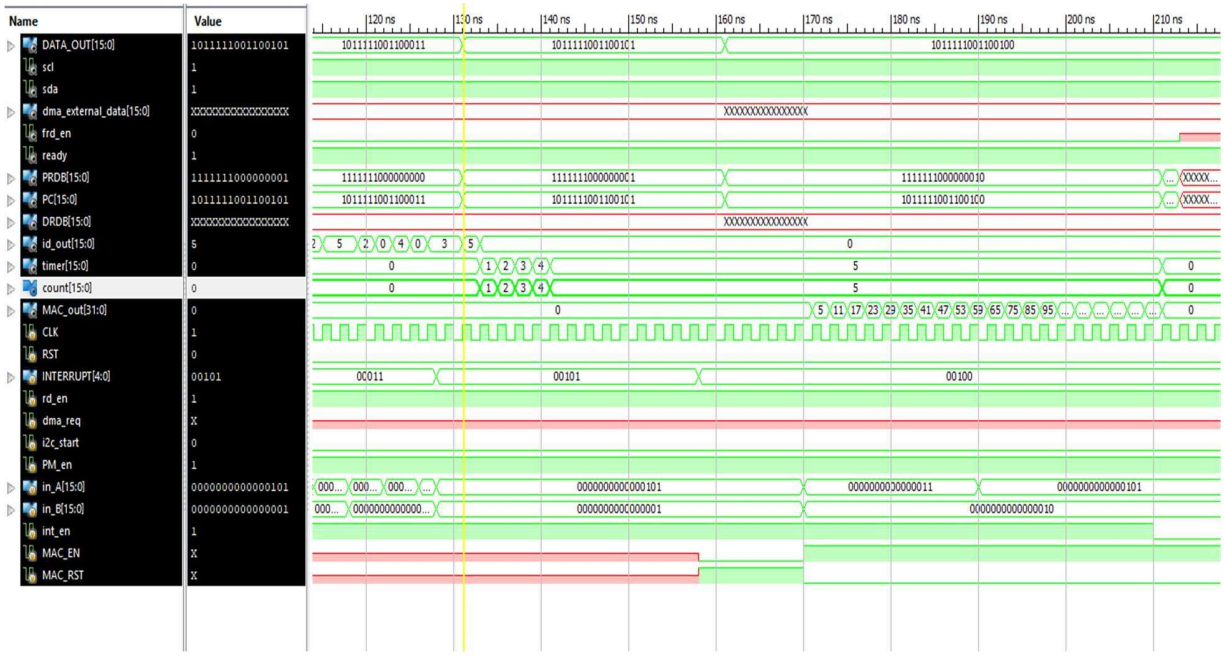
INTERRUPT TIMER



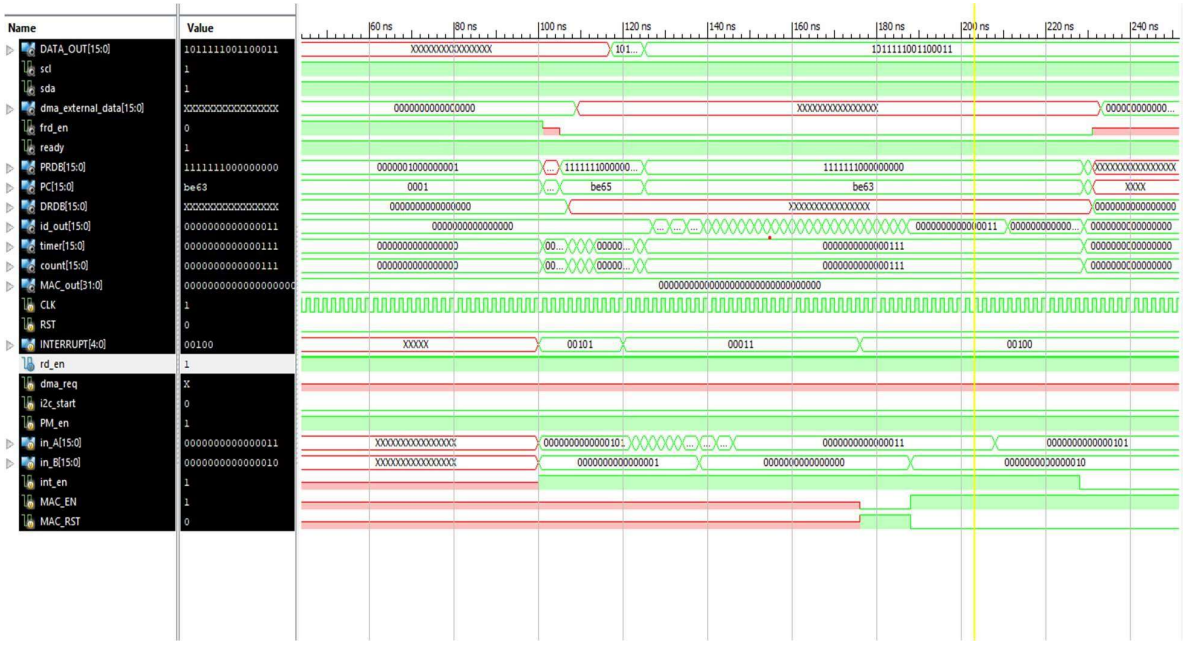
FLAGS FOR PIPELINING



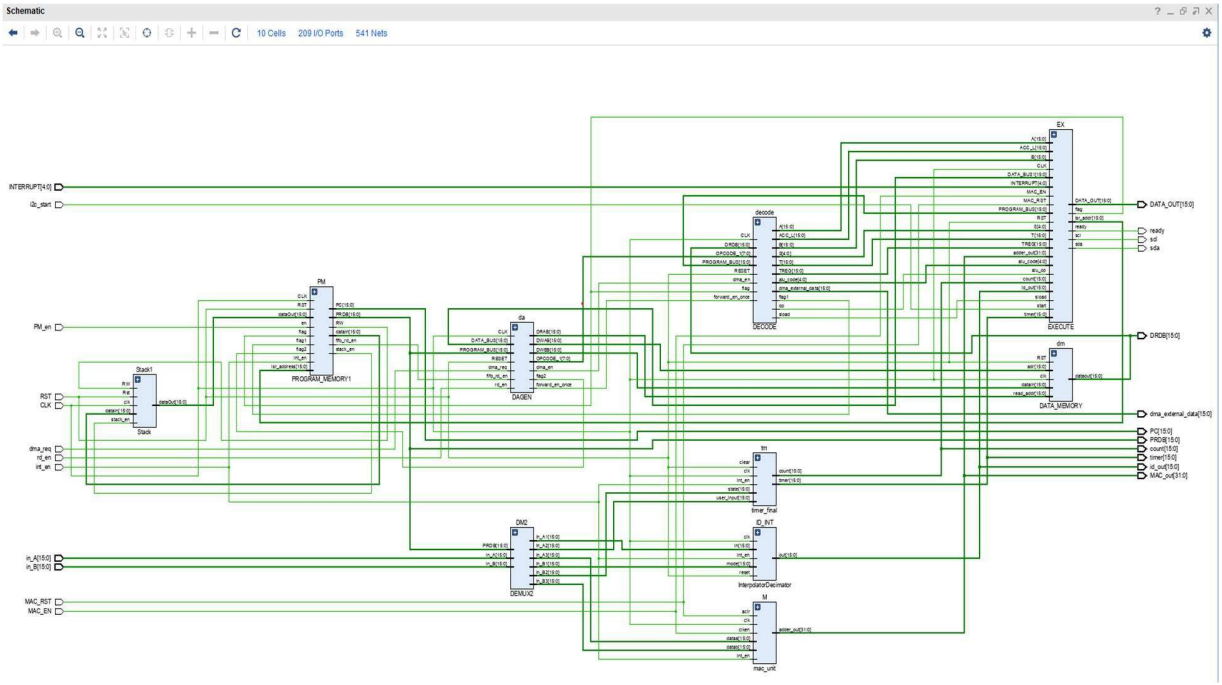
ISE SIMULATION OUTPUT



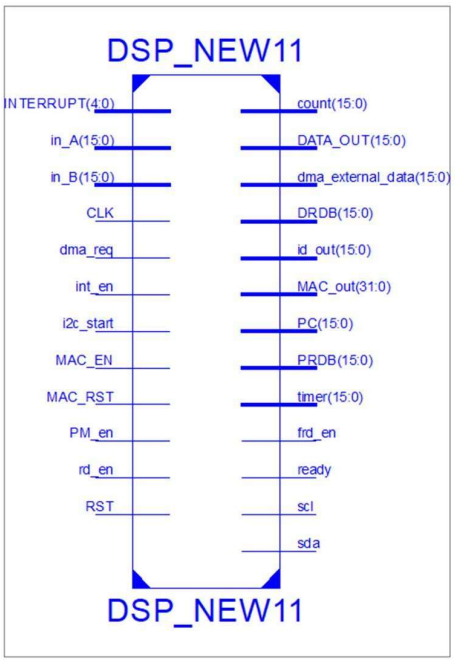
INTERRUPT PRIORITY



VIVADO RTL SCHEMATIC



ISE RTL SCHEMATIC



Summary and Reports:

ISE SUMMARY

DSP_NEW11 Project Status (07/28/2023 - 15:55:54)			
Project File:	NEW_DSP.xise	**Parser Errors:**	No Errors
Module Name:	DSP_NEW11	**Implementation State:**	Placed and Routed
Target Device:	xc7vx485t-2ffg1761	**• Errors:**	No Errors
Product Version:	ISE 14.7	**• Warnings:**	386 Warnings (366 new)
Design Goal:	Balanced	**• Routing Results:**	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	**• Timing Constraints:**	All Constraints Met
Environment:	System Settings	**• Final Timing Score:**	0 (Timing Report)

Performance Summary			[-]
Final Timing Score:	0 (Setup: 0, Hold: 0)	**Pinout Data:**	Pinout Report
Routing Results:	All Signals Completely Routed	**Clock Data:**	Clock Report
Timing Constraints:	All Constraints Met		

Detailed Reports					[-]
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	Fri Jul 28 15:52:35 2023	0	164 Warnings (144 new)	21 Infos (21 new)
Translation Report	Current	Fri Jul 28 15:52:50 2023	0	0	0
Map Report	Current	Fri Jul 28 15:54:24 2023	0	216 Warnings (216 new)	7 Infos (7 new)
Place and Route Report	Current	Fri Jul 28 15:55:30 2023	0	6 Warnings (6 new)	3 Infos (3 new)
Power Report					
Post-PAR Static Timing Report	Current	Fri Jul 28 15:55:53 2023	0	0	4 Infos (4 new)
Bitgen Report					

Secondary Reports		[-]
Report Name	**Status**	**Generated**
ISIM Simulator Log	Out of Date	Fri Jul 28 15:47:48 2023

Date Generated: 07/28/2023 - 16:16:38

SYNTHESIS REPORT

```
=========================================================================
*                      Synthesis Options Summary                        *
=========================================================================
---- Source Parameters
Input File Name                    : "DSP_NEW11.prj"
Ignore Synthesis Constraint File   : NO

---- Target Parameters
Output File Name                   : "DSP_NEW11"
Output Format                      : NGC
Target Device                      : xc7vx485t-2-ffg1761

---- Source Options
Top Module Name                    : DSP_NEW11
Automatic FSM Extraction           : YES
FSM Encoding Algorithm             : Auto
Safe Implementation                : No
FSM Style                          : LUT
RAM Extraction                     : Yes
RAM Style                          : Auto
ROM Extraction                     : Yes
Shift Register Extraction          : YES
ROM Style                          : Auto
Resource Sharing                   : YES
Asynchronous To Synchronous        : NO
Shift Register Minimum Size        : 2
Use DSP Block                      : Auto
Automatic Register Balancing       : No

---- Target Options
LUT Combining                      : Auto
Reduce Control Sets                : Auto
Add IO Buffers                     : YES
Global Maximum Fanout              : 100000
Add Generic Clock Buffer(BUFG)     : 32
Register Duplication               : YES
Optimize Instantiated Primitives   : NO
Use Clock Enable                   : Auto
Use Synchronous Set                : Auto
Use Synchronous Reset              : Auto
Pack IO Registers into IOBs        : Auto
Equivalent register Removal        : YES

---- General Options
Optimization Goal                  : Speed
Optimization Effort                : 1
Power Reduction                    : NO
Keep Hierarchy                     : No
Netlist Hierarchy                  : As_Optimized
RTL Output                         : Yes
Global Optimization                : AllClockNets
Read Cores                         : YES
Write Timing Constraints           : NO
Cross Clock Analysis               : NO
Hierarchy Separator                : /
Bus Delimiter                      : <>
Case Specifier                     : Maintain
Slice Utilization Ratio            : 100
BRAM Utilization Ratio             : 100
DSP48 Utilization Ratio            : 100
Auto BRAM Packing                  : NO
Slice Utilization Ratio Delta      : 5

=========================================================================
```

DEVICE UTILIZATION SUMMARY

```
Device Utilization Summary:

Slice Logic Utilization:
  Number of Slice Registers:                    441 out of 607,200    1%
    Number used as Flip Flops:                  441
    Number used as Latches:                       0
    Number used as Latch-thrus:                   0
    Number used as AND/OR logics:                 0
  Number of Slice LUTs:                       1,824 out of 303,600    1%
    Number used as logic:                     1,786 out of 303,600    1%
      Number using O6 output only:            1,557
      Number using O5 output only:               36
      Number using O5 and O6:                   193
      Number used as ROM:                         0
    Number used as Memory:                       34 out of 130,800    1%
      Number used as Dual Port RAM:              34
        Number using O6 output only:              2
        Number using O5 output only:              0
        Number using O5 and O6:                  32
      Number used as Single Port RAM:             0
      Number used as Shift Register:              0
    Number used exclusively as route-thrus:       4
      Number with same-slice register load:       1
      Number with same-slice carry load:          3
      Number with other load:                     0

Slice Logic Distribution:
  Number of occupied Slices:                    856 out of  75,900    1%
  Number of LUT Flip Flop pairs used:         1,970
    Number with an unused Flip Flop:          1,551 out of   1,970   78%
    Number with an unused LUT:                  146 out of   1,970    7%
    Number of fully used LUT-FF pairs:          273 out of   1,970   13%
    Number of slice register sites lost
      to control set restrictions:                0 out of 607,200    0%

  A LUT Flip Flop pair for this architecture represents one LUT paired with
  one Flip Flop within a slice.  A control set is a unique combination of
  clock, reset, set, and enable signals for a registered element.
  The Slice Logic Distribution report is not meaningful if the design is
  over-mapped for a non-slice resource or if Placement fails.
  OVERMAPPING of BRAM resources should be ignored if the design is
  over-mapped for a non-BRAM resource or if placement fails.

IO Utilization:
  Number of bonded IOBs:                        210 out of     700   30%

Specific Feature Utilization:
  Number of RAMB36E1/FIFO36E1s:                   0 out of   1,030    0%
  Number of RAMB18E1/FIFO18E1s:                   1 out of   2,060    1%
    Number using RAMB18E1 only:                   1
    Number using FIFO18E1 only:                   0
  Number of BUFG/BUFGCTRLs:                       1 out of      32    3%
    Number used as BUFGs:                         1
    Number used as BUFGCTRLs:                     0
  Number of IDELAYE2/IDELAYE2_FINEDELAYs:         0 out of     700    0%
  Number of ILOGICE2/ILOGICE3/ISERDESE2s:         0 out of     700    0%
  Number of ODELAYE2/ODELAYE2_FINEDELAYs:         0 out of     700    0%
  Number of OLOGICE2/OLOGICE3/OSERDESE2s:         0 out of     700    0%
  Number of PHASER_IN/PHASER_IN_PHYs:             0 out of      56    0%
  Number of PHASER_OUT/PHASER_OUT_PHYs:           0 out of      56    0%
  Number of BSCANs:                               0 out of       4    0%
  Number of BUFHCEs:                              0 out of     168    0%
  Number of BUFRs:                                0 out of      56    0%
  Number of CAPTUREs:                             0 out of       1    0%
  Number of DNA_PORTs:                            0 out of       1    0%
  Number of DSP48E1s:                             0 out of   2,800    0%
  Number of EFUSE_USRs:                           0 out of       1    0%
  Number of FRAME_ECCs:                           0 out of       1    0%
  Number of GTXE2_CHANNELs:                       0 out of      56    0%
  Number of GTXE2_COMMONs:                        0 out of      14    0%
  Number of IBUFDS_GTE2s:                         0 out of      28    0%
  Number of ICAPs:                                0 out of       2    0%
  Number of IDELAYCTRLs:                          0 out of      14    0%
  Number of IN_FIFOs:                             0 out of      56    0%
  Number of MMCME2_ADVs:                          0 out of      14    0%
  Number of OUT_FIFOs:                            0 out of      56    0%
  Number of PCIE_2_1s:                            0 out of       4    0%
  Number of PHASER_REFs:                          0 out of      14    0%
  Number of PHY_CONTROLs:                         0 out of      14    0%
  Number of PLLE2_ADVs:                           0 out of      14    0%
  Number of STARTUPs:                             0 out of       1    0%
  Number of XADCs:                                0 out of       1    0%
```

TIMING REPORT

```
Clock to Setup on destination clock CLK
---------------+---------+---------+---------+---------+
               | Src:Rise| Src:Fall| Src:Rise| Src:Fall|
Source Clock   |Dest:Rise|Dest:Rise|Dest:Fall|Dest:Fall|
---------------+---------+---------+---------+---------+
CLK            |    6.756|         |    0.754|         |
---------------+---------+---------+---------+---------+

Pad to Pad
---------------+---------------+---------+
Source Pad     |Destination Pad|  Delay  |
---------------+---------------+---------+
CLK            |scl            |    7.617|
RST            |PRDB<0>        |    9.546|
RST            |PRDB<1>        |    8.435|
RST            |PRDB<8>        |    7.094|
RST            |PRDB<9>        |    6.792|
RST            |PRDB<10>       |    6.784|
RST            |PRDB<11>       |    6.852|
RST            |PRDB<12>       |    7.136|
RST            |PRDB<13>       |    7.140|
RST            |PRDB<14>       |    6.797|
RST            |PRDB<15>       |    7.041|
RST            |frd_en         |    7.038|
RST            |ready          |   12.013|
---------------+---------------+---------+
```

Various Summary and Reports generated by Xilinx ISE for synthesis are studied.

Conclusion:

The modified DSP processor has set of instructions defined in initial block inside Program memory with pipelining and Interrupt controller module with Priority Interrupts. Then it has been synthesized in Xilinx ISE software successfully with no errors. In this project, I learnt a lot about this DSP processor and its architecture with good understanding in program memory, data memory, decode block, execute block, Interrupt controller unit, and interrupt priority. Also, I learnt about how Interrupts and ISR address works. Finally, This project has helped me to improve my knowledge in Verilog and use of engineering software Xilinx vivado and ISE. This modified 16-bit fixed point DSP processor can be implemented widely and has so many applications.