IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p- ISSN: 2278-8735.Volume 14, Issue 1, Ver. I (Jan.-Feb. 2019), PP 31-40
www.iosrjournals.org

Verilog Implementation of a MIPS RISC 32-bit Pipelined Processor Architecture

P. Indira[1], Dr. Ved Vyas Dwivedi[2], Dr. M. Kamaraju[3]
[1]*(ECE Department, CU Shah University, India)*
[2]*(ECE Department, CU Shah University, India)*
[3]*(ECE Department, Gudlavalleru Engg. College, JNT University, India)*
Corresponding Author: P. Indira

Abstract: *Pipelining is a design implementation concept, where elements of a pipeline are often executed in parallel (or) in time-segmented fashion with pipeline registers used as buffer storage. In this work, 3 stage pipelined architecture with 32-bit MIPS RISC processor is used to optimize the design throughput. The high performance features compromise the trade-off between power and speed requirement. In this paper, the possible hazards, and issues with remedies are discussed. In comparative study, different device parameters are compared especially, Power is very much minimized and speed is enhanced when compared to its counterparts. The simulation is carried out with Xilinx 14.3 ISE suite with Verilog HDL coding. MATLAB tool is employed to represent the relationship of various parameters involved in it.*

Keywords: *MIPS RISC Processor, Verilog HDL coding, MATLAB tool, Xilinx.*

Date of Submission: 23-01-2019 Date of acceptance: 07-02-2019

I. Introduction

Multitasking with high performance and optimal throughput are the design requirements (or) expectations of present generation VLSI models [1]. Power, Area and Speed are the critical design parameters and have always a tradeoff between them [2]. A change in one parameter changes the other parameters and makes the system complex. As design complexity increases, it demands innovative research to meet this challenge.

Pipelining is an implementation technique to improve the capacities of the processor and better utilization of hardware and time [3]. The rationale behind using pipeline is to improve the performance goal with increased throughput [4].

MIPS (Microprocessor without Interlocked Pipeline Stage) is a type of RISC (Reduced Instruction Set Computing) processor based Instruction set architecture, which has successfully used in research, industries and many more applications [5]. As low power and high speed are the design goals, one has to prefer RISC processor than CISC (Complex Instruction Set Computing) processor, as the RISC processor has simplified instruction set and balances the speed consideration with low power obligation [6][7][8][9]. With RISC processor, 80% of jobs are performed with 20% of instructions.

In this work, the Virtex-7 FPGA is used to implement the pipelined action. It has the highest performance of the 7-series FGPA devices such as Artix7 and Kintex7. And it is capable of handling the Virtex6 FPGA series.

II. Pipeline Stages of a Processor

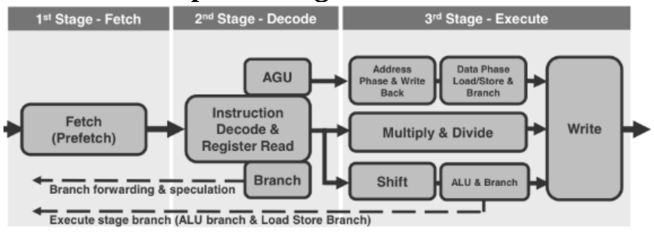

1st Stage - Fetch
2nd Stage - Decode
3rd Stage - Execute
AGU
Fetch (Prefetch)
Instruction Decode & Register Read
Branch
Address Phase & Write Back
Data Phase Load/Store & Branch
Multiply & Divide
Shift
ALU & Branch
Write
Branch forwarding & speculation
Execute stage branch (ALU branch & Load Store Branch)

Fig 1. Three stages of Pipeline connections

2.1 MIPS RISC Processor

The MIPS RISC processor possesses low power, high performance, and enhanced capacity, is reflected in wide range of applications. These include high-end wired communication, test and measurement, advanced RADAR, high performance computing, etc. Its package contains high performance flip chip package, fourth generation sparse chevron pin pattern, speeds up to 2.133 Gbps for parallel I/O, speeds up to 28.05 Gbps for MGT and Discrete Substrate decoupling capacitors.

It enables substantially larger devices and adopts 28nm process Technology FGPA Die. The largest Virtex-7 device is almost 3 times the size of the largest Virtex-6 device. Its growth is higher than Moore's Law dictates. It supports two distinct types of I/O blocks. One is high range type, which has standards up to 3.3V and the other one is high performance type, which has, the more I/O delay capability, and supports standards up to 1.8V. It consists of a flexible pipeline, carries in and out 25 x 18 multipliers, 25-bit pre-adder, 17-bit shifter, 48-bit ALU pattern, and detects dynamic operations with expanded eco-system support.

2.2 Three pipeline stages in processing an instruction:

Pipelining is an implementation technique where the pipeline stages are hooked together to perform part of the operation of an instruction. All the stages must be ready to proceed at the same time. The pipeline designers' goal is to balance the length of each pipeline stage.

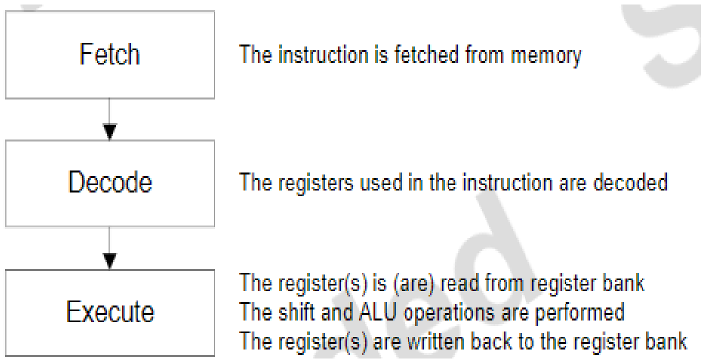

Fetch
The instruction is fetched from memory
Decode
The registers used in the instruction are decoded
Execute
The register(s) is (are) read from register bank
The shift and ALU operations are performed
The register(s) are written back to the register bank

Fig 2. Flow chart of Pipelining stages

3 stage pipelining

To enable several operations to take place at the same time in pipelining, the simultaneous implementation of devices such as memory, integer unit and other units are essential.

Instruction Fetch: In this stage instruction opcode is being fetched from instruction memory and the buffer unit is ready to take the next instruction opcode.

Instruction Decode: In the Instruction Decode opcode is being decoded by the decoder unit. The register bank is also present with this stage to find the effective address. For indirect addressing mode, two clock cycles are needed. Identification of operands for different addressing modes is located through the register bank and handover to the next stage for computation work.

Execute Stage: This is the stage where the combination of the three functions is performed: execution of the operation is performed in Integer Unit and the results are stored in data memory and registers. Load store instructions are also performed in this stage. The integer unit consists of the ALU, shifter, and Multiplier devices. The arithmetic logic unit performs arithmetic and logical operations. The shifter performs the shift/rotate operations. The multiplier performs the multiplication/division operations. All the computation operations are performed at this stage.

2.3 MIPS RISC Processor Description:

Main components

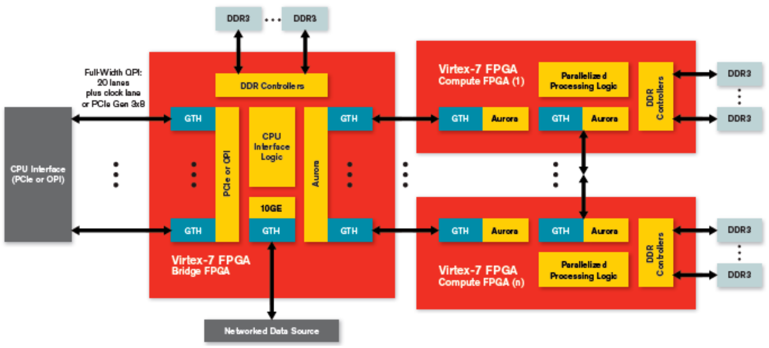

DDR3
DDR3
Full-Width QPI: 20 lanes plus clock lane or PCIe Gen 3x8
DDR Controllers
CPU Interface (PCIe or QPI)
GTH
GTH
PCIe or QPI
CPU Interface Logic
10GE
GTH
Aurora
GTH
GTH
Virtex-7 FPGA Bridge FPGA
Networked Data Source
Virtex-7 FPGA Compute FPGA (1)
Parallelized Processing Logic
GTH
Aurora
GTH
Aurora
DDR Controllers
DDR3
DDR3
Virtex-7 FPGA Compute FPGA (n)
Parallelized Processing Logic
GTH
Aurora
GTH
Aurora
DDR Controllers
DDR3
DDR3

Fig 3. Virtex 7 Processor components

1. DDR3 SDRAM Controller unit [10][11]:

Double data rate Type 3 synchronous dynamic random access Memory is a higher speed successor to DD2 SDRAM controller.

DDR3 memory uses less power (1.35V or 1.5V) than DD2 memory (1.8V or 1.9V). The power consumption of individual SDRAM chips varies with speed, usage, voltage, etc.

It has a pre-fetch buffer of 8-burst-deep, DDR2 pre-fetch buffer has 4-burst deep and with DDR, it is 2-burst deep. This enables the Technology in DDR3 which increases the transfer speed.

DDR3 latencies are numerically higher because the I/O bus clock cycles by which they are measured are shorter: the actual time interval is similar to DDR2 latencies, around 10ns.

For better reliability, detecting the major and minor errors is important. ECC (Error correcting code) bits are used not only to correct minor errors but also used to detect major errors.

READ and WRITE calibration, the introduction of the asynchronous RESET pin, dynamic ODT (On-Die-Termination) allow different termination values for Reads and Writes. The register or buffer memory is an added feature to it.

2. PCI Express:

Peripheral component interconnect (PCI) express is an interface standard for connecting high speed components. Every desk PC motherboard does have a number of PCIe slots to add GPUs, RAID cards, Wi-Fi cards or SSD (Solid State Drive) and add-on cards.

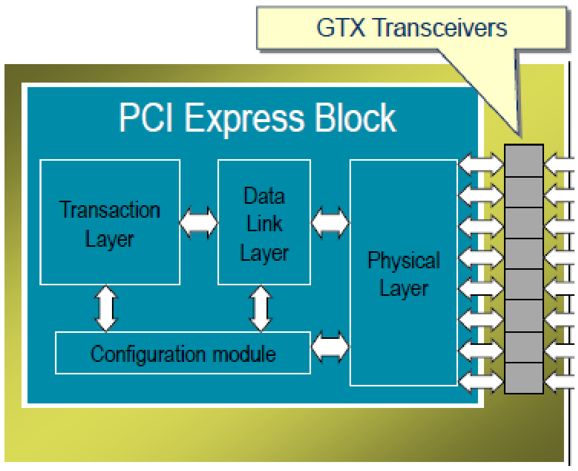


Fig 4. PCI Express Block

CPU Interface Logic: The CPU interface has a special feature that allows and coordinates different logic chips to work together on this platform. Similarly, a chip that contains CPU and a RAM block serves the same purpose.

***Parallelized Processing Logic*:** A parallel bus is a way of mixing dry signal with an effected signal in varying amounts. The two streams should be truly parallel and therefore are independent of each other giving you the freedom to mix them in any manner we needed. To achieve this, we can send all the channels we want to process to a dry subgroup, using the output option. We can utilize the same channel to send to the other subgroup to process.

With the two buses created, you can now add any processing you like, to the second group. This can then be mixed to taste, creating a balance of original, dynamic, untreated signal and the processed secondary group. Its applications are used in compression, limiting, distortion, bit crushing, and even in modulation effects.

Auxilary components

1. Low Power Unit:

Low power design: 28nm high performance low power process technology is used in this processor to achieve the best performance, compromise between critical factors such as speed, delay, area and power. The design of the processor itself implicitly consumes low power without compromising to bandwidth. Further, an intelligent clock gating algorithm is included in Xilinx design tool to reduce the power.

2. Pipeline (Low Power Pulse-triggered flip-flop) Registers:

A new implicit pulse-triggered flip-flop is designed to solve the common transistor stacking problem with the gated Pull-up control scheme. When compared to the other implicit PTFF designs, power is minimized with low layout area and PDP. These Flip-flops are used in pipeline registers to make the Pipelining devour low power.

III. Hazard

Situations that demand in pipeline stages would crave for the required information to continue their operation is called a hazard. It will happen when the next instruction is launched during its allotted clock cycle.

3.1 Structural hazards

This is caused by resource scarcity. If two or more instructions needed the same resource at the same time, this hazard occurs.

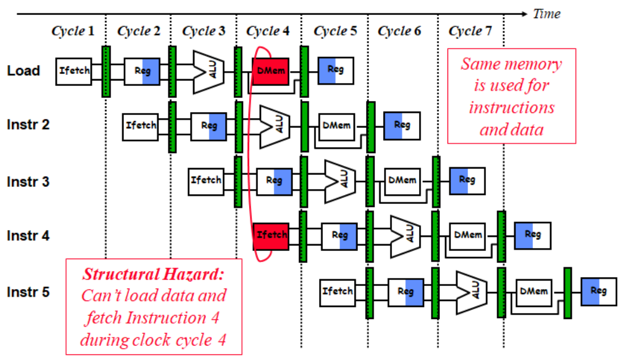

Time
Cycle 1
Cycle 2
Cycle 3
Cycle 4
Cycle 5
Cycle 6
Cycle 7
Load
Instr 2
Instr 3
Instr 4
Instr 5
Ifetch
Reg
ALU
DMem
Same memory is used for instructions and data
Structural Hazard: Can't load data and fetch Instruction 4 during clock cycle 4

Fig. 5 Conflict due to one memory port

Resolving Structural Hazards:

1. **Wait**: Must detect the hazard and create a mechanism to delay instruction access to that resource.

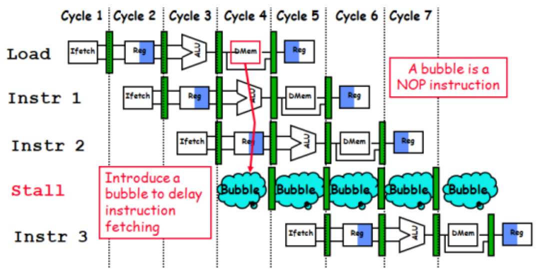


Fig. 6 Detect hazard and delay

2. **Redesign the pipeline**: Add more hardware to eliminate this hazard with two memory ports as instruction cache and data cache.

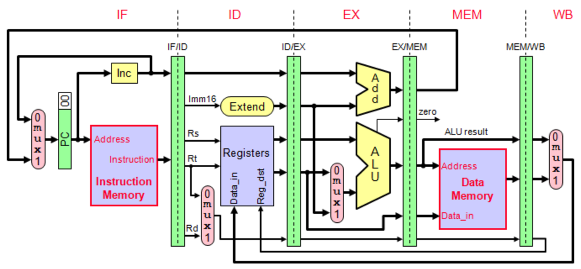
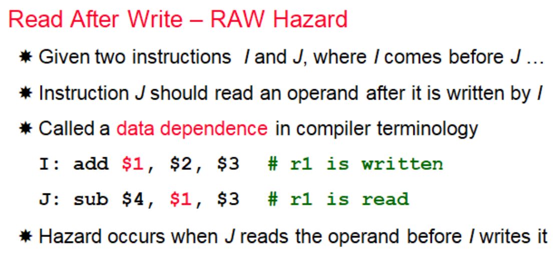


Fig. 7 Add two memory units

3.2 Data Hazards:

Some instructions depend on one another about data. If the data is not reached timely to that stage, then the data hazard occurs.



Read After Write – RAW Hazard

- Given two instructions *I* and *J*, where *I* comes before *J* …
- Instruction *J* should read an operand after it is written by *I*
- Called a data dependence in compiler terminology

```
I: add $1, $2, $3   # r1 is written
J: sub $4, $1, $3   # r1 is read
```

- Hazard occurs when *J* reads the operand before *I* writes it

Resolving the data hazards:

Either create stalls wherever the discrepancies occur about data or forward the required data to that device directly by creating a forwarding unit.

Otherwise, reorder the instructions with compiler scheduling.

3.3 Control Hazards:

Branch instructions always need return address and the branch location to jump on it. If any of the information is missing, it creates a hazard, which is called as a Control Hazard.

The branch instruction is not detected until the ID stage where a new instruction has already been fetched or effective address is not calculated until execute stage – this hazard arises.

Resolving the control hazard:

Delaying the instruction stage to protect the instruction from flushing or to create stalls or reorder the instructions depends upon the dependency of the instructions.

Branch target is predicted by branch prediction buffer to store the prediction bits for the branch instruction.

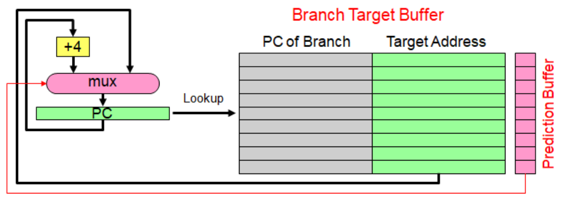


Fig. 8 Branch Preduction Buffer Unit

IV. Issues In Pipelining

Theoretically, in the pipelining process, the total time to complete the number of instructions is less. That means, it is not reducing the individual clock time of the stages. In fact, practically due to some problems, individual clock time may increase. Like that, several issues are hindering the pipelining process. These limitations of controlling the pipeline practically are called 'Issues in the Pipelining'. Here, we are mentioning three different issues.

1) Pipeline latency: This is about the delay introduced in each stage (or) the extra time required to complete a stage.
2) The balance between pipeline stages: The various stages of operation require various clock times and some of them are routine tasks which require very less time and some need more time than the allotted time.
3) In the pipeline stages, different devices propagate different delays according to the instructions and clock skews. The different clock timings and latch delays cause a burden to the pipeline stages and throw them in vain.

V. Results and analysis

Pipelining allows several operations to perform simultaneously (Fetching, processing memory system utilization, computation, storage etc). The 3-stage pipelining consists of fetch-decode-execute stages.

A. *Instruction Fetch (IF) stage:*

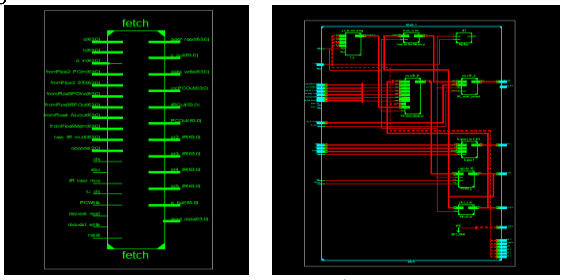


Fig. 9 RTL schematic of Fetch Stage

The first stage of the instruction pipeline is fetch stage. Here the program counter points to the instruction being fetched and stored in the buffer after that PC is incremented to the next instruction to search in address registers. Address selector, address incrementor, address register and instruction memory are used in this stage.

Table no1: Power consumption of fetch stage

Frequency	Power (W)	Time (ns)	Delay (ns)
250MHz	0.00101	3.25	0.283
500MHz	0.002026	4.37	0.326
750MHz	0.003038	5.00	0.444
1000MHz	0.004051	5.23	0.609

B. *Instruction Decode (ID) stage:*

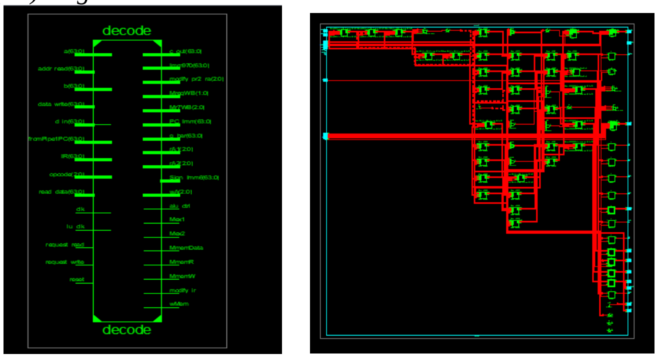


Fig. 10 RTL Schematics of Decode Stage

At this stage, control signals are generated by using control logic and data registers are also accompanied to find the effective address of the operand. The decoder decodes the opcode and passes the operand along with opcode to the next stage.

Table no2: Power consumption of decode Stage

Frequency	Power (W)	Time (ns)	Delay (ns)
250MHz	0.00017	3.00	0.333
500MHz	0.00035	3.36	0.421
750MHz	0.00052	4.02	0.451
1000MHz	0.00070	4.75	0.552

C. Execute stage:

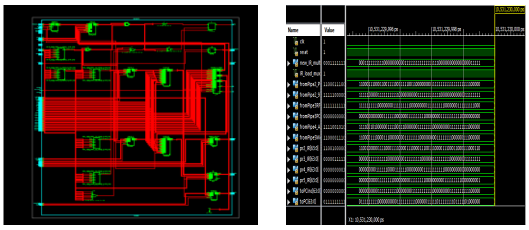


Fig. 11 RTL schematic of Executive Stage

This is the last stage of the 3-stage pipelined architecture where not only arithmetic logic operations are performed but branch operations are also performed by calculating the branch address. Load/store instructions are operated to give and receive the data from the data memory. A shifter and multiplier are also there at this stage to perform the shift/ rotate and multiply/division operations. The results are stored in register bank which are hidden in this stage.

Table no3: Power consumption of Execution Stage

Frequency	Power (W)	Time (ns)	Delay (ns)
250MHz	0.02211	6.59	0.527
500MHz	0.04422	7.77	0.655
750MHz	0.06634	8.67	0.730
1000MHz	0.08845	9.00	1.385

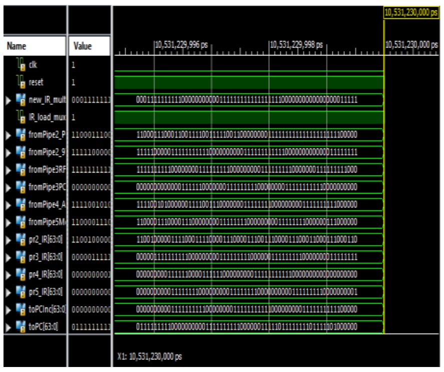
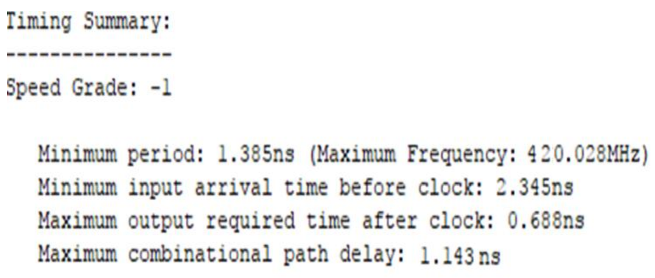


Fig. 12 Output of Pipelining

VI. Software Tool

The software tool to implement the 3-stage pipelining through Virtex7 family processor is 'Xilinx 14.3 instruction set architecture suite'.

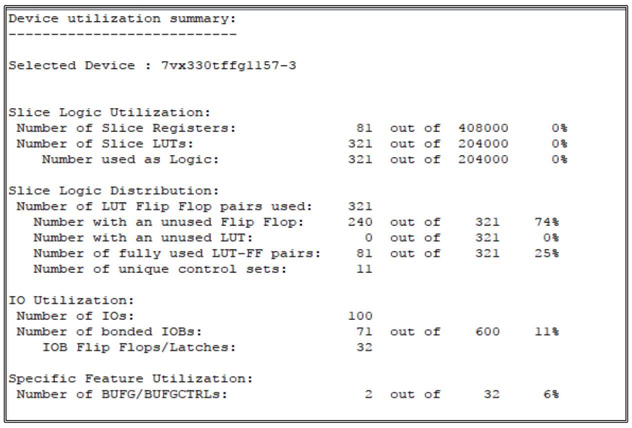


```
Timing Summary:
---------------
Speed Grade: -1

   Minimum period: 1.385ns (Maximum Frequency: 420.028MHz)
   Minimum input arrival time before clock: 2.345ns
   Maximum output required time after clock: 0.688ns
   Maximum combinational path delay: 1.143ns
```

Fig. 13 Time and delay summary report

The timing diagram shows the operating frequency of the pipelining as 420.028 MHz and minimum time to complete the operations as 1.385ns and the maximum path delay as 1.143ns. The clock time duration to perform the operations is also shown in the time and delay summary report.



```
Device utilization summary:
---------------------------

Selected Device : 7vx330tffg1157-3


Slice Logic Utilization:
 Number of Slice Registers:              81  out of  408000     0%
 Number of Slice LUTs:                  321  out of  204000     0%
    Number used as Logic:               321  out of  204000     0%

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:    321
   Number with an unused Flip Flop:     240  out of     321    74%
   Number with an unused LUT:             0  out of     321     0%
   Number of fully used LUT-FF pairs:    81  out of     321    25%
   Number of unique control sets:        11

IO Utilization:
 Number of IOs:                         100
 Number of bonded IOBs:                  71  out of     600    11%
    IOB Flip Flops/Latches:              32

Specific Feature Utilization:
 Number of BUFG/BUFGCTRLs:                2  out of      32     6%
```

The device utilization summary shown above gives the registers I/O devices, flip-flops and other clock-related elements details. The number of slice registers is 81 and slice LUTs are 321 out of 20400, which are very less.

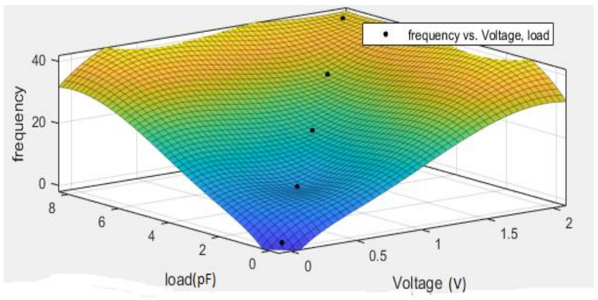


Fig. 14. Fixed voltage and load vs. variable frequency

MATLAB tool is used in this work to draw 3D and 2D graphs for different parameters relations. The above fixed voltage and load versus variable frequency gives the relationship between these three parameters.

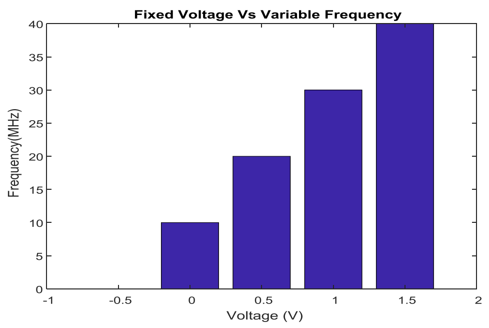


Figure 15. Fixed voltage vs. variable frequency

Voltage increases with frequency variation linearly as shown in the curve as well as in results of the pipeline stages

VII. Comparitive Analysis.

Table no4: Parameter Comparison of various Devices

Parameters [12]	Process Technology	LUTs	Frequency (MHz)	Power (W)
Spartan3: XC3S1500L-4FG676	90nm	417	98.09	0.144
Virtex5: XC5VFX30T –3FF665	65 nm	300	321.048	0.777
Virtex6: XC6VLX75T – 3FF784	40 nm	307	401.881	1.440
Virtex6Low Power: XC6VLX75TL–IL-FF784	40 nm	307	335.233	0.920
Our Model (Virtex 7)	28 nm	321	420.028	0.0233

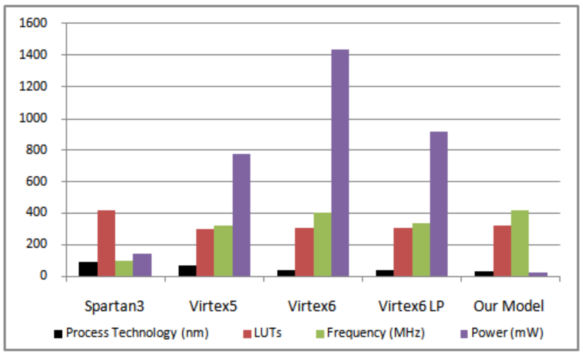


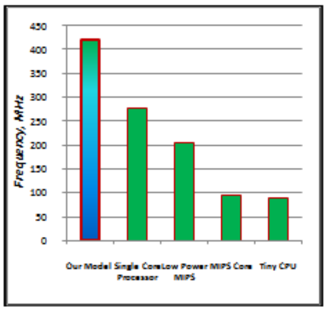
Fig. 16 Device parameter comparison

In power comparison study, our model produces better results in terms of leakage power, dynamic power, and total power with respect to other existing models.

Table no5: Frequency comparison of various pipeline models

Parameters	Our Model	Single Core Processor [13]	Low Power MIPS [14]	MIPS Core [15]	Tiny CPU [16]
Max. Frequency	420.028 MHz	277.9	205.7 MHz	95.5 MHz	89 MHz
LUT	321	1168	1890	2340	336

In the frequency comparison study our model produces the highest speed which is 60% more when compared to the nearest counterpart; similarly, the number of LUTs used is also less.



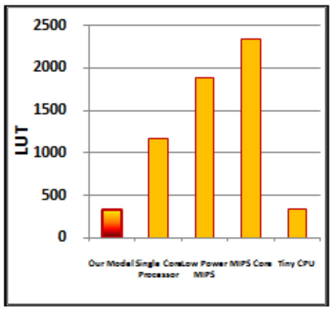


Fig. 17 Frequency comparison **Fig. 18** LUT comparison

VIII. Conclusion

In this research, a 32-bit, three stage pipeline implementation is carried out with virtex 7 family processor core. The advanced features with auxiliary features augment the model performance in terms of speed increase and low power consumption. At each stage, time, delay and power consumption are noted. The hazards and issues along with remedial measures are described. Xilinx software tool with Verilog coding is used to compute the results. MATLAB tool is used to depict the relationship of the process parameters in 2D and 3D graphs.

In a comparison study, our model proves supremacy in terms of frequency, power and area occupied, with other counterparts.

References

[1]. Rashidah F Olanrewaju, Fawwaz E Fajingbesu, Junaid SB, Farhat Anwar, Bisma Rasool Pampori. Design and Implementation of a 5-stage pipelining architecture simulator for RISC-16 instruction set. Indian Journal of Science and Technology. 2017; 10(3).
[2]. Raja Krishnamoorthy, Sarvanan S. A novel flip-flop based error free, area efficient and low power pipeline architecture for finite impulse recursive system. Cluster Computing: Springer. 2018
[3]. Zulkifli M, Yudhanto YP, Soetharyo NA, Adiono T. Reduced Stall MIPS architecture using pre-fetching accelerator. International conference on Electiral engineering and Informatics. 2009
[4]. Saranya Krishnamurthy, Ramani Kannan, Emran Azwan Yahya, Kishore Bingi. Design of FIR Filter using Novel Pipelined Bypass Multiplier. Third International Symposium on Robotics and Manufacturing Automation(ROMA): IEEE.2017
[5]. Trivedi P, Tripathi RP. Design & analysis of 16 bit RISC processor using low power pipelining. International conference on Computing, Communication & Automation (ICCCA): IEEE. 2015; 1294-1297.
[6]. Sneha Mangalwedh, Roopa Kulkarni S, Kulkarni Y. Low Power Implementation of 32-Bit RISC Processor with Pipelining. Proceeding of the Second International Conference on Microelectronics, Computing & Communication Systems. 2017; 307-320.
[7]. Husainali S Bhimani, Hitesh N Patel, Abhishek A Davda. Design of 32-bit 3-Stage Pipelined Processor based on MIPS in Verilog HDL and Implementation on FPGA Virtex7. International Journal of Applied Information Systems (IJAIS). 2016.
[8]. Rakesh MR. RISC Processor Design in VLSI Technology Using the Pieline Technique. International Journal of Innovative Research in Electrical, Electronics, Instrumentation and control Engineering. 2014; 2(4).
[9]. Indu M, Arun Kumar M. Design of Low Power Pipelined RISC Processor. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering. 2013; 2(8).
[10]. Vineeta Singh, Ajinkya Raut, Apurva Lankar, Kanchan Salvekar. Design of DDR3 SDRAM controller. International Journal of Electronics and Computer Science Engineering.
[11]. Komala M, Suvarna D, Nataraj KR. Design and Implementation of High Performance DDR3 SDRAM controller. International journal of Engineering Research and Technology. 2015; 4(5): 857-861.
[12]. Narender Kumar and Munish Rattan. Implementation of embedded RISC Processor with Dynamic Power Management for Low Power embedded system on SOC. Proceedings of 2015 RAECS UIET conference, IEEE.2015.
[13]. Nishant Kumar, Ekta aggrawal. General purpose Six-Stage Pipelined Processor. International Journal of Scientific & Engineering Research. 2013; 4(9).

[14]. Gautham P, Parthasarathy R, Karthi Balasubramanian. Low Power Pipelined MIPS Processor Design. 2009 ISIC proceedings.
[15]. Mamum Bin Ibne Reaz, Shabiul Islam, Mohd S Sulaiman. A single Clock Cycle MIPS RISC Processor Design using VHDL. ICSE 2002 Proceedings,Penang, Malaysia. 2002.
[16]. Koji Nakano, Kensuke Kawakami, Koji Shigemoto, Yuki Kamada, Yasuaki Ito. A Tiny Processing System for Education and Small embedded Systems on the FPGAs. IEEE/IFIP International Conference on Embedded and Ubiquitous Computing. 2008.

IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) is UGC approved Journal with Sl. No. 5016, Journal no. 49082.

P. Indira. "Verilog Implementation of a MIPS RISC 32-bit Pipelined Processor Architecture." IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) 14.1 (2019): 31-40