DESIGN AND ANALYSIS OF A 32-BIT PIPELINED MIPS RISC PROCESSOR

P. Indira[1], M. Kamaraju[2]and Ved Vyas Dwivedi[3]

[1,3]Department of Electronics and Communication Engineering, CU Shah University, Wadhwan, Gujarat, India
[2]Department of Electronics and Communication Engineering,Gudlavalleru Engineering College, JNT University, Kakinada, Andhra Pradesh, India

ABSTRACT

Pipelining is a technique that exploits parallelism, among the instructions in a sequential instruction stream to get increased throughput, and it lessens the total time to complete the work. . The major objective of this architecture is to design a low power high performance structure which fulfils all the requirements of the design. The critical factors like power, frequency, area, propagation delay are analysed using Spartan 3E XC3E 1600e device with Xilinx tool. In this paper, the 32-bit MIPS RISC processor is used in 6-stage pipelining to optimize the critical performance factors. The fundamental functional blocks of the processor include Input/Output blocks, configurable logic blocks, Block RAM, and Digital clock Manager and each block permits to connect to multiple sources for the routing. The Auxiliary units enhance the performance of the processor. The comparative study elevates the designed model in terms of Area, Power and Frequency. MATLAB2D/3D graphs represents the relationship among various parameters of this pipelining. In this pipeline model, it consumes very less power (0.129 W),path delay (11.180 ns) and low LUT utilization (421). Similarly, the proposed model achieves better frequency increase (285.583 Mhz.), which obtained better results compared to other models.

KEYWORDS

MATLAB, SPARTAN3E, MIPS RISC processor, Xilinx, Digital Clock Manager.

1. INTRODUCTION

Currently, the VLSI digital systems design overburdened with many complex features. Multitasking, parallelism makes the system slow and consumes more power to meet these customer requirements; and designers have to compromise with the critical factors [1].

One best way to overcome this problem is an implementation of the pipelining technique in VLSI system design[2]. Pipelining is an implementation technique that several operations of instruction are performed simultaneously to optimize speed, area and throughput of the work [3]. By applying the instruction pipelining, decrease of power, delay, time and enhancement of speed occurs along with the complete utilization of hardware.

The objective of this pipeline process is to minimize the power, increase the speed and to get all benefits of high performance. For that, each element used in this design is power optimized. In

addition to that, low power and high speed techniques enhance the results. Instead of 32-bit architecture, 64-bit and 128-bit complexities of architecture can be tried for challenge.
The problem definition in this proposed architecture is to design a low power high speed pipeline model to achieve less power and latency with low power high performance.

RISC processors are efficient in various ways compared to CISC processors as they consume less power, execute faster as the number of instructions is less andhas simplified addressing modes with simpler designs etc. [4–9].

By using the MIPS RISC processor, besides millions of instructions are performed concurrently without interlocking, qualitatively many advanced desirable functions are also can be executed.

The Spartan 3E FPGAs are the successors of Spartan 3 family with advanced features containing high volume, cost-sensitive, more logic per I/O units and updated programming features without hardware replacement, which is impossible with ASICs designs.

The applications include broadband access, home networking, display/projection, and digital television equipment and simply it is a superior alternative to mask-programmed ASICs.

In this Research implementation, MIPS RISC Spartan 3E family processoris used to evaluate the various parameters through the pipeline to get optimum throughput. The Verilog HDL coding is used to implement the instruction pipelining process on Xilinx platform.

The organization of this paper is as follows: Brief discussion about the low power importance, processor details, scope of the paper, problem definition and objectives are reflected in section 1-Introduction. Section 2 reviewsthe related works of different authors. Section 3 describes the proposed methodology which contains stages of pipelining and main elements of the processor. Section 4 explains the low power high speed techniques to enhance the results. Section 5 is about Hazards and their remedies. Section 6 presents the simulation results of all six stages of pipelining and their analysis. Section 7 describes about the software tools used and the related parameters are narrated through 3D and 2D graphs. The comparative analysis is carried out in Section 8, where frequency, power, LUT and Process technologyare compared. Finally, conclusion and future scope comprehend the process resultsand the additional room of further work, respectively.

2. PROPOSED METHODOLOGY:

In this proposed methodology, 6 stages of pipelining process have been carried out.They are Instruction Fetch Stage, Instruction Decode Stage, Register Read Stage, Memory Access Stage, Data Memory Stage, and Write Back Stage. The instructions are executed, deliberately and systematically going through all the stages.Each stage performs its pre-determined tasks and contributes to the whole task. Registers between each stage helps in buffering during the pipeline process. The time allotted for each stage is given as one clock cycle. If at all any mismatches occur hazards may take place. Thecontrol unit produces NOP signals (stalls) to avoid flushing from pipelining. The expected hazards are taken care by using different techniques with hardware and software protection.

For minimizing the power, each element in pipeline process is carefully selected and implemented. In addition to that, low power technique,i.e., power gating (fine grain method) is applied to all the

devices in pipeline to minimize the power consumption. Similarly balancing pipelining reduces the latency and increases the speed of the process.

The processor used in this architecture is of Spartan 3E family, XC3 1600e device with package FG484 Verilog HDL programming language is used for coding to implement the pipeline process. Xilinx software tool is used for simulation to get the Dynamic power, leakage power and total power for different frequencies. Load/ Store instructions are used for read/write instructions which is associated with data memory.

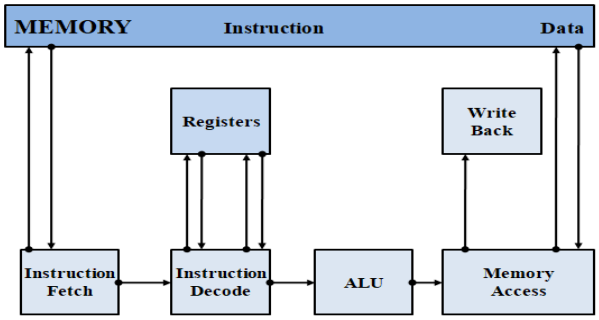

Figure 1. A Pipeline Data path

2.1. Stages of Pipelining

To gain the speed up and further ease of operations, the pipeline has more stages. All the instructions in the pipeline follow the same sequence of simultaneous operations.

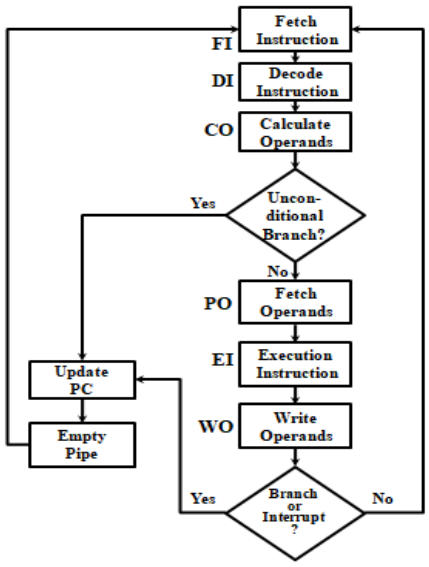

Figure 2. Flow chart of Pipelining stages

2.1.1. Instruction fetch (IF) Stage:

In this stage, the Opcode of the instruction is retrieved from the Instruction Memory (or) Instruction Cache. There is a buffer unit attached at this stage to collect thenext five instruction Opcodes to enhance the performance of the pipelining.

2.1.2. Instruction Decode (ID) Stage:

In this phase, once the Opcode is determined by decoding operation, operand specifierspush this stage to the operand searching. Generally operands are available in Register Bank (or) data Memory.

2.1.3. Calculate Operands (CO) Stage:

The effective address of each operand is estimated in this stage. Depending on the Addressing modes operands are being located. For indirect addressing mode, to locate the operand, it takes 2 clock cycle time. For Direct &Immediate addressing modes only one cycle time is enough to capture the data.

2.1.4. Fetch Operands (FO) Stage:

Generally all operands reside in big Data Memory.To save the time & for ease of operation, Data Cache is used. Operands may be also fetched from registers.

2.1.5. Execute Instruction (EI) Stage:

In this stage, actual indicated operationsare performed with Arithmetic Logic Unit. Obviously this Unit consumes more power and needs high processing speed.

2.1.6. Write back (WB) Stage:

The results are transferred/stored in Data Cache/Data Memory and update the Registers with new information with flag status.

2.2. Fundamental programmable functional elements:

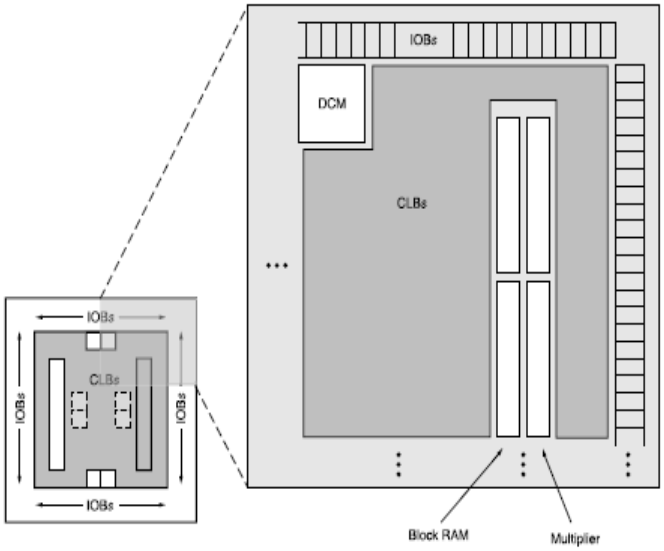

Figure 3. Block Diagram of Processor Core

2.2.1. Main components

2.2.1.1. Configurable Logic Blocks (CLB) :

TheCLBs are meant for performing logic functions and for data storage.

2.2.1.2. Input and output Blocks (IOBs) :

It manages the data flow between input-output pins and internal devices. There are four IOBs Top (B0), Right (B1), Bottom (B2), and Left (B3). Each I/O permits input flow/ output flow together with a three state bus buffer. It consists of unidirectional and bidirectional interfaces with FPGA internal logic. The unidirectional input is the only block that has the sub-set of the full IOB capabilities. Thus there are no connections or logic for an output path.

2.2.1.3. Block RAM:

This unit stores the data in single port and dual port blocks. It also consists of dual port RAM conflicts and resolution block which resolves the data hazards and structural hazards.

2.2.1.4. Multiplier Blocks:

The multiplication can be performed here with 18 bit binary numbers.

2.2.1.5. Digital Clock Manager (DCM):

It is a “self-calibrating”system, which can perform functions such as delaying, multiplying, dividing and phase shifting.

2.2.2. Auxiliary components

2.2.2.1. Package Marking:

The “quad flat packages” are used in this core processor.

2.2.2.2. Input Delay Option:

The “programmable delay block” delays the signal whenever it requires. This adjusts path delay when input flip-flops are used with a global clock.

The delay values are assigned as follows:

IBUF_DELAY_VALUE

2.2.2.3. Storage Element Functions:

There are 3 pairs of storage elements (edge triggered D-flip-flops or a level sensitive latch) work together with a special multiplexer to produce double data rate transmission.

There is a register cascaded feature which intends to simplify the operation and to enhance the speed.

2.2.2.4. Keeper Circuit:

It holds last logic level even though all drivers have been turned off. Pull-up Pull-down resistors overwrite the keeper settings.

2.2.2.5. ESD Protection :

Electro-static Discharge (ESD) effect and voltage fluctuation damages can be eliminated by clamp diodes to protect all the device pads in the system.

2.2.2.6. JTAG Boundary scan capabilities:

It allows the debug / emulation functions regardless of mode pin settings. Selecting JTAG mode cancels the other modes to perform its operation.

The JTAG interface is easily cascaded to any number of FPGAs by connecting TDO output of one device to the other TDI input of the next device in the chain.

2.2.2.7. Program boundary to third party support:

The main system boundary is extended to third party utilization for programs, data, etc. which are connected through socket adopter.

2.2.2.8. Power Distribution System (PDS):

In this, PDS is designed nicely by bypass/ decoupling capacitors. The power on reset (POR) circuit in PDS holds the reset state until the V_{ccINT},V_{CCAUX},and V_{CCO} Bank2 reach their respective threshold levels.

2.2.2.9. No internal charge pumps:

It is a system protection feature. This feature allows FPGA to reject the analog noise when the CCLK configuration clock is ON.

2.2.2.10. Production Stepping:

allows advanced features of stepping 1 amalgamated to the previous version of stepping 0 into it.

2.2.2.11. Simultaneous Switching Outputs:

This feature allows the maximum number of concurrently connected outputs to operate without getting into the danger of switching noise.

2.3. Main components of pipeline model:

2.3.1. Low Power ALU [10]:

The ALU performs all complex tasks, which need more power. A low leakage ALU is designed so that wastage of power is controlled and overall power is reduced. For that, low leakage ALU model is implemented in this pipeline.

2.3.2. Caches:

Instead of using data memory and instruction memory, data cache and instruction cache are used in pipeline process. These small memories are effective in locating the operands and thereby it saves time. Most frequently used & important data is readily utilized by caches in pipeline process than big memories.

2.3.3. Data forwarding Unit and Branch and Jump Prediction Unit [11]:

Data Hazard can be resolved by using Data forwarding unitwhich sends the result in advance. Usage of forwarding unit is an efficient technique that predicts the data required and bypasses some of the stages. A 2 bit branch and jump prediction unit is used to store the destination address of the branch instruction and the current branch status.

2.3.4. DDR4 SDRAM Controller [12]:

DDR4 SDRAM controller is a recent version after DDR3 SDRAM, which is widely used in PC, high-end servers, smartphones, etc. It is basically an interface which bridges the gap between SDRAM devices and Processor sub-system. The main advantage of DDR4 family is reduced power, parallel bank group, faster burst access and better enablement for large capacity memory sub-systems. It has more advanced features: CRC generator, Calibration Unit, Command generation Unit, etc.

2.3.5. Pipeline Registers [13]:

The pipeline registers consist of dual edge triggered implicit Flip-flops (DIFF_CGS). The main advantage of using these types of Flip-flops in pipeline registers is to get the benefit of low power. Almost 10% of the power is getting reduced by using these Flip-flops in registers.

3. LOW POWER AND HIGH SPEED TECHNIQUES:

3.1. Power Gating [14]:

Power gating is a well-known Technique used in this work. Whenever the system is in the inactive state, the circuit gets turned-off automatically, by using this technique. This saves the leakage power in standby mode. The low power units used in this work implicitly reduces the power to provide their contribution in power minimization. Asynchronous blocks (especially ALU) are connected to synchronous blocks with handshake signals and are used in this design to maintain high speeds and utilize time optimally.

3.2. Deeper Pipeline process:

Pipeline technique is used for power saving, effective utilization of hardware&time and to get maximum speed. Actually pipelining doesn't reduce power by itself, it reduces the critical path delay by inserting registers between the combinational logic. At the same time, speed will be increased amd hence the total process time gets reduced.

For Deeper pipeline process, speed can be enhanced more effectively than normal pipelining. For a 6-stage pipeline process, suppose each stage is given 10 units of time. Each stage (task) is

independent and they can be finished in their own time. For example, Instruction Decode stage may take only two units to complete the task while the Execute stage may need 9 units to complete the task. Giving equal time to each stage is not appropriate to obtain maximum speed. And hence, deeper pipelining can be introduced to increase the speed. For that, two units of slots are allocated to the entire pipeline. In that, IF stage uses just two slots (4 units) to complete the task, where as Execute stage uses all 5 slots (9 units) to accomplish the task, which is shown in the Figure 4.

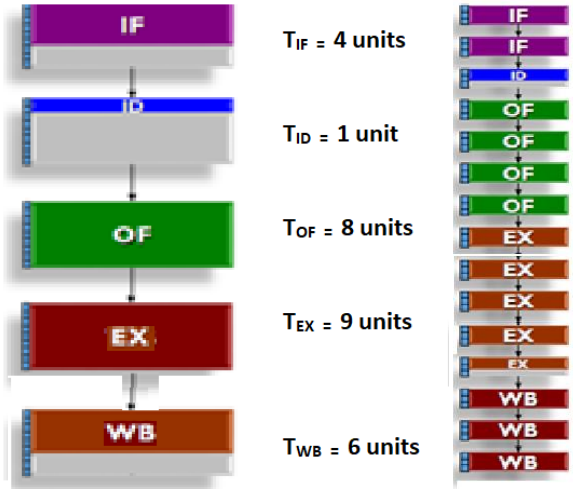

Figure 4. Deeper Pipelining

Before Deeper pipelining, total time taken to complete the task$T_{cyc} = T_{IF} + T_{ID} + T_{OF} + T_{ES} + T_{OS}T_{cyc} = 4 + 1 + 8 + 9 + 6 = 28$sec.After Deeperpipeline:
Speed up = 28/2 = 14 sec.

By using Deeper pipeline method, the speed of the pipeline process can be enhanced by 14 times.

4. HAZARDS

Pipeline Hazards occur, where one instruction cannot immediately follow another in a concurrent instructions execution. Hazards can always be resolved by waiting. Hazards limit the performance of the computers.

4.1. Structural Hazards:

This Hazard occurs, when two (or) more instructions need to utilize the same resource at atime.
Example:

1. One memory unit is used for instruction fetch and datafetch.
2. Since many floating point instructions require many cycles it is easy for them to interfere with each other.

Dealing with structural hazards:

1. Releasing of Stalls:This is a low cost, simple, but increases clock cycles per instruction. Using stalls should be avoided because stalling is a performance effect.

2. Separate Hardware Resources must be allocated, as they are useful for multi-cycle instructions with for good performance,however sometimes it is complex too.
3. Replicate the resources for good performance, however it increases the cost and may introduce inter-connected delay.

4.2. Data Hazards:

Data hazards occur when data is used before it is ready. As shown in the figure below, the ADD instruction releases its result after the execute stage. But that is needed by the SUB instruction even before it is released. The solutions for a data hazards are stalling, forwarding, and reordering.

4.2.1. The Hazard Unit Releases the Stalls to Avoid the Flushing.

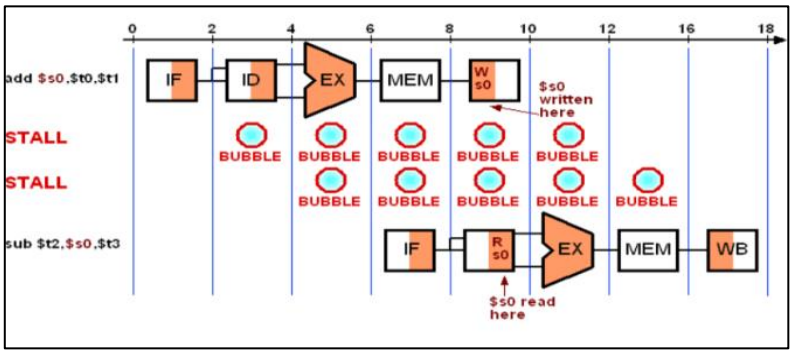

Figure5.Introducing Stalls

4.2.2. The Key Idea to Resolve the Data Hazard Through Forwarding the Data Directly to Next Stage as Shown in the Figure

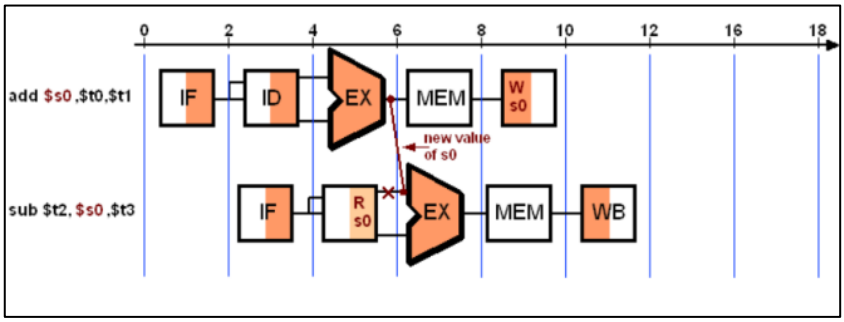

Figure6. Data Forward

Reordering:

It has been addressed only potential data hazards, where the forwarding unit is able to detect and resolve them without affecting the performance of the pipeline. There are also unavoidable data hazards which the forwarding unit cannot resolve. Either stalls or reordering of the instructions can be done by the compiler. The compiler gives the preference to independent instructions to introduce the delay.

4.3. Control Hazards:

If the main program is branching towards a sub-program, it needs the return address in main memory. Then only it jumps towards subroutine. The control hazard occurs if the instruction is not specified in the return address.

Remedies:

- Stop loading instructions until the result is available.
- Assume an outcome and continue fetching but one may lose cycles if it is a mis-prediction.
-

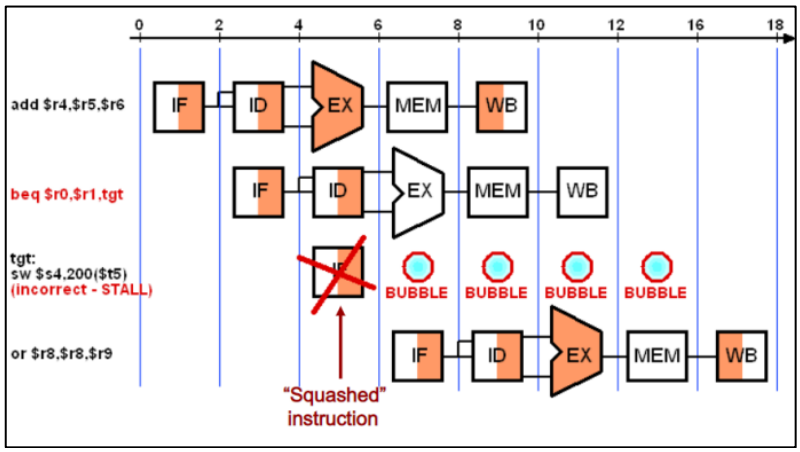

Figure7. Control Hazard – Incorrect prediction

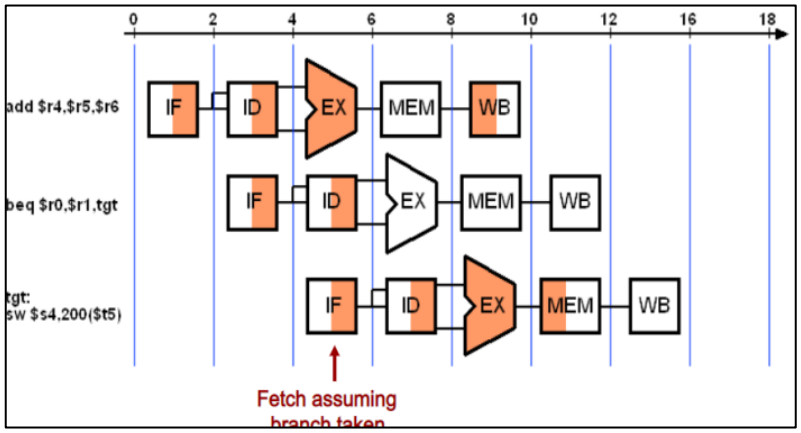

Figure8. Control Hazard – Correct prediction

For each branch encountered during execution, branch predictor predicts whether the branch will be taken or not.

Delayed branches:

The compiler coding is arranged in such a way that preference is given to independent instructions prior to each branching so as to introduce delay.

5. RESULTS AND ANALYSIS:

5.1. Instruction Fetch(IF) Stage:

Figure9. RTL schematics of Fetch Stage

This is the first stage in the pipelining stages, where instruction memory is used to retrieve the opcode of an instruction. The PC is then incremented to the next address fetch by PC +4.

Table 1. Power consumption of Fetch Stage

Frequency (MHz)	Leakage Power (W)	Dynamic Power (W)	Total Power (W)
250	0.0203	0.0002	0.0205
500	0.0204	0,0005	0.0209
750	0.0205	0,0007	0.0212
1000	0.0206	0.0010	0.0216

5.2. Instruction Decode (ID) Stage:

Figure10. RTL Schematics of Decode Stage

In this stage a decode operation is performed by a decoder unit and it is generally attatched with the register banks.

Table 2. Power consumption of Decode Stage

Frequency (MHz)	Leakage Power (W)	Dynamic Power (W)	Total Power W)
250	0.0203	0.0001	0.0204
500	0.0203	0.0002	0.0205
750	0.0203	0.0003	0.0206
1000	0.0203	0.0003	0.0206

5.3. Register Read (RR) Stage:

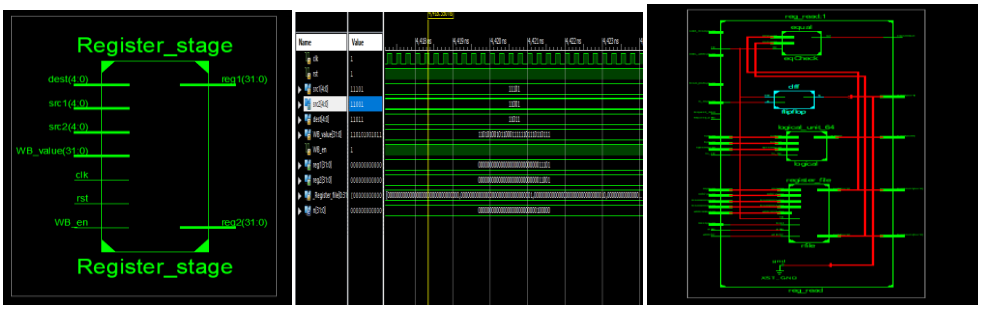

Figure11. RTL schematics of Register Read Stage

In this stage, effective address of the operand is calculated using different addressing modes and locating the required data in register bank or data memory.

Table 3. Power consumption of Register Read Stage

Frequency (MHz)	Leakage Power (W)	Dynamic Power (W)	Total Power (W)
250	0.0204	0.0024	0.0228
500	0.0204	0.0048	0.0253
750	0.0205	0.0073	0.0278
1000	0.0206	0.0097	0.0302

5.4. Execute(EXE) Stage:

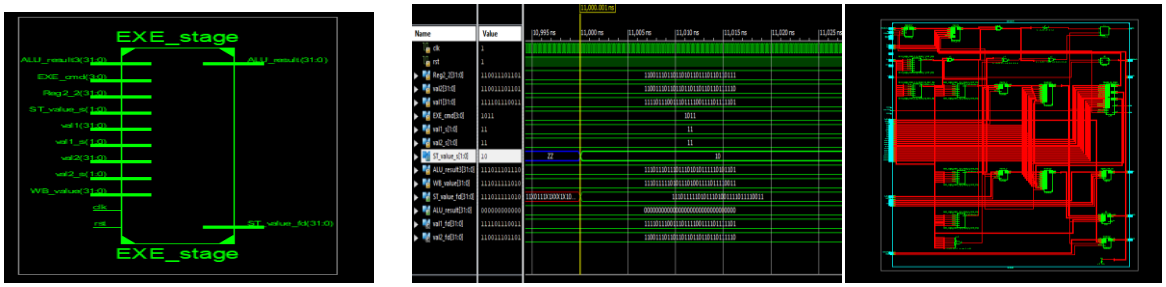

Figure 12. RTL schematics of Execute Stage

In this stage all types of instructions are executed. Depending on the addressing mode this stage may take one or two clock cycles.

Table 4. Power consumption of Execute Stage

Frequency (MHz)	Leakage Power (W)	Dynamic Power (W)	Total Power W)
250	0.0211	0.0025	0.0236
500	0.0217	0.0028	0.0245
750	0.0221	0.0031	0.0252
1000	0.0228	0.0038	0.0266

5.5. Memory –Access Stage:

In this stage load / store instructions are used especially to retrieve the data or to supply the data to the data memory.

Either results or stored data is exchanged between ALU and data memory.

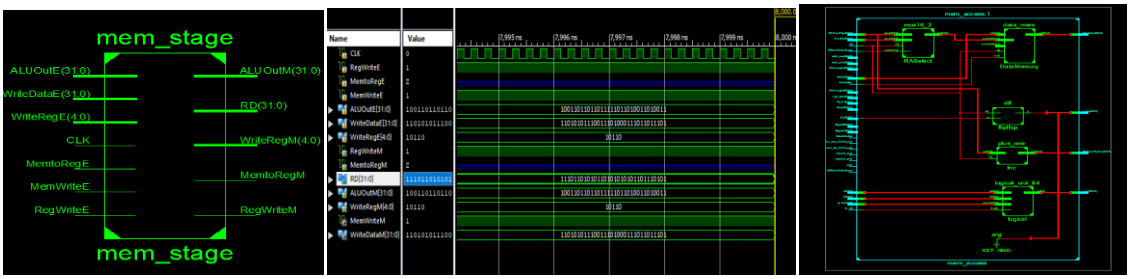

Figure 13. RTL Schematics of Memory Access stage

Table 5. Power consumption of Memory Access Stage

Frequency (MHz)	Leakage Power (W)	Dynamic Power (W)	Total Power (W)
250	0.0209	0.0006	0.0215
500	0.0203	0.0012	0.0216
750	0.0204	0.0018	0.0222
1000	0.0204	0.0025	0.0228

5.6. ‘Write_Back (WB) Stage:

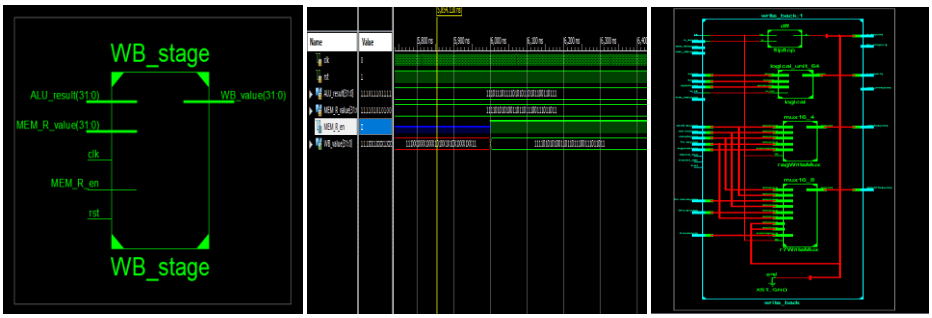

Figure 14. Outputs of Write-back

The results obtained in the execution stage are necessarily stored in the register banks in this stage.

Table 6. Power consumption of Write Back Stage

Frequency (MHz)	Leakage Power (W)	Dynamic Power (W)	Total Power (W)
250	0.0203	0.0004	0.0207
500	0.0203	0.0007	0.0211
750	0.0203	0.0011	0.0215
1000	0.0203	0.0015	0.0218

6. SOFTWARE TOOL AND RELATED REPRESENTATION

Spartan 3E applications can be processed using the Xilinx ISE 8.1i software, which also implements critical bit stream generator updates.

Verilog HDL coding is usedto obtain the data in all the stages. MATLAB Tool is also supported in connected to the parameter relations.

```
Timing Summary:
---------------
Speed Grade: -2

   Minimum period: 3.502ns (Maximum Frequency: 285.583MHz)
   Minimum input arrival time before clock: 3.191ns
   Maximum output required time after clock: 2.395ns
   Maximum combinational path delay:11.180ns
```

Figure 15. Time and delay summary report

Figure 15 shows the clock arrival time and finished time with maximum operating frequency and path delay of the pipelining.

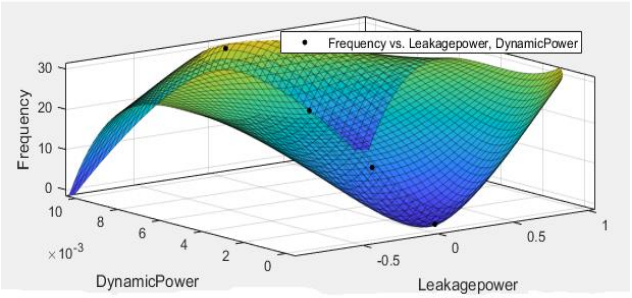

Figure 16. Frequency vs. Leakage Power & Dynamic Power

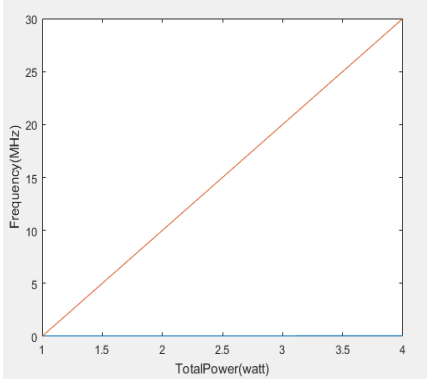

Figure 17. Frequency vs Power

Spartan 3E family FPGA device XC3E1600E is used in this research to obtain the informationabout the parameters in six stages of pipelining with package of FG484. The above curve (Figure 16) represents the relationship with Dynamic power, Leakage power and Frequency and Figure 17 represents the relationship between Frequency and Power.

7. COMPARATIVE ANALYSIS

Table 7.Power comparison of various pipeline models

Parameters [15]	Process Technology	LUTs	Frequency (MHz)	Power (W)
Spartan3: XC3S1500L-4FG676	90nm	417	98.090	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
Proposed Model	90nm	421	285.583	0.129

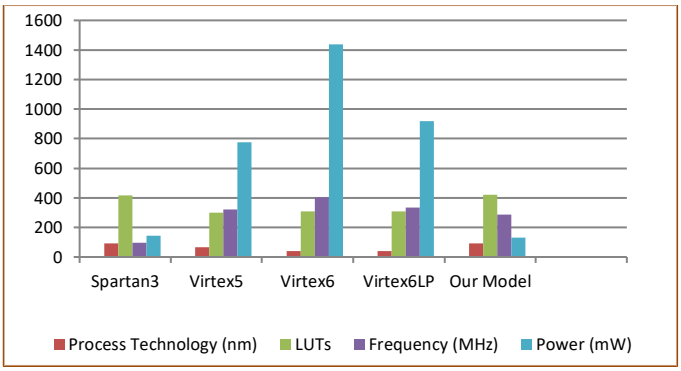

Figure18. Performance Comparison of various process cores

In device comparison graph, comparison is done with the different devices with different parameters such as Process Technology, LUTs, Frequency and Power. The proposed model, (Spartan3E) consumes less power 0.129 W when compared to other device powers.

Table 8. Frequency and LUT comparison of various pipeline models

Parameters	Proposed Model	GPPM[16]	Low Power MIPS [17]	MIPS Core [18]	Tiny CPU [19]
Max. Frequency (MHz)	285.583	277.9	205.7	95.5	89
LUT	421	1168	1890	2340	336

In Frequency comparative analysis, proposed model has obtained the maximum operating frequency (285.583MHz) with the LUTs utilized is 421.When compared to the other models,proposed model gains better results in terms of increased speed and less utilization of slices.

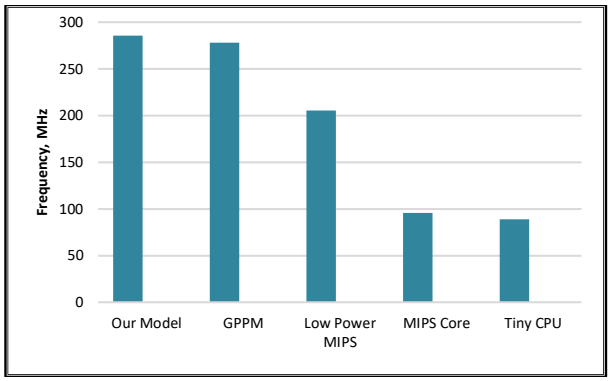

Figure19. Frequency Comparison

8. CONCLUSION

In this research, Spartan 3E family device of MIPS RISC Processor is employed to implement the 6-stage pipelining. Low power high speed techniques were employed to reduce the power and to increase the speed. The proposed model utilizes a total power of 0.129W,which is less than its counterparts. Similarly, the speed of this process is 285.583 MHzwhich is 22% higher than GPPM model, which has the same Spartan 3E device. Moreover, the utilization of number of elements is also less, there by optimizing the area with this design.

Various hazards are analyzed, for which possible remedies are also presented. The dynamic power, leakage power and total power are measured at various frequencies. Various parameters relating to this pipeline process are shown through 3D and 2D graphs by using MATLAB tool. The time taken to complete the whole pipeline processwith propagation delay is also noted, which is in the order of Nano seconds. Verilog HDL coding with Xilinx software tool is used to obtain the simulation results.

In comparison to various pipelining models,the proposedmodel obtains the best results in terms of power, speed, and area utilization.

9. FUTURE SCOPE

The proposed pipeline model uses Spartan 3E Processor with normal pipeline process. Whereas, Virtex 7 processors can use super scalar pipeline which is faster than normal pipelining and super pipelining. In this work, power gating technique is used to minimize the power. Some other power minimization techniques can be combined with this, to further reduce the power and to increase the speed. Spartan 7 processor has more advanced features than Spartan 3E and can be used with more number of pipeline stages, which eases the process.

REFERENCES

[1] Rashid F. Olanrewaju, Fawwaj E Fajingbesi, S.B. Junaid, Ridzwan Alahudin, Farhat Anwar & Bisma Rasool Pampori (2017) "Design and Implementation of a Five Stage Pipelining Architecture Simulator for RiSC-16 Instruction Set", *Indian Journal of Science and Technology*, Vol. 10, No. 3, pp 1-9.

[2] Vijaykumar J, Nagaraju B, Swapna C & Ramanujappa T (2014 April) "Design and Development of FGPA based Low Power pipelined 64-bit RISC processor with Double precession Floating point Unit", International Conference on Communication and Signal processing.

[3] Saranya Krishnamurthy, Ramani Kannan, Erman Azwan Yahya & Kishore Bingi (2017) " Design of FIR Filter using Novel pipelined Bypass Multiplier",*IEEE 3rd International Symposium on Robotics and Manufacturing Automation*, pp1-6.

[4] Sneha Mangalwedhe, Roopa Kulkarni & S. Y. Kulkarni (2017) "Low Power Implementation of 32-bit RISC Processor with pipelining. 2nd International Conference on Microelectronics", *Computing & Communication Systems (MCCS-2017)*, at Bangalore.

[5] Husainali S Bhimani, Hitesh N. Patel & Abhishek A Davda (2016) "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,* Vol. 10, No. 9.

[6] Rakesh M.R. (2014 April) "RISC Processor Design in VLSI Technology Using the Pipeline Technique", *International journal of innovative research in Electrical, Electronics, Instrumentation and control Engineering*, Vol. 2, No. 4.

[7] Indu M& Arun Kumar M. (2013 August) "Design of Low Power Pipelined RISC Processor",*International Journal of Advanced Research in Electrical Electronics and Instrumentation Engineering*, Vol. 2, No. 8.

[8] Priyanka Trivedi &Rajan Prasad Tripathi (2015) "Design & Analysis of 16 bit RISC Processor Using Low Power Pipelining",*International Conference on Computing, Communication and Automation*, pp 1294-1297.

[9] Charu Sharma & Gurupreet Singh Saini (2017 June) "Design and Analysis of High Performance RISC Processor using Hyperpipelining Technique",*IJASRE*, Vol. 3, No. 5, pp 200-206.

[10] Meera S & Umamaheshwari D "Genetic Algorithm for Leakage Reduction through IVC using Verilog" *International Journal of Microelectronics Engineering*, Vol. 1, No. 1, pp 51-62.

[11] Zulkifli.M, Yudhanto.Y.P , Soetharyo N.A, and Adiono.T (2009, August), "Reduced Stall MIPS Architecture using Pre-Fetching Accelerator",*International Conference onElectrical Engineering and Informatics*, IEEE.

[12] Md. Ashraful Islam, Md. Yeasin Arafath, Md. Jahid Hasan (2014, December) "Design of DDR4 SDRAM Controller",*8th International Conference on Electrical and Computer Engineering*, Dhaka, Bangladesh.

[13] Liang Geng, Ji-zhong Shen, Cong-yuan Xu (2016) "Power-efficient dual-edge implicit pulse-triggered flip-flop with an embedded clock-gating scheme", *Frontiers of Information Technology and Electrical Engineering*,Vol. 17, No. 9. PP 962-972.

[14] Aruljothi K, Prajitha PB & Rajaprabha R (2014) "Leakage Power reduction using Power gating and Multi-v_t technique", *International Journal of Advanced research in Computer Engineering and Technology*, Vol. 3, No. 1.

[15] Narender Kumar & Munish Rattan. (2015 December) "Implementation of Embedded RISC processor with Dynamic Power Management for Low-Power Embedded system on SOC", *IEEE Proceedings of 2015 RAECS.*

[16] Nishant Kumar & Ekta Aggrawal (2013, September), "General Purpose Six-Stage Pipelined Processor",*International Journal of Scientific & Engineering Research*, Vol. 4, No.9.

[17] Mamum Bin IbneReaz, Shabiul Islam & Mohd. S. Sulaiman (2002 December) "A single Clock Cycle MIPS RISC Processor Design using VHDL",*In proceedings of IEEE International Conference on Semiconductor Electronics, Penang, Malaysia,* pp 199-203.

[18] Gautham P, Parthasarathy R. & KarthiBalasubramanian (2009 December) "Low Power Pipelined MIPS Processor Design", *In proceedings of IEEE International Conference on Integrated circuits*, pp 462-465.

[19] Koji Nakano, Kensuke Kawakami, Koji Shigemoto, Yuki Kamada & Yasuaki Ito (2008) "A Tiny Processing System for Education and Small embedded Systems on the FPGAs", *In IEEE International Conference on Embedded and Ubiquitous Computing*.

AUTHORS

PONUGUMATLA INDIRA,
M. Tech., MBA, M.Sc. (Psych), (Ph.D.).
Assistant Professor, GITAM University, Hyderabad

DR. M. KAMARAJU,
M. Tech., Ph.D.
Professor, ECE Dept.,
Gudlavalleru Engineering College, JNTUK,
Krishna District, Andhra Pradesh, India

DR. VED VYAS DWIVEDI,
M. Tech., Ph.D.
Professor, ECE Dept.
Pro-Vice Chancellor, CU Shah University,
Wadhwan – Gujarat, India