General Purpose Six-Stage Pipelined Processor

Nishant kumar, Ekta aggrawal

Abstract— This paper proposes design of six stage pipelined processor. The architecture is modified to increase the speed of operation. The architecture of the processor includes the ALU, Pipelined data-path, Data forwarding unit, Control logic, data and program memories and Hazard control unit. Hazard detection unit and data forwarding unit have been included for efficient implementation of the pipeline. Design and verification of processor has been done using Verilog on Xilinx 14.1 platform and ASSEMBLER is written in PERL language which decrease the complexity of instruction writing in program memory is used in this design. As a result this design uses 1168 LUTs and achieves 277.9MHz frequency and it achieves 20% better performance on single-thread program than conventional pipelined processor.

Index Terms— Processor, Pipelining, Data hazard, Perl, Data- forwarding, FPGA, Verilog

1 Introduction

THIS paper proposes a pipelined architecture which improves the maximum operating frequency and throughput of processors by instruction level pipelining. This paper presents the processor architecture having six stages of pipelining with separate data and program memory.

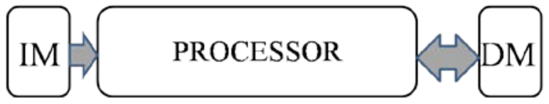

Fig. 1. Processor Architecture

Pipelining is an implementation technique in which multiple instructions are overlapped in execution [5]. Every instruction is divided into several parts and each part is called as a stage. This design have six stage pipeline architecture, namely instruction fetch(IF) ,instruction decode(ID), register read(RR), execution(EXEC), Data memory(MEM) and write back(WB) as shown in Fig.2. GPR (General purpose register) used in the RR stage is a Dual-port RAM so that we can read two register at the same time.

IF
ID
RR
EXEC
MEM
WB

Fig. 2. Stages of pipelined Architecture.

Instruction set of any general purpose processor consists of three types namely Register, Immediate, and Jump. A subset of instruction set has been implemented in this design, total 32 instructions are implemented in this design.

- *Nishant kumar is currently pursuing M.Tech in VLSI & Embedded System Design in Maulana Azad National Institute of Tehnology, Bhopal,India.*
  E-mail:nishantseth07@gmail.com
- *Ekta aggrawal is currently pursuing M.Tech in VLSI & Embedded System Design in Maulana Azad National Institute of Tehnology, Bhopal,India.*
- *Email:aggrawal.ekta@gmail.com*

The organization of this paper is as follows. Section II summarizes the related work. Section III gives the detail of implemented processor architecture; Section IV shows the result and comparison of implemented design. We conclude the paper in section V.

2 Relared Work

Earlier processor are single cycle processor, a single cycle processor is a processor which executes one instruction in one clock cycle, and the period of this clock signal depends upon the maximum time taken by an instruction. In any processing unit maximum time taken by an instruction is due to memory read, memory write and ALU operations. If all these operations are performed in a single cycle then it will take larger time to execute one instruction, and maximum clock frequency that can be applied is reduced. Therefore single clock cycle processors operate at lower frequency [3,6].

To improve the clock speed operation of processor is divided into sub units by applying pipelining. In pipelining execution of each instruction is split into a sequence of dependent steps. [9].These steps are IF (instruction Fetch), ID (Instruction Decode), EXEC (Execution) and WB (Write back to memory). In pipelined architecture multiple instructions executes simultaneously. Pipelining does not reduce the total time taken by an instruction, but it increases the number of instruction that can be executed together, and instruction throughput is increased. Even pipelining introduces latency in the output but the maximum frequency of operation is increased [7].

Performance of any pipelined processor is degraded when an instruction depends on the result of previous instruction or any data which is not yet generated. In any pipelined processor this situation often comes because multiple instruction are executing simultaneously. In this case pipeline is stalled and processor send NOP instruction until the result for which the instruction was waiting is generated. This condition is known as RAW (read after write) Data Hazard [8]. To avoid these hazards, there is a need to forward data which is done by the forwarding unit. Thus Data-forwarding and Hazard detection units are added with the processor to remove hazards [8].

3 Processor Architecture

This section describes the architecture of the implemented processor. The advantages of implemented pipelined processor, which support its practicality, are following.

- *Compatibility*
  In this Pipelined processor 32 instructions are implemented. These instructions are mainly simple instruction such as product-sum operation, logical instruction and jump instruction so that most of the program can execute on this processor.

- *High performance and area efficient*
  This is single–thread pipelined processor. Number of pipelined stages is increased to increase the maximum operating frequency of processor. Comparisons are given in table3.

- *Implementation Scheme*
  This pipelined processor is implemented in Synthesizable Verilog-HDL. Implementation is done for targeting Xilinx Sparten 3E FPGA.

3.1 Instruction Set Architecture

Instruction set of any processor tell about processor complexity. Instruction set of implemented processor consists of register, immediate, jump and branch type instructions. List of implemented instructions for this design are given in Table.1.

TABLE 1

IMPLIMENTED INSTRUCTIONS

INSTRUCTION TYPE	INSTRUCTION
REGISTER	ADD,SUB,AND,OR,XOR,SL,SR, INC,DEC,MUL,DEC,INC,SL,SR,SLC,SRC,DIV,
IMMEDIATE	SUBI,MVIH, MVIL,ADDI,ANDI,ORI,XORI
MEM	LOAD, STORE
JUMP	JZ,JNZ,JC,JUMP
NOP AND HALT	NOP,HLT

3.2 Architecture of pipelined processor

This section describes the architecture of processor. The basic policy of this design is to provide enough practicality of a processor core. The pipelined architecture of implemented processor is shown in Fig.3. The main pipeline stages are

1) IF-Instruction Fetch
2) ID-Instruction decode
3) RR-register read
4) EXEC-Execution
5) MEM-Data Memory
6) WB-write back

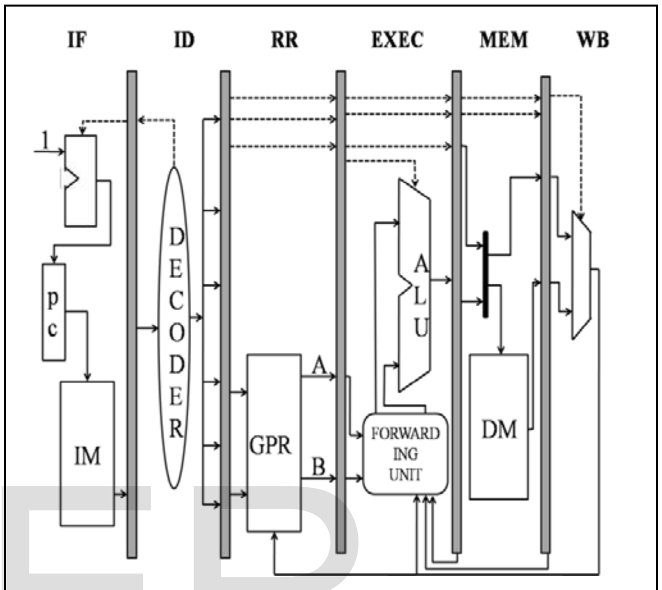

Fig. 3. Single Core Architecture.

Other than these a hazard unit is introduced to remove data hazard, when a data requested by an instruction has not become available it causes data hazards. In earlier to resolve this type of data hazard, pipelining was stalled by the hazard unit until that data become available. In this design data-forwarding is used to remove stalling of pipelining [7].

Working of the pipeline stages are explained below:

3.2.1 Instruction Fetch Unit

In the IF stage instructions are fetched one by one from the instruction memory according to the PC value. Program counter (PC) keeps the track of instruction that is being fetched. Instructions are fetched at every clock cycle from instruction memory.

3.2.2 Instruction Decode

This unit read instruction from instruction register and decodes the operands according to operation. The 16 bit instruction will be divided into several parts.

3.2.3 Register Read Unit

In this unit we calculate the address of read register and this address is send to the GPR. Data from GPR (general purpose register) is read and forwarded to the next unit. GPR is dual ports RAM in which read and write operations can be done simultaneously at different address.

3.2.4 Execution Unit

The main function of this stage is arithmetic calculation. This unit contain ALU unit. The inputs to the ALU are selected by forwarding unit. It decides that from where data is forwarded whether from EXEC-to-EXEC unit forwarding or MEM-to-EXEC forwarding or WB-to-EXEC forwarding [10].

3.2.5 Data Memory

If there is any data to be written or read from the data memory then this unit is used. So this stage is only used for loading and storing instruction, which read and writes the data memory respectively. For other Instruction this unit is not used. The result of the ALU can be directly stored in the data memory. This unit interface with the data memory.

3.2.6 Write Back

This stage is used when we need to write back to the GPR. It is used for writing any data from instruction or storing result of the ALU to the GPR.

3.3 Hazard Handling

In any pipelined architecture, three types of hazards occur [7], they are control hazard, data hazard and structural hazard.

3.3.1 Structural Hazard

This type of hazard occurs when two instructions require same hardware, and there is insufficient hardware [7]. To remove such type of hazard this design is a linearly pipelined and GPR, data memory and instruction memory are separated by pipelining. That's why structural hazard does not occur in this design.

3.3.2 Control Hazard

Control hazard mainly occur when the branch instruction comes and the instruction in the IF and ID are to be rewritten [7]. And the instructions that are in this stage have to be flushed out. But this type of hazard does not occur in this design because in instruction set we are not using any branch instruction.

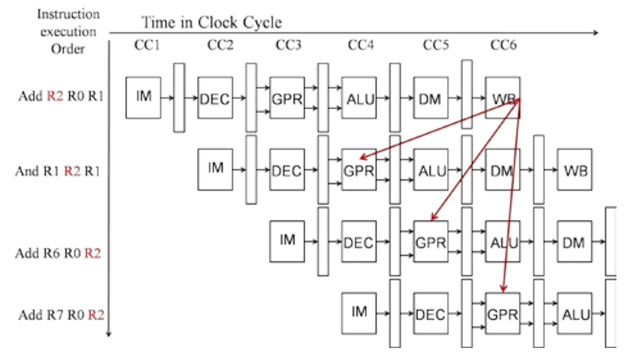

Fig. 4. *Data dependencies in pipelining.*

3.3.3 Data Hazard

This type of hazard commonly comes when an instruction depends on the result of previous instruction or any data which is not yet generated. In Fig.4 one instruction will generate a result which will written in the R2 in Clock Cycle 6(CC6) but next instruction need this result in CC4 and a false data will read from R2 in CC4, to remove such problem we have to wait until the result will be written to the register R2 in CC6. Pipelining is stall for some clock cycle and it will decrease the performance of the processor [7].Fig.4 shows the example pipelined data dependencies and Fig.5 and Fig.6 shows how these dependencies are resolved.

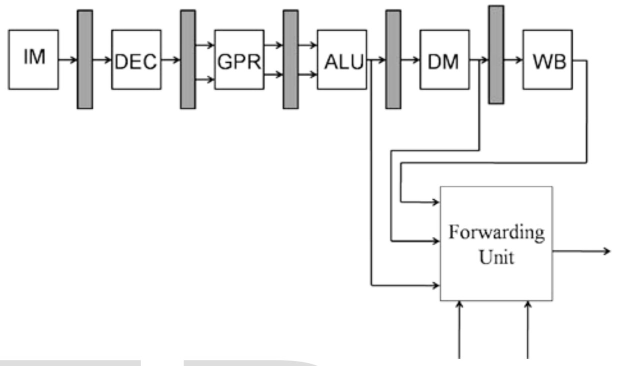

Fig. 5. *Forwarding Unit.*

To remove such type of hazard data forwarding technique is used shown in Fig 5. Hazard unit Compare the addresses of the Source registers of Current instruction decoded in ID unit with the address of destination register of the previous 3 instruction, according to these address Hazard unit send control signal to the forwarding unit. Current result of ALU and previous result of ALU which are in next pipelined stage are forwarded to Data forwarding unit so that according to control signal forwarding unit send data to ALU.

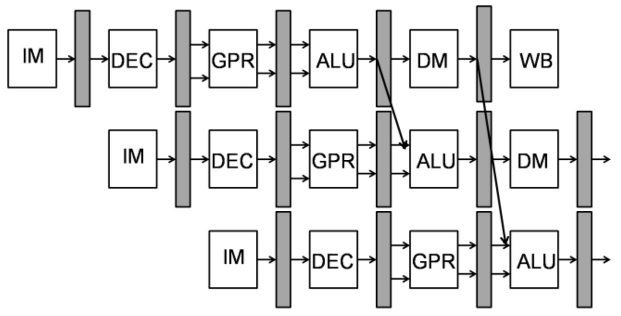

Fig. 6. *Data forwarding.*

Hazard unit Compare the addresses of the Source registers of Current instruction decoded in ID unit with the address of destination register of the previous 3 instruction, according to these address Hazard unit send control signal to the forwarding unit. Current result of ALU and previous result of AlU which are in next pipelined stage are forwarded to Data-forwarding unit so that according to control signal forwarding unit send data to ALU Show in Fig.5. In this case if result of the first instruction is generated by the ALU then this result is feed back to the input of the ALU. And there is no need to wait for result written in the R2. A forwarding unit is used for data forwarding, it forward the result to EX stage from EX, MEM and WB [8].

3.4 Memory Initialization

Writing of any program code in Instruction memory (which is in machine language) is difficult and there are chances of error. Assembler written in *PERL* language is used to make it easier. It takes a Text file as a input in which instructions are written in assembly language and convert it into machine code and generate a output Text file [10].

4 RESULTS

Proposed design analysis has been done for calculation of delay and LUTs using Xilinx 14.1.

Synthesis using Xilinx

We design this architecture in Verilog-HDL, for synthesis we use Xilinx 14.1 and calculate LUTs used and maximum operation frequency. A sample program is written to check the working of implemented processor. The complete Synthesis results to Spartan-3 FPGA are presented in Table 2, whose hardware was fit in an XS3S500E FG320 device. Maximum operating frequency for the design is found to be 277.9 MHz . Table 3 shows the comparison between present work and the mips-core[10], low power MIPS[6] and Tiny-CPU[8].

TABLE 2

XILINX 14.1 SYNTHESIS RESULT

DIVECE	SPARTEN3E XC3S500E FG320
4 I/P LUTs	1168
Flip Flop	618
Max freq	277.9 MHz

The maximum frequency in present design is higher in comparison with the low power MIPS [6], MIPS core [10], Tiny-CPU [8]

TABLE 3

SINGLE-CORE PERFORMANCE COMPARISON WITH MIPS-CORE [10] , LOW POWER MIPS[6] AND TINY-CPU[8].

	Present Paper	Low power MIPS[6]	MIPS-core[10]	Tiny-CPU[8]
Max. Frequency	277.9MHz	205.7MHz	95.5 MHz	89
LUT	1168	1890	2340	336

5 Conclusion

This paper presents a study and design of Scalable six stage pipelined architecture of processor. The design achieves high performance by using pipelining. The speed of operation is increased by 20%, Data Hazard Unit is also used in this design to remove the data hazards by using data forwarding technique.

Reference

[1] P.Bulic,V.Gustin,D.Sonc,andA.Strancar," An FPGA based integrated environment for computer architecture," Comput. Appl. Eng. Educ. 2010.
[2] S.-L.L.Lu,P.Yiannacouras,T.Suh,R.Kassa, and M. Konow, "A desktop computer with a reconfigurable Pentium," Trans. Reconfig.Technol. Syst., vol. 1, pp. 1–15, 2008
[3] A. Clements, "The undergraduate curriculum in computer architecture, "IEEE Micro, vol. 20, no. 3, pp. 13–21, May–Jun. 2000.
[4] D. Patterson and J. Hennessy, Computer Organization and Design: The Hardware/ Software Interface, 3rd ed., 2007.
[5] J. Hennessy and D. Patterson, Computer Architecture: A Quantitative Approach, 4th ed., 2007.
[6] M. S. I. Mamun Bin Ibne Reaz and M. S. Sulaiman, "A single cycle mips risc processor design using vhdl ,"in procedding of IEEE international Conference on Semiconductor Electronics, Dec 2002, pp. 199–203.
[7] A. A. S. Pejman Lotfi Kamran, Amir Mohammad Rahmani and A. A.Kusha, "Stall power reduction in pipelined architecture processors," in Proceedings of the 21st International Conference on VLSI Design, 2008,pp. 541–546.
[8] Gautham.p "Low Power Pipelined MIPS Processor," in proceeding of IEEE international conference on Integrated circuits, Dec 2009,pp 462-465.
[9] Jong Hyuk Lee, Seung Eun Lee, Heon Chang Yu, and Taeweon Suh,"Pipelined CPU Design With FPGA in Teaching Computer Architecture," IEEE Transaction on Education, VOL.55, NO. 3,AUGUST 2012.
[10] Koji Nakano, Kensuke Kawakami, Koji Shigemoto, Yuki Kamada, Yasuaki Ito "A Tiny Processing System for Education and Small Embedded System on the FPGs, " in IEEE International Conference on Embedded and Ubiquitous Computing, 2008.
[11] MicroBlaze Processor Performance, Xilinx, http://www.xilinx.com/products/design/resources/proc_central/microblaze_per.html
[12] Yuhta Wakasugi, Naoki Fujieda Shinya Takamaeda and Kenji Kise, "MipsCoreDuo: A Multifunction Dual-Core Processor, " in IEEE international Symposium on International Signal Processing and Communication System, Dec 2009.