MD.Shabeena Begum, M.Kishore Kumar / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 1, Issue 3, pp.1148-1151

FPGA BASED IMPLEMENTATION OF 32 BIT RISC PROCESSOR

M.Kishore Kumar
Associate Professor, Department of ECE
Sri Vasavi Engg College,
Tadepalligudem.

MD.Shabeena Begum
M.Tech Student, Department of ECE
Sri Vasavi Engg College,
Tadepalligudem.

Abstract

In this paper, a design of general purpose processor with a 5 stage pipeline, to incorporate programmable resources in to a processor. RISC processors have a CPI (clock per instruction) of one cycle. This is due to the optimization of each instruction on the CPU and a technique called pipelining. This technique allows each instruction to be processed in a set number of stages. This in turn allows for the simultaneous execution of a number of different instructions, each instruction being at a different stage in pipeline. The development approach of the overall system design depends on the design specification, analysis and simulation. The RISC Processor core is high performance 32-bit microprocessor. This processor make it especially suited to embedded control applications.

Keywords: Pipeline, clock per instruction, control applications

1. INTRODUCTION

As the high capacity ,low –cost FPGA devices train continues its revolutionary journey through the electronics design Landscape, an ever-increasing number of design landscape, an ever-increasing number of designers are jumping on board trading their traditional hardware-based systems for the attractive ness of the FPGA's 'soft' programmability.

Creation of soft processor based systems, destined to run within a chosen
Target FPGA device, becomes second nature – utilizing one of the many supported flavors of 32 –bit RISC processor, wired up to access peripheral I/O and memory over a standard bus interface.' Soft processors are processors that are defined as part of the FPGA design that is programmed into the physical FPGA device, rather than physical, discrete devices connected to the FPGA, or processors that are immersed as part of the physical FPGA's makeup. Such processors are typically 32-bit and have simple, RISC architectures.

Embedded software refers to the code- the software 'smarts' – that gets downloaded to the physical FPGA device and which will run on a soft processor defined within the FPGA design. The beauty of using 'soft' processors in FPGA designs is that you are not locked to a physical device. We can change processor or modify the code running on it simply by reprogramming the physical FPGA device with a modified hardware design or updated embedded code – leading to true ' field upgradeable hard ware and software'.

2. MICRO ARCHITECTURE

It includes the CPU core, an instruction cache and data cache. You can select an optimum data and instruction cache configuration among a variety of possible configurations.

Micro architectures can be pipelined to different categories. Thus the degree of pipelining is a micro architectural decision.

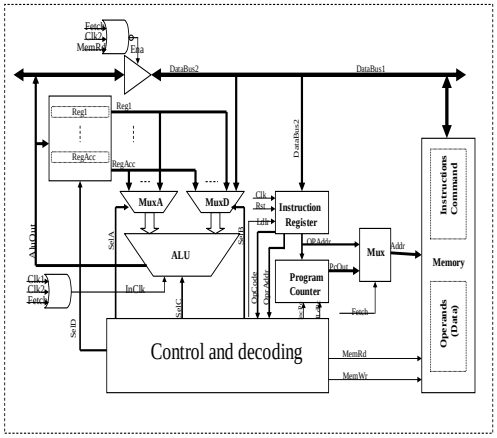


Figure 1: Block diagram of the RISC Processor Core.

Final clock frequency of a specific processor pipeline on a given silicon process technology depends heavily on how deeply the processor is pipelined. When designing a new processor, a key design decision is the target design frequency of operation. The frequency target determines how many gates of logic can be included per pipeline stage in the design. This helps to determine how many pipeline stages there are in the machine.

There are trade offs when the designing for higher clock rates. High clock rates need deeper pipelines so the efficiency at the same clock rate goes down. Deeper pipelines make many things take more clock cycles, such as mispredicted branches and cache misses, but usually more than make up for the lower per-clock efficiency by allowing the design to run at a much higher clock rate. 50%
Increase in frequency might buy only a 30% increase in net performance, nut this frequency increase still provides a significant overall performance increase.

High-frequency design also depends heavily on circuit design techniques, design methodology, design tools, Silicon process technology, power and thermal constrains, etc. At higher frequencies, clock skews and jitter and latch Delay becomes a much bigger percentage of the clock cycles, reducing the percentage of the clock cycle usable by actual logic. The deeper pipelines make the machine more complicated and require it to have deeper buffering to cover the longer pipelines.

The CPU core comprises the following blocks:

CPU register: General purpose register, program counter

CP0 register: Registers for system control coprocessor (cp0) functions

ALU/Shifter: Computational unit

MAC: Computational unit for multiply/add

Bus interface unit: Control bus interface between CPU core and external circuit.

Memory management unit: Direct segment mapping memory management unit.

3. INSTRUCTIONSET OVERVIEW

All Processor Core instructions are 32 bits in length. These are three instruction formats: Immediate (I-type), Jump (j-type) and register (R-type). Having just three instruction formats simplifies instruction decoding.

I-type (Immediate)

31 26	25 21	20 16	15
OP	rs	rt	Immediate

J-type (Jump)

31 26	25
OP	Target

R-type (Register)

31 26	25 21	20 16	15 11	10 6	5
OP	rs	rt	rd	sa	funct

3.1. Load/Store machine with a large number of internal registers:

RISC design philosophy typically uses a large number (commonly 32) of registers. Most instructions operate on these registers, with access to memory made using a very limited set of Load and Store instructions. This limits the need for continuous access to slow memory for loading and stroring data.

3.2 Separate Data Memory and Instruction Memory access paths:

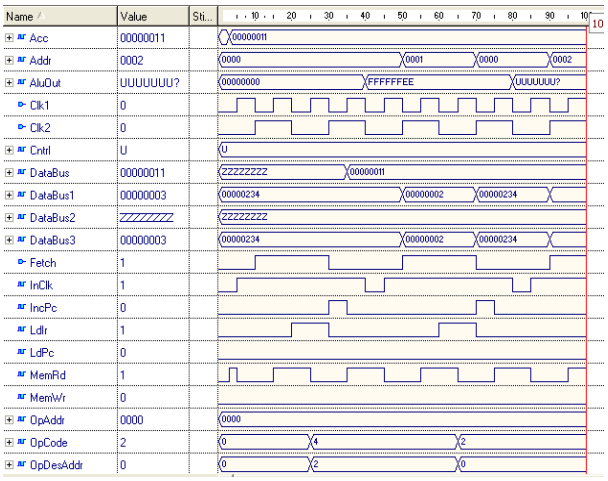
Different stages of the pipeline perform simultaneous accesses to memory. This Harvard style of architecture can either be used with two completely different memory spaces, a single dual-port memory space with separate data and instruction caches for the two pipeline stages.

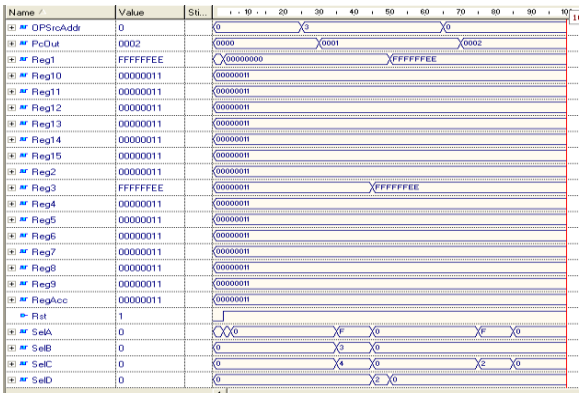
This high-speed ALU core is kept as small as possible to minimize the metal length and loading. Only the essential hardware necessary to perform the frequent ALU operations is included in this high-speed ALU execution loop. Functions that are not used very frequently, for most integer programs, are not put in this key low-latency ALU loop but are put elsewhere. Some examples of integer execution hardware put elsewhere are the multiplier, shifts, flag logic, and branch processing.

The processor does ALU operations with an effective latency of one-half of a clock cycle. It does this operation in a sequence of three fast clock cycles (the fast clock runs at 2x the main clock rate) as shown in Figure 7. In the first fast clock cycle, the low order 16-bits are computed and are immediately available to feed the low 16-bits of a dependent operation the very next fast clock cycle. The high-order 16 bits are processed in the next fast cycle, using the carry out just generated by the low 16-bit operation. This upper 16-bit result will be available to the next dependent operation exactly when needed. This is called a staggered add.

4 SYNTHESIZED RESULTS

The ISA of the 32 bit RISC processor was described using the VHDL .The tool chain including the Active HDL simulator; it was synthesized using Xilinx 9.2i. The total memory usage is 81408 kB. Maximum combinational path delay is 2.677ns. Operating clock frequency about 300 MHZ.





Simulation Results of RISC Processor

5 CONCLUSION

32-bit RISC Processor core has been design and implemented in hardware on Xilinx Spartan 3E FPGA. The design has been achieved using VHDL and simulated with Xilinx 9.2i. Spartan 2E development board has been used for the hardware part. Most of the goals were achieved and simulation shows that the processor is working perfectly, Future work will be added by increasing the number of instructions and make a pipelined design with less

clock cycles per instruction and more improvement can be added in the future work.

REFERENCES

[1]Imyong lee, Dongwook Lee, Kiyoung choi "*ODALRISC: A Small, Low power and Configurable 32-bit RISC processor*" International SOC design conference 2008.
[2] R. Gonzalez, "*Xtensa: a configurable and extensible processor*," IEEE Micro, v.20 n.2, pp.60-70, March 2000.
[3] S. Pees, A. Hoffmann, V. Zivojnovic, and H. Meyr, *"LISA – machine description language for cycle-accurate models of programmable DSP architectures*," in proc. Design Automation Conference, pp.933-938, June 1999.
[4] A. Abd-alla and D. Kartlgaard, "*Heuristic Synthesis of Micro programmed Computer Architectures*," IEEE Trans. on Computers Vol. 23, No. 8, Aug. 1974, pp. 802-807.
[5]P. Liu and F. Mowle, "*Techniques of Program Execution with a Writable Control Memory*," IEEE Trans. on Computers, Vol. 27, No. 9, Sept. 1978, pp. 816-827.
[6] R. Razdan and M.D. Smith, "*A High-Performance Micro architecture with Hardware-Programmable Functional Units*,"Proc. Micro-27, IEEE Computer Society, 1994, pp. 172-180.
[7] S. Hesley et al., "*A 7th-generation x86 Microprocessor*," Proc. IEEE Int'l. Solid-State Circuits Conf., Vol. 42, IEEE Press, 1999, pp. 182-183.
[8] V. iivojnovit, S. Pees, and H. Meyr, "*LISA - machine description language and generic machine model for HW/SW co-design,*" in Proceedings of the IEEE Workshop on VLSI Signal Processing, (San Francisco), Oct. 1996.
[9] R. W. Cook and M. J. Flynn, "*System design of a dynamic microprocessor*," IEEE Trans. Computer., vol. C-19, pp. 213-222, Mar. 1970.
[10] S. Santhanam et al., "*A Low-Cost 300-MHz RISC CPU with Attached Media Processor*,"Proc. IEEE Int'l Solid-State Circuits Conf., Vol. 41, IEEE Press, 1998, pp. 298-299.
[11] R. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, and A. Wang. "*MIS:Multiple-Level Logic Optimization System*".IEEE Transactions on CAD, CAD-6(6):1062-1081, Nov.1987.
[12] G. Hadjiyiannis, S. Hanono, and S. Devadas, "*ISDL: An instruction set description language for retarget ability*," in Proc. of the ACM/IEEE Design Automation Conference (DAC), Jun. 1997.
[13] P. Athanas and H. Silverman. "*Processor Reconfiguration Through Instruction-set Metamorphosis"*. IEEE Computer, 26(3):11-18, Mar. 1993.
[14] A. Kalavade and E. Lee, *"A hardware–software codesign methodology for DSP applications*," IEEE Design €4 Test of Computers, pp. 16-28, Sept. 1993.
[15]V. Zivojnovid, S. Tjiang, and H. Meyr, "*Compiled simulation of programmable DSP architectures*," in Proc. of IEEE Workshop on VLSI in Signal Processing, Osaka, Japan, pp. 187-196, Oct. 1995.
[16]S. Oberman, "*Floating point division and square root algorithms and implementation in the AMD-K7 microprocessor,*" in Proc. 14th IEEE Symp. Computer Arithmetic, Apr. 1999, pp. 106–115.
[17] T. Rauscher and A. Agrawala.
"*Dynamic Problem oriented Redefinition of Computer Architecture via Microprogramming*". IEEE Transactions on Computers, C-27(1 I): 1006-1014, Nov. 1978.
[18]Michael Gschwind, Valentina Salapura, and Dietmar Maurer "*FPGA prototyping of a RISC Processor Core For Embedded Applications*" IEEE transaction on VLSI systems, vol 9, no.2, April 2001.