International Journal of Engineering Research and Science & Technology

ISSN : 2319-5991
Vol. 4, No. 2
May 2015

www.ijerst.com

Email: editorijerst@gmail.com or editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 4, No. 2, May 2015

Research Paper

AN EFFICIENT IMPLEMENTATION OF POWER AND AREA OPTIMIZED ON-CHIP NETWORK COARSE-GRAINED PROCESSOR

Yalavarthi Ramakrishna Paramahamsa[1]*

*Corresponding Author: Yalavarthi Ramakrishna Paramahamsa ✉ ylrkph@gmail.com

Coarse Grained Arrays (CGAs) with run-time reconfigurability play an important role in accelerating reconfigurable computing applications. It is challenging to design On-chip Communication Networks (OCNs) for such CGAs with dynamic run-time reconfigurability whilst satisfying the tight budgets of power and area for an embedded system. This project presents a silicon-proven design of a circuit-switched OCN fabric with a dynamic path-setup scheme capable of supporting an embedded coarse-grained processor array. The paper involves design of a RISC core processor and simulating it. A Reduced Instruction Set Compiler (RISC) is a microprocessor that had been designed to perform a small set of instructions, with the aim of increasing the overall speed of the processor while executing the instruction along with reduction of area(gate count) and power consumption. The RISC architecture follows the philosophy that one instruction should be performed every cycle. This work presents the design and implementation of a 32 bit RISC soft core processor intended for computer architecture introduction considered to be an effective solution for computer comprehension. The idea of this project was to create a microprocessor as a building block in VHDL than later easily can be included in a larger design. The processor gets the data from the serial device uart which can reduce the transmission cost. It will be useful in systems where a problem is easy to solve in software. However at a high level of complexity it is easier to implement the function along with reduced power consumption and area (gate count). In this project XILINX ISE 12.3i is used for logical verification and further synthesizing.

Keywords: Risc processor, Uart, Sipo and Piso

INTRODUCTION

Granularity is the extent to which a system is broken down into small parts, either the system itself or its description or observation. It is the extent to which a larger entity is subdivided. For example, a yard broken into inches has finer

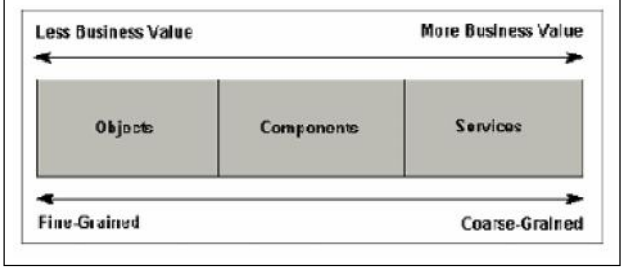
[1] Professor, Brindavan Institute of Technology and Science (BITS), Kurnool 518218, AP, India.

granularity than a yard broken into feet. Coarse-grained systems consist of fewer, larger components than fine-grained systems, a coarse-grained description of a system regards large subcomponents while a fine-grained description regards smaller components of which the larger ones are composed.

Service composition lets develop business processes, applications, and complex services using simpler services from diverse environments. Here's a look at the key role played by coarse-grained interfaces. One of the primary benefits of Service-Oriented Architecture (SOA) is the ability to compose applications, processes, or more complex services from other less complex services. This activity, sometimes called service composition, allows developers to compose applications and processes using services from heterogeneous environments without regard to the details and differences of those environments. This SOA feature depends greatly on the services being constructed and exposed with course grain interfaces. Service granularity refers to the scope of functionality a service exposes. Fine-grained services might be services that provide a small amount of business-process usefulness, such as basic data access. Slightly more coarse-grained services might provide rudimentary operations, which are valuable to system experts, but are not of much value to a business-process expert. Services of the most value to business experts are constructed from lower-level services, components, and objects that are intelligently structured to meet specific business needs. These coarse-grained services can be created from one or more existing systems by defining and exposing interfaces that meet business-process requirements.

As Figure 1 illustrates, the level of granularity generally depends on the purpose of the software entity. The level of granularity for services tends to be coarser than the level of granularity for objects or components. A service typically exposes a single, discrete business process.

Figure 1: Degrees of Granularity



Coarse Grained Reconfigurable Architectures

In contrast to FPGA use (fine grain reconfigurable) the area of Reconfigurable Computing mostly stresses the use of coarse grain Reconfigurable Arrays (RAs) with path widths greater than 1 bit, because fine-grained architectures are much less efficient because of a huge routing area overhead and poor routability. Since computational data paths have regular Structure, full custom designs of Reconfigurable Data Path Units (RDPUs) can be drastically more area-efficient, than by assembling the FPGA way from single-bit CLBs. Coarse-grained architectures provide operator level CFBs, word level data paths, and powerful and very area-efficient data path routing switches. A major benefit is the massive reduction of configuration memory and configuration time, as well as drastic complexity reduction of the P&R (placement and routing) problem. Several architectures will be briefly outlined Some of them introduce multi-granular solutions, where more coarse granularity can be achieved by bundling of resources, such as e.g., 4 ALUs of 4 bits each to obtain a 16 bit ALU.

Network-On-Chip

To meet the growing computation-intensive applications and the needs of low-power, high-performance systems, the number of computing resources in single-chip has enormously increased, because current VLSI technology can support such an extensive integration of transistors. By adding many computing resources such as CPU, DSP, specific IPs, etc to build a system in System-on-Chip, its interconnection between each other becomes another challenging issue. In most System-on-Chip applications, a shared bus interconnection which needs arbitration logic to serialize several bus access requests, is adopted to communicate with each integrated processing unit because of its low-cost and simple control characteristics. However, such shared bus interconnection has some limitation in its scalability because only one master at a time can utilize the bus which means all the bus accesses should be serialized by the arbitrator. Therefore, in such an environment where the number of bus requesters is large and their required bandwidth for interconnection is more than the current bus, some other interconnection methods should be considered. Such scalable bandwidth requirement can be satisfied by using on-chip packet-switched micro-network of interconnects, generally known as Network-on-Chip (NoC) architecture. The basic idea came from traditional large-scale multi-processors and distributed computing networks.

RISC PROCESSOR

The design of 32-bit RISC processor incorporates various design blocks like Arithmetic Logic Unit (ALU), Accumulator, Program Counter (PC), Instruction Register (IR), Memory, Control Unit (CU), and additional logic which can handles 32 bit data, 28 bit address and Uses fixed instruction format of length 32 bit.Size of opcode is of 4 bit which can handling 15 instructions with a 256 memory locations.

Let us consider an instance when some information is stored in the memory. Now when the system is switched on, CPU is initialized. In order to fetch an instruction, as a result the program goes to the location in the memory that is pointed out by the program counter. After some instance, the instruction from the memory is put on the data bus. This cycle is called the instruction fetch cycle. The instruction is now available at the data bus. at next instance; the instruction is loaded into the instruction register. This is called the instruction load. In this cycle the 4 msb's of the instruction are separated and put in the opcode register and are loaded to control unit as well as ALU. The rest of the bits are sent out as Irout. The outputs of the instruction register and the program counter are connected to a mux. During the negative edge of the fetch signal, the output of the instruction register is selected, while the output from the program counter is selected during the positive edge of fetch cycle.

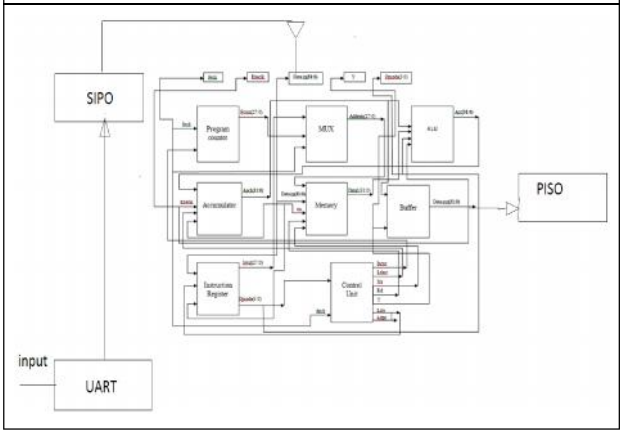
Now when the fetch signal goes low the mux selects the output from the instruction register and it points to the location of the operand. Now the operand present in the location is placed on the data bus. After an instruction is fetched the program counter is incremented. It points to the next location.Now the operand is available at the ALU. The operand is taken in by the ALU and operates on it. Now the result is available at acc1 at positive edge of execlk. During the negative edge of execlk, the result at the Acc1 register is placed on the data bus, which is sent and loaded into the accumulator for any further operations. If

the data has to be stored into the memory, then during this clock cycle, Rd and Wr has to be 0 and 1 respectively. As a result the accumulator is connected to the memory and the value in the accumulator is sent back to a location in the memory through a module named Buffer. A characteristic of RISC processor is their ability to execute one instruction per clock cycle.

DESIGN ISSUES

The UART takes bytes of data and transmits the individual bits in a sequential fashion. UART is serially based device. The sender and receiver must agree on timing parameters in advance and special bits are added to each word, which is used to synchronize the sending and receiving units. In general, UART contains of two main block, the transmitter and receiver block. The transmitter sends a byte of data bit by bit serially out from UART while UART receiver receives the serial in data bit by bit and converts them into a byte of data.

Figure 2: Block Diagram of the on Chip Network Processor



UART starts the data transmission by asserting a bit called the "Start Bit" to the beginning of each data that is to be transmitted. The Start Bit is also used to inform the receiver that a byte of data is about to be sent. After the Start Bit, the individual bits of the "byte" of data are sent, with the Least Significant Bit (LSB) being sent first. Each bit in the transmission is transmitted for exactly the same amount of time as all of the other bits. On the other, UART the receiver will need to sample the logic value that being received at approximately halfway through the period assigned to each bit to determine if it is logic 1 or logic 0.

When a byte of data has been sent, the transmitter may add a "Parity Bit". The receiver to perform simple error checking may use the Parity Bit. In this project, parity bit is not being implemented. After this, a "Stop Bit" is sent by the transmitter to indicate the transmitter has completed the data transmission. If another byte of data is to be transmitted, the Start Bit for the new data can be sent as soon as the Stop Bit for the previous word has been sent.As we discussed earlier that UART is serially based device, the output also generates in serial fashion. RISC has a relatively few instructions, few addressing modes and few instruction formats. As a result, a relatively small and simple executing hardware subsystem of the CPU is required.

This yields the following results, the chip area, dedicated to the realization of the control unit is considerably reduced. As a result of the reduction , the RISC designer can fit a large number of CPU registers on the chip which enhances the throughput for a large class of programs. RISC is able to process parallel input at a time, so we make use of SIPO register to convert serial output coming out of UART to parallel. Thus RISC processor gives parallel output. As serial interfaces have certain advantages over parallel interfaces. The most significant advantage is simpler wiring. In addition, serial interface cables

can be longer than parallel interface cables, because there is much less interaction (crosstalk) among the conductors in the cable. So the output coming out of RISC is converted to serial using PISO register.

RESULTS

The waveform which is shown above describes the functionality of the designed module. The RTL schematic describes about the user view of the module and the TECHNOLOGY schematic describes the chip view of the module.

Figure 3: Top Module Waveform

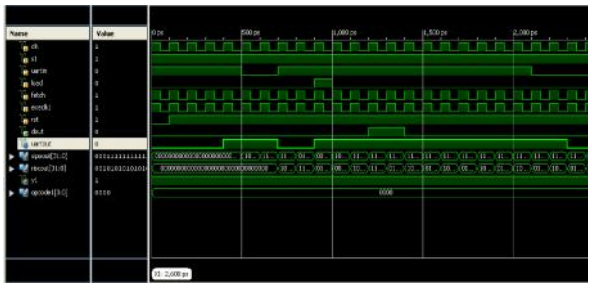


Figure 4: Schematic

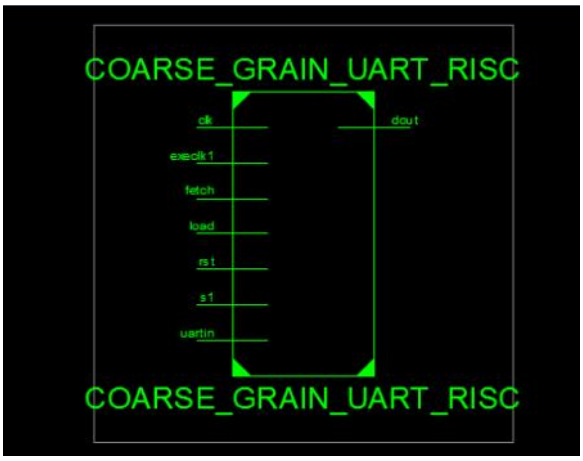


Figure 5: RTL Schematic

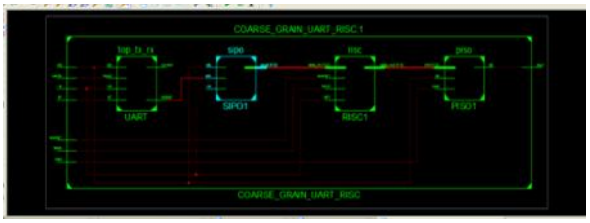


Figure 6: Technology Schematic

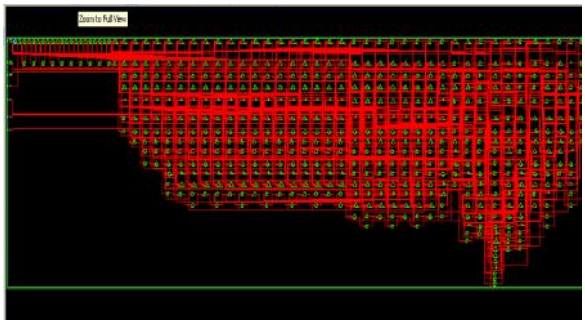
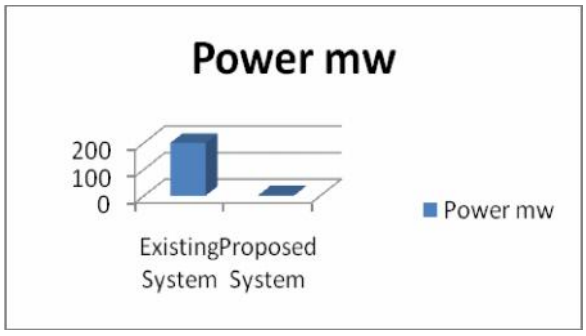


Figure 7: Power Table

	Power
Existing System	200mw
Proposed System	2.706mw

Figure 8: Power Graph



CONCLUSION

This paper presents the design and implementation of a 32 bit RISC soft core processor intended for computer architecture

introduction considered to be an effective power with reduced area solution for computer comprehension. Verilog language is used to develop and RTL. From the power graph obtained this paper concludes the developed chip is utilized in low power applications where power is the main criteria. However at a high level of complexity it is easier to implement the function along with reduced power consumption and area (gate count). In this project XILINX ISE 12.3i is used for logical verification and further synthesizing.

REFERENCES

1. Clermidy F, Bernard C, Lemaire R, Martin J, Miro-Panades I, Thonnart Y, Vivet P and Wehn N (2010), "A 477 mW NoC-Based Digital Basebandfor MIMO 4G SDR", in *ISSCC Dig. Tech. Papers*, pp. 278-279.
2. Hartenstein R (2001), "A Decade of Reconfigurable Computing: A Visionary Retrospective", *Proc. Design, Autom., Test Euro. (DATE)*, pp. 642-649.
3. Pham P-H, Mau P and Kim C (2009), "A 64-PE Folded-Torus Intra-Chip Communication Fabric for Guaranteed Throughput in Network-on-Chip Based Applications", in Proc. IEEE Custom Integr. Circuits Conf. (CICC), pp. 645-648.
4. Platzner M, Teich J and Wehn N (2010), "Dynamically Reconfigurable Systems: Architectures, Design Methods and Applications", Springer, New York.
5. Rodrige S, Silva I S and Azevedo A (2004), "When Reconfigurable Architecturemeets Network-on-Chip", in *Proc. Symp. Integr. Circuits Syst. Design*, pp. 216-221.
6. Rossi D, Campi F, Spolzino S, Pucillo S and Guerrieri R (2010), "A Heterogeneousdigital Signal Processor for Dynamically Reconfigurable Computing", *IEEE J. Solid-State Circuits*, Vol. 45, No. 8, pp. 1615-1626.
7. Truong D N, Cheng W H, Mohsenin T, Zhiyi Y, Jacobson A T, Landge G, Meeuwsen M J, Watnik C, Tran A T, Zhibin X, Work E W, Webb J W, Mejia P V and Baas B M (2009), "A 167-Processor Computational Platform in 65 nm CMOS", *IEEE J. Solid-State Circuits*, Vol. 44, No. 4, pp. 1130-1144.

International Journal of Engineering Research and Science & Technology
Hyderabad, INDIA. Ph: +91-09441351700, 09059645577
E-mail: editorijlerst@gmail.com or editor@ijerst.com
Website: www.ijerst.com