Special Issue of Engineering and Scientific International Journal (ESIJ)
Technical Seminar & Report Writing - Master of Computer Applications - S. A. Engineering College (TSRW-MCA-SAEC) – May 2015
ISSN 2394-187(Online)
ISSN 2394-7179 (Print)

Approaches on Operating System using Reconfigurable Architecture

P.Priya[#1], V.Rekha[*2], M.Vaitheswari[*3]

Department of Computer Applications, S.A Engineering College, Chennai
Pripriya04@gmail.com, rekhavenkatesan23@gmail.com, avinabi3@gmail.com

***Abstract*—**Reconfigurable architecture deals as an introduction to operating systems, their main functions and what types exist, as well as possible benefits gained by placing typical software functionality in hardware. By introducing hardware support in the form of reconfigurable architecture, such as programmable logic devices, not only new functionality and better performance can be achieved but also improved determinism and predictability. These properties make it easier for developers of real-time systems when trying to both calculate and predict timing constrains. Operating systems in hardware are still in the development phase and one version of how an operating system in hardware could look like will be presented, including layers and message passing.

Keywords*—Operating System, Field programmable gate array, Configurable logic block.*

1. Introduction

The architecture of a computer can be very scary to look at. There are instruction sets, memory, registers, buses etc. Operating systems were originally developed to provide a set of common system services, such as I/O, communication, and persistent storage, to simplify application programming. With the advent of multiprogramming, this charter expanded to include abstracting shared resources so that they were as easy to use and sometimes easier as dedicated physical resources.

2. FPGA

FPGA stands for Field Programmable Gate Array and can be programmed using an HDL. FPGAs are made of many configurable logic blocks *(CLBs)* and these blocks are then made of gates, latches, multipliers etc. FPGAs with over one million gates are not unusual today. A matrix of wires and connections that are reconfigurable link these CLBs together with each other and I/O ports As FPGA density increases with VLSI feature sizes below 0.15 micron, the need for time and space optimization of reconfigurable designs will give way as the major focus of users to the need for tools to manage the complexity of systems incorporating in excess of 20 million gate equivalent designs. This will lead to demand for better design tools but will also open the way for the introduction of system software for the management of pre-designed reconfigurable cores, as shown in (Fig. 1) In the area of traditional computing the latter is the preserve of an operating system (OS). Therefore OS like software will be needed for reconfigurable computing (RC).

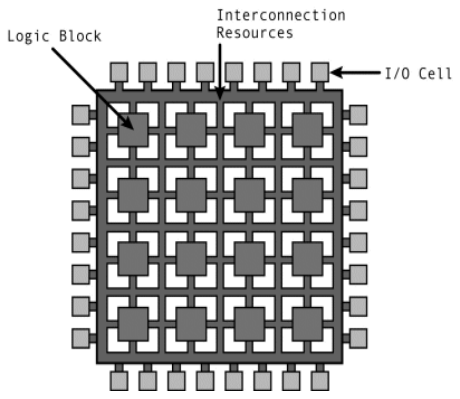

Fig. 1: General FPGA architecture

As the number of system gates available on reconfigurable platforms increase beyond 20 million, the issue of the management of these resources and their sharing among may applications and users will become more of a concern.

2.1 Reconfigurable functional unit

Many reconfigurable systems use commercial FPGAs as a reconfigurable fabric. These commercial FPGAs contain many three to six input lookup tables, each of which can be thought of as a very fine-grained functional unit, as shown in (Fig.2).

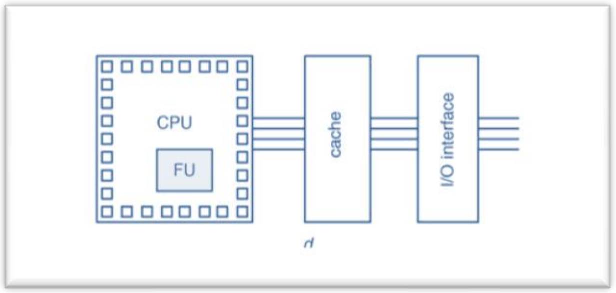

Fig.2: Reconfigurable functional unit

Reconfigurable functional units can be classified as either coarse-grained or fine-grained. A fine-grained functional unit can typically implement a single function on a single (or small number) of bits. The most common kind of fine-grained functional unit is the small lookup tables that are used to implement the bulk of the logic in a commercial field programmable gate array. A coarse-grained functional unit on the other hand, is typically much larger, and may consist of arithmetic and logic units (ALUs) and possibly even a significant amount of storage.

2.2 Reconfigurable fabric

The heart of any reconfigurable system is the reconfigurable fabric. The reconfigurable fabric consists of a set of reconfigurable functional units, a reconfigurable interconnects, and a flexible interface to connect the fabric to the rest of the system. In this Section, we review each of these components, and show how they have been used in both commercial and academic reconfigurable systems, as shown in (Fig. 3)A common theme runs through this entire section: in each component of the fabric, there is a tradeoff between flexibility and efficiency. A highly flexible fabric is typically much larger and much slower than a less flexible fabric. The RC research community thus needs to further investigate the requirements and technical implementation of reconfigurable system software and in particular an operating system for reconfigurable computing.

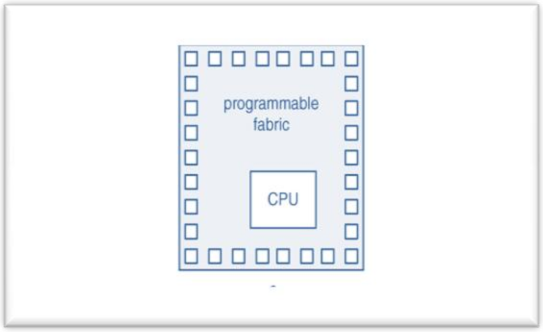

Fig. 3: Processor embedded in a Reconfigurable fabric

On the other hand, a more flexible fabric is better able to adapt to the application requirements.

2.3 FPGA Background

An FPGA is a reconfigurable device composed of configurable logic blocks and programmable interconnects. Configurable logic blocks can be programmed to implement any desired logic functions that use a memory technology to store output values. Programmable interconnects surround the individual logic blocks and allow user-defined connections. Hence, the combination of configurable logic blocks and programmable interconnects is able to create any logic gate network.

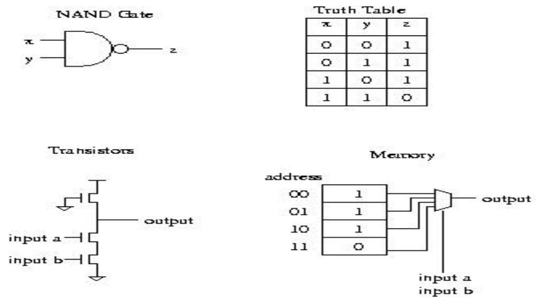

Fig.4: FPGA logic gate network

For an FPGA, a specific architecture is adopted for targeted computations as opposed to computing on a fixed architecture of a scalar processor. Therefore, FPGA technology implies the use of dedicated hardware for performing computations. The process is similar to the design of an application-specific integrated circuit (ASIC) with difference of reprogrammable functionality in FPGAs. Unfortunately, lower clock frequency is the cost of having a high flexible computing fabric, as shown in (fig 4).Mainstream FPGAs are built on static random access memory (SRAM) technology and operate at associated speeds (on average, 400 MHz). This lower speed is misleading when comparing the technology to scalar processors.

2.4 Programmable Logic Devices

Today many types of programmable logic devices (PLDs) are available which can be used in different fields. In is a list of names with of devices that hardware and software can be downloaded to

1) Programmable Logic Device (PLD)
2) Simple Programmable Logic Device (SPLD)
3) Complex Programmable Logic Device (CPLD)
4) Field Programmable Gate Array (FPGA)
5) Field Programmable Inter Connect (FPIC)

2.5 View of FPGA

Programmable logic devices (PLD) are integrated circuits with internal logic gates that are connected together through fuses. A process that is called programming defines the functionality of the chip. ROMs (Read Only Memories), PALs (Programmable array logic) and PLAs (Programmable Logic Arrays) are examples of PLDs. The main difference between these devices is the position of the fuses and the fixed connection between gates. Inside each PLD is a set of fully connected macro cells. These macro cells are typically comprised of some amount of combinatorial logic (AND and OR gates, for example) and a flip-flop. A PLA consists of two levels of logic gates: a programmable "wired" AND plane followed by a programmable "wired" OR-plane. A PLA is structured. So that any of its inputs (or their complements) can be AND'ed together in the AND-plane .For large logic circuits**,** complex programmable logic devices (CPLD) can be used. A CPLD consists of a Set interconnection network. Thus hardware resources remain static for the life of the design whereas static reconfiguration allocates logic for the duration of an application, dynamic reconfiguration (often referred to as run time reconfiguration) uses a dynamic allocation scheme that re-allocates hardware at run-time

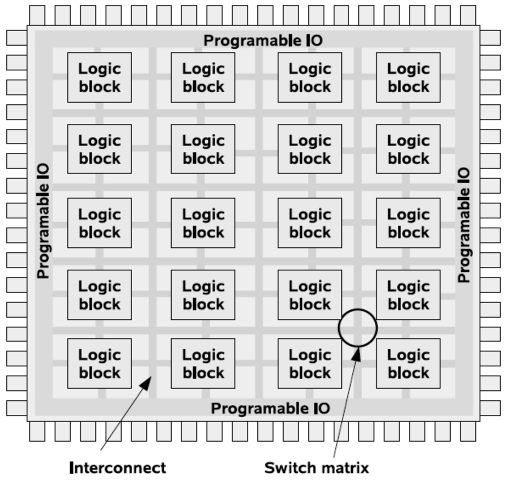

Fig.5: Programmable logic devices (PLD)

The connection between the input/output blocks and the macro cells and those between macro cells and macro cells can be made through the programmable interconnection network. In practical view point, there are two differences between FPGA and CPLD, as shown in (fig 5).first, the FPGAs with thousand gate capacity have more facilities instead of CPLDs to design more complex and huge digital systems. Second, FPGAs use more programmable switches for FPGA's interconnection blocks that have more delay. Therefore FPGAs have more delays instead of CPLDs and PALs. Also FPGAs are implementable by using Computer Aided Design (CAD) tools such as Very Log hardware design language (VHDL).

3. Design Flow

A possible design flow that takes in account the hardware or software-co-design methodology based on the detailed tasks analysis of the application, taking into account the system partitioning, software (cpu) and hardware (FPGAs coprocessors) components. The basic principles that must conduct the design flow are the applications specification and its implementation in terms of software and hardware components. In this direction, the application designer must need the possible minimum information on the software as the hardware components implementation. Although currently to be possible to incorporate matrix of floating-point blocks in FPGAs, the current demand for such high precision architecture, does not yet justify, the inclusion of such hard-cores inside commercial devices, probably because of its high implementation cost.

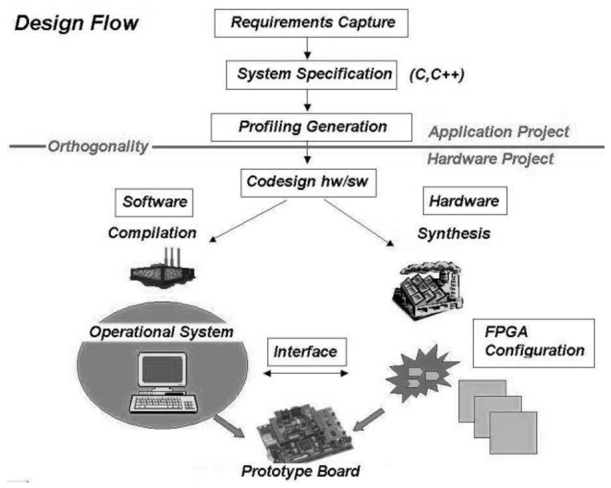

Fig.6: Flow of software co-Design

It shows the gather of system requirements according to the hardware satisfaction of software design, as shown in (fig 6).The co-Design of hardware or software can be implemented of synthesis of FPGA configuration and shows how they interface with the operating system views of softwares. The FPGA helps to identical with reconfigure architecture.

3.1 FPGA Hardware function call

The function calls, that abstraction in hardware can be implemented. Portable, Extensible Toolkit for Scientific Computation (PET SC).A function in the PET SC library is usually implemented in software. To carry through its acceleration in FPGA it would be necessary to the library with calls implemented by a new library, the reconfigurable PET SC library (RC-PET SC).

The new modules forms the library in hardware are generated through high level synthesis methods (HLS - High Level Synthesis) [79]. For each function from PET SC will be implemented, a new extension in hardware, called RC function.

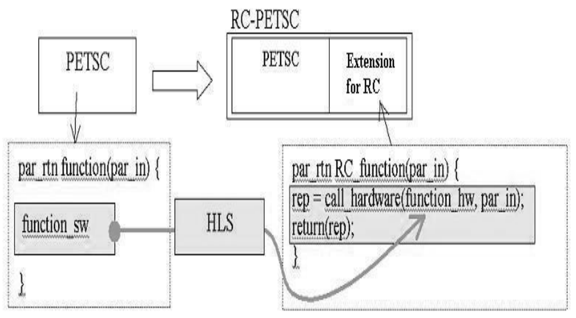
RC-PETSC
PETSC
PETSC
Extension for RC
par rtn function(par in) {
function sw
}
HLS
par rtn RC function(par in) {
rep = call hardware(function hw, par in);
return(rep);
}

Fig.7: FPGA hardware function call

This new function carries through the same task of its version in software, but now, the computation uses the module in hardware, as shown in (fig 7). TheRC function keeps the same call form, send parameters and return variables. Internally, the RC function must activate the reconfigurable hardware (call hardware). This function comprises three steps:

- Sends the core for the FPGA (function hardware).
- The data to be processed (par in), and returns the results (rep).
- The PET SC means Portable, Extensible Toolkit for Scientific Computation and constitutes a software library that contains data structures and functions for scaled and parallel scientific applications.

3.2 Architectural model based on layers for Aquarius device drivers' platform

The device drivers are OS components that must carry through the following functions to control a specific hardware component to provide access services to other software components to deal with errors in the manipulation of hardware devices. Analyzing the device drivers. We verify that the HAL layer (the Hardware Abstraction Layer) is the layer next to the physical device. This layer supplies access functions that depend of the hardware characteristics. The next layer is the kernel, considered the "brain" of the device drivers. This layer is responsible by IP-Select Map operation mode, involving its control and the registers access sequence.

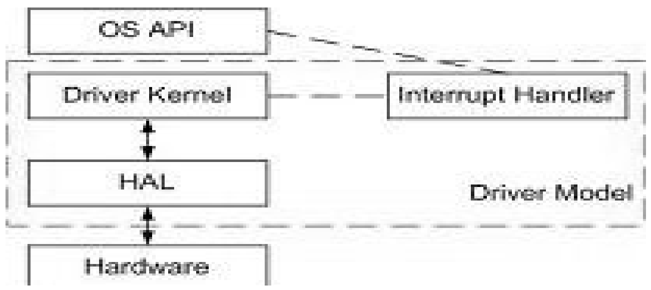

Fig.8: layers for Aquarius device drivers' platform

These parameters define the services offered for the application layers as shown in (fig 8).The kernel is also responsible by the services consistency verification. The driver in the upper layer is called API (Operating System Application Program Interface). The API controls communication between applications and the kernel. These resources provide an easy way to the users to have access to all services offered by the FPGA co-process.

4. Conclusion

Reconfigurable architecture has examined research issues of an operating system for a reconfigurable computer. To reconfigure a new hardware, it requires having Sample space to place the new hardware. The component placement issue becomes complex if the component needs to be placed near special resources like built- in memory, I/O pins or DLLs on the FPGA. Newly configured hardware must meet the timing requirement for the efficient operation of the circuit. Longer wires between components may affect the timing. Or under timing the new added design may yield erroneous result.

References

[1] Today's Operating Systems and their Future with Reconfigurable Architecture JesperMelin IDT, Mälardalen University, Västerås, Sweden Kopparbergsvägen 27B 72213 Västerås SWEDEN jmn06007@student.mdh.se Daniel Boström IDT, Mälardalen University,Västerås,Sweden.
[2] Research Issues in Operating Systems for Reconfigurable ComputingGrant B. Wig and David A. Kearney Reconfigurable Computing Laboratory (RCL)Advanced Computing Research Centre University of SouthAustralia.
[3] Reconfigurable computing: architectures and design methodsT.J. Todman, G.A.Constantinides, S.J.E. Wilton, O. Mencer, W. Luk and P.Y.K. Cheung Reconfigurable Computing Architecture Survey and introduction Ali Azarian Department of computer Engineering Payame NoorUniversity(PNU)Shushtar,Iran.
[4] Reconfigurable Platforms for High Performance ProcessingS´ergio B.Nascimento1,3, Jordana Seixas1, Edson Barbosa1,2, Stelita Silva1, Abner Correa1, Viviane Lucy1,Victor Medeiros1, Arthur Rolim1, Dercy Lima1andManoelEusebiodeLima1.
[5] Customising processors: design-time and run-Time opportunities, Luk,W,Lect.NotesComput.Sci.,2004.3133.
[6] Customising hardware designs for ellipticcurve cryptographyTelle, N., Cheung, C.C., and Luk, W,Lect. Notes Comput. Sci., 2004, 3133 A quantitative analysis of the speedup Factors of FPGAs over processor.
[7] Adaptive Multiuser Online Reconfigurable Engine D. Rakhmatov, S. Vrudhula, T. Brown, and A. Nagarandal " in IEEE Design & Test of Computers, vol. 17, 2000, pp. 53-67.
[8] Dynamic Reconfiguration to SupportConcurrent ApplicationsJ.Jean, K.Tomko, V. Yavagal, J. Shah, and R.Cook IEEE Transactions onComputers, vol. 48, pp. 591602, 1999.
[9] FPGA Hardware In Configurable Computing Technology and its uses in High Performance Computing DSP and Systems Engineering,Bellingham, S. Guccione and D. Levi.
[10] A Bridging Layer for Run - Time Reconfigurable Hardware Operating Systems. Villalobos, Ricardo &Abielmona,Rami&Grozal,

Voicu EEE International Instrumentation and Measurement Technology Conference

P.Priya is holding under graduation degree in B.Sc computer science from Bhaktavatsalam Memorial College for women and pursuing post graduation in master of computer applications from S.A Engineering College. This paper is a part of curriculum convered under in (MC7413) “TECHNICAL SEMINAR AND REPORT WRITING”

V.Rekha is holding under graduation degree in B.Sc computer science from Bhaktavatsalam Memorial College for women and pursuing post graduation in master of computer applications from S.A Engineering College.This paper is a part of curriculum convered under in (MC7413) “TECHNICAL SEMINAR AND REPORT WRITING”

M.Vaitheswari is holding under graduation degree in BCA computer applications from Thirumurugan Arts and Science College For Women pursuing post graduation in master of computer applications from S.A Engineering College.This paper is a part of curriculum convered under in (MC7413) “TECHNICAL SEMINAR AND REPORT WRITING”