Multi-agent based Reconfigurable Architecture for Future Wireless Networks

T.Suresh, Joseph Henry
R.M.K Engineering College
Kavaraipettai, TamilNadu, India
fiosuresh@yahoo.co.in

P.Rangarajan
R.M.D Engineering College
Kavaraipettai, TamilNadu, India

Abstract—**The key behind successful resolution of all possible communication challenges in future wireless systems is in the interoperability. The concept of interoperability will yield the necessity of reconfigurability and cooperativeness in various communications systems paving the road towards the 4G paradigms. In this paper, the analysis is conducted for the design of reconfigurable architectures for emerging wireless communication systems. This paper also proposes an innovative multi-agent based reconfigurable system approach by which a unified hardware/software co-design methodology is accomplished to improve the system performance in terms of adaptability, efficiency and reliability.**

Keywords— **Interoperability, reconfigurable, multi-agent, wireless network.**

I. Introduction

Commercial wireless network standards are continuously evolving from 2G to 3G and then further onto 4G. Each generation of networks differ significantly in link-layer protocol standards causing problems to subscribers, wireless network operators and equipment vendors. The air interface and link-layer protocols differ across various standards like WLAN, Bluetooth, GSM, GPRS, CDMA, and UMTS. This problem has inhibited the deployment of global roaming facilities causing great inconvenience to subscribers who travel frequently from one continent to another. Subscribers are forced to buy new handsets whenever a new generation of network standards is deployed. Wireless network operators face problems during migration of the network from one generation to next due to presence of large number of subscribers using legacy handsets that may be incompatible with newer generation network. To achieve seamless communication, future wireless communication systems will have to incorporate and integrate different wireless access technologies. The integration should offer seamless interoperability of different types of wireless networks with the wireline backbone. So it requires the design of a single wireless user terminal able to autonomously operate in different heterogeneous access networks. The design of this multistandard/multimode user terminal faces several problems. There must be reductions in cost, size, power consumption and circuit complexity that will lead to a reconfigurable user terminal that must be cheap and efficient. For this purpose, advanced reconfigurable architectures which can dynamically adapt to the changes in access technologies are currently hot research topics[1].

Hardware/Software partitioning provides a way of customizing the hardware and software architectures to complement one another in ways which improve system functionality, performance, reliability, survivability, and cost effectiveness[15]. There are several issues need to be addressed for the co-design. The issues are hardware/software partitioning, and implementation of agent-based architecture in hardware and software. In general more regularly structured agents that have highly repetitive and extensive time consuming operations are suitable for implementation in reconfigurable hardware, whereas the more complex and irregularly structured agents should be programmed in software. The hardware agent implementation is similar to the software , all the parameters of the agents can be transmitted by function calls in C/C++, while in hardware we have to specifically define all of the hardware agent entities, their associate ports and parameters in VHDL.

A. *Coarse-grained Reconfigurable Architecture*

During recent years , a number of research efforts are focused on the design of new reconfigurable systems for particular area of application. The work in the area of developing new reconfigurable architectures covers the research on fine-grained and coarse-grained reconfigurable architectures[2]. Coarse-grained reconfigurable architectures offer a high degree of parallelism, sufficient to achieve high throughput. In general, coarse-grained architectures are made upon a set of hard macros (8-bit, 16-bit or even a 32-bit ALU), usually called Processing Element (PE). The PEs are able to carry few operations such as addition, subtraction or even multiplication. The interconnection is realized either through switching matrices or dedicated busses. The configuration is done by defining the operation mode of the PEs and programming the interconnection between the Processing Elements.

II. Reconfigurable Architectures

The evolving of current and future broadband access techniques into the wireless domain introduces new and flexible network architectures with difficult and interesting challenges. The system designers are faced with a challenging

978-1-4244-4711-4/09/$25.00 ©2009 IEEE

set of problems that stem from access mechanisms, energy conservation, error rate, transmission speed characteristics of the wireless links and mobility aspects. This paper analyzes the major challenges in realizing flexible microelectronic system solutions for future mobile communication applications. Based thereupon, the various architecture designs of flexible system-on-chip solutions in the digital baseband processing for future mobile radio devices are discussed. The motivation of this paper is to find a new parallel and dynamically reconfigurable hardware/software architecture adaptable to various wireless network standards.

A. Montium Architecture

At the University of Twente in the Chameleon project[1] the steps have been made to define an energy efficient heterogeneous reconfigurable System-on-Chip architecture. Chameleon Systems created one of the first practical commercial implementation of coarse-grained reconfigurable technology for data intensive Internet, DSP and other high performance telecommunication applications[8]. This architecture contains a coarse-grained reconfigurable part called MONTIUM Processor Tiles. The MONTIUM tile is designed to execute highly regular computational intensive DSP kernels using its Multiply and Accumulate operations(MAC). This architecture yields a combination of performance, flexibility and energy-efficiency.

Fig.1 depicts a single MONTIUM Processor Tile. The hardware organization within a tile is very regular and resembles a very long instruction word (VLIW) architecture. The five identical Arithmetic and Logic units (ALU1_ _ _ALU5) in a tile can exploit spatial concurrency to enhance performance. This parallelism demands a very high memory bandwidth, which is obtained by having 10 local Memories (M01_ _ _M10) in parallel. The ALU input registers provide an even more local level of storage. Locality of reference is one of the guiding principles applied to obtain energy efficiency in the MONTIUM. A vertical segment that contains one ALU together with its associated input register files, a part of the interconnect and two local memories is called a Processing Part (PP). The five Processing Parts together are called the Processing Part Array(PPA). A relatively simple sequencer controls the entire PPA. The Communication and Configuration Unit (CCU) implements the interface with the world outside the tile. The interconnect provides flexible routing within a tile. The configuration of the interconnect can change every clock cycle.

An HiperLAN/2 receiver was implemented in this architecture, which is adaptable depending on the used modulation scheme, as well as a flexible Bluetooth receiver on the same architecture. But the set of multi-standard algorithms is rather incomplete. So more adaptive algorithms are to be developed for the various wireless standards.

B. Reconfigurable Baseband Processor

This architecture focuses on increasing the granularity of the configurable units without compromising flexibility. Fig.2 shows Reconfigurable baseband processor. This architecture contains five different configurable units Dominant configurable unit (DOF), CORDIC, ML, ALU and Interconnect unit. The DOF consists of four multipliers, five adders, two accumulators, two shifters , eight two's complement operations and two multiplexers. The CORDIC configurable unit can be configured to perform either polar to rectangular transformation or rectangular to polar transformation. The core building block is the CORDIC stage which is composed of adders and shifters. Any miscellaneous operation is supported by two general-purpose 16 bit ALUs running at the highest speed. The two ALUs are serving the real and the complex components of the datapath respectively. Using Multiple Instruction stream, Multiple Data steams (MIMD) approach with shared memory, the flexibility can be provided.

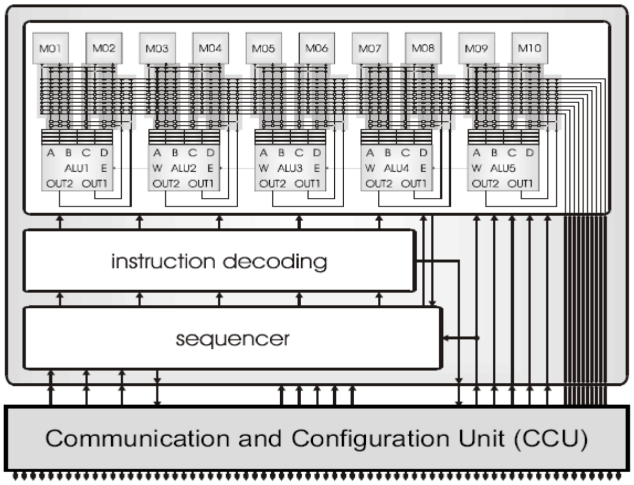

Figure 1. MONTIUM Processor Tile

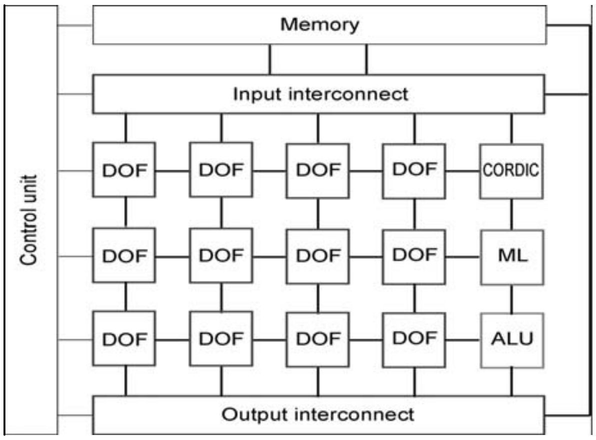

Figure 2. Reconfigurable baseband processor

The processor has two memory banks: data memory and coefficient memory. The data memory has simultaneous read write ports, while the coefficient memory has a read/write port. Using time-multiplexed cross-bar interconnect the number of wires are reduced. It provides a more compact interconnection, but it requires extra circuitry for the multiplexing and demultiplexing of signals and it incurs latency. This architecture was tested in IEEE 802.11 standard and power consumption was estimated for each module, since power efficiency is more critical in wireless communications applications.

C. *FleXilicon architecture*

The architecture proposed in reference[3] addresses three critical design issues with the loop-level-parallelism, wide memory bandwidth, reconfigurable controller, and the support of flexible word length and mapping critical loops into processing elements. Processing element of an instruction-oriented architecture performs sequence of operations, which are defined by instructions, micro-code and/or control signals. Instructions are stored in a configuration memory and fetched by a controller to control the associated processing element. FleXilicon has an array of Processing Elements Slices(PES) and each PES consists of local memories, crossbar switch network, 16 processing elements and multipliers and a RC(Reconfigurable Controller). This architecture is also a coarse -grained architecture which intends to address various problems faced by the existing architectures like PADDI, Chameleon, AVISPA and RAW.

D. *Reconfigurable Modem architecture*

In this paper a Rreconfigurable Modem (RM) Architecture targeting 3G multi-standard wireless communication system was proposed. This architecture targeted two 3G wireless standards WCDMA and CDMA 2000 and the design objectives are scalability, low power dissipation and low circuit complexity. This architecture includes cell searcher, rake receiver and viterbi decoder. Cell searcher requires Registers , Adders, Comparators and Multipliers. Rake receiver contains Registers, Counters, Logic gates, Adders, Subtractors , Accumulators, Multipliers, Shifters and Comparators. Viterbi decoder requires Registers, Adders and Comparators. Based on the above four different types, PEs which consists of many functional units are constructed. Each PE is dedicated for one of four basic operations, Bit manipulation, One-Bit Correlation, MAC or ACS (ADD Compare and Select). Then PEs are grouped into four different types of PE modules. PEs inside the PE module communicate with each other reconfigurable bi-directional data paths.

It was proved that this architecture achieves flexibility, scalability and low circuit complexity. It can support multiple standards and easily adapt to rapid change of specification. This architecture has a critical path delay of 13nsec and that for the ASIC implementation is 7.5 nsec. The longer critical path delay is due to long data path inside PEs.

III. Multi-agent Based Reconfigurable Architecture

Multi-agent systems are systems that contain several entities that are called agents[13]. Agents are intelligent autonomous stand-alone processes, continuously sense the environment and over time perform actions which are designed to alter that environment. These agent-based systems have been successfully applied to a number of problem domains, with one of the most active areas of research being in the area of mobile communication. In the past, employing agent-based techniques has been restricted to software-only implementations. But now with the advent of large-scale commercially available reprogramable logic, it is now possible to extend this agent technology to both software and selected hardware of the system such as Field Programmable Gate Arrays (FPGA) and Reconfigurable System-On-Chip (SOC). By applying this agent-based model into a reconfigurable system-on-chip platform, the overall system flexibility and efficiency will be significantly improved. Implementation of agent techniques in reconfigurable hardware allows for creation of high-speed systems that can exploit a much finer grained parallelism[18].

The focus of this paper is to illustrate how agent based techniques can be employed within reconfigurable hardware design environments. We call these new agents that run inside reconfigurable logic ‘Hardware Agents’. Such design environments often utilize object-oriented hardware description languages such as VHDL to capture the design and use synthesis tools to translate this high level description of the system into a low-level bit stream that can be used to configure the reconfigurable devices. In the following sections, this paper will introduce the basic concepts associated with the hardware multi-agent paradigm.

A. *The BDI Agent Model*

The formalism used for the hardware agents is derived from the well-known Beliefs-Desires-Intentions (BDI) architecture that is described extensively in the technical literature [16,17]. In the Beliefs-Desires-Intentions architecture each agent maintains a set of beliefs, a set of desires, and a set of intentions. The set of beliefs indicates what the agent currently believes to be true concerning its environment. It is a localized view where what an agent believes to be true may or may not in fact be true. For a system as a whole the set of desires is a set of outcomes that each agent would like to cause in its own environment. Note that the agent may or may not be able to bring its desires about. Achieving its desire may require action from other agents or may not be possible at all. Finally, the set of intentions is a set of actions that the agent intends to take to attempt to bring about its desires. The architecture of the BDI agent model is shown in Fig.3, which has three external ports: inter-agent communication, control, and input/output. The control port is used for the system to activate/deactivate the agent, and for the agent to inform the system when it finishes its job, and also a clock signal for the agent to synchronize with the system. The input/output port is used to send and receive information to and from the host environment. The inter-agent communication port allows the agents to send/receive information with other agents and cooperate with each other.

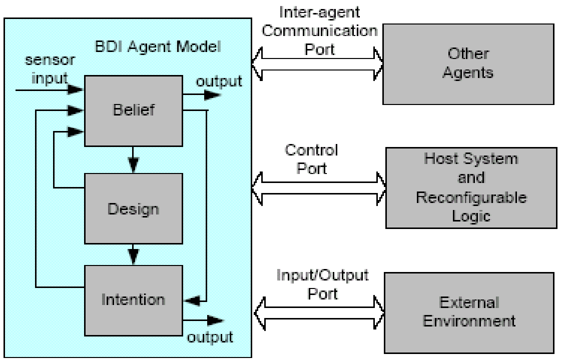

Figure 3. The architecture of BDI agent model

B. *Hardware agent model*

In general more regularly structured agents that have highly repetitive and extensive time consuming operations are suitable for implementation in reconfigurable hardware, whereas the more complex and irregularly structured agents should be programmed in software. Hardware agents can act as an interface between the environment and the reconfigurable system. They can be viewed as a top layer in reconfigurable architecture. Hardware agents are placed in the configuration memory of the reconfigurable system. In this environment, such hardware agents can be located in more than one configuration memory; also , more than one hardware agent could exist within a single configuration memory. The hardware agent implementation is similar to the software , all the parameters of the agents can be transmitted by function calls in C/C++, while in hardware we have to specifically define all of the hardware agent entities, their associate ports and parameters in VHDL. In practice, agents can each be modeled as separate modules in the same hardware description file. The most straightforward method in VHDL is to instantiate each agent as a component and then interconnect the components together using signals in a structural model. The block diagram of hardware agent is shown in Fig. 4. As we see in this block diagram each agent has four kinds of ports and six kinds of signals. The index values (n1, n2, n3 and n4) in this figure illustrate the variable number of ports and signals that can enter and exit each agent. The exact number and size of each port is dependent upon how the agents are interconnected with one another and with the outside environment to support the given application.

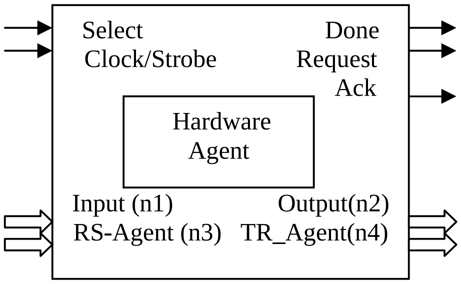

Figure 4. Hardware Agent block diagram

Agents receive and send information to and from one another through the RS_Agent and TR_Agent ports, respectively, thereby allowing the agents to communicate and cooperate with each other. In a similar manner, the agents send and receive information to and from the host environment through the Output and Input ports, respectively. When the system wants to use a specific agent, it activates that agent by setting the Select signal. When the agent reaches a particular goal, the Done signal will be set. The Clock signal will connect to the system clock to synchronize this agent with other agents as well as portions of the system. In this model, the Strobe, Request and Ack signals form the handshake signals needed to ensure the reliable transmission of data between the agents in asynchronous communication. In most cases the Request signal is used to request that another agent in the system perform a desired operation. The receiving agent will be informed that data is available by monitoring its Strobe input signal. The receiving agent then at its discretion can activate the Ack (acknowledge) when it is ready or has received information from the other agents or the environment.

C. *Advantages of hardware agents*

Hardware agents have many possible advantages. The first advantage of hardware agents is their speed of operation. The processing time of a hardware agent can be one or two orders of magnitude greater than an equivalent software agent due to the speed of hardware compared to the speed of microprocessor-based software. This speed could be especially desirable in a real-time system and real time processing requiring high-speed signal conditioning. Another potential advantage is lower power consumption. Hardware agents can be better suited for monitoring the environment and placing different system elements into low-power standby mode. Using low-power standby modes more efficiently could reduce power consumption. Also, implementation of a system entirely in reconfigurable hardware might use less power than the use of microprocessors for the same functionality, since only the necessary functions will be implemented at any one time.

IV. Performance Comparison

Table I shows simulation results for a Viterbi decoder. The performance of FleXilicon architecture was compared with ARM920T processor. FleXilicon reduces the number of cycles significantly for Viterbi decoder. Table II summarizes the synthesis results of Reconfigurable Modem(RM). The estimated area for the RM is 6.9M equivalent NAND2 gates while that for ASIC is 5.5M NAND2 gates. By optimizing the bit width, the circuit complexity of RM will be lowered significantly. The complete HiperLAN/2 receiver processing would require about 11μs while mapping the HiperLAN/2 receiver algorithms on the MONTIUM Architecture. Reference [2] shows that the Reconfigurable baseband processor implementation is two orders of magnitude more power efficient than Digital Signal Processors (DSP).

TABLE I. PERFORMANCE COMPARISON FOR VITERBI DECODER

	Frequency	*Cycles*	*Execute Time(μs)*
FleXilicon	200MHz	12	0.07
ARM 920T	200MHz	22706	27.47

TABLE II. PERFORMANCE COMPARISON OF AN ASIC AND RM

	Circuit Complexity (NAND2 gates)	*Critical path delay(ns)*
ASIC	5.5M	7.53
RM	6.9M	13.02

In Reference[14] the agent-based platform was implemented for face detection and tracking algorithm and the experiment results showed that the average detection time for software-only is 33.05msec, while the average time for hardware/software co-design approach is 8.375msec. It is obviously that the agent based co-design approach can improve the system performance significantly.

V. CONCLUSION

In this paper some Reconfigurable Architectures adaptable to multistandard wireless communication systems are analyzed and their performances are compared. In this paper we also addressed the application of the multiagent technology to reconfigurable hardware/software systems. Experiment results of applying agent-based architecture to a real-time face detection and tracking task showed that this architecture provides a very flexible platform and much more efficient compared to other approaches. So our future research will focus on multiagent hardware/software reconfigurable architecture adaptable for wireless networks.

REFERENCES

[1] G.K. Rauwerda, G.J.M. Smit, L.F.W. van Hoesel, P.M. Heysters , "Mapping Wireless Communication Algorithms to a Reconfigurable Architecture", Proceedings, International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA'03), pages: 242-251, 2003

[2] Ada S.Y. POON, "An Energy Efficient Reconfigurable baseband processor for Wireless Communications", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 15, No.3, pages: 319-327 Mar. 2007

[3] Jong-Suk Lee and Dong Sam Ha , "FleXilicon: a Reconfigurable Architecture for Multimedia and Wireless Communications", Proceedings, International Symposium on Circuits and Systems, 2006

[4] Jina Kim and Dong Sam Ha, "A New Reconfigurable Modem Architecture for 3G Multi-Standard Wireless Communication Systems" Proceedings , IEEE International Symposium on Circuits and Systems 2005, Pages: 1051-1054

[5] Hooman Parizi, Afshin Niktash, Amir Kamalizad, Nader Bagherzadeh, "A Reconfigurable Architecture for Wireless Communication Systems", Proceedings, Third International Conference on Information Technology: New Generations, 2006, Pages: 250-255

[6] Alsolaim.A,Becker.J, Glesner.M, Starzyle.J, "Architecture and Application of a dynamically reconfigurable hardware array for future mobile communication systems", Proceedings, IEEE Symposium on Field-Programmable Custom Computing Machines, 2000, Pages: 205-214

[7] Carl Ebeling, Chris Fisher, Guanbin Xing, Manyuan Shen, Hui Liu, "Implementing an OFDM Receiver on the RaPiD Reconfigurable Architecture", IEEE Transactions on Computers ,Vol.53, Issue 11, Pages:1436-1448 ,Nov.2004

[8] Reiner Hartenstein, "Coarse Grain Reconfigurable Architectures", Proceedings, Conference on Asia South Pacific Design Automation, Pages:564-570, Jan.2001

[9] Frank Hannig, Hritam Dutta,and Jurgen Teich, "Regular Mapping For Coarse-Grained Reconfigurable Architectures", Proceedings, IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol.V,pages:57-60,May 2004

[10] Jurgen Becker,Manfred Glesner, "A parallel Dynamically Reconfigurable Architecture Designed for Flexible Application-Tailored Hardware /Software Systems in Future Mobile Communication", The Journal of Supercomputing, Vol.19, Issue 1,Pages:105-127,May 2001

[11] Scott Hauck,Thomas W.Fry, Matthew M.Hosler,and Jeffrey P.Kao, "The Chimaera Reconfigurable Functional Unit", IEEE Transactions on Very Large Scale Integration(VLSI) Systems,Vol.12,No.2,Feb. 2004

[12] Andrew A.Gray,Clement Lee, Payman Arabshahi, Jeffrey Srinivasan, "Object-Oriented Reconfigurable Processing for Wireless Networks",Proceedings, IEEE Intenational Conference on Communications, Vol.1, Pages:497-501,2002

[13] Hamid Reza Naji,B.Earl Wells, "On Incorporating Multi Agents in Combined Hardware/Software based Reconfigurable Systems – A General Architecture Framework", Proceedings ,Thirty Fourth Southeastern Symposium on System Theory,Pages:344-348, 2002

[14] Yan Meng, "A Mobile Vision System with Reconfigurable Intelligent Agents",Proceedings, International Joint Conference on Neural Networks, July 2006

[15] Yan Meng, "An Agent–based Reconfigurable System-on-Chip Architecture for Real-time Systems",Proceedings, Second International Conference on Embedded Software and Systems,2005

[16] H.R. Naji, B.E. Wells, M. Aborizka, "Hardware agents", Proceedings of the ISCA 11th International Conference on Intelligent Systems on Emerging Technologies (ICIS-2002),Boston, MA, July 2002.

[17] A. Rao, M. Georgeff, "BDI agents: from theory to practice", Proceedings of the ICMAS'95 First International Conference on Multi-agent System, 1995.

[18] Hamid Reza Naji, B.Earl Wells, Letha Etzkorn, "Creating an adaptive embedded system by applying multi-agent techniques to reconfigurable hardware", Future Generation Computer Systems, Vol.20, Issue 6, Pages:1055-1081, Aug. 2004