Reconfigurable Systems: A Survey

Toshiaki Miyazaki

NTT Optical Network Systems Laboratories
A1-329S, 3-1 Morinosato Wakamiya, Atsugi, 243-01 JAPAN
e-mail: miyazaki@aecl.ntt.co.jp

Abstract— Recently, a lot of reconfigurable systems (RSs) utilizing FPGAs have been reported. The aim is to achieve high performance with low production cost, or to produce non-"von Neumann" machines. In this paper, we overviews RSs. FPGAs, the key components of RSs, are introduced. Second, we categorize RSs into several types. Next, applications are described. Then, hot topics related to RSs are mentioned. Finally, some open problems are described.

I. Introduction

Reconfigurable or *programmable* systems have been around a long time. For example, the Burroughs B1700 computer [1] had multiple instruction sets for different targets, such as COBOL and FORTRAN, and could be switched to the suitable machine by installing different microprograms. Video games can also be said to reconfigurable, because they can be changed to different game machines by exchanging ROM cartridges or CD-ROMs [2][3]. In addition, today's MPU-embedded systems, such as those found in vehicles and almost all home electric appliances, can be categorized in the same group, because different functions are provided by changing program ROMs. However, for all of these examples, the reconfigurability is only in the software not the hardware.

On the other hand, since the introduction of FPGAs (Field Programmable Gate Arrays) [4][5], a number of hardware reconfigurable systems have been developed. Compared with software reconfigurable systems, FPGA-based systems have more potential in terms of performance and adaptability for target problems. In this paper, we call these systems "hardware reconfigurable systems" or "reconfigurable systems (RSs)" for short. They have been given other names, such as custom computing machines (CCMs), or FPGA-based custom computing machines (FCCMs), depending on the application area.

One of the goals of RSs is to provide high performance with low production cost, as exemplified by "Splash-2", performs comparably to a Cray YMP super-computer in some applications but costs only $40,000 [15].

Another aspect of RSs, or CCMs, is that some of them could possibly offer an alternative to "von Neumann" machines [8][17][35]. *Partially reconfigurable* FPGAs like Xilix XC6200 [37] and Atmel AT6000 [36], which are motivating the creation of new architecture paradigms and applications of RSs, are especially interesting.

The rest of this paper is organized as follows: Section II introduces FPGAs and categorizes them into several types. In Section III, RSs are surveyed and grouped into three types. Section IV describes applications of RSs. Section V mentions new ideas in this area. Section VI concludes by mentioning future work.

II. Reconfigurable Logic [14]

Today, many kinds of reconfigurable logic devices are available, i.e., FPGAs and CPLDs (Complex Programmable Logic Devices), and they are the keys to constructing RSs. Almost all RSs use commercially available FPGAs/CPLDs, but some utilize custom reconfigurable chips [31][8][34][33]. In this section, FPGAs/CPLDs are categorized according to the characteristics [1].

Configurable logic: This is a general term for logic devices that can be customized one time. Anti-fuse FPGAs, such as the Actel ACT3/3200DX series, Quicklogic pASIC1/pASIC2 series, and Crosspoint CP20K series are examples of this type.

Reconfigurable logic: This refers to logic devices that can be customized many times. These devices often adopt EPROM, EEPROM, or FLASH technology. They can be reprogrammed after being mounted on a PCB, which known as "in-system programming" or ISP. The Altera MAX7000/9000 series, AMD MACH1/2/3/4 series, Philips PZ3000/5000 series, and GateField GF100K series are in this category.

Dynamically reconfigurable logic: This supports on-the-fly programming capability after mounting on a system board. It is often called "in-circuit reconfiguration", or ICR. Almost all devices of this type use SRAM technology to realize the ICR mechanism. The Xilnix XC3100/4000/5200 series, Lucent ORCA-2 series, and Altera FLEX8000/10K series are typical devices in this category.

Dynamically reconfigurable interconnect: This is a general term for interconnect devices that can be pro-

[1] Almost all FPGA/CPLD vendors have their own WWW home pages, and we can get their product information from them[6][7].

grammed pin-to-pin connections after mounting on a system board. They are similar to the above SRAM-based FPGAs, but do not have any programmable logic blocks. The Aptix FPIC (Field Programmable Interconnect Chip) and I-Cube PSID (Programmable Switching and Interconnect Device) are typical examples.

Virtual logic: This is a kind of dynamically reconfigurable logic device, but it features partial reconfiguration capability. This mechanism allows part of the device to be reprogrammed dynamically while the rest of the device is executing user defined logic. In other words, different logic circuits can time-share the same part of this device. From this mind, this device has been referred to as "adaptive logic", "cache logic", and "swappable logic." The Xilinx XC6200 series [37] and Atmel AT6000 series [36] have partial reconfiguration mechanisms.

III. Reconfigurable Systems

A lot of RSs have been developed in the last decade. More than eighty are listed in Ref. [32], and there are certainly many more in the world today. The features of some of them are summarized in Table I. From the architecture point of view, they can be categorized into three types: attached processors, coprocessors, and special-purpose machines (see also Fig. 1).

A. Attached processors

These systems are often connected to workstations (WSs) or personal computers (PCs), and they perform heavy tasks that the WS or PC itself cannot handle very well.

PeRLe-1 [27] is a pioneering system of this type. It has been applied to solving a variety of problems in RSA encryption, video compression, high-energy physics, and sound synthesis. The system is programmed using C++. To enable this, *PeRLe-1* programming environment has several libraries. The runtime library provides a device driver interface, and the netlist library enables one to describe the netlists of synchronous circuits using C++.

Splash-2 [15] is another well-known attached system. It consists of several reconfigurable boards connected to each other by a futurebus+ backplane and has an S-Bus interface for a Sparcstation. Each reconfigurable board, called an array board, has 17 processing elements. Each element contains a Xilinx XC4010 FPGA and 512 Kbytes memory. Crossbar switches connect processing elements, while there are dedicated connections between two neighboring elements. *Splash-2* is basically a SIMD (Single-Instruction Multi-Data stream) machine. It exhibits exceptional performance: it is more than 10,000 times faster than a conventional workstation when used, for instance, to search a genetic database or for DNA pattern matching. A commercial version of *Splash-2*, called *WILDFIRE*, is also available [39].

TERAMAC [31] is a rather large system. It has multi-chip modules (MCMs) using custom FPGA called *PLASMA*. The *TERAMAC* system has a hierarchical structure: Several MCMs are mounted on a board, and the MCM-mounted boards make up the system. The PLASMA chips are connected by hierarchical crossbar switches, and they provide the user with a huge programmable logic plane.

B. Coprocessors

This type of system is connected to an MPU bus, and performs as a "reconfigurable" coprocessor. Unlike the functions of ordinary coprocessors, such as floating-point multipliers, the functions of a reconfigurable coprocessor can be customized according to applications.

In *PRISM-II* [16], a reconfigurable module containing three Xilinx XC4010's is attached to an AMD29050 processor via bus. More than one reconfigurable module can be connected to the bus. *Pamette* [28] is a descendant of the *PeRLe-1*, and compacted as a PCI board. In this type, reducing the number of data transactions through the MPU bus is the key to improving performance, because this often becomes a performance bottleneck.

C. Special-purpose machines

The systems in this category perform special purpose tasks and can often be booted without any help from a WS or PC.

GANGLION [40] is a reconfigurable board for neural network applications. *YARDS* [33] has a RISC-type MPU card, FPGAs, 2-port SRAMs and I-Cube PSIDs on a VME board. With a telecommunications interface board, it realizes a high-speed ATM (Asynchronous Transfer Mode) network control function. In addition, it performs remote reconfiguration; that is, the hardware behavior of the receiving system can be changed according to reconfiguration data sent via the ATM network. *ProBoard* [34] is also in this category. *P4* [42] is tuned to providing a reconfigurable platform of the protocol boosters in ATM networks.

IV. Applications

RSs feature both flexibility, as in software, and high speed based on parallel execution, as in hardware. In other words, RSs can fit each target application and provide sufficient performance by examining and extracting the parallelism and regularity of the target application. Thus, the applications of RSs are in fields requiring a lot of parallel or bit-level operations that ordinary MPUs have difficulty handling. Typical applications are as follows.

Image and signal processing: Typical examples are the video processing and finger print matching in *Splash-2* [15]. *Splash-2* maps fine-grain parallel operations into

TABLE I
RECONFIGURABLE SYSTEMS

Name	Developer	FPGAs†	On-board RAM	External bus	Interconnect	Notes††
PeRLe-1 [27]	DEC Paris Lab (F)	XC3090 x24	4MB SRAM	DEC TURBOchannel	Fixed mesh	Lempel-Ziv, RSA encryption, NN
Splash-2 [15]	SRC (A)	XC4010 x16	(128Kx16b)x16 RAM	S-Bus	Linear array & crossbar	PM, IP
AnyBoard [11]	North Carolina State Univ. (A)	XC3042 x5	(128Kx8b)x3	ISA	Linear array & fixed buses	RP
ArMen [12]	Univ. de Bretagne Occidentale (F)	XC3090 x1	1, 2 or 4Mb/node	S-Bus	Cube	MIMD machine
RM-II [13]	Kobe Univ. (J)	XC4005 x9	32Kx16b	N.A.	crossbar& XC4005	Emulation
TERAMAC [31]	HP (A)	PLASMA (custom) x108	32MB SRAM	SCSI	crossbar, MCM	ACC
RPM [29]	USC (A)	XC4013 x7	2MB SRAM, 8MB SRAM, 96MB DRAM	SCSI	N.A.	RP for multiprocessors
Transmogrifier-2 [9]	Univ. of Toronto (C)	Altera 10K50 x2	(256K x64b) x2 banks	Parallel port to Sun, 512b general I/O	I-cube IQ320 x2	RP
Xputer/ MoM-3 [8]	Univ. of Kaiserslautern (D)	XC4000E(X) x2, XC4010, rDPA-1 (custom) x8	2MB SRAM	VME	Local bus (MoM-bus)	Procedurally data-driven CCM
Riley-2 [41]	Imperial College (UK)	XC6216 x4	(128Kx32b) x4	PCI	Linear array & local bus	Codesign, DR
PCI Pamette V1 [28]	DEC Paris (F)	XC4000E x5	256MB DRAM, 128KM x2 SRAM	PCI	N.A.	I/O-oriented CoP, CP(PeRLe-1)
PRISM-II [16]	Brown Univ. (A)	XC4010 x3	(128Kx32b) x3	AMD29050 bus	Inverted tree, or none	CoP
EVC1 [38]	VCC (A)	XC4013 x1	2MB SRAM	S-Bus	None	CoP, CP
WILDFIRE [39]	Annapolis Micro Sys. (A)	XC4000 series x16	8MB SRAM	VME, PCI	Linear array & crossbar	PM, CP(splash-2)
H.O.T Works [38]	VCC (A)	XC6216 x1, XC4013E x1	2MB SRAM	PCI	Fixed	Codesign, DR, CP
ACEcard [46]	TSI TelSys (A)	XC6264 x2	64MB DRAM, 1MB SRAM	PCI	Fixed	CoP, DR, CP
RC1000-II [47]	Embedded Solutions Ltd. (UK)	XC4010E x1	256KB, 1MB SRAM	ISA	Fixed, Daughter card	CoP, CP
GANGLION [40]	IBM (A)	XC3090 x24	24KB PROM	VME/ Datacube MAXbus	Fixed	NN
YARDS [33]	NTT Labs. (J)	XC4010 x2/ MAX9000x2/ PROTEUS (custom) x2	(32Kx8b 2-port SRAM) x4	VME	I-cube IQ320 x2, Daughter card	Telecom
DISC [35]	Brigham Young Univ. (A)	CLAy31 x2	N.A.	ISA	N.A.	DR

† The number of FPGAs per board is shown if the system can have more than one board.
†† ACC: Accelerator; RP: Rapid Prototyping; PM: Pattern matching; IP: Image Processing; NN: Neural Network; DR: Dynamic Reconfigurability; CoP: Coprocessor; CP: Commercial Product

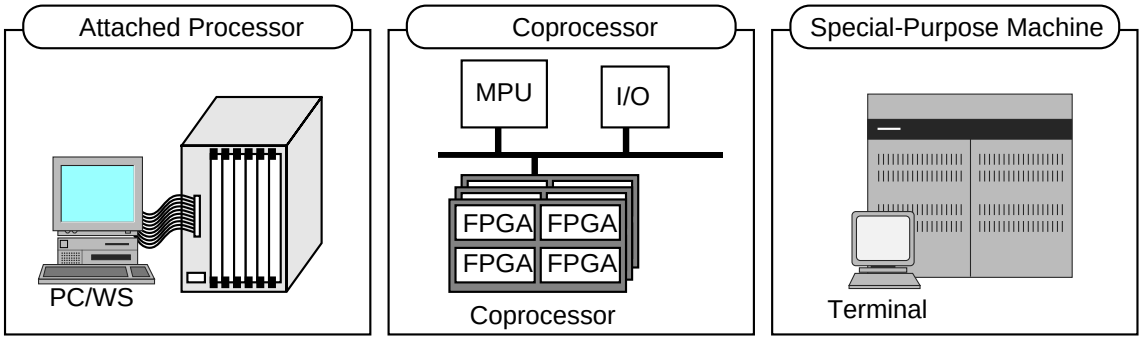

Fig. 1. Three types of architecture for reconfigurable systems.

its SIMD architecture, and exhibits good performance. We can find numerous papers on signal processing applications utilizing RSs or FPGAs in the proceedings of FPGA-related symposiums and workshops, such as FCCM[2], FPGA[3], and FPL[4].

High-speed data handling and control: Telecom applications [34][33][42] are in this category. These RSs contain many bit-level pattern matching circuits, because data checking is always required in order to synchronize telecommunications data streams and extract data structures, i.e., packets and cells.

There are some neural network (NN) applications [40][27]. Their aims are not only to construct a NN itself in a RS, but also to apply high-speed NNs to some specific applications like data clustering.

Satisfiability, which is known as the *NP-complete* problem, solver is another application in this area [44][45]. In addition, a system called *RM-II* [13] has been applied to accelerating logic diagnosis.

Rapid prototyping and emulation: Historically, this is the first application field of FPGA-based RSs [9][29]. In the ASIC design area, ASIC emulators such as Quickturn [43] were developed just after FPGAs became commercially available. However, the target of these emulators is to accelerate chip design not system-level prototyping. If we want to emulate system-level architecture efficiently, the RSs should have some functional-level modularity and a mechanism by which some function blocks can be replaced according to applications. This is because it is extremely difficult to realize all circuits from a sea of reconfigurable gates, i.e. FPGAs, without violating the performance constraints. In addition, system-level emulations often require specific peripherals. Thus, it is important to be able to change a part of the system smoothly.

[2]IEEE Symposium on FPGAs for Custom Computing Machines
[3]ACM/SIGDA International Symposium on Field Programmable Gate Arrays
[4]International Workshop on Field-Programmable Logic and Applications

V. New Ideas

A. Dynamic Reconfiguration

Compared with ASICs, FPGAs have disadvantages in both area and performance [49]. To overcome these problems, some new ideas have been presented [35][25][24][17][30]. The common thread among them is efficient time-sharing of the finite reconfigurable circuit area, from which arises the terms "dynamic reconfiguration", or "run-time reconfiguration". For example, *DISC* (Dynamic Instruction Set Computer) utilizes the virtual logic device described in Section II and changes its instruction set according to applications [35]. In other words, as shown in Fig. 2, if there is no corresponding circuit in *DISC* when an instruction is latched, a circuit is loaded or swapped into the *DISC* and executed. Thus, an infinite number of instructions can be executed in *DISC*. Similar ideas can be found in Refs. [25] and [24]. On the other hand, *WASMII* [17] architecture has a dynamic reconfiguration function whose mechanism is based on a dataflow machine concept.

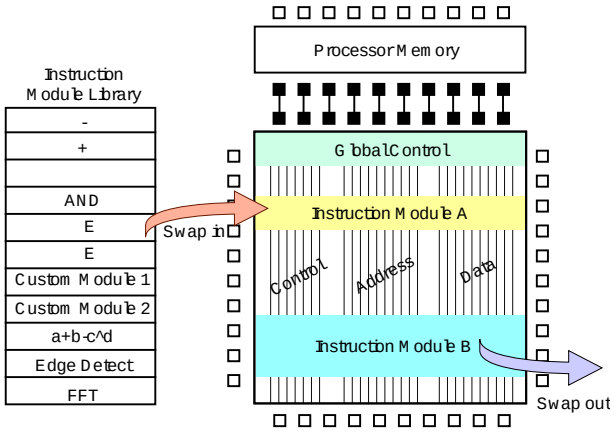

Fig. 2. Concept of DISC (Dynamic Instruction Set Computer).

In implementing dynamic or run-time reconfigurability

in RSs or FPGAs themselves, we have to deeply consider the following items.

Configuration speed: In conventional FPGAs, the configuration speed is much slower than the execution speed. (The ratio is about 1:10^6.) There are two ways to improve the configuration time. One is to increase the amount of configuration data bits per clock. The other is to reduce the total amount of configuration data itself. The parallel memory writing adopted in the XC6200 is an example of the former and the *Colt* [30] approach is an example of the latter. *Colt* reduces the number of control points, which are directly related to the amount of configuration data, by adopting a coarse-grain FPGA architecture.

Global registers: Some global registers, which hand out temporary values from *t* time circuits to *t+1* time circuits, must be prepared.

Implementation cost: We have to prepare more additional circuits to realize the dynamic reconfiguration mechanism compared with the conventional FPGAs. Of course, it is meaningless if the cost becomes more expensive than the cost in the case of tiling the conventional FPGAs. Recently, a DRAM-FPGA has been reported [26]. This approach reduces implementation cost, by reducing the area of configuration memories, which often occupies a large portion of the die area in an FPGA.

B. Evolvable Hardware

Evolutionary algorithms are search and optimization procedures that have their origin in the biological world. The Genetic Algorithm (GA) and Genetic Programming (GP) are two of the most widely used evolutionary algorithms [18][19]. Evolvable hardware is a general term for hardware that contains a mechanism based on evolutionary algorithms [20]. The emergence of FPGAs stimulated work in this area. *BIODULE* [21] contains an FPGA and emulates a cell. By connecting many *BIODULEs*, a cellular array that demonstrates self-reproduction and self-repair functions can be constructed. The AdAM system utilizes FPGAs to accelerate GA operations [22].

VI. Concluding Remarks

Reconfigurable systems (RSs) utilizing FPGAs have many of the same advantages as MPU-based systems, such as upgradeability, standardization of platforms, and amortization of development costs, while providing the high-performance capability that only well-tuned hardware can. Thus, RSs have a lot of potential to produce new evolutions or, in some cases, revolutions in many technology fields. As shown in Table I, there are a variety of commercially available RSs, and they are stimulating activities in this area. Many people are especially interested in RSs utilizing XC6200 as a platform in order to explore new applications and non-"von Neumann" architectures.

However, people involved in this area are struggling to find their "killer-application." In other words, creating a new reconfigurable system is relatively easy, but proving its superiority to MPU-based systems is quite hard. A general or target-less reconfigurable system would fail, because its functions could not be so drastically changed even if the system were *reconfigurable*. For example, a simply-implemented state-of-the-art MPU as FPGAs never exceeds the performance of the original MPU, because FPGAs do not perform as well as ASICs realized with the same fabrication technology. In other words, some architectural innovations, the ideas for which could come from target applications, should be provided in each RS.

From the application development point of view, the current status of the development environment for RSs is poor compared with the ordinary software development environment for MPUs. The user must often perform time-consuming FPGA configuration tasks, such as logic partitioning, placement, and routing. This is somewhat analogous to using assembly language in the software development. Of course, people who have recognized this situation are attacking these problems [10][48]. In fact, many RSs mentioned in this paper adopt subsets with some extensions of standard software languages, such as C/C++ and JAVA, to describe their applications. Different from software compilers, the compilers for RSs produce netlists for FPGAs instead of assembly codes. *Handel-C* [48], originally developed at Oxford University, is commercially available as a "hardware" compiler development environment [47]. Furthermore, for dynamic reconfigurable systems, the concept of the "virtual operating system" has been introduced [23]. It handles logic swapping and task scheduling while an ordinary OS in a computer performs memory and task management. Nevertheless, it will take much effort to improve the usability of RSs in order to popularize them.

References

[1] W. T. Wilner, "Design of the Burroughs B1700," AFIPS Proc. the 1972 Fall Joint Computer Conference, pp. 489-497, Montvale, N.J., 1972.

[2] Nintendo of America, Inc., http://www.nintendo.com/

[3] Sony Computer Entertainment America Inc., http://www.playstation.com/index.html

[4] Brown S. D., Francis R. J., Rose J. and Vranesic Z. G., "Field-Programmable Gate Arrays," Kluwer Academic Publishers, 1992.

[5] S. Brown, "FPGA Architecture Research: A Survey," IEEE Design & Test of Computers, Vol. 13, No. 4, pp. 9-15, 1996.

[6] —, http://www.reconfig.com/

[7] —, "FPGA related WWW links," http://www.www.mrc.uidaho.edu/fpga/fpga.html

[8] R. W. Hartenstein et al., "A New FPGA Architecture for Word-Oriented Datapaths," Proc. FPL'94 (Springer LNCS 849), pp.144-155, September 1994.

[9] D. M. Lewis et al., "The Transmogrifier-2: A 1 Million Gate Rapid Prototyping System," Proc. ACM/SIGDA FPGA'97, pp.53-61, Feb. 1997.

[10] D. Galloway et al., "The Transmogrifier C Hardware Description Language and Compiler for FPGAs," Poc. FCCM'95, April 1995.

[11] D. E. Van den Bout et al., "AnyBoard: An FPGA-Based, Reconfigurable System," IEEE Design & Test of Computer, Vol. 9, No. 3, pp.21-30, September 1992.

[12] F. Raimbault et al., "Fine grain parallelism on a MIMD machines using FPGAs," Proc. FCCM'93, pp.2-8, 1993.

[13] N. Suganuma et al., "Reconfigurable machine and its application to logic diagnosis," Proc. ICCAD92, pp.373-376, 1992.

[14] T. Sueyoshi, "Present Status and Problems of the Reconfigurable Computing Systems —Toward the Computer Evolution —," IEICE, Technical Report, VLD96-79, pp.111-118, December 1996. (in Japanese)

[15] D. Buell et al., "Splash 2: FPGAs in a Custom Computing Machine," IEEE Computer Press, 1996 (ISBN0-8186-7413-X)

[16] M. Wazlowski et al., "PRISM-II Compiler and Architecture," Proc. FCCM'93, pp.9-16, 1993.

[17] X-P. Ling, and H. Amano, "WASMII: a Data Driven Computer on a Virtual Hardware," Proc. FCCM'93, pp.33-42, 1993.

[18] —, "Evolutionary Electronics," http://watson.open.ac.uk/∼monty/EHW.html

[19] E. Sanchez at al. eds. "Towards Evolvable Hardware," Springer Lecture Notes in Computer Science (LNCS) 1062, 1996.

[20] T. Higuchi et al. eds., "Evolvable Systems: From Biology to Hardware," Springer Lecture Notes in Computer Science (LNCS) 1259, 1997.

[21] D. Mange at al., "Embryonics: A New Family of Coarse-Grained Field-Programmable Gate Array with Self-Repair and Self-Reproducing Properties," in Springer Lecture Notes in Computer Science (LNCS) 1062, pp. 197-220, 1996.

[22] T. Hikage, H. Hemmi, and K. Shimohara, "Hardware Evolution System Introducing Dominant and Recessive Heredity," in Proc. ICES'96 (Springer Lecture Notes in Computer Science (LNCS) 1259, 1997).

[23] G. Brebner, "A Virtual Hardware Operating System for Xilinx 6200," Proc. FPL'96,(LNCS 1142), pp.327-336, 1996.

[24] S. Trimberger at al., "A Time-Multiplexed FPGA," Proc. FCCM'97, pp.22-28, 1997.

[25] K. Ishii, A. Tsutsui, and T. Miyazaki, "An Architecture of Dynamic Reconfigurable FPGA," Proceedings of IEICE General Conference, D-138, (p.146), Mar. 1995. (in Japanese)

[26] M. Motomura et al., "An Embedded DRAM-FPGA Chip with Instantaneous Logic Reconfiguration," Proc. 1997 Symposium on VLSI Circuit, 8-1, June 12-14, Kyto, 1997.

[27] J. E. Vuillemin at al., "Programmable Active Memories: Reconfigurable Systems Come of Age," IEEE Trans. on VLSI Systems, Vol. 4, No. 1, pp. 56-69, March 1996.

[28] Digital Equipment Corporation., http://www.research.digital.com/SRC/pamette

[29] K. Onre et al., "The design of RPM: an FPGA-based multiprocessor emulator," Proc. FPGA'95, pp.60-66, 1995.

[30] R. Bittner, and P. Athanas, "Wormhole Run-time Reconfiguration," Proc. FPGA'97, pp.79-85, Monterey, February 1997. http://www.ee.vt.edu/athanas/whrtr

[31] R. Amerson et al., "Teramac—Configurable Custom Computing," Proc. FCCM'95, Napa, April 1995.

[32] S. Guccione, "List of FPGA-based Computing Machine," http://www.io.com/∼guccione/HW_list.html

[33] A. Tsutsui, and T. Miyazaki, " YARDS: FPGA/MPU Hybrid Architecture for Telecommunication Data Processing," Proc. ACM/SIGDA FPGA'97, pp.93-99, Feb. 1997.

[34] K.Hayashi, T. Miyazaki, K. Shirakawa, K. Yamada, and N. Ohta, "Reconfigurable Real-Time Signal Transport System using Custom FPGAs," Proc. FCCM'95, pp. 68-75, Napa, April 1995.

[35] M. J. Wirthlin, and B. L. Hutchings, "A Dynamic Instruction Set Computer," Proc. FCCM'95, 1995.

[36] Atmel Corporation, "AT6000/LV Series: Coprocessor Filed Programmable Gate Arrays," Data Sheet., http://www.atmel.com/

[37] Xilinx Inc., "XC6200 Field Programmable gate Arrays," Data sheets., http://www.xilinx.com

[38] Virtual Computer Corporation., http://www.vcc.com/

[39] Annapolis Micro Systems, Inc., http://www.anapmicro.com/

[40] C. E. Cox, and W. E. Blanz, "GANGLION—A Fast hardware Implementation of a Connectionist Classifier," IEEE Proc. Custom Integrated Circuits Conference (CICC'91), 6.5.1, 1991.

[41] P. I. Mackinlay et al., "Riley-2: A Flexible Platform for Codesign and Dynamic Reconfigurable Computing Research," Proc. FPL'97,(LNCS 1304), pp.91-110, 1997.

[42] I. Hadžić, and J. M. Smith, "P4: A Platform for FPGA Implementation of Protocol Boosters," Proc. FPL'97 (Springer LNCS 1304), pp.438-447, September 1997.

[43] Quickturn Design Systems, Inc., "System Realizer M3000/M250", Brochure, (http://www.qcktrn.com/).

[44] M. Abramovici, and D. Saab, " Satisfiability on Reconfigurable Hardware," Proc. FPL'97 (Springer LNCS 1304), pp.448-456, September 1997.

[45] T. Suyama et al., "Solving Satisfiability Problems on FPGAs," Proc. FPL'96 (Springer LNCS 1142), September 1996.

[46] TSI Telsys, Inc., http://www.tsi-telsys.com/

[47] Embedded Solutions Limited, http://www.embedded-solutions.ltd.uk

[48] I. Page et al., "Handel-C Language Reference Guide," Oxford University Computing Lab., August 1996. http://www.comlab.ox.ac.uk/oucl/users/ian.page/handel/handel-c.html

[49] O.T. Albaharna, P.Y.K. Cheung, and T.J. Clarke, "Area & Time Limitations of FPGA-based Virtual Hardware," Proc. ICCD'94, pp. 184-189, 1994.