CHAPTER 3

RECONFIGURABLE COMPUTING SYSTEMS

Steven A. Guccione
Cmpware, Inc.

Like most technologies, reconfigurable computing systems are built on a variety of existing technologies and techniques. It is always difficult to pinpoint the exact moment a new area of technology comes into existence or even to pinpoint which is the first system in a new class of machines. Popular scientific history often gives simple accounts of individuals and projects that represent a turning point for a particular technology, but in reality the story is usually more complicated. A number of individuals may arrive at similar conclusions, at very nearly the same time, and the details of their research are nearly always different. It is in the investigation of these details that a better understanding of the technology, and its development, can be reached.

While it is satisfying to say that Thomas Edison invented the lightbulb in 1879, the real story is much more complex and much more interesting. Such is the case with reconfigurable computing hardware systems, as it is with most technologies. In the short time that these systems have been in existence, a relatively large number of them, developed by many highly trained and talented individuals from diverse fields, have evolved very quickly. In approximately a decade the number of implemented reconfigurable systems went from a small handful to hundreds.

The large number of exotic high-performance systems designed and built over a very short time makes this area particularly difficult to document, but there is also a problem specific to them. Much of the work was done inside various government agencies, particularly in the United States, and was never published. In these cases, all that can be relied on is currently available records and publications.

3.1 EARLY SYSTEMS

The generally agreed on criterion for a reconfigurable computing system is that it be built from reconfigurable computing devices such as field-programmable gate arrays (FPGAs) or FPGA-like devices. In general, these devices must be reprogrammable and permit hardwarelike levels of performance, if not hardwarelike structures. Moreover, they should permit orders of magnitude speedup over traditional microprocessors for similar clock speeds, silicon

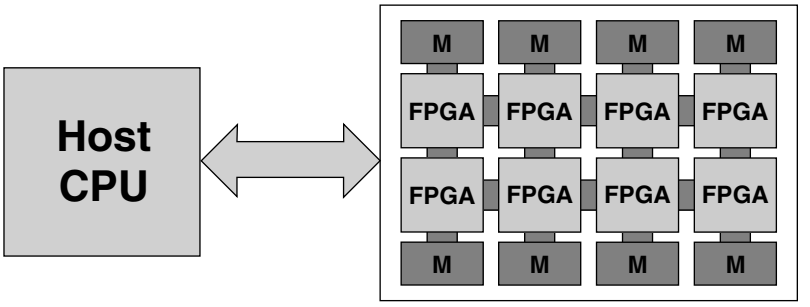

FIGURE 3.1 ■ The traditional processor/coprocessor arrangement for reconfigurable computing hardware.

area, and technology. Most significantly, however, the system must be reprogrammable and able to perform with a variety of applications. Systems that use a fixed hardware design, even if they use reconfigurable computing elements, are viewed more as using this design in place of traditional hardware for cost savings or convenience. It is in the ability to use reconfigurable devices for more general-purpose computing that makes them "reconfigurable."

Reconfigurable systems are likewise distinguished from other cellular multiprocessor systems. Array processors, in particular Single Instruction Multiple Data Processors (SIMDs), are considered architecturally distinct from reconfigurable machines, in spite of many similarities. This distinction arises primarily from the programming techniques. Array processors tend to have either a shared or dedicated instruction sequencer and take standard instruction-style programming code. Reconfigurable machines tend to be programmed spatially, with different physical regions of the device being configured at different times. This necessarily means that they will be slower to reprogram than cellular multiprocessor systems but should be more flexible and achieve higher performance for a given silicon area.

One of the earliest acknowledged reconfigurable computing machines, although it is frequently referenced under "distributed computing," is the Fixed-Plus-Variable (F+V) computer developed by Estrin and his colleagues at the University of California at Los Angeles in the mid-1960s [17–20]. The F+V consisted of a standard processor unit that controlled many other "variable" units. It had several limitations, including the need to manually change wiring as part of the reconfiguration process, but it did offer relatively mature software tools for its time. Generally because of its use of reconfigurable computing concepts, the F+V system is acknowledged to be the forerunner of modern reconfigurable computing architectures.

After the F+V, there was a gap of nearly two decades before more modern reconfigurable computing systems began to be explored. The rise of the modern era began in the mid-1980s, when commercially available FPGA devices from companies such as Xilinx and Altera as well as several smaller companies became widely available.

These devices were generally based around small lookup tables (LUTs) and a programmable interconnection network. The LUTs were typically 8- or 16-bit memories configured to implement arbitrary logic functions, taking their inputs from and sending their outputs to a programmable interconnection network. While this network could not provide arbitrary interconnections, software tools were usually able to produce operational digital circuits for a wide range of popular designs.

Even by 1990, however, the largest FPGA devices supported designs on the order of 10K logic gates. This is a very small number and barely suitable for a parallel multiplier circuit. Even worse, the FPGAs were in competition with modern microprocessors, which were doubling in performance every 18 months and providing a simpler programming model, more mature tools, and a larger base of experienced users. For these reasons, the early work in reconfigurable systems necessarily concentrated on two areas, often simultaneously:

- The systems would have to use relatively large numbers of FPGAs, sometimes hundreds, to achieve sufficient computing power to be of use when compared to microprocessor-based systems.
- They would attack problems that were naturally ill suited to modern microprocessors, including bit-oriented algorithms that did not map efficiently to word-oriented microprocessors and highly structured and repetitive algorithms such as graphics that mapped well to the hardwarelike structures of reconfigurable systems.

The 1990s also marked the beginning of an explosive growth in circuit density following Moore's Law, with a doubling in FPGA density approximately every 18 months. As the density increased, the typical application went from simple interface or "glue" logic circuits to more complex designs, eventually supporting large custom coprocessors, typically for digital signal processing (DSP) or other data-intensive applications. With large, high-quality, commercially available FPGA devices now in use, and with the ongoing rapid increase in density, FPGA-based reconfigurable computing machines quickly became widely available.

3.2 PAM, VCC, AND SPLASH

In the late 1980s, PAM, VCC, and Splash—three significant general-purpose systems using multiple FPGAs—were designed and built. They were similar in that they used multiple FPGAs, communicated to a host computer across a standard system bus, and were aimed squarely at reconfigurable computing.

3.2.1 PAM

The Programmable Active Memories (PAM) project at Digital Equipment Corporation (DEC) initially used four Xilinx XC3000-series FPGAs as shown in Figure 3.2 [8]. The original Perle-0 board contained 25 Xilinx XC3090 devices

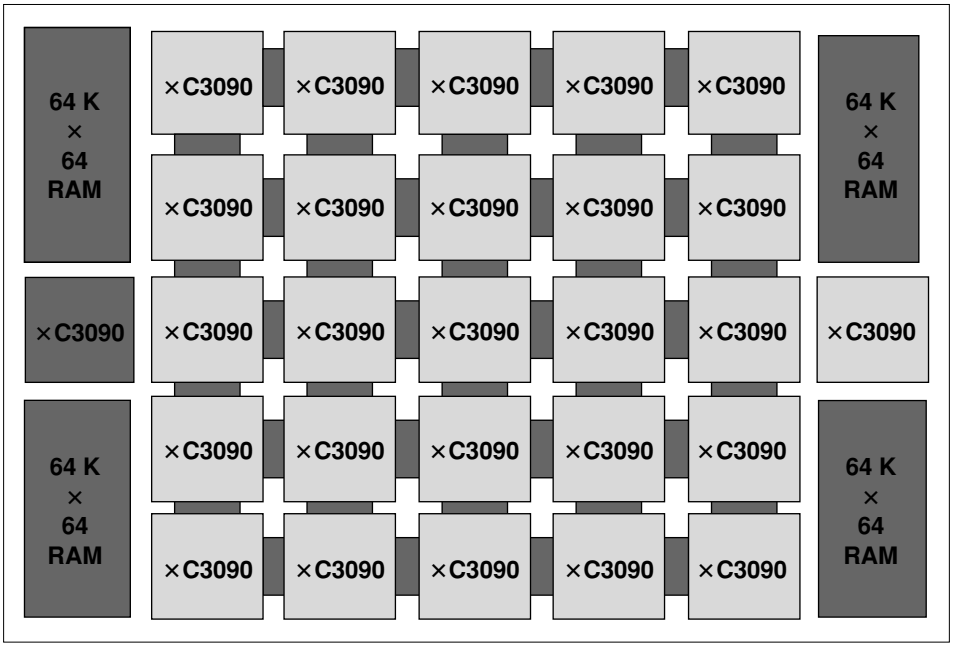

FIGURE 3.2 ■ Digital Equipment Corporation's PAM Perle-0.

in a 5×5 array, attached to which were four independent banks of fast static RAM (SRAM), arranged as 64K $\times$ 64 bits, which were controlled by an additional two XC3090 FPGA devices. This wide and fast memory provided the FPGA array with high bandwidth. The Perle-0 was quickly upgraded to the more recent XC4000 series. As the size of the available XC4000-series devices grew, the PAM family used a smaller array of FPGA devices, eventually settling on 2×2.

Based at the DEC research lab, the PAM project ran for over a decade and continued in spite of the acquisition of DEC by Compaq and then the later acquisition of Compaq by Hewlett-Packard. PAM, in its various versions, plugged into the standard PCI bus in a PC or workstation and was marked by a relatively large number of interesting applications as well as some groundbreaking work in software tools. It was made available commercially and became a popular research platform.

3.2.2 Virtual Computer

The Virtual Computer from the Virtual Computer Corporation (VCC) was perhaps the first commercially available reconfigurable computing platform. Its original version was an array of Xilinx XC4010 devices and I-Cube programmable interconnect devices in a checkerboard pattern, with the I-Cube devices essentially serving as a crossbar switch as shown in Figure 3.3 [11]. The topology of the interconnection for these large FPGA arrays was an important issue at this time: With a logic density of approximately 10K gates and input/output (I/O) pins on the order of 200, a major concern was communication across FPGAs. The I-Cube

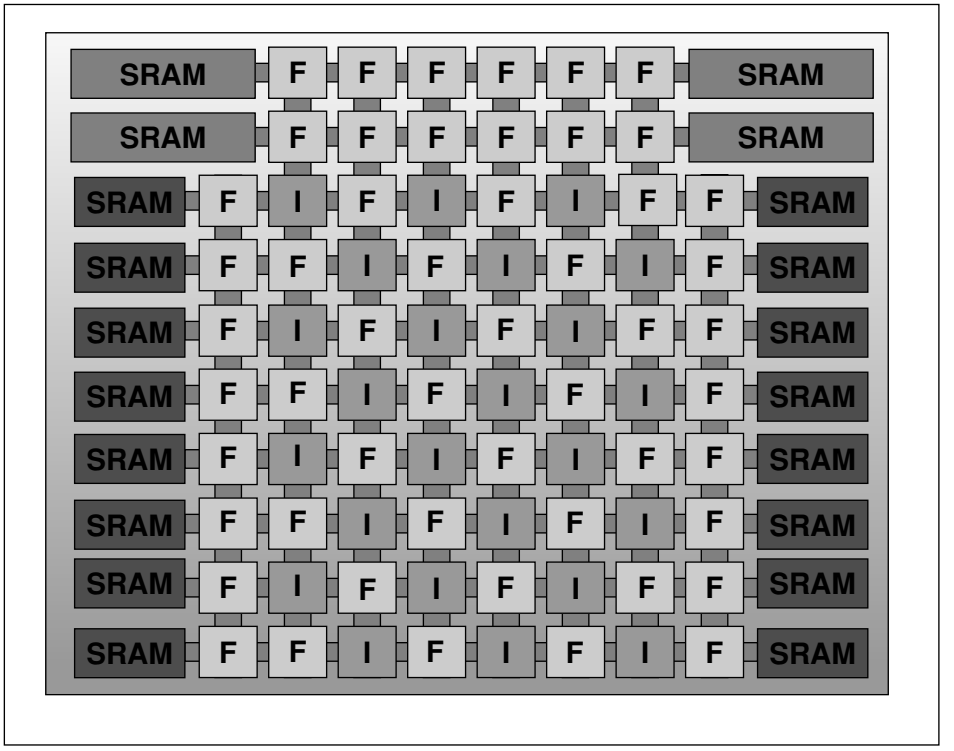

FIGURE 3.3 ■ VCC's Virtual Computer.

devices were perceived as providing more flexibility, although each switch had to be programmed, which increased the design complexity.

The first Virtual Computer used an 8×8 array of alternating FPGA and I-Cube devices. The exception was on the left and right sides of the array, which exclusively used FPGAs, which consumed 40 Xilinx XC4010 FPGAs and 24 I-Cubes. Along the left and right sides were 16 banks of independent 16×8K dual-ported SRAM, and attached to the top row were 4 more banks of standard single-ported $256\text{K} \times 32$ bits SRAM controlled by an additional 12 Xilinx XC4010 FPGAs. While this system was large and relatively expensive, and had limited software support, VCC went on to offer several families of reconfigurable systems over the next decade and a half.

3.2.3 Splash

The Splash system, from the Supercomputer Research Center (SRC) at the Institute for Defense Analysis, was perhaps the largest and most heavily used of these early systems [22,23,27]. Splash was a linear array consisting of XC3000-series Xilinx devices interfacing to a host system via a PCI bus. Multiple boards could be hosted in a single system, and multiple systems could be connected together. Although the Splash system was primarily built and used by the Department of

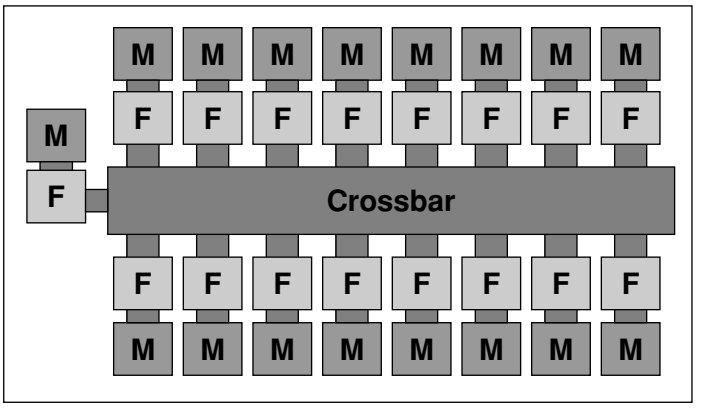

FIGURE 3.4 ■ SRC's Splash 2.

Defense, a large amount of information on it was made available. A Splash 2 system quickly followed and was made commercially available from Annapolis Microsystems [30].

The Splash 2 board consisted of two rows of eight Xilinx XC4010 devices, each with a small local memory attached as shown in Figure 3.4. These 16 FPGA/memory pairs were connected to a crossbar switch, with another dedicated FPGA/memory pair used as a controller for the rest of the system. Much of the work using Splash concentrated on defense applications such as cryptography and pattern matching, but the associated tools effort was also notable, particularly some of the earliest high-level language (HLL) to hardware description language (HDL) translation software targeting reconfigurable machines [4]. Specifically, the data parallel C compiler and its debug tools and libraries provided reconfigurable systems with a new level of software support.

PAM, VCC, and Splash represent the early large-scale reconfigurable computing systems that emerged in the late 1980s. They each had a relatively long lifetime and were upgraded with new FPGAs as denser versions became available. Also of interest is the origin of each system. One was primarily a military effort (Splash), another emerged from a corporate research lab (PAM), and the third was from a small commercial company (Virtual Computer). It was this sort of widespread appeal that was to characterize the rapid expansion of reconfigurable computing systems during the 1990s.

3.3 SMALL-SCALE RECONFIGURABLE SYSTEMS

PAM, VCC, and Splash were large and relatively expensive machines. Because a single high-density FPGA device could cost several thousand dollars, their circuit boards were perhaps some of the most expensive built at that time. Around 1990, a cost of approximately $1 per reconfigurable logic gate in a reconfigurable

system was not unusual. Of course, this cost dropped rapidly as FPGAs of higher densities and lower prices became available.

Because of their cost, the use of FPGAs was somewhat limited, and none of these systems achieved widespread success as consumer products. While work on them was ongoing, smaller-scale FPGAs were beginning to appear that would have a major impact on the direction of the field, especially because their rapidly increasing density meant that the large multichip systems of yesterday would very soon fit within a single device.

3.3.1 PRISM

One of the smaller-scale experiments with reconfigurable computing was PRISM, developed at Brown University [5]. This was an unusual project in that it used a single small FPGA as a coprocessor in a larger distributed system. This distributed processor/coprocessor arrangement was unique for its time and would reappear many years later in more mainstream reconfigurable supercomputers. It permitted small but often complex calculations to be offloaded from the central processing unit (CPU) to the reconfigurable coprocessor. The circuits implemented in the coprocessor may not have been large, but the tighter coupling to the processor gave this architecture an advantage in places where larger and more expensive arrays would not have been appropriate.

Also of note are PRISM's software development tools. Its compiler technology used was advanced for its era and was one of the earlier experiments in high-level language programming of reconfigurable systems. In particular, it addressed a more fine-grained form of coprocessing where the host CPU and the reconfigurable coprocessor shared the workload. Larger systems tended to have vastly more powerful reconfigurable units and often used the host only for simple control, input/output, and display. The workload was seldom shared with the host CPU in any meaningful way in these larger systems.

3.3.2 CAL and XC6200

Perhaps the most interesting project of this era came from Algotronix, a small Scottish company with connections to the University of Edinburgh [28], which created its own FPGA exclusively targeted at reconfigurable computing [1]. The Configurable Array Logic (CAL) featured very simple logic cells compared to other commercial FPGAs. What was unique about CAL was that each cell could be individually addressed and reconfigured dynamically—something that no other FPGA device at the time could manage. CAL also featured a fairly standard bus interface that permitted it to be easily used with a microprocessor in a coprocessor arrangement. Algotronix also offered some fairly traditional graphical tools to program CAL, as well as a small board containing multiple CAL devices.

While CAL was unique and influential, it was not until the acquisition of Algotronix by Xilinx in the early 1990s that its ideas would become more widespread. Xilinx began development of a second-generation CAL device it called the XC6200 [12]. Many of its features were the same as those of the earlier CAL

devices, but the backing of Xilinx gave the XC6200 a high level of acceptance, at least in the research community.

Perhaps the most groundbreaking aspect of CAL, and later the XC6200 family, was largely nontechnical. Because these devices supported fine-grained and dynamic reconfiguration, they required that the configuration bitstream be openly documented. Thus, all of the internal details of the logic circuitry and the configuration process were fully documented and made publicly available. Unlike other programmable hardware, the internal programming codes for most FPGAs have never been published, largely for historical and practical reasons.

3.3.3 Cloning

Early in the history of FPGAs, there was some concern that lower-cost "cloned" FPGA devices could be made by third parties. For instance, a company could produce a device that was functionally identical to a Xilinx XC4000 and sell it to Xilinx customers. This was common practice for older and smaller silicon devices and was sometimes encouraged by manufacturers. However, the large investment FPGA vendors had in software tools and silicon intellectual property made them resistant to releasing any more information than necessary about their silicon architectures. Also, as long as the high-level design tools were available for a reasonable price and worked well, most users did not have any particular need to examine an FPGA device's internal workings.

Today it is unlikely that a device as complex as an FPGA could be "cloned." Even if the technical challenges could be overcome, legal barriers to using such intellectual property probably could not be successfully challenged.

Another concern of some customers was that knowledge of the internal workings of FPGA devices could permit their designs to be compromised. While most people familiar with the issues tended to dismiss the idea of reverse-engineering, especially as FPGAs have increased in size, it was still a concern to some customers.

For these reasons, FPGA bitstreams have traditionally been, and still remain, a tightly held trade secret. The XC6200 broke ranks by publicizing its configuration data and permitting a new level of experimentation with tools and applications that could make use of these powerful new modes of operation. Commercial success for the XC6200 would be elusive in the fiercely competitive and rapidly changing FPGA market of the 1990s, but it remained a favorite of researchers even long after its cancellation by Xilinx.

3.4 CIRCUIT EMULATION

One of the early large-scale uses of reconfigurable logic was for circuit emulation. Because FPGAs can, in theory, implement any arbitrary digital logic circuit, some people realized that they could be used as a form of simulation accelerator. At the time, digital circuit simulation had become a bottleneck in the design process. As integrated circuit designs became larger and more complex, the

time necessary to simulate them also grew. Without accurate simulation, design errors inevitably crept into the final design, often requiring another expensive and time-consuming redesign cycle. In spite of the high cost of FPGA devices, the ability to quickly and accurately evaluate the function of large and complex digital circuits became very valuable. Also, for chip designs that included programmable processors (or that were the processors themselves), an FPGA-based prototype provided a development platform for testing the software that would eventually run on the production device.

Interestingly, the larger and more complex the circuit, the more difficult and time consuming simulation became and the more valuable FPGA-based emulation would be to designers. For this reason, some of the largest and most expensive FPGA-based machines have traditionally been digital circuit emulators. Some purists may point out that such machines were highly application specific and did not necessarily constitute reconfigurable computing. While these machines did often simulate only a single design in their lifetimes, they were usually reconfigured as much as several times per day to perform different functions. Also, in some cases users would go on to realize that the emulation platforms could be employed for more general-purpose computing.

Emulation using reconfigurable logic quickly became popular, with very large-scale systems becoming commercially available in rapid succession. PiE and QuickTurn in the United States announced their machines, as did the smaller InCA in Europe. The machines were very similar, all attempting to put as many high-density FPGAs as possible into a single system. Because they were highly scalable, and their densities and prices were changing rapidly, it is difficult to gauge what a typical large FPGA emulation system would be. However, a system on the order of 1 million programmable logic gates built from devices with approximately a 10K-gate capacity would be representative of a large, but early FPGA emulation. While large and expensive, these systems were very valuable to integrated circuit designers, who knew the high cost of designs with bugs. One place in particular where they had a large impact was in the design of microprocessors.

3.4.1 AMD/Intel

Because microprocessors were very complex and had strict deadlines to meet, emulation became very important at places such as Advanced Micro Devices (AMD) and Intel. And because the new microprocessor parts often had to be compatible with older models, emulation was a very good way to guarantee that systems would be compatible across generations. One event in the 1990s would help further drive emulator popularity. AMD and Intel began a decades-long competition to produce the latest high-performance device compatible with a x86 instruction set for the desktop PC market.

Initially, AMD was in the “follower” position and was attempting to create functionally identical versions of Intel devices. This was no small challenge, and with new products being released almost yearly, the value to AMD of getting a functionally correct Intel work-alike device as soon as possible was very high.

Emulators played a very large role in verifying the functional correctness of the AMD designs against the Intel device. While the emulated designs would run at perhaps a few hundred kilohertz as compared to the tens of megahertz of the final silicon devices, being able to run test vectors at this rate, and even eventually booting entire operating systems, was crucial in proving the compatibility of these microprocessor designs.

Emulation is still widely used in digital design, but the increasing size and decreasing cost of FPGA devices has led to a smaller market for very large emulation machines such as the ones offered by QuickTurn and PiE. In fact, QuickTurn and PiE merged in 1993 after a short legal battle. The merged company was acquired in 1998 by Cadence, a CAD software vendor.

3.4.2 Virtual Wires

Although emulation was largely a commercial endeavor, one research project in this area warrants special mention—the Virtual Wires Project (see Chapter 30, Logic Emulation on Multi-FPGA Systems) at M.I.T., which produced an emulator that helped overcome one of the most serious limitations of emulators of the time [6]. Whereas the logic density of FPGAs grew rapidly, chip-to-chip interconnect soon became the limiting factor in large, multi-FPGA designs such as emulation. In fact, many emulated designs used only a fraction of the logic in the FPGAs while consuming all of the input/output resources. Then along came Virtual Wires with a pin multiplexing scheme to share I/O pins on FPGA devices transparently, permitting their higher utilization. This technology would be licensed to another logic emulation company, Ikos, which would eventually be bought by another of the large CAD software vendors, Mentor Graphics.

Emulation had perhaps two major impacts on reconfigurable computing. First, it was an early large-scale user of reconfigurable logic that was commercially successful. This helped drive similar work in the field. Perhaps just as important, many of the researchers involved in the emulation work saw the value of more general-purpose computing using reconfigurable logic and would go on to lead advancements in other areas of reconfigurable systems.

3.5 ACCELERATING TECHNOLOGY

After the success of digital circuit emulators and the research results of the early systems, reconfigurable computing was poised to expand. Three factors helped drive this expansion. First, the ever-increasing density of FPGA devices was making larger and larger amounts of reconfigurable logic available at an increasingly lower price. In just a few short years, the million-gate systems that took several large boards could be built with a single device. This in itself led to widespread experimentation with reconfigurable computing as dozens of research projects at universities and research labs across the globe sprang up.

The second factor is one that has become more obvious in retrospect. By the mid-1990s the decades-long increase in microprocessor computing power was

beginning to ebb. Late in the decade, it was clear that manufacturing technology constraints, power consumption issues, and architectural limitations such as memory performance were bringing an end to the long era of microprocessor dominance. In the past, new solutions to high-performance computing had had to contend with the yearly appearance of a new microprocessor with double the performance of the previous generation and a consumer-friendly price. This made it difficult for custom high-performance systems to be competitive. With the end of the steep growth in microprocessor performance in sight, however, other solutions to high performance were beginning to look more attractive. Reconfigurable computing technology happened to be emerging just at this critical juncture and would be considered by many as a top contender for the future of high-performance computing.

The third factor was the Department of Defense's new funding program, named Adaptive Computing Systems (ACS), which invested more than $100 million in reconfigurable computing research during the mid- to late 1990s. It is always difficult to judge the effect of such a program, but it is clear that ACS not only led to an increased level of research in this field but also provided a useful forum for researchers, both academically and commercially. While the program funded exclusively U.S. researchers, it also appears to have spurred reconfigurable computing research in other places, particularly the United Kingdom and Japan [31, 32].

The era of expansion in reconfigurable computing technology was marked by a rapid growth in the number of systems being constructed. An accurate count of projects in this area is difficult, but certainly dozens and perhaps hundreds of reconfigurable systems were constructed at this time [25]. However, the increased density of FPGA devices led to a shift away from large, expensive systems like those of the first generation and toward smaller systems, often containing a single FPGA device on a standard board to be plugged into a personal computer or workstation.

The new systems tended to be primarily for research and were more often than not hobbled by two problems. First, the tools to program a reconfigurable computing platform were not standardized and often amounted to two completely decoupled design flows. Hardware design tools provided by the FPGA vendor were used to construct a circuit in the FPGA coprocessor, while standard software development tools were used to program the host PC or workstation. This hardware/software codesign style was inefficient and inflexible, and required highly skilled engineers. For those reasons, although there were a few notable software and tools projects at this time, they were more the exception than the rule. None achieved widespread popularity.

3.5.1 Teramac

Among the projects to come out of this era, Teramac [3, 14], a product of the Hewlett-Packard research laboratories, bears special mention, for three reasons. First, it went against the trend by creating a large multi-FPGA machine. Second, it straddled different markets by being aimed at both circuit emulation and

reconfigurable computing. Lastly, it was constructed of custom-integrated circuits instead of commercially available FPGA devices.

Teramac was originally designed to perform emulation for a large microprocessor design that was being developed jointly by Hewlett-Packard and Intel. It was to be the first 64-bit Intel processor and at the time went by the name "Merced." The joint Intel/HP project was announced in 1994 and was expected to produce its first silicon device by 1999.

All of this was taking place just as large circuit emulators from vendors such as QuickTurn were emerging as the new tools for large microprocessor development. The HP/Intel venture decided to also produce its own emulator, which would not use commercial FPGAs but rather an HP custom-designed reconfigurable logic device [2]. This was not as unusual an idea as it may seem. Intel and HP certainly had the resources to produce such a machine, and the current FPGA-based offerings were far from perfect.

The three biggest problems associated with emulators at this time were cost, low circuit density, and tools. In fact, the tools problem was perhaps the most severe of the three. Large designs needed massive computing resources on their own to be converted into configuration bitstreams for the many FPGA devices. If we assume that the emulator hardware consisted of hundreds of FPGA devices, each taking several hours of time on a standard personal computer or workstation to produce a configuration bitstream, it is clear that a large computational resource was required just to produce the data used by the emulator.

This part of design and test was often the bottleneck, and there appeared to be little that could be done to accelerate the process. Additionally, commercial FPGA devices were aimed at a more general-purpose logic design market and were not explicitly aimed at emulation. A special-purpose device more tailored to the needs of circuit emulation could provide the higher density and performance required by emulation users.

Teramac was announced in 1995 and had some unique features. First, it successfully overcame many of the limitations of the commercial FPGA devices of that era. Its custom FPGA (called Plasma) focused on fast compilation times via very flexible crossbar-based interconnects. This was in contrast to commercial FPGA's focus on logic density and performance, and it meant that the placement and routing of a design for a single Plasma device took seconds, not minutes to hours. Perhaps more interesting, Plasma made good use of defect tolerance. Boards and devices that would otherwise have been thrown away could be used in the Teramac; an analysis phase would test the system to log defects and permit the faulty portions of the system to be bypassed. While regular array architectures such as FPGAs lend themselves naturally to such defect and fault tolerance, it had not traditionally been used in commercial reconfigurable logic devices.

In addition to its emulation duties, Teramac was used for applications such as image processing, bioinformatics, search, and CAD, making it a true reconfigurable computing platform. However, while Teramac was successful, the chip it was built to emulate, the IA-64 family, was somewhat less so. The IA-64 devices were late to market, but they did eventually ship and found their way

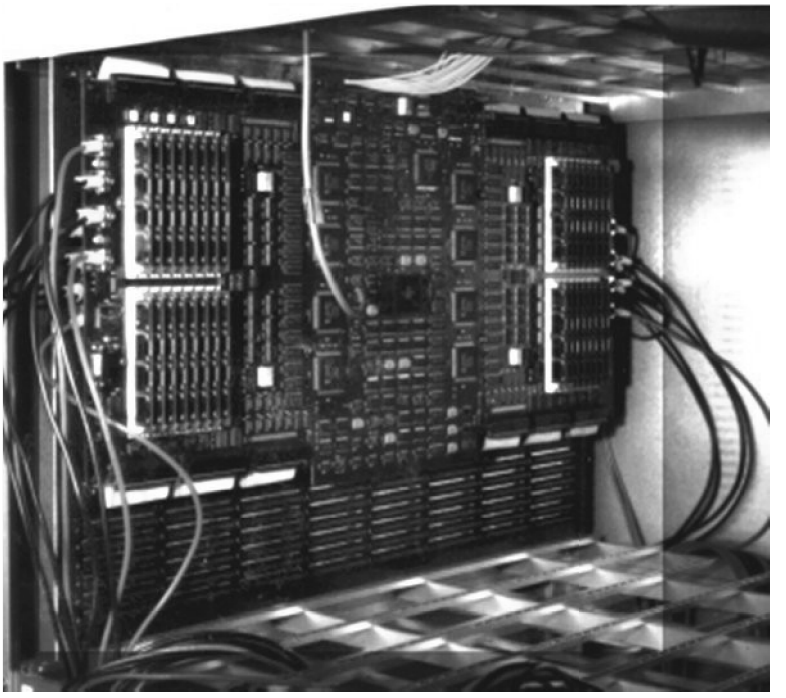

FIGURE 3.5 ■ A Hewlett-Packard Laboratories Teramac board.

into commercial products—just not enough to justify the massive investment by HP and Intel, which would not jointly produce other architectures. Thus, Teramac became an early casualty of the HP/Intel microprocessor design partnership. Figure 3.5 shows a picture of one of the boards from a Teramac system.

3.6 RECONFIGURABLE SUPERCOMPUTING

While the number of small reconfigurable coprocessing boards would continue to proliferate as commercial FPGA devices became denser and cheaper, other new hardware architectures were produced to address the needs of large-scale supercomputer users. Unlike the earlier generation of boards and systems that sought to put as much reconfigurable logic as possible into a single unified system, these machines took a different approach. In general, they were traditional multiprocessor systems, but each processing node in them consisted of a very powerful commercial desktop microprocessor combined with a large commercial FPGA device. Another factor that made these systems unique is that they were all offered by mainstream commercial vendors. By 2005 the three largest

makers of traditional supercomputer systems—Cray Research, SRC, and Silicon Graphics—were all producing systems of this type.

3.6.1 Cray, SRC, and Silicon Graphics

The first reconfigurable supercomputing machine from Cray, the XD1, is based on a chassis of 12 processing nodes, with each node consisting of an AMD Opteron processor. Up to 6 reconfigurable computing processing nodes, based on the Xilinx Virtex-4 devices, can also be configured in each chassis, and up to 12 chassis can be combined in a single cabinet, with multiple cabinets making larger systems. Hundreds of processing nodes can be easily configured with this approach.

SRC, a company with historic connections to Cray, takes a more aggressive approach to reconfigurable computing [34]. Both of their multiprocessor systems feature traditional processor and reconfigurable processing units that share a common buslike structure and may be mixed in various configurations [21]. Like the Cray system, the SRC machines also use large Xilinx Virtex-series FPGAs and x86-family desktop processors. SRC also offers smaller personal workstation systems for development.

Finally, Silicon Graphics offers its Reconfigurable Application-Specific Processor (RASP) family of systems [36], which also use high-density Xilinx Virtex FPGAs as its reconfigurable computing elements, but in dual-device configurations on a "blade"-style module. These are very small boards that can be plugged into large racks, often with the system still operating. They interface to the more traditional Silicon Graphics workstation and multiprocessor systems, which also use high-performance desktop microprocessors but are based on the MIPS architecture.

The Cray, SRC, and Silicon Graphics machines point to a clear direction for large-scale reconfigurable computing systems. They combine a more distributed array of FPGA elements with an emphasis on floating-point arithmetic. As FPGA densities continue to increase, the ability to perform large floating-point calculations, even multiple floating-point calculations, in a single device becomes significant. Also, as the performance of commodity microprocessors remains plateaued, it is likely that acceleration techniques such as those used in these reconfigurable machines will continue to be used.

3.6.2 The CMX-2X

A discussion of distributed, floating-point FPGA-based supercomputing would not be complete without a mention of the CM-2X [13]. This machine predates the current crop of reconfigurable supercomputers by over a decade and consists of a Connection Machine 2 from Thinking Machines supplemented with FPGA coprocessors instead of the standard floating-point devices typically used. The CM-2X was a defense-related project, and little information is available on it. However, along with the PRISM system, it is clearly the forerunner of this family of distributed multiprocessor reconfigurable supercomputers.

3.7 NON-FPGA RESEARCH

Although the vast majority of reconfigurable computing systems were based on commercially available FPGA devices, there are some notable exceptions. A small number of projects designed and built custom-reconfigurable silicon devices as the basis of their designs [7, 15, 16, 24, 26, 33, 35]. The general trend was to replace the smaller-grained LUTs in the FPGA architecture with coarser-grained structures more amenable to computing. Typically this meant arithmetic logic units (ALUs) that mapped more closely to traditional programming languages.

Such a coarser-grained approach raises the issue of categorizing non-FPGA devices. Large numbers of ALU-like structures quickly begin to resemble multiprocessors or very long instruction word (VLIW) machines more than they do FPGAs. The way routing is performed may further differentiate non-FPGA from FPGA devices. In general, non-FPGAs are computation, not circuit, oriented. They can easily produce the larger and more complex circuits used by typical arithmetic-based computations, but may not be able to efficiently implement arbitrary digital logic functions.

These systems may have broken new and interesting ground, but the problem with them may ultimately be a practical one. Because commercial FPGAs are very popular, they tend to use the latest silicon processes and are very efficiently designed. The software support for such devices is also decidedly nontrivial. To produce a custom reconfigurable computing device that can compete with both the dense, efficient circuitry and the large body of available software tools of modern FPGAs is a daunting prospect. Given these barriers, no serious contenders to commercial FPGAs as the basis for reconfigurable computing machines have arisen. While the ideas behind these novel architectures are sound and the advantages tangible, it has proved difficult to offer them as a viable alternative to FPGA-based reconfigurable systems.

3.8 OTHER SYSTEM ISSUES

In spite of nearly two decades of intensive research and commercial activity, and the potential to provide orders of magnitude performance, reconfigurable logic-based computing systems have not yet begun to displace conventional systems in any significant way. There are perhaps many factors in this lack of acceptance, but technical details at the hardware level certainly appear to be one of the most serious.

One unavoidable architectural problem involves the necessary use of reconfigurable logic in a processor/coprocessor arrangement, which ties an inherently serial host system to the high-performance and highly parallel reconfigurable processing unit. This connection is necessarily made across a system bus of some sort, which is guaranteed to serialize access to the coprocessor. Thus, the reconfigurable coprocessor can only be “fed” data at a relatively low and fixed

rate. Such a drawback resembles the "von Neumann bottleneck" in conventional uniprocessor systems, where access to memory over a similar bus restricts performance. In the case of reconfigurable systems, the bus interface is the same but the processor is connected to the reconfigurable unit instead of to a memory unit.

By a similar analogy, Amdahl's Law states that an algorithm's parallel performance is eventually dominated by its serial portions. If, for instance, an algorithm is 90 percent parallelizable, the limit on speedup is 10. This implies that even if the parallel portion of the algorithm can be executed in zero time, the serial portion will still take the same fixed amount of time to execute. Similarly, no matter how much work can be offloaded to the reconfigurable coprocessors, the portions that cannot will tend to dominate the computation time.

In this sense, the same problems that limit the ability to parallelize algorithms also limit the ability to use reconfigurable computing. While there are other issues that limit acceptance of reconfigurable systems, including the lack of mature software development tools and competition from other, more conventional architectures, the basic inability to exploit the parallelism in general-purpose reconfigurable computing will always be a serious concern.

The conventional desktop or server approaches to reconfigurable systems have their difficulties, but reconfigurable computing may still find an agreeable environment in embedded systems, which tend to have streaming data inputs and outputs and may not be at the mercy of the bandwidth of existing system buses. In addition, there may be other attractive features of reconfigurable logic in such embedded systems, including lower overall power consumption and the ability to dynamically adapt to external conditions.

3.9 THE FUTURE OF RECONFIGURABLE SYSTEMS

There appear to be some clear trends in the relatively brief, but active, history of reconfigurable computing. Commercial FPGA devices have continued to be dominant in such systems, but FPGA architectures are also evolving, beginning to incorporate coarser-grained resources. Block memory units and multiplier units have become standard, and even multiple microprocessor cores have found their way onto FPGA devices. Morover, this trend has been mirrored in the coarser-grained research efforts in more recent reconfigurable logic devices. Clearly there is a trend toward coarser-grained elements, as well as a heterogeneous variety of elements.

Perhaps in a related way, large-scale high-performance computing, or supercomputing, has clearly embraced reconfigurable logic. Reconfigurable computing appears to be the path to the higher levels of performance desired by these architectures, particularly as traditional microprocessor architectures have reached a performance plateau. Still, while the manufacturers of supercomputing equipment have clearly embraced reconfigurable computing, it remains to be seen if end users will do so as well.

References

[1] Algotronix, Ltd. *CAL1024 Datasheet*, 1990.

[2] R. Amerson, R. Carter, W. Culbertson, P. Kuekes, G. Snider. Plasma: An FPGA for million gate systems. *Proceedings of the ACM/SIGDA Fourth International Symposium on Field-Programmable Gate Arrays*, February 1996.

[3] R. Amerson, R. Carter, W. Culbertson, P. Kuekes, G. Snider. Teramac-configurable custom computing. *IEEE Symposium on FPGAs for Custom Computing Machines*, April 1995.

[4] J. M. Arnold. The Splash 2 software environment. *Proceedings of the IEEE Workshop on FPGAs for Custom Computing Machines*, April 1993.

[5] P. M. Athanas, H. F. Silverman. Processor reconfiguration through instruction-set metamorphosis. *IEEE Computer* 26(3), March 1993.

[6] J. A. Babb, R. Tessier, A. Agarwal. Virtual wires: Overcoming pin limitations in FPGA-based logic emulators. *Proceedings of the IEEE Workshop on FPGAs for Custom Computing Machines*, April 1993.

[7] V. Baumgarten, F. May, A. Nuckel, M. Vorbach, M. Weinhardt. PACT XPP—A self-reconfigurable data processing architecture. *First International Conference on Engineering of Reconfigurable Systems and Algorithms* (*ERSA)*, Las Vegas, June 25–28, 2001.

[8] P. Bertin, D. Roncin, J. Vuillemin. Introduction to programmable active memories. *Technical Report 3*, DEC Paris Research Laboratory, 1989.

[9] D. H. Brown Assoc. Cray XD1 brings high-bandwidth supercomputing to the mid-market (*http://www.cray.com/downloads/dhbrown_crayxd1_oct2004.pdf*), October 2004.

[10] D. A. Buell, K. L. Pocek, eds. *Proceedings of the IEEE Workshop on FPGAs for Custom Computing Machines*, IEEE Computer Society Press, 1993.

[11] S. Casselman. Virtual computing and the virtual computer. *IEEE Workshop on FPGAs for Custom Computing Machines*, April 1993.

[12] S. Churcher, T. Kean, B. Wilkie. XC6200 FASTMAP^{TM} processor interface. *Proceedings of the Fifth International Workshop on Field-Programmable Logic and Applications, FPL 1995*, August/September 1995.

[13] S. A. Cuccaro, C. F. Reese. The CM-2X: A hybrid CM-2/Xilinx prototype. *IEEE Workshop of FPGAs for Custom Computing*, April 1993.

[14] W. B. Culbertson, R. Amerson, R. J. Carter, P. J. Kuekes, G. Snider. Teramac configurable custom computer. *Field-Programmable Gate Arrays (FPGAs) for Fast Board Development and Reconfigurable Computing, Proceedings of International Society of Optical Engineering,* October 1995.

[15] C. Ebeling, D. C. Cronquist, P. Franklin. RaPiD—Reconfigurable pipelined datapath. *Field-Programmable Logic: Smart Applications, New Paradigms and Compilers*, R. W. Hartenstein, M. Glesner, eds., Springer-Verlag, September 1996.

[16] C. Ebeling, D. C. Cronquist, P. Franklin, J. Secosky, S. G. Berg. Mapping applications to the rapid configurable architecture. *IEEE Symposium on FPGAs for Custom Computing Machines*, April 1997.

[17] G. Estrin. Organization of computer systems—The fixed plus variable structure computer. *Proceedings of the Western Joint Computer Conference*, May 1960.

[18] G. Estrin, B. Bussell, R. Turn, J. Bibb. Parallel processing in a restructurable computer system. *IEEE Transactions on Electronic Computers* 12(5), December 1963.

[19] G. Estrin, R. Turn. Automatic assignment of computations in a variable structure computer system. *IEEE Transactions on Electronic Computers* 12(5), December 1963.

[20] G. Estrin, C. R. Viswanathan. Organization of a "fixed-plus-variable" structure computer for eigenvalues and eigenvectors of real symmetric matrices. *Journal of the ACM* 9(1), January 1962.

[21] O. D. Fidanci, D. Poznanovic, K. Gaj, T. El-Ghazawi, N. Alexandritis. Performance overhead in a hybrid reconfigurable computer. *Reconfigurable Architecture Workshop*, April 2003.

[22] M. Gokhale, W. Holmes, A. Kosper, D. Kunze, D. Lopresti, S. Lucas, R. Minnich, P. Olsen. SPLASH: A reconfigurable linear logic array. *International Conference on Parallel Processing*, 1990.

[23] M. Gokhale, A. Kosper, S. Lucas, R. Minnich. The logic description generator. *Proceedings of the International Conference on Application Specific Array Processing*, 1990.

[24] S. C. Goldstein, H. Schmit, M. Budiu, S. Cadambi, M. Moe, R. Taylor. PipeRench: A reconfigurable architecture and compiler. *IEEE Computer* 33(4), April 2000.

[25] S. A. Guccione. List of FPGA-based computing machines (*http://www.io.com/~guccione/HW_list.html*), 1994.

[26] R. W. Hartenstein, M. Herz, T. Hoffmann, U. Nageldinger. Using the KressArray for configurable computing. *Proceedings of the International Society of Optical Engineering Conference on Configurable Computing: Technology and Applications*, November 1998.

[27] M. W. Holmes, A. Kosper, S. Lucas, R. Minnich, D. Sweely. Building and using a highly parallel programmable logic array. *IEEE Computer* 24(1), January 1991.

[28] T. A. Kean. *Configurable Logic: A Dynamically Programmable Cellular Architecture and Its VLSI Implementation*, Ph.D. thesis, University of Edinburgh, January 1989.

[29] T. A. Kean. Déjà vu, all over again. *IEEE Design and Test of Computers* 22(2), March/April 2005.

[30] J. T. McHenry, R. L. Donaldson. WILDFIRE custom configurable computer. *Field Programmable Gate Arrays (FPGAs) for Fast Board Development and Reconfigurable Computing, Proceedings of the International Society of Optical Engineering*, October 1995.

[31] T. Miyazaki, T. Murooka, M. Katayama, A. Takahara. Transmutable telecom system and its application. *IEEE Symposium on FPGAs for Custom Computing Machines*, April 1999.

[32] T. Miyazaki, K. Shirakawa, M. Katayama, T. Murooka, A. Takahara. A transmutable telecom system. *Field-Programmable Logic: From FPGAs to Computing Paradigms*, Springer-Verlag, August/September 1998.

[33] M. Moe, H. Schmit, S. Copen Goldstein. Characterization and parameterization of a pipeline reconfigurable FPGA. *IEEE Symposium on FPGAs for Custom Computing Machines*, April 1998.

[34] D. S. Poznanovic. Application development on the SRC Computers, Inc. systems. *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium*, 2005.

[35] H. Schmit, D. Whelihan, A. Tsai, M. Moe, B. Levine, R. Reed Taylor. PipeRench: A virtualized programmable datapath in 0.18 micron technology. *Proceedings of the IEEE Custom Integrated Circuits Conference*, 2002.

[36] Silicon Graphics, Inc. Extraordinary acceleration of workflows with reconfigurable application-specific computing from SGI (*http://www.sgi.com/pdfs/3721.pdf*), 2004.