

# Parallel Processing Architectures for Reconfigurable Systems

Kees A. Vissers  
CTO, Chameleon Systems Inc,  
Research Fellow, UC Berkeley  
[kees@cmln.com](mailto:kees@cmln.com), [vissers@eecs.berkeley.edu](mailto:vissers@eecs.berkeley.edu)

## Abstract

*Novel reconfigurable computing architectures exploit the inherent parallelism available in many signal-processing problems. These architectures often consist of networks of compute elements that have an ALU-like structure with corresponding instructions. This opens opportunities for rapid dynamic reconfiguration and instruction multiplexing. The field of computer architectures has significantly contributed to the systematic and quantified exploration of architectures. Novel reconfigurable architecture exploration should learn from this approach.*

*Future System-on-a-Chip platforms will consist of a combination of processor architectures, on-chip memories, and reconfigurable architectures. The real challenge is to design those architectures that can be programmed efficiently. This requires that first a programming environment and benchmarks be created and then that the reconfigurable architectures be systematically explored.*

## 1. Reconfigurable Systems

The rapid progress of silicon technology provides new implementation options for system designers. System implementation was historically seen as a combination of a program on a programmable processor or DSP and a number of dedicated IP blocks. The field of hardware software co-design addressed the quantification of the design trade-offs. The flexibility of programs running on a processor is in general preferred over the fixed hardware implementation in a dedicated IP block. However, the implementation of a high-performance processing task, e.g. pixel processing in TV systems, is expensive on a processor system, which may be a general-purpose processor, a DSP, or a dedicated media processor. The total silicon area and power consumption is 10–50 times higher than that of a dedicated fixed implementation in the form of an IP block in a System on Chip.

Bit-oriented FPGAs have shown a dramatic increase in number of usable gates over the last few years. This progress with FPGAs is an excellent illustration of the very rapid advancement of the silicon technology. The FPGAs have made drastic changes possible to the method

of system prototyping. The flexibility of configuring or programming the system is an enormous advantage and provides a unique time to market. However, the 20–40 times higher cost compared to dedicated implementations in IP blocks has limited the usage to low-volume applications and has prevented the penetration of high-volume, cost-sensitive consumer electronics.

Furthermore, the traditional implementation of a function on an FPGA using logic synthesis based on VHDL or Verilog has provided a radically different programming environment than a conventional C, C++ or JAVA programming language, which is the standard practice for general purpose processors, DSPs or media processors.

Recently reconfigurable computing systems have been proposed that consist of a large number of ALUs. Often these ALUs with corresponding register files are hierarchically interconnected. These reconfigurable computing systems limit the cost of the implementation, while maintaining the flexibility of programmable or reconfigurable systems.

## 2. Exploiting Parallelism

The only interesting hardware in reconfigurable computing is the hardware that has a high level of abstraction as a programmer's view. The challenge is to exploit the inherent parallelism that is often present in large signal processing applications. Compilers for conventional programming languages like C, C++ and Java have difficulty extracting the parallelism that was present in the original problem. The major difficulties are in index expression analysis and loop analysis. Conventional compilers have shown reasonable success extracting an instruction-level parallelism in the range of 4–10 for VLIW architectures and superscalar architectures.

Signal processing systems can conveniently be expressed in a graphical block diagram fashion. Illustrations of these systems are the Mathworks' Simulink, Cadence's SPW, Synopsys' System Studio and UC Berkeley's Ptolemy. Recently FPGA vendors have started to support dedicated libraries for the Simulink tools. Application programmers can express their signal processing functions in a convenient manner. Dedicated

implementations of multipliers and adders in the FPGA fabric can be exploited. The Simulink environment often uses logic synthesis as 'a back-end' tool for the FPGA synthesis.

Currently there are several approaches to augment C-like programming languages to support the extraction of fine-grain parallelism. This is sometimes used as the entry for new ALU-based reconfigurable computing systems.

### 3. Reconfigurable Architectures in Terms of Computer Architectures

The field of building multi-processor systems is not new. Numerous computer architectures have been proposed for multi-processing systems, including vector-processing machines, message passing machines, and cache-coherent shared-memory machines. The specific focus of signal-processing systems for reconfigurable computing creates new opportunities. In computer architecture terms, these networks of interconnected ALUs are "distributed register file architectures", often without any cache architectures or memory hierarchy. The result is that the programmer has to manipulate the transport of data and programs, called "configuration" in FPGA terms. In terms of computer architecture, this is a back to the times of overlay programming, first made popular for FORTRAN.

Most bit-oriented FPGAs have a very large number of bits for a single configuration. This stems from the history of logic-synthesis-driven general, bit-oriented flexibility. Any processor-based system uses instructions for ALU-like architectures. The systematic design of instruction sets for processors can also be used for configurable systems. Code size can be traded against special flexibility. The introduction of a systematic quantification and exploration of computer architectures has led to the design of RISC and VLIW processors. The field of reconfigurable computing can and should apply many lessons from the computer architecture research to the quantification and systematic design of systems. Building a suite of benchmarks and a retargetable compiler or mapping system are essential for this systematic approach.

### 4. Reconfiguration During Run-time

The programmer of conventional computer architectures is completely abstracted from the memory hierarchy. Explicit understanding and manipulation of the location of the program code is irrelevant. The latter is not true for reconfigurable systems.

Dynamic reconfiguration in bit-oriented FPGAs has often been limited by the speed of the internal bus architectures and by the very large number of bits required

for a configuration. The emergence of new, ALU-based reconfigurable computing architectures opens the possibility of rapid dynamic reconfiguration, since the instruction-based design is effectively a compression of the configuration bits.

Conventional computer architectures time multiplex the complete calculation over one processor. In FPGAs often the complete opposite is done: a spatial unrolling of the complete calculation is done with just one large configuration. An ALU-based reconfigurable computing structure can easily support dynamic configuration and limited-time multiplexing of several instructions. This will remove the limitation of "maximum number of usable gates" in conventional FPGAs. The systematic analysis and exploitation of dynamic reconfiguration will be a major step forward for reconfigurable computing platforms. This will allow a trade-off in space and time, without the need to recode or rewrite the algorithm. Furthermore, the contents of the next configuration can depend on the result of the current calculation. This is similar to the branch instruction in conventional architectures.

### 5. Future Systems on a Chip

Complete systems must often perform several tasks, such as searching an electronic TV program database, processing audio pixel processing for high-performance video, and system functions like allocating memory and keeping the administration of the streaming of real-time data. Silicon technology today already allows the integration of several processors and dedicated IP functions. Future systems will shift towards more programmable and reconfigurable integrated Systems on a Chip (SoCs). These will include embedded memories and several processors, including a general-purpose processor that will run an operating system. More and more dedicated IP blocks will be replaced by reconfigurable solutions. Understanding the unique costs, benefits, and application domains of *all* of the various implementations will lead to novel, highly effective, very flexible Systems on a Chip. The increasing cost of masks and the increasing risks of building a "first time right" Silicon implementation will drive the industry to platforms that contain forms of reconfigurable logic and reconfigurable compute structures. The great challenge is the integration of the various programming views of the subsystems.