MATRIX: Coarse-Grain Reconfigurable Computing (Abstract)

Ethan Mirsky
eamirsky@ai.mit.edu

André DeHon
andre@ai.mit.edu

MIT AI Lab
545 Technology Square, Cambridge, MA 02139
May 1, 1995

Overview
Modern computing substrates are, in general, either fixed-functionality (microprocessors) or reconfigurable at a very fine grain (*e.g.* FPGAs, such as the offerings from Xilinx [Xil93], and DPGAs [TEC$^+$95]). Both types of substrates offer advantages and disadvantages to the computation they are performing. For example, microprocessors offer extremely fast processing for a set of operations, but force all computations to express themselves in terms of this limited set. FPGAs and DPGAs, on the other hand, give computations the opportunity to adapt the substrate to perform their specific operations, but pay a high price on reconfigurable overhead (in terms of on-chip resources). The fine-grained nature of these devices requires that configuration data be stored for every bit of logic and interconnect, while many applications operate on data in larger, more uniform aggregates. MATRIX (*Multiple ALU Architecture with Reconfigurable Interconnect*) is a hybrid architecture which embeds coarse-grain, microprocessor-style, functional units in a reconfigurable array. MATRIX bridges the gap between the two extremes and provides us with a better understanding of the effect that the nature of the computational substrate has on the efficiency of computation.

Preliminary Design
The MATRIX reconfigurable array consists of relatively small, basic functional units (or BFUs) connected through a hierarchical, reconfigurable, interconnect system. Figure 1 shows one possible architecture for the BFUs. Each BFU is designed so that it can serve as an 8-bit, general purpose processor (ALU + register file), a special purpose element (by fixing the function port), or a 256x8 memory block (by disabling the ALU).

One of the great advantages of this system is that large amounts of memory is closely coupled with the processing elements. In addition, the pure-memory option allows a user to deploy large memory blocks wherever they are needed, with only small area overhead (the ALU will be relatively small compared to the memory block).

This ability allows a user to deploy BFUs as instruction stores for other BFUs wherever these are needed. Figure 2 shows two such possible configurations. The left side of Figure 2 shows BFUs arranged as a single processing unit. Units I, II, and III serve as instruction store for unit IV, which is a processing unit. If several of these configurations are programmed into a single chip, the MATRIX array can act as a MIMD-style (Multiple Instruction, Multiple Data) parallel processor.

The right side of Figure 2 shows BFUs arranged in a SIMD-style (Single Instruction, Multiple Data) configuration. In this case, units I and II serve as instruction stores (for the function and address ports), while units III,IV, and V serve as processing units. Another option allows the carry signals in each of processing unit ALUs to be chained together. This configuration allows the right side of Figure 2 to function as a 24-bit wide data-path.

MATRIX Advantages and Goals
The ability of a MATRIX array to act as a SIMD or a MIMD machine, or any combination of the two, is a main source of its advantage over conventional computing systems. While a SIMD machine is extremely inefficient on problems with low data-parallelism and MIMD machines pay a very high resource overhead on highly data-parallel problems, a MATRIX array can efficiently handle both.

MATRIX also has significant advantages over more fine-grained reconfigurable logic, especially in terms of achievable performance and resource utilization. While a traditional FPGA pays routing and interconnect overhead for

Acknowledgments: This work is conducted under the direction of Principal Research Scientist Thomas Knight Jr. This research is supported by the Advanced Research Projects Agency of the Department of Defense under Rome Labs contract number F30602-94-C-0252.

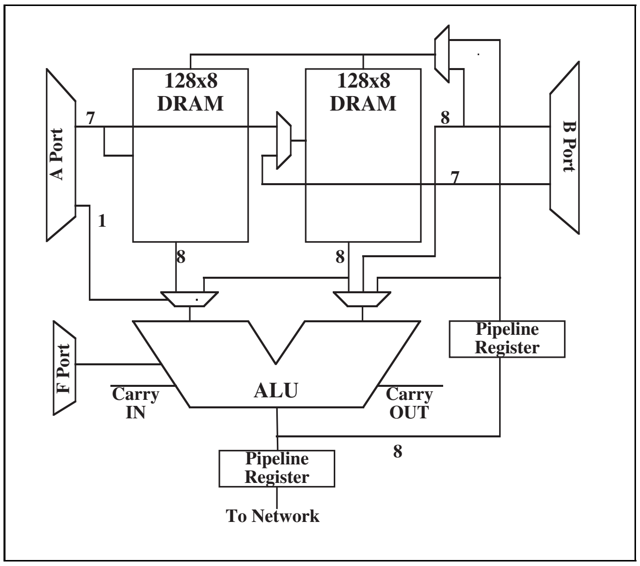

Figure 1: MATRIX Basic Functional Unit

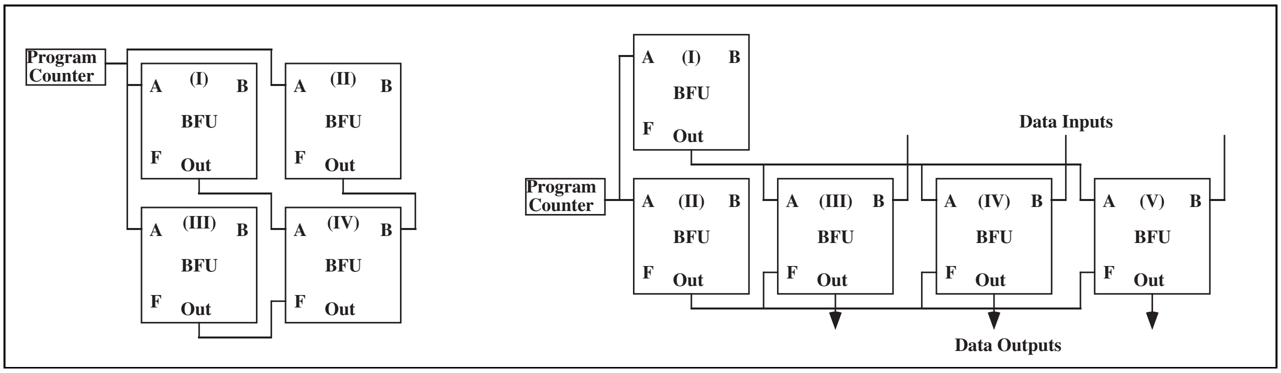

Figure 2: MIMD and SIMD Style BFU Configurations

each bit in a datapath, MATRIX only pays overhead interconnecting 8-bit units, lowering the routing and configuration overhead. Factored over all of the switches on a single die, the configuration overhead represents a significant area savings.

Like any configurable logic element, MATRIX is best suited for computations which have a highly consistent set of operations (always performing the same specialized operations). Combined with the advantages described above, this means that MATRIX is ideally suited for operations such as:

- Digital Signal Processing (DSP)
- Video Processing
- Encryption

By exploring the ability of a prototype MATRIX array to serve as a substrate for these, and other, computational processes, we will develop a better understanding of the advantages of coarse-grain reconfigurable computing. Future work will explore hybrid designs, combining MATRIX-style BFUs with more fine-grained logic and coupling MATRIX arrays with more conventional microprocessors.

References

[TEC+95] Edward Tau, Ian Eslick, Derrick Chen, Jeremy Brown, and André DeHon. A First Generation DPGA Implementation. Transit Note 114, MIT Artificial Intelligence Laboratory, January 1995. Anonymous FTP `transit.ai.mit.edu:transit-notes/tn114.ps.Z`.

[Xil93] Xilinx, Inc., 2100 Logic Drive, San Jose, CA 95124. *The Programmable Logic Data Book*, 1993.