Reinventing Computing

A First Generation DPGA Implementation (Abstract)

Ian Eslick   Derrick Chen   Edward Tau
{beethovn,kuang,edtau}@ai.mit.edu
Jeremy Brown   Ethan Mirsky   André DeHon
{jhbrown,eamirsky,andre}@ai.mit.edu

MIT AI Lab
545 Technology Square, Cambridge, MA 02139
April 29, 1995

Dynamically Programmable Gate Arrays (DPGAs) represent a hybrid architecture lying between traditional FPGAs and SIMD arrays. Notably, these arrays can efficiently support computations where the function of the array elements needs to vary both among array elements during any single cycle and within any single array element over time. We describe our minimal, first generation DPGA. This DPGA uses traditional 4-LUTs for the basic array element, but backs LUT and interconnect programming cells with a 4-context memory implemented using dynamic RAM. Additionally, this DPGA supports non-intrusive background loads of non-active contexts and automatic refresh for the dynamic memory cells. We draw several lessons from this design experience which may be relevant to future DPGA and FPGA designs.

Traditional Field-Programmable Gate Arrays (FPGAs) have a set of programmable elements which can be configured to personalize the FPGA to implement a user-defined function. Reprogrammable FPGAs implement these programmable elements as memory cells, commonly static RAMs. These cells are configured during a slow programming phase which typically takes 10's of milliseconds due to limited bandwidth to off-chip memories [Xil93] [Atm93]. Once configured, the array personality is static until reloaded. Each array element, consequently, performs the same function throughout the epoch of operation for the device. Devices such as the AT6000 from Atmel can support partial reloads during operation, but the reload still takes a millisecond or more due to bandwidth limitations, and cells cannot be used during the reload operation.

Dynamically Programmable Gate Arrays (DPGAs) [BDK93] differ from traditional FPGAs by providing on-chip memory for multiple array personalities. The configuration memory resources are replicated to contain several configurations for the fixed computing and interconnect resources. In effect, the DPGA contains an on-chip cache of array configurations and exploits high, local on-chip bandwidth to allow reconfiguration to occur rapidly, on the order of nanoseconds instead of milliseconds. Loading a new configuration from off-chip is still limited by low off-chip bandwidth. However, the multiple contexts on the DPGA allow the array to operate on one context while other contexts are being reloaded.

We describe our minimal DPGA implementation. This design represents our first generation effort and contains considerable room for optimization. Nonetheless, the design demonstrates the viability of DPGAs, underscores the costs and benefits of DPGAs as compared to traditional FPGAs, and highlights many of the important issues in the design of programmable arrays.

Our DPGA Features:

- 4 on-chip configuration contexts
- DRAM configuration cells
- non-intrusive background loading
- automatic refresh of dynamic memory elements
- wide bus architecture for high-speed context loading
- two-level routing architecture

Figure 1 shows the architectural composition of our prototype DPGA. Table 1 summarizes the area usage on the prototype. [TEC+95] provides a concise overview of the prototype including a summary of key component sizes and timings. [BCE+94] provides more detailed documentation on the design and implementation of the prototype.

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.

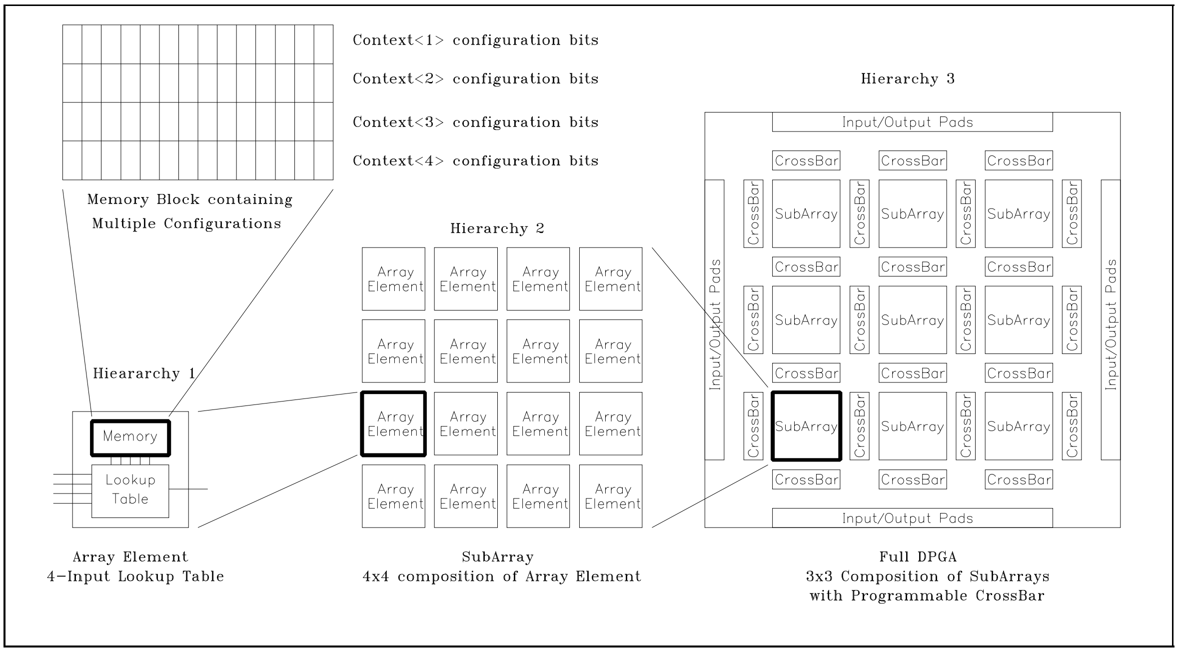

Figure 1: Architecture and Composition of DPGA

Function	Elements	Percent
Logic	Total	16
	Memory array	10
	Memory decode	3
	Fixed Logic	3
Network	Total	66
	Memory array	15
	Memory decode	5
	Switching	19
	Wiring	27
Blank		18
Total		100

Table 1: Array Core Area Breakdown by Programmable Function

References

[Atm93] Atmel Corporation, 2125 O'Nel Drive, San Jose, CA 95131. *Configurable Logic Design and Application Book*, 1993.

[BCE+94] Jeremy Brown, Derrick Chen, Ian Eslick, Edward Tau, and André DeHon. A 1μ CMOS Dynamically Programmable Gate Array. Transit Note 112, MIT Artificial Intelligence Laboratory, November 1994. Anonymous FTP `transit.ai.mit.edu:transit-notes/tn112.ps.Z`.

[BDK93] Michael Bolotski, André DeHon, and Thomas F. Knight Jr. Unifying FPGAs and SIMD Arrays. Transit Note 95, MIT Artificial Intelligence Laboratory, September 1993. Anonymous FTP `transit.ai.mit.edu:transit-notes/tn95.ps.Z`.

[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.