MATRIX
A Reconfigurable Computing Device with Configurable Instruction Distribution

André DeHon <andre@acm.org>

Dan Hartman <omv@ai.mit.edu>

Ethan Mirsky <eamirsky@ai.mit.edu>

Reinventing Computing
MIT Artificial Intelligence Laboratory

Acknowledgments: This research is supported by the Advanced Research Projects Agency of the Department of Defense under Rome Labs contract number F30602-94-C-0252.

MIT AI Lab

Outline

- Problem Identification
- Benefits of Configurable Instruction Distribution
- MATRIX Architecture
- Usage Example – Convolution
- Implementation Details
- Summary

The Problem

We want to have:

- Computing solutions which are programmable
  (post fabrication and in-system)

- Efficient (application oriented) Silicon Usage
  (minimize size and cost necessary to solve problem)

Traditional Approaches

Traditional Solutions (*e.g.* Processors and FPGAs)

- Fix Instruction Distribution

- Limits application range where architecture is efficient

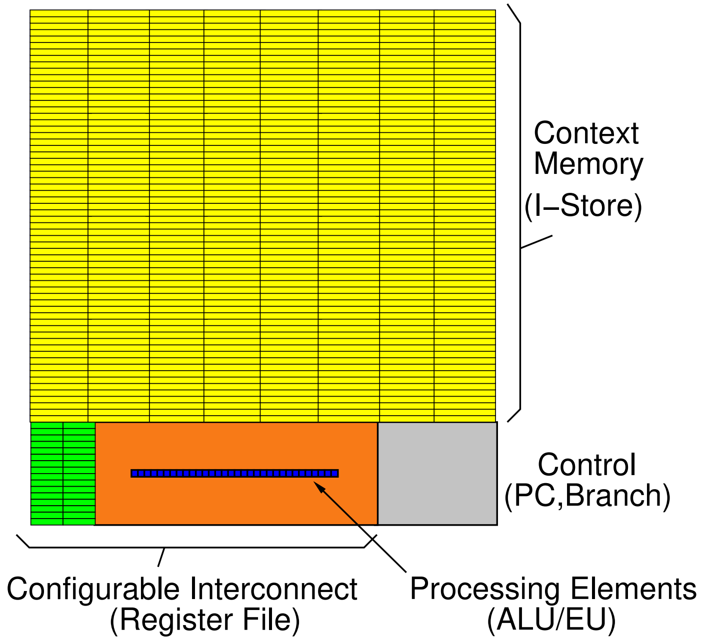
Processors

Dedicate significant area to instruction and data memory

$\Rightarrow$ efficient for heavily multiplexed operation

$\Rightarrow$ inefficient on regular tasks

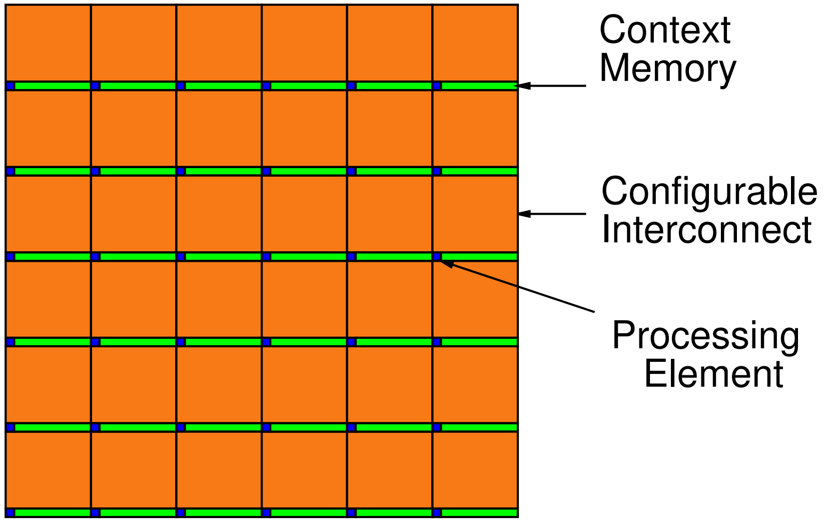
$\Rightarrow$ lower yielded capacity on fine-grained data



Field-Programmable Gate Arrays

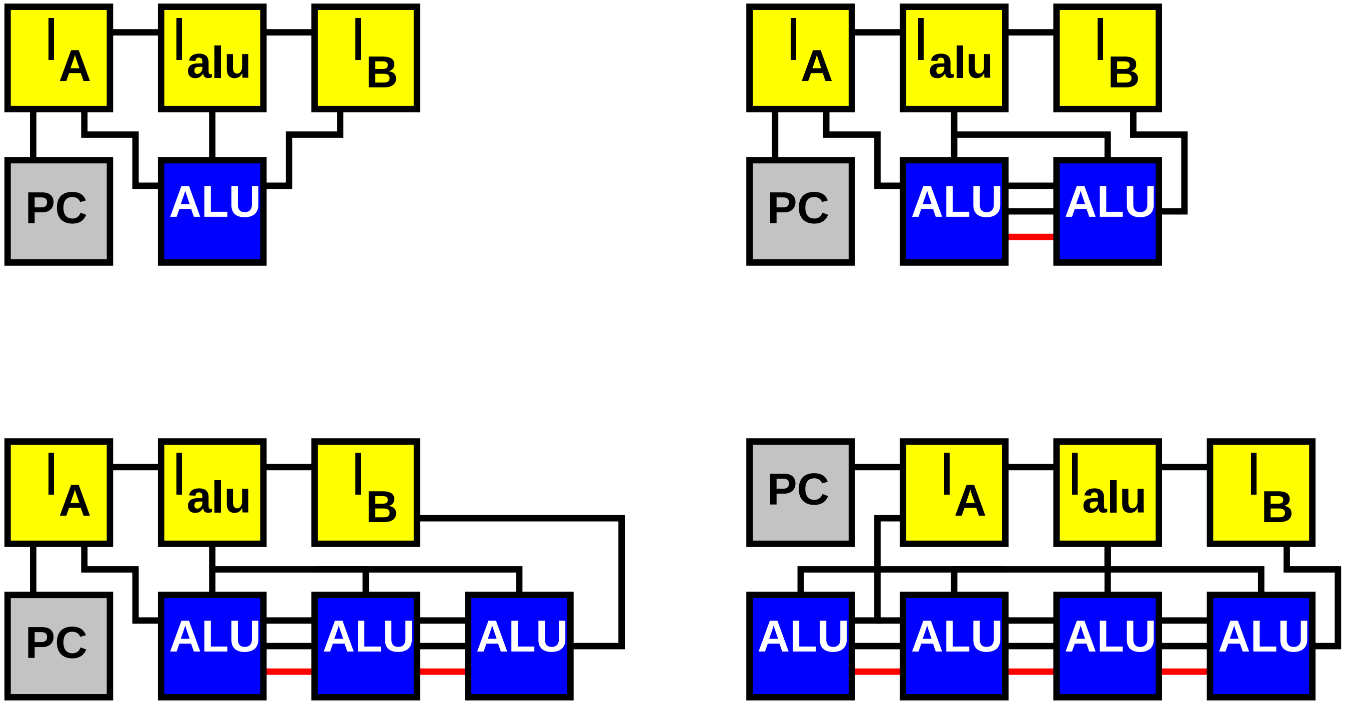
Dedicate most area to fine-grained interconnect

- $\Rightarrow$ **efficient on very regular operations**
- $\Rightarrow$ **high fine-grain yield**
- $\Rightarrow$ **low yielded capacity on irregular tasks**

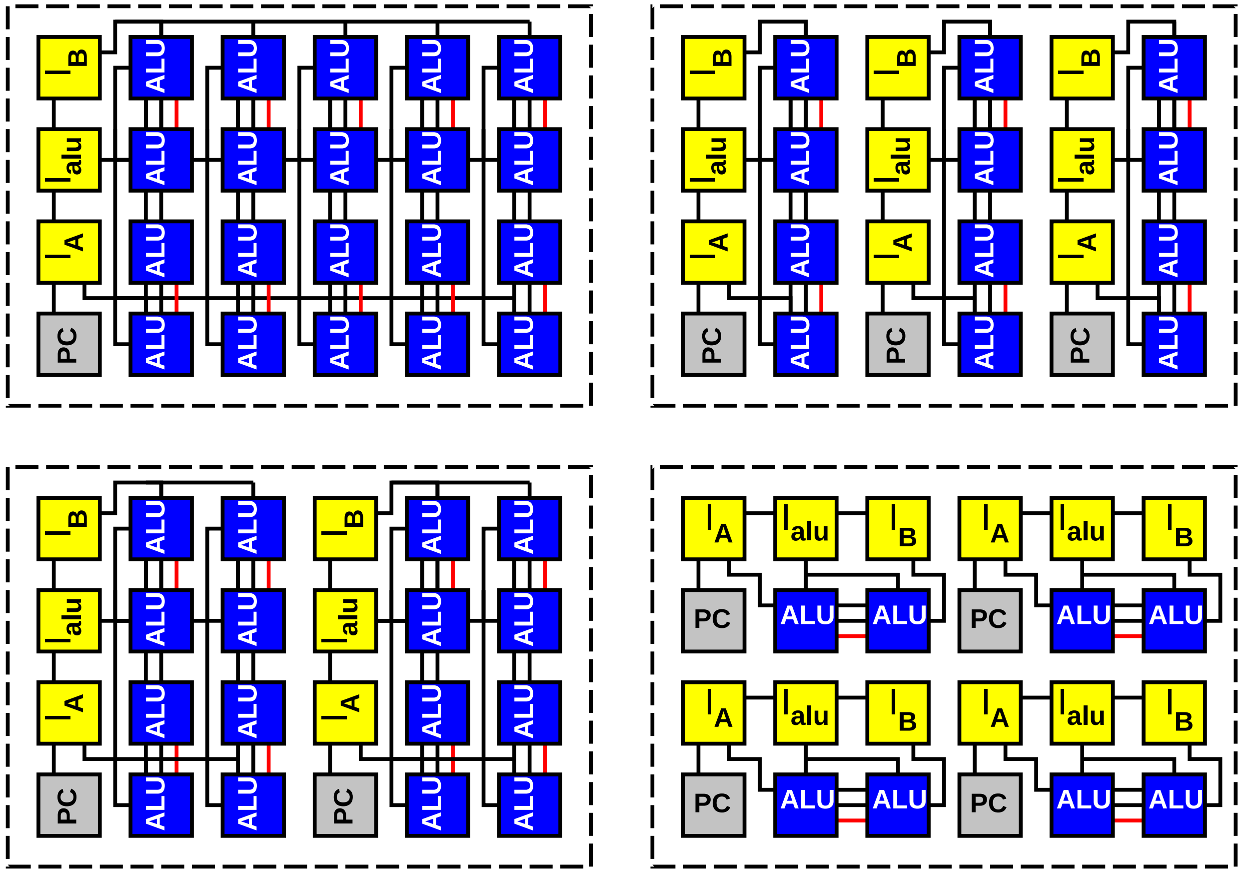


MATRIX Benefits:
Configurable Granularity

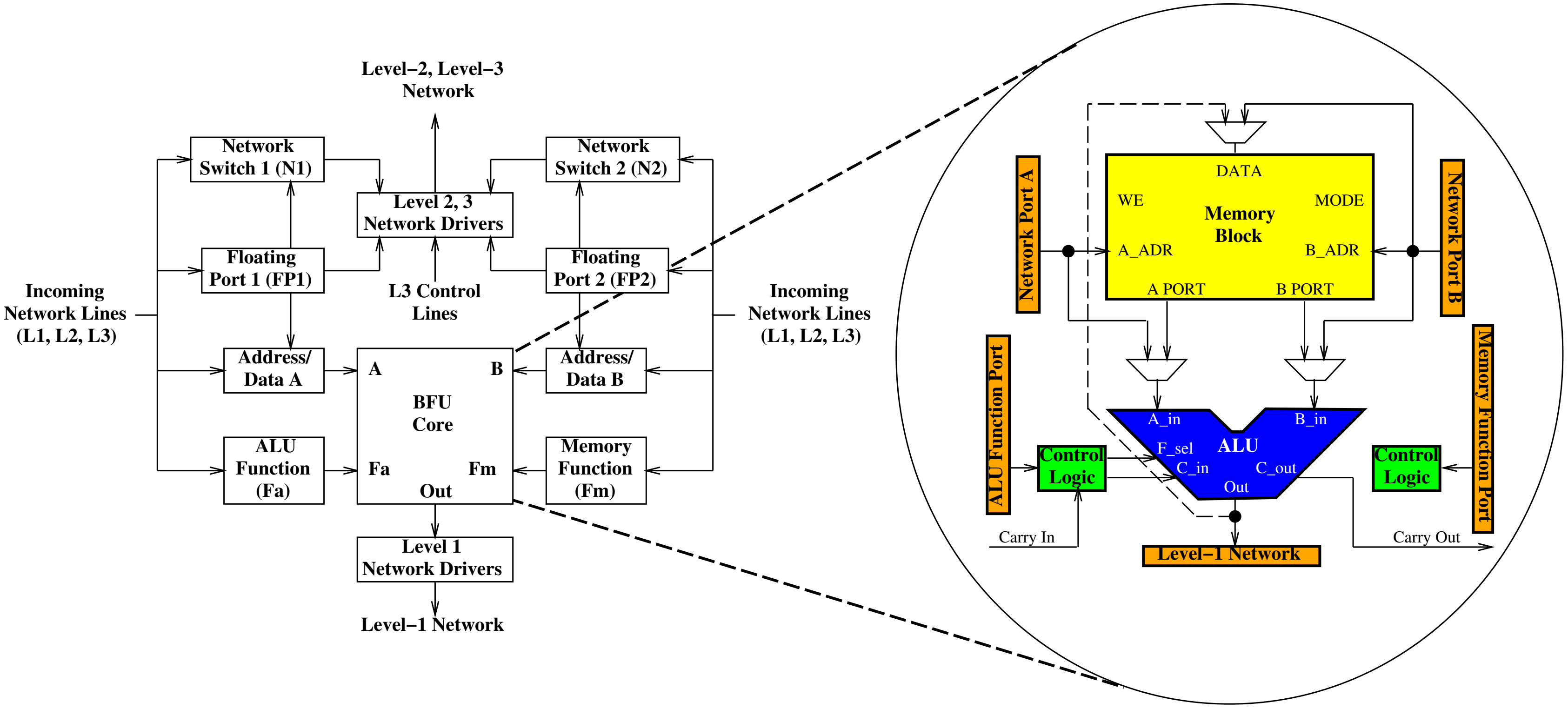
Select granularity according to needs of the application.



MATRIX Benefits: Configurable Instruction Distribution



MATRIX Architecture
Basic Functional Unit (BFU)

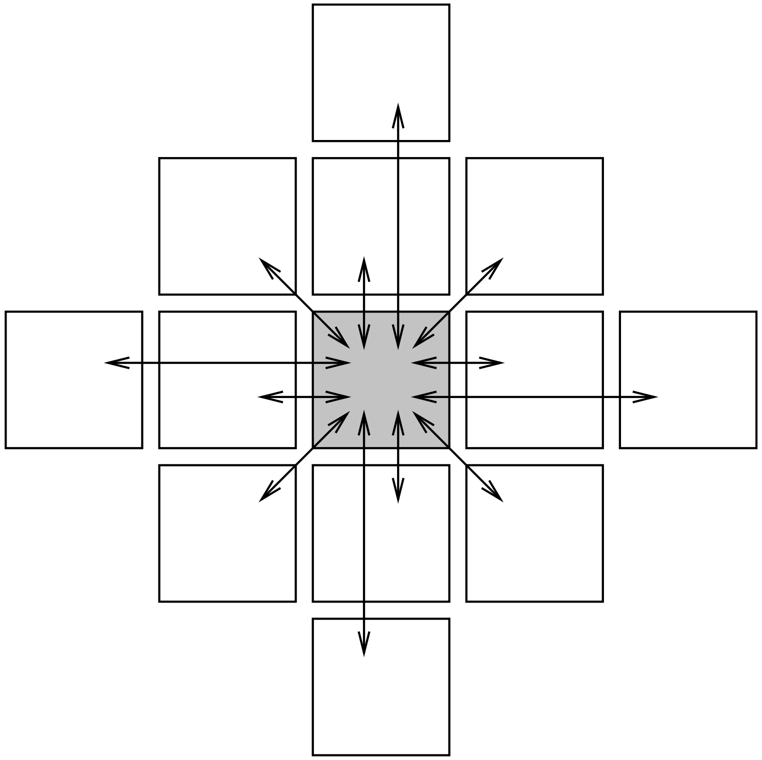


- Same network servers data, instructions, and addresses
- Can use BFU as Instruction Store, Data Memory, or Compute Unit

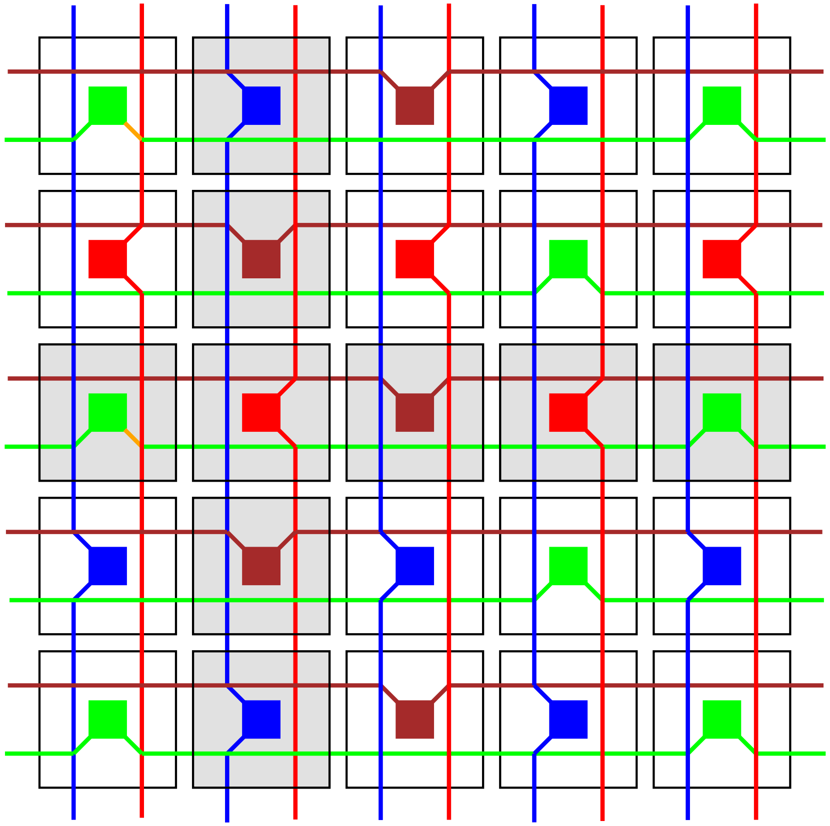
MIT AI Lab

MATRIX Architecture (Network)

Nearest Neighbor Interconnect



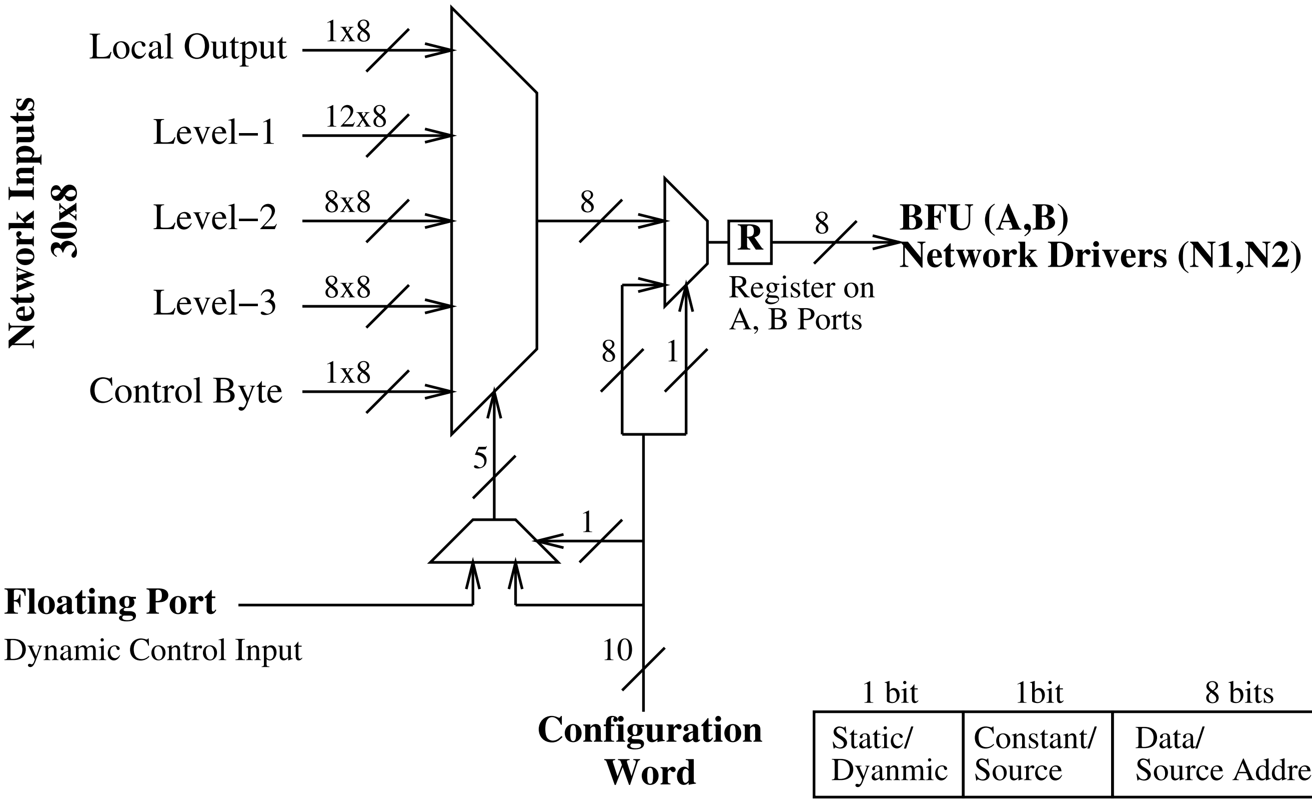
Length Four Bypass Interconnect



Global – 4 Lines per Row/Column

MIT AI Lab

MATRIX Port Architecture



1 bit	1bit	8 bits
Static/ Dyanmic	Constant/ Source	Data/ Source Address

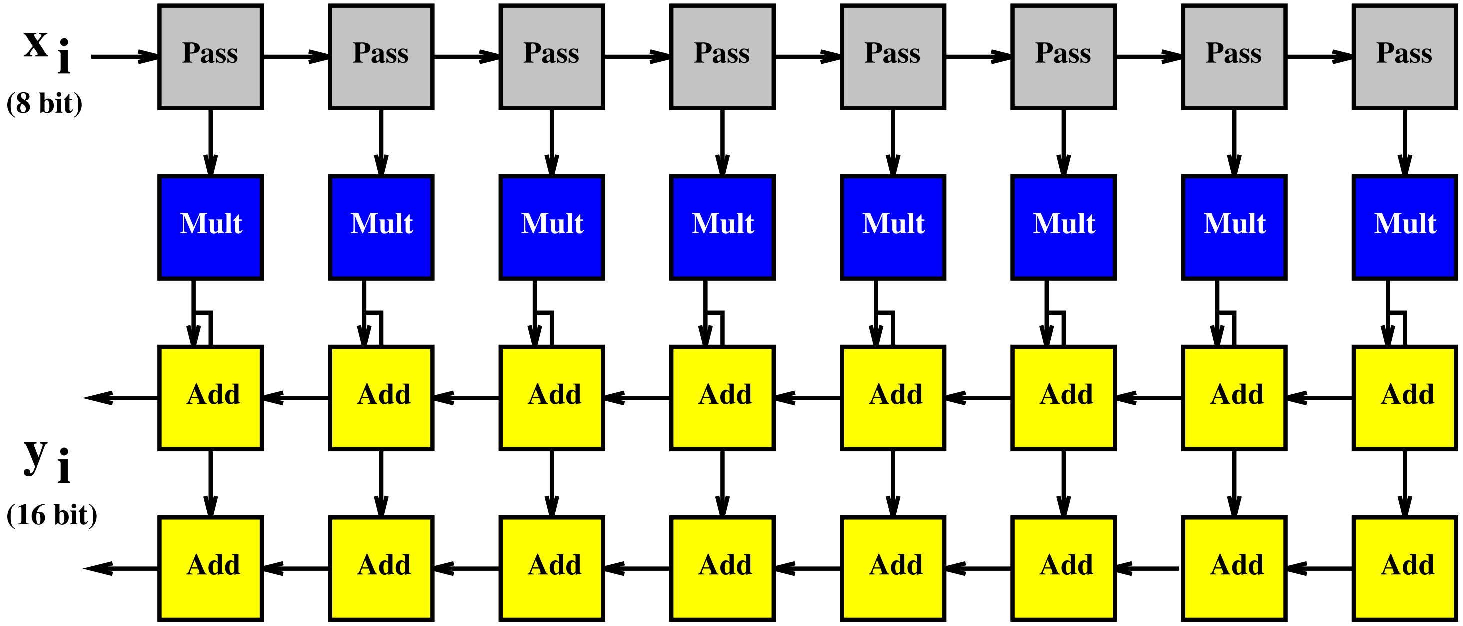
Usage Example
Finite Impulse Response Computation

$$y_i = w_1 \cdot x_i + w_2 \cdot x_{i+1} + \cdots + w_k \cdot x_{i+k-1}$$

Examples based on:

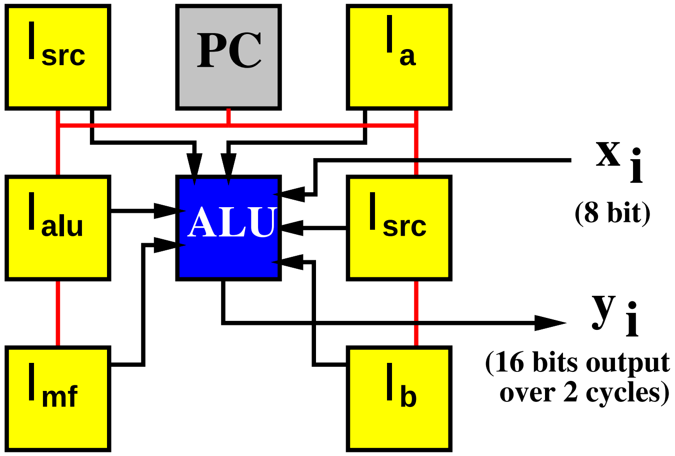
- 8-bit samples
- 16-bit accumulate

MATRIX FIR – Systolic (Spatial)



Area:	$4k$ BFUs as shown ($2k+4$ in practice)
	(k == number of filter weights)
Throughput:	2 cycles per 16-bit result (25MHz)

MATRIX FIR – Microcoded (Space Limited)



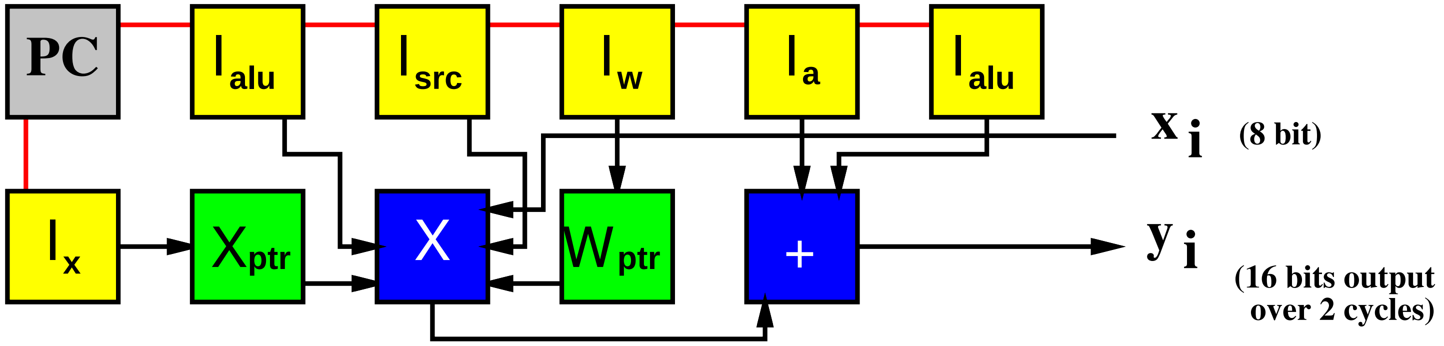
Label	ALU Op	PC
inn	Rs $\leftarrow$ Rs $\times$ Rw	
	Rw $\leftarrow$ $\times$-continue	
	Rl $\leftarrow$ Rs + Rl	
	Rh $\leftarrow$ Rw +-continue Rh	
ent	Rxp $\leftarrow$ Rxp + 1 ; Match $(k+1)$	BNE x2
	Rs $\leftarrow$ $<$ Rxp $>$	(br slot)
	Rxp $\leftarrow$ 65	
	Rs $\leftarrow$ $<$ Rxp $>$	
x2	Rwp $\leftarrow$ Rwp + 1 ; Match $(k+1)$	BNE inn
	Rw $\leftarrow$ $<$ Rwp $>$	(br slot)

Area: 8 BFUs

Throughput: $(8k+9)$ cycles per 16-bit result

(k == number of filter weights)

MATRIX FIR – VLIW



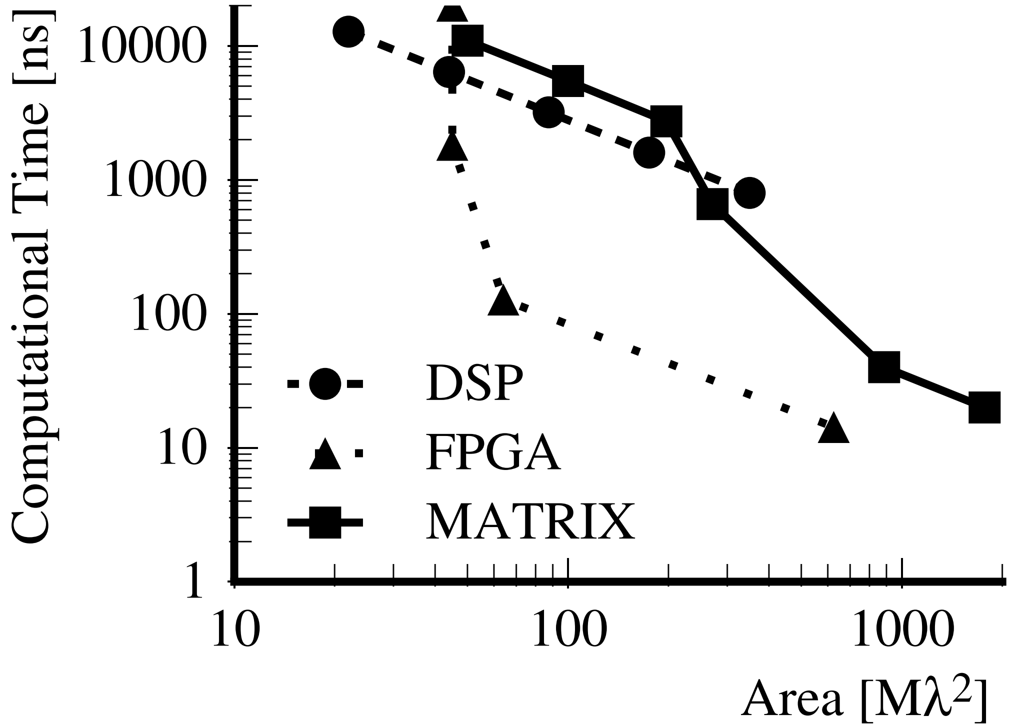
Label	Xptr unit	Wptr unit	PC	MPY unit	+-unit
innerloop	Xptr++ MOD k | 64 output Xptr	Wptr++; Match k output Wptr	BNE innerloop (pipelined branch slot)	$<$ Xptr $> \times <$ Wptr $>$ $\times$-continue	Rhigh$\leftarrow$ Rhigh + MPY-result Rlow $\leftarrow$ Rlow + MPY-result

Area: 11 BFUs

Throughput: $(2k + 1)$ cycles per 16-bit result

(k == number of filter weights)

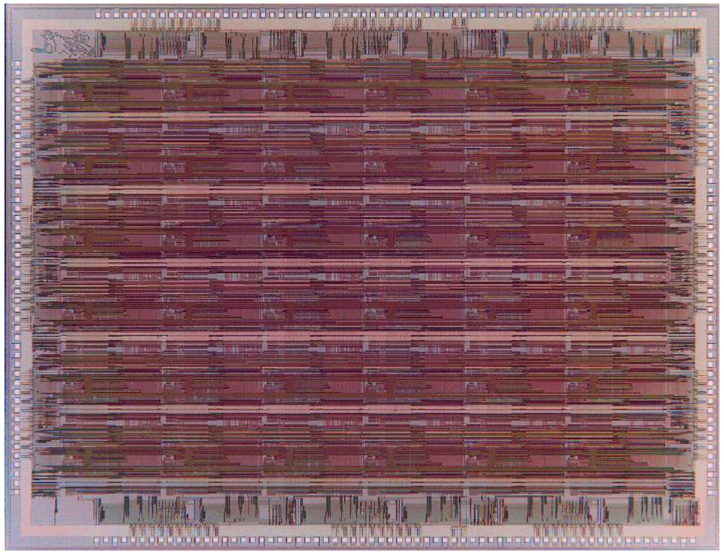
MATRIX Area-Time Comparison



Area-Time for 16 TAP Filter

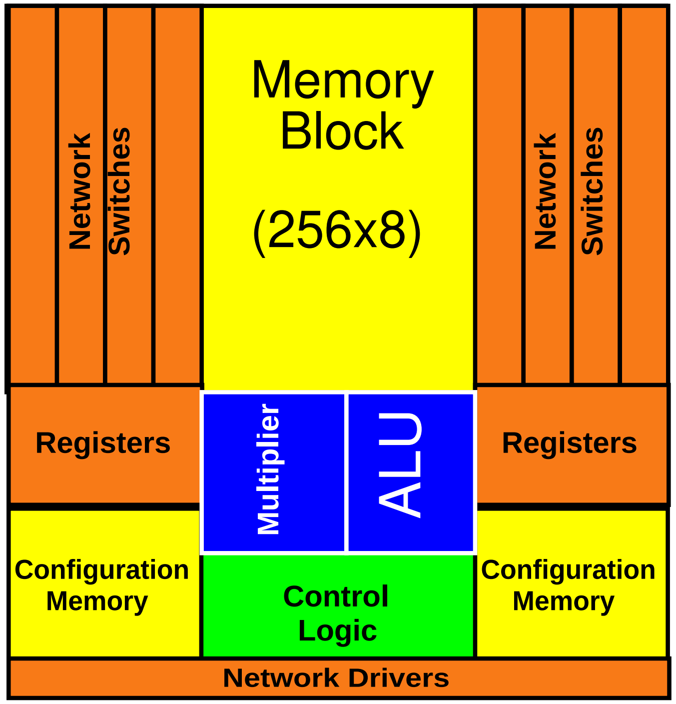
MATRIX Implementation

Technology	0.6μm CMOS	(3 metal)
Die Size	13mm$\times$10mm	
Pin Count	228 (124 I/O)	
Clock Rate	50 MHz	
BFUs	36 (prototype)	(100's in typical devices)
Performance	1.8 Gops/sec	(8-bit Ops)



MATRIX BFU Details

- BFU Size 1.3mm$\times$1.7mm
- Features
  - 8-bit ALU
  - 8$\times$8$\rightarrow$16 Multiplier
  - 256 Byte SRAM (dual port)
  - 8 Byte-wide input ports
  - 28 input sources
  - 11 Byte-wide output drivers
  - Config. Memory: 90 Bytes



Summary

- **Traditional Architecture fix instruction distribution**
  - Large amounts of silicon may sit idle $\Rightarrow$ inefficient
- **MATRIX is a novel architecture which addresses this problem with:**
  - Resource Balance
  - Configurable Instruction Distribution
  - Deployable Resources
- **Resources may be deployed in an application specific manner**
  - Datapaths, Memory, Instruction Distribution