Re-Configurable Computing

Dr. Muhammad Elrabaa

March 2007

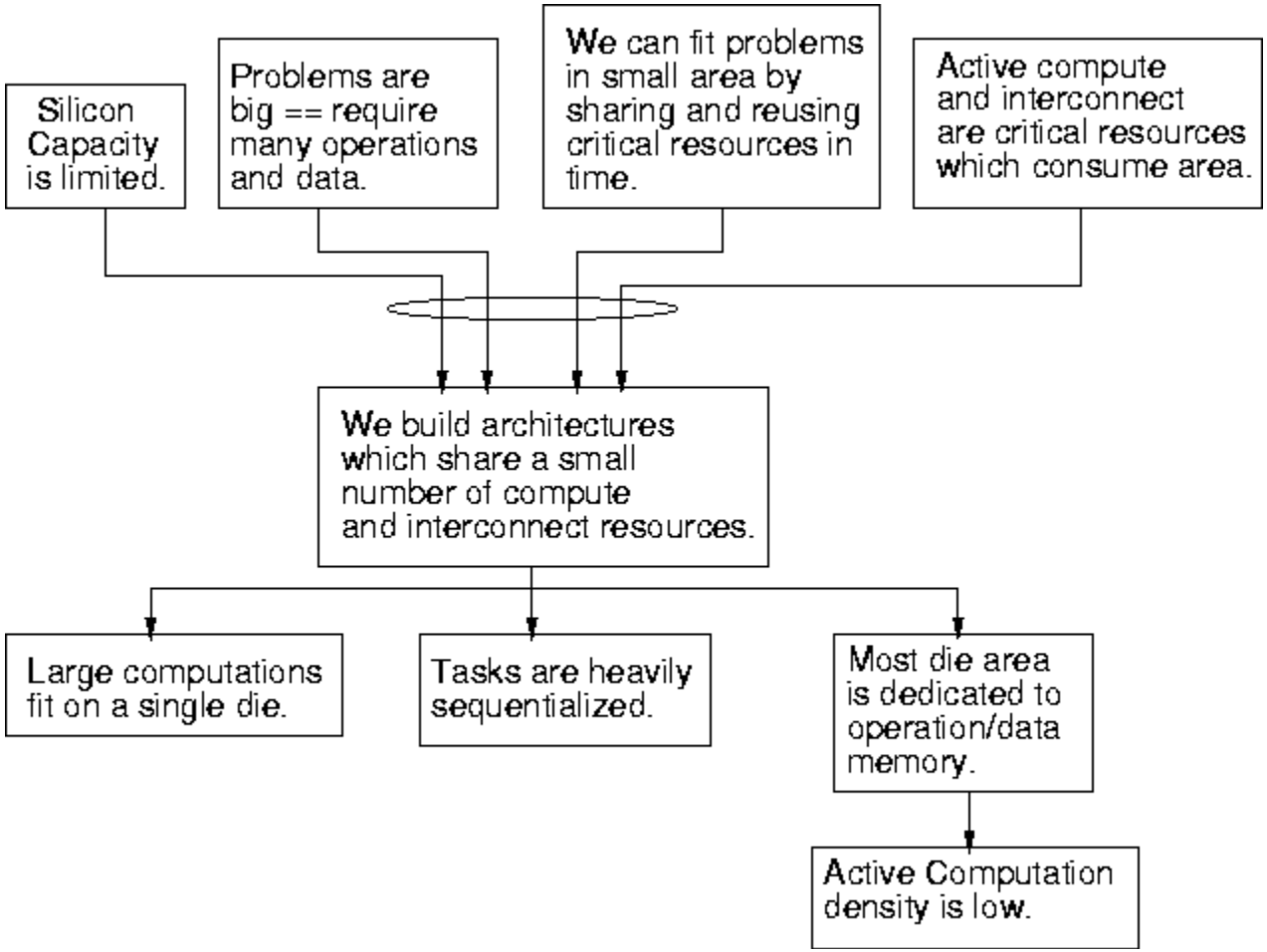
Big Idea

The Biggest Idea here is perhaps the simplest:

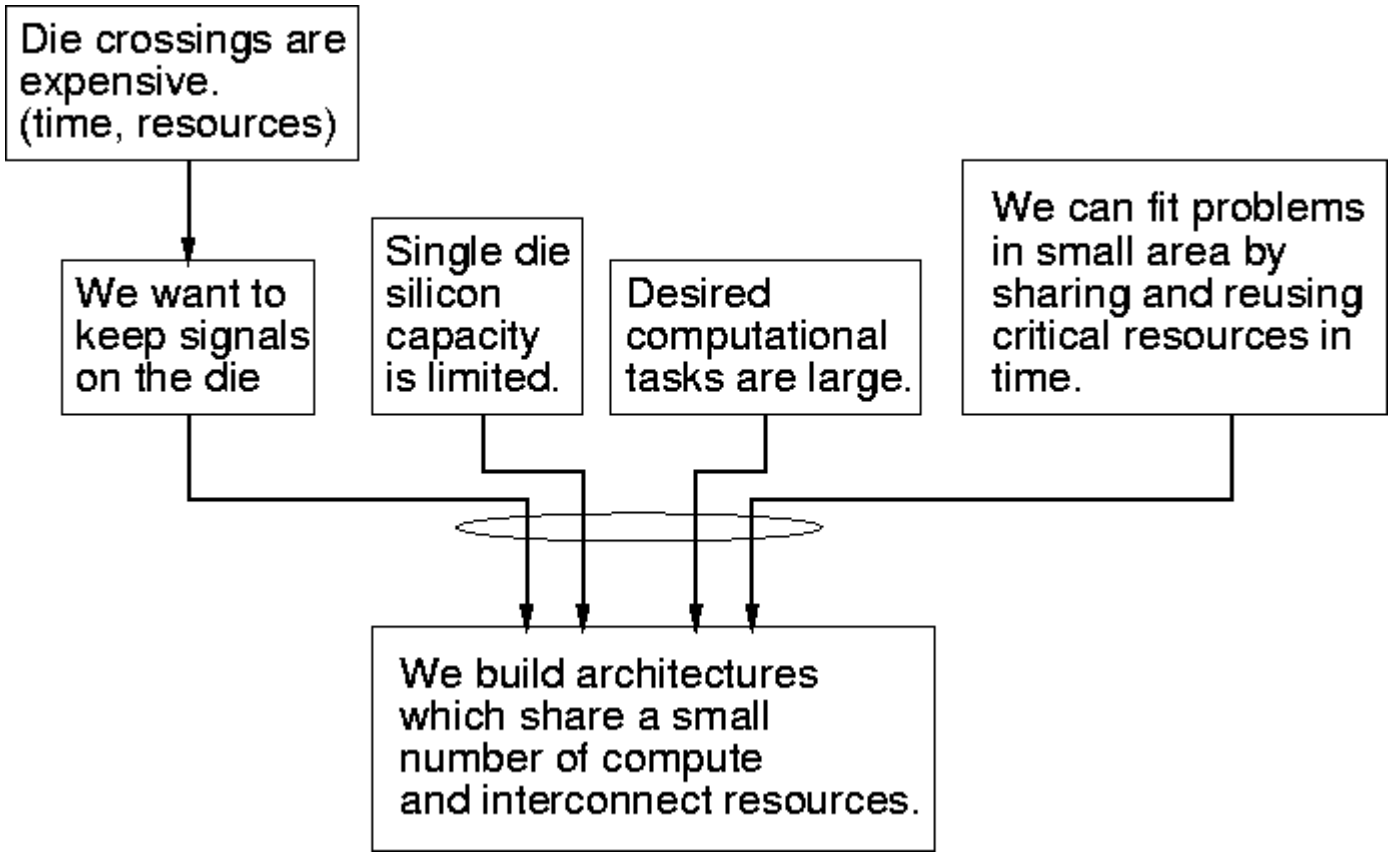
When we have 1000x the resources,
we design computers differently.

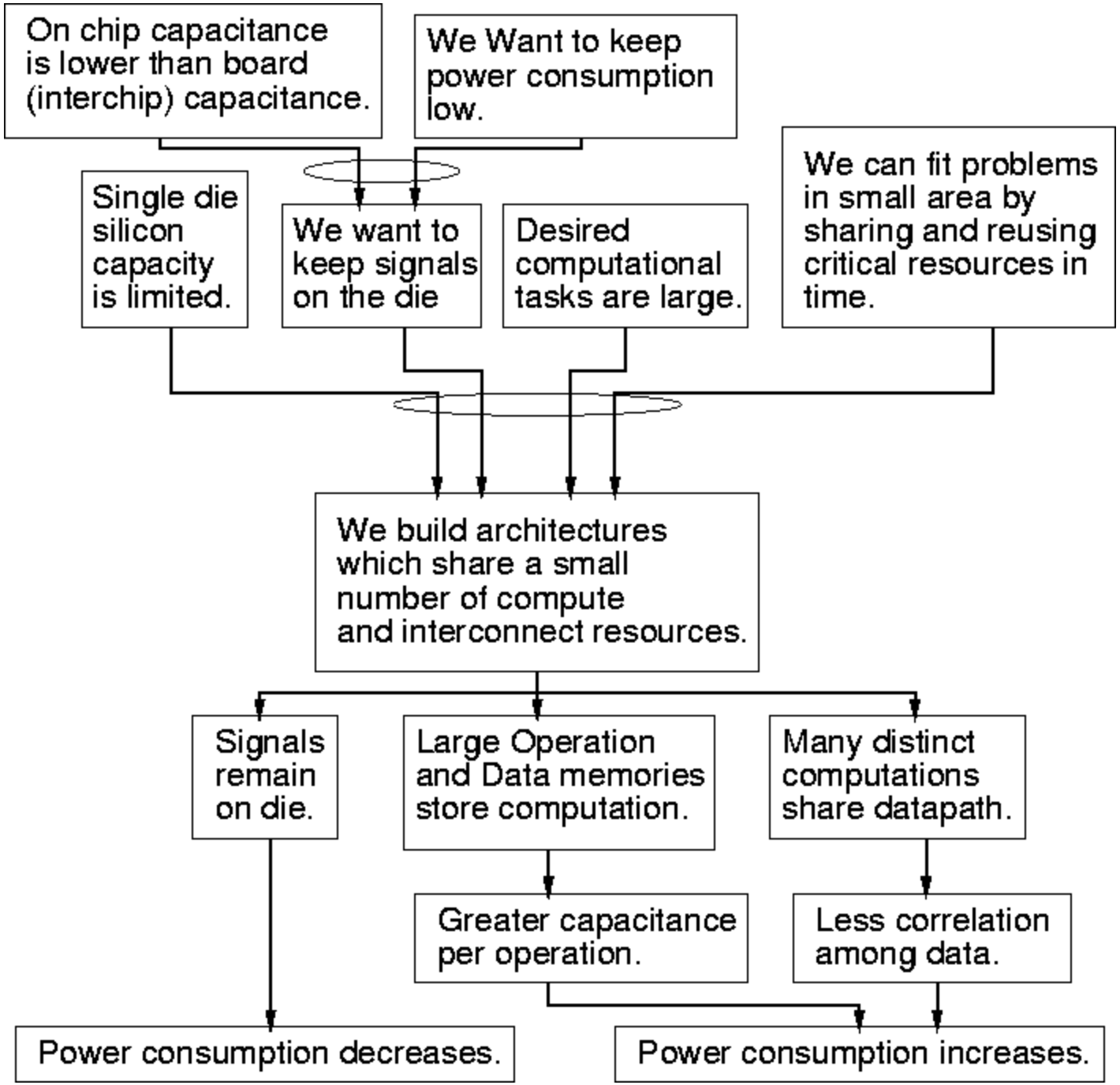
(Good architecture depends on costs.)

In 1983?


Silicon Capacity is limited.
Problems are big == require many operations and data.
We can fit problems in small area by sharing and reusing critical resources in time.
Active compute and interconnect are critical resources which consume area.
We build architectures which share a small number of compute and interconnect resources.
Large computations fit on a single die.
Tasks are heavily sequentialized.
Most die area is dedicated to operation/data memory.
Active Computation density is low.

Cont…


Die crossings are expensive. (time, resources)
We want to keep signals on the die
Single die silicon capacity is limited.
Desired computational tasks are large.
We can fit problems in small area by sharing and reusing critical resources in time.
We build architectures which share a small number of compute and interconnect resources.


On chip capacitance is lower than board (interchip) capacitance.
We Want to keep power consumption low.
Single die silicon capacity is limited.
We want to keep signals on the die
Desired computational tasks are large.
We can fit problems in small area by sharing and reusing critical resources in time.
We build architectures which share a small number of compute and interconnect resources.
Signals remain on die.
Large Operation and Data memories store computation.
Many distinct computations share datapath.
Greater capacitance per operation.
Less correlation among data.
Power consumption decreases.
Power consumption increases.

Challenging our Assumptions

- General-purpose computing machines don’t have to look like processors.
- 1000x increase in single-chip silicon capacity changes the underlying design costs.

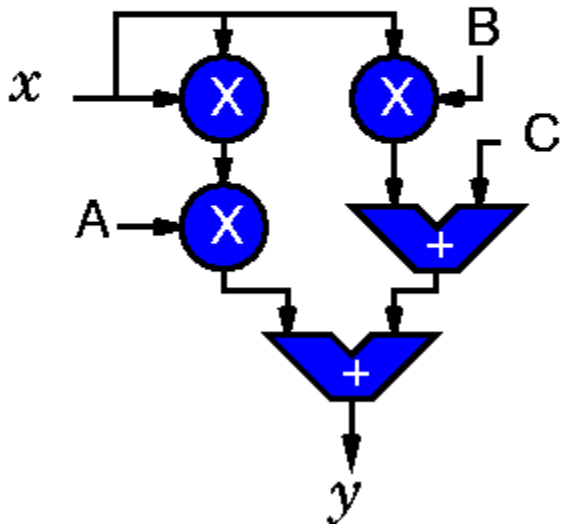
Early RC Successes

- Fastest RSA implementation is on a reconfigurable machine (DEC PAM)
- Splash2 (SRC) performs DNA Sequence matching 300x Cray2 speed, and 200x a 16K CM2
- Many modern processors and ASICs are verified using FPGA emulation systems
- For many signal processing/filtering operations, single chip FPGAs outperform DSPs by 10-100x.

What is Configurable Computing?

Short answer: Computing via post-fabrication, spatially programmed connection of processing elements.

$$y = Ax^2 + Bx + C$$



Defining Terms

Fixed Function:

- Computes one function (e.g. FP-multiply, divider, DCT)
- Function defined at fabrication time

Programmable:

- Computes “any” computable function (e.g. Processor, DSPs, FPGAs)
- Function defined after fabrication

“Any” Computation? (Universality)

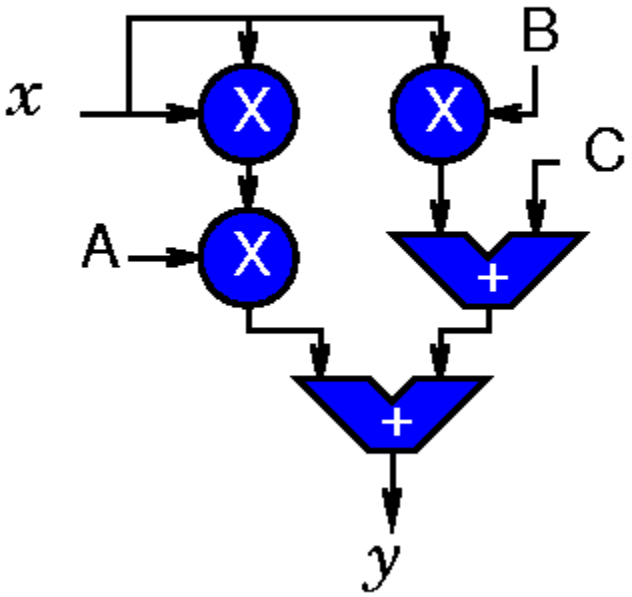
- Any computation which can “fit” on the programmable substrate
- Limitations: hold entire computation and intermediate data

Benefits of Programmable

- Non-permanent customization and application development after fabrication
- economies of scale (amortize large, fixed design costs)
- time-to-market (evolving requirements and standards, new ideas)

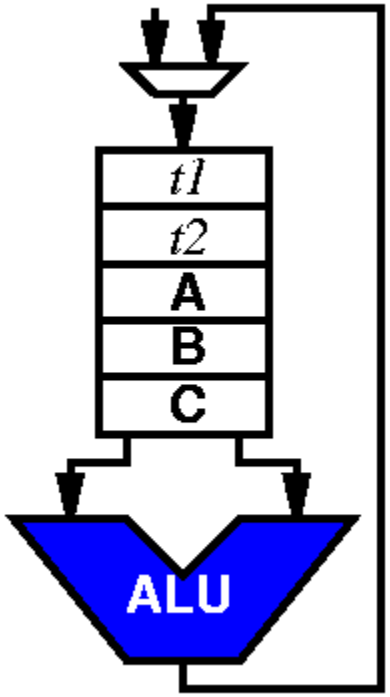
Spatial vs. Temporal Computing

Spatial

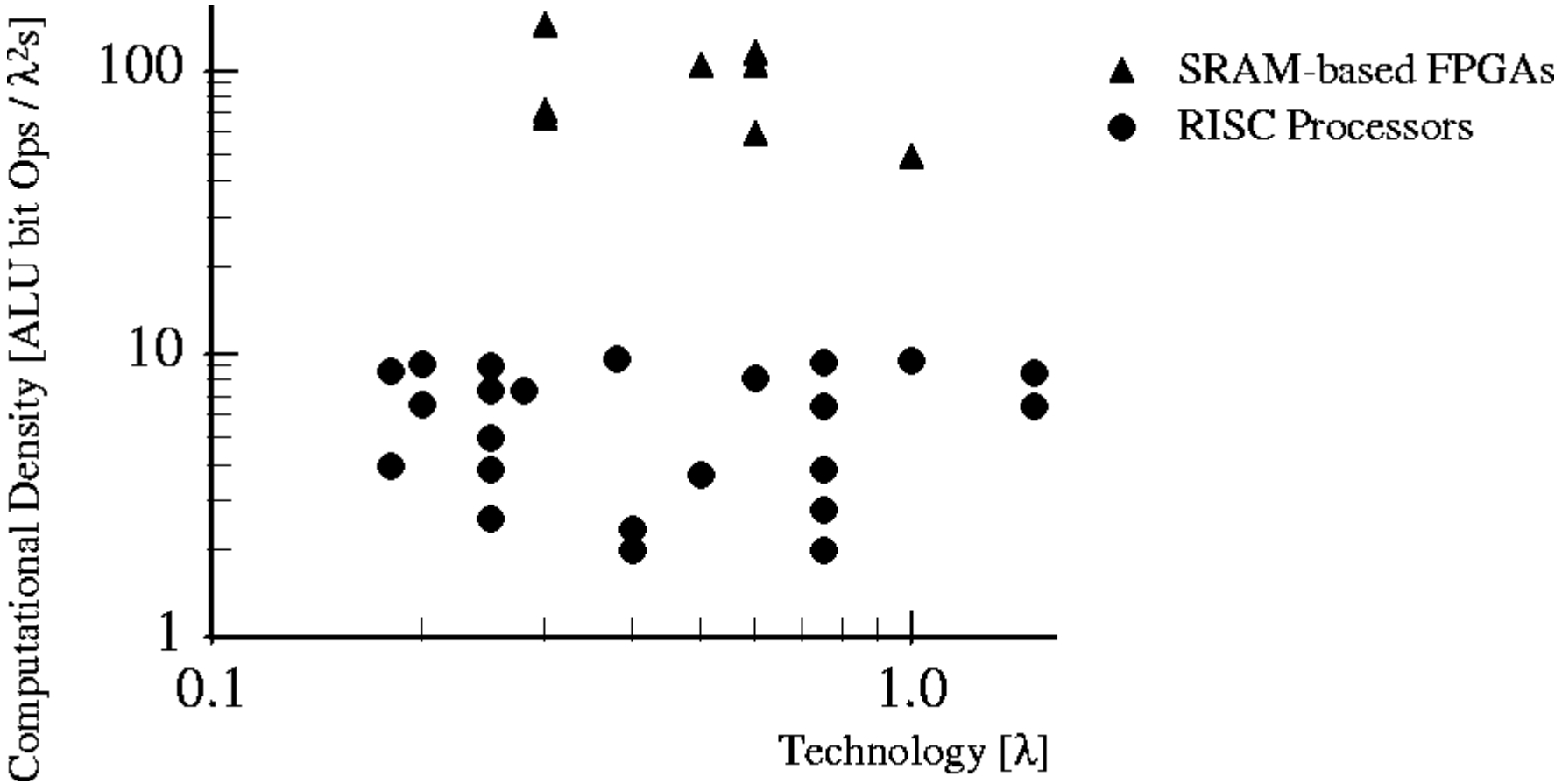


Temporal

$t1 \leftarrow x$
$t2 \leftarrow \mathbf{A} \times t1$
$t2 \leftarrow t2 + \mathbf{B}$
$t2 \leftarrow \mathbf{t2} \times t1$
$y \leftarrow \mathbf{t2} + \mathbf{C}$



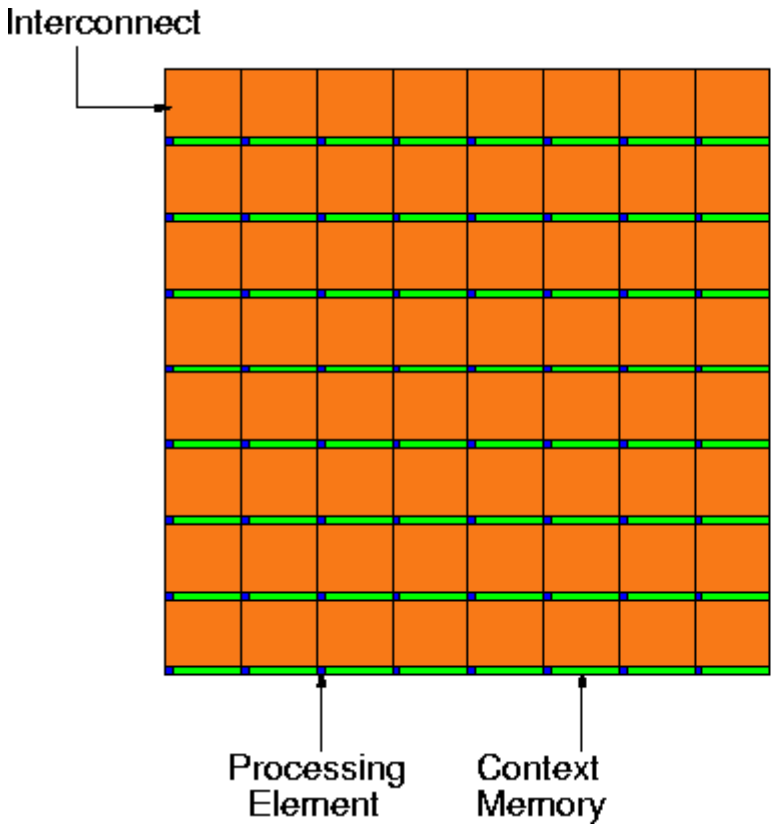
Density Comparison

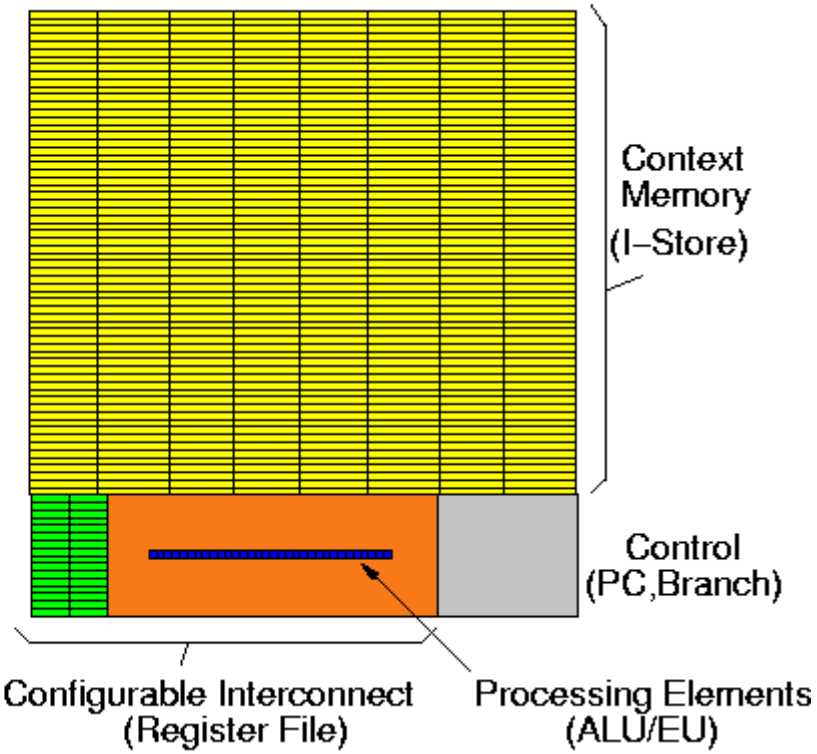


Spatial/Configurable Benefits

- 10x raw density advantage over processors
- potential for fine-grained (bit-level) control --- can offer another order of magnitude benefit

Processor vs. FPGA Area





Configurable Drawbacks

- Each compute/interconnect resource dedicated to single function
- Must dedicate resources for every computational subtask
- Infrequently needed portions of a computation sit idle → inefficient use of resources → Dynamic re-configuration can help

Where CC interesting?

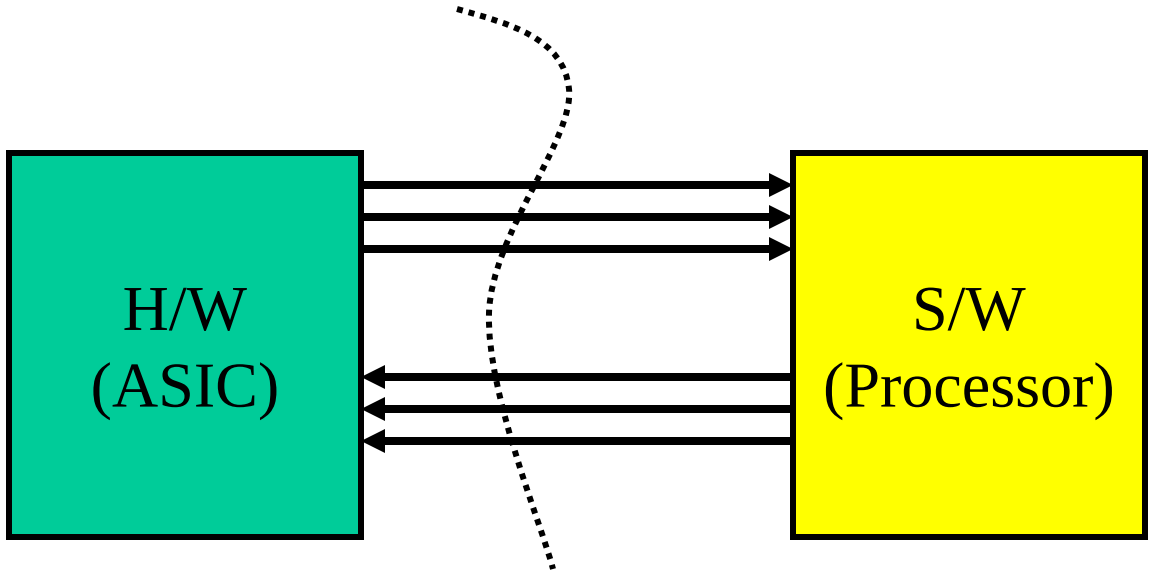
- Regular applications -- need same operation repeatedly
- High concurrency -- large number of operations can occur simultaneously
- Fine-grained data -- small operand data widths

Implications?

- Post-fabrication programmable computing space >> processor arch.
- With $1T\lambda^2$ dies now and $10T\lambda^2$ on the horizon, a much wider space of computing architectures opens up.
- Major feature: more spatial processing, less multiplexing/sharing of resources.

Potential for Embedded Systems

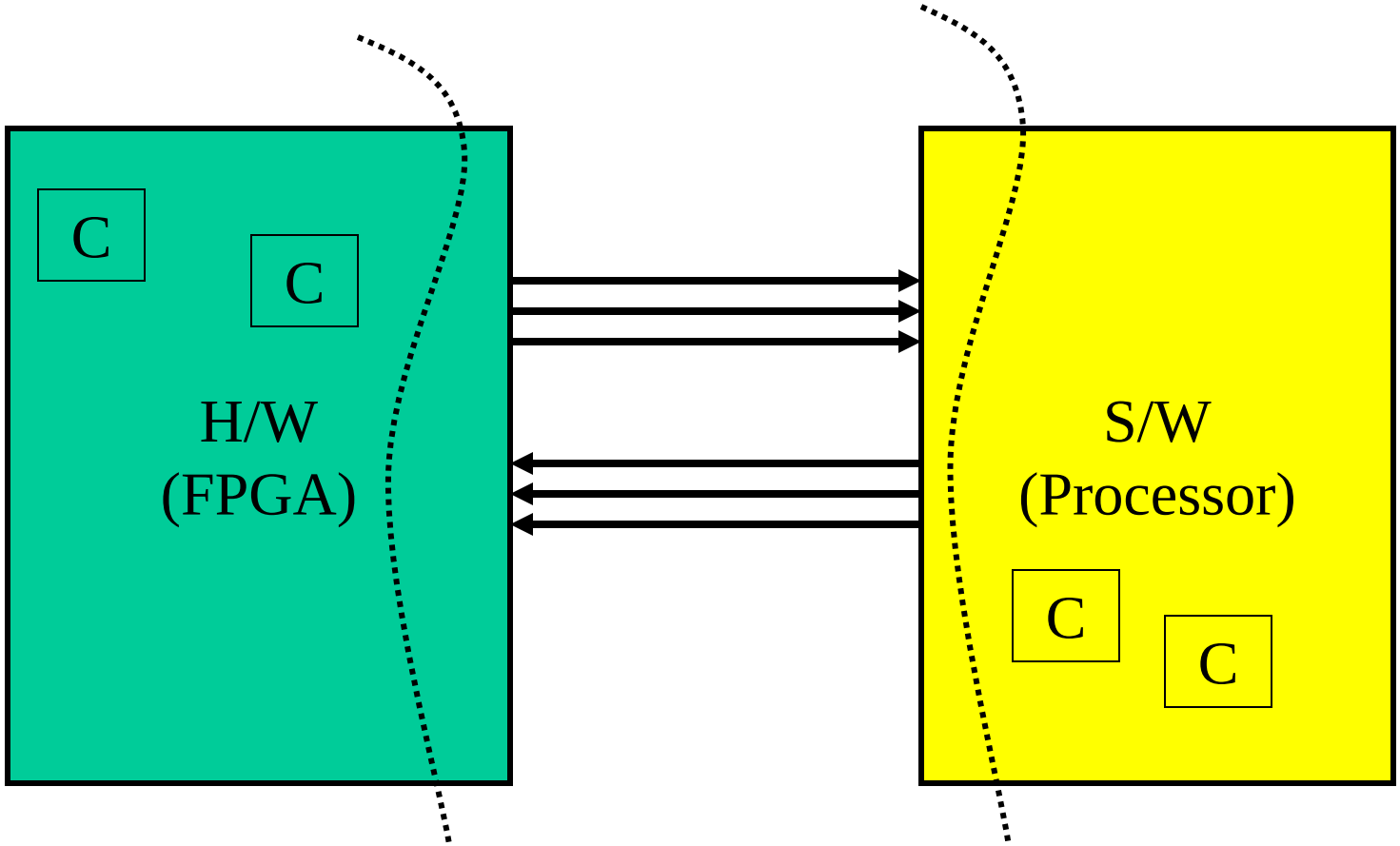
- General Architecture of current Embedded Systems:



- H/W – S/W borders frozen at design time → rigid and require re-design for new products/operating conditions

Cont….

- With RC, the borders need not be frozen … It can change during application life time or even operation time … opening new possibilities … Dynamic tradeoffs (power vs performance or resolution) … Evolvable computing!


C
C
H/W
(FPGA)
S/W
(Processor)
C
C