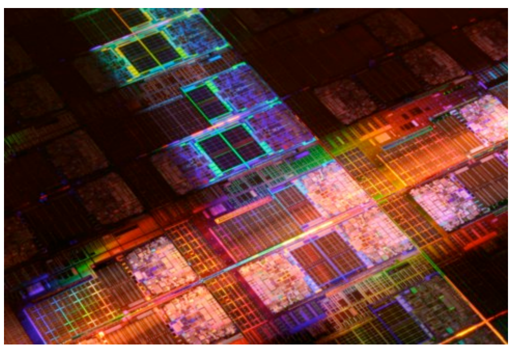
Hardware Parallelism vs. Software Parallelism

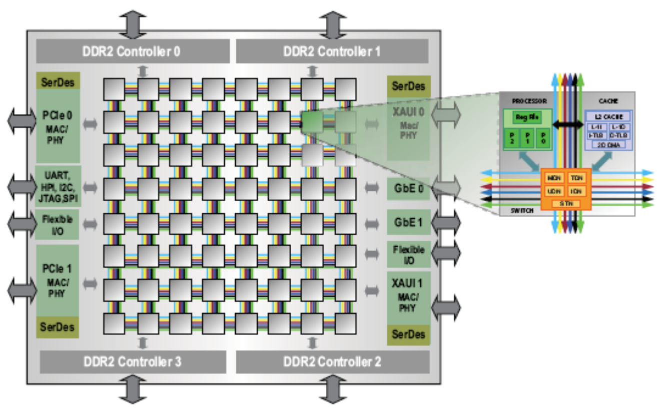
John A. Chandy
Janardhan Singaraju
Department of Electrical and Computer Engineering
University of Connecticut

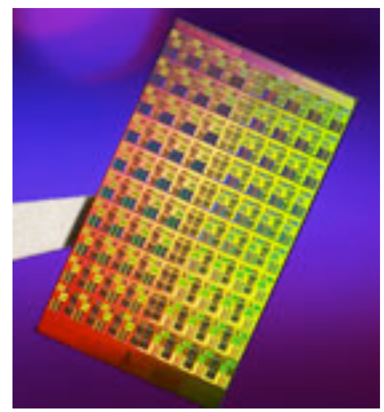
USENIX Workshop on Hot Topics in Parallelism
March 30, 2009

Billions of transistors

- Clock scaling has stopped
- Transistor scaling hasn't stopped
- Billions of transistors – what do we do with them?
  - Multiple cores







Multicore Architectures

- Intel Dunnington) 6 cores, 45nm, ~3GHz
- Tilera 64 core, 90nm, ~700MHz
- Intel Tera, 80 cores, 45nm, 3-5GHz
- Specialized GPUs
  - Nvidia GeForce 280, 240 SPs, 65nm, 1.3GHz

Multicore Architectures

- 8-80 cores at 45nm
- 11nm in 5 years => 100-1000 cores
- What's the problem?
  - Are there enough applications that can drive commodity processors with 100s of cores?
  - How do we get enough data into the processor to feed that many cores?

Multicore Architectures

- Applications
  - Multitasking – doesn’t scale past 8 cores
  - Gaming – GPUs with specialized cores
  - Data intensive computing – scales with data size
    - Not really mainstream
- Mainstream applications drive commodity processors

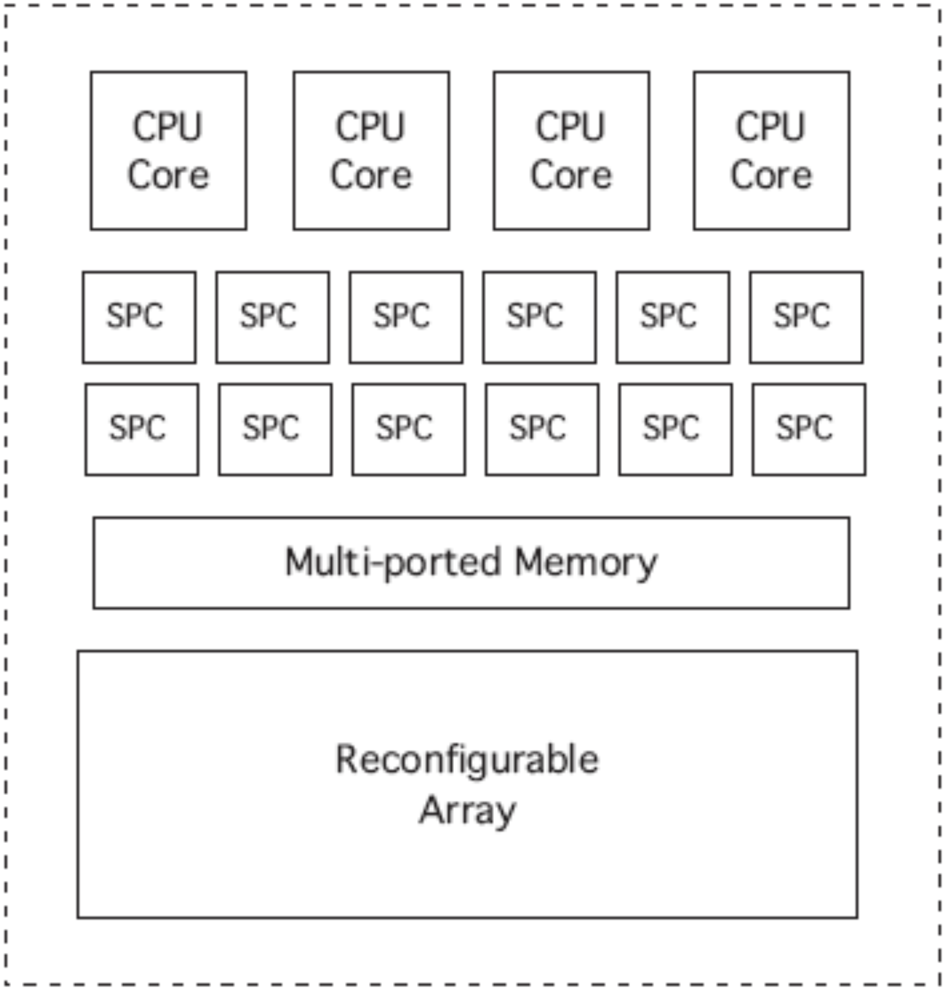
Multicore Architectures

- Mainstream applications do not scale to 1000 cores
- Heterogeneous architectures
  - Mix specialized cores with CPU cores
    - Cell Broadband Engine
      - 1 POWER CPU, 8 SPEs
  - What's practical with 1000 cores?
    - 8-16 CPU cores, 200 GPU cores, DSP cores
    - What do we do with the rest?

Reconfigurable HYbrid Multicore Architecture

- Instead of looking for parallelism in software, look for it in hardware
- Along with the CPU, GPU, and DSP cores, add some reconfigurable hardware cores
- Reconfigurable HYbrid Multicore Architecture (RHYMA)

Reconfigurable HYbrid Multicore Architecture



Hardware computing

Algorithm	Speedup	FPGA	CPU
DES Encryption	24	GARP 133MHz	SPARC 167MHz
Number Factoring	6.8	Xilinx XC4085 16MHz	UltraSPARC 200MHz
Intrusion Detection	27.8	Xilinx Virtex2 303MHz	Intel P4 1.7GHz
Numerical Simulation	5.69	Xilinx Virtex4 50MHz	Intel P4 3.0GHz
Genome Sequencing	100	Xilinx Virtex4 125MHz	AMD Opteron 2.2GHz

Computational Density

Device	Bit-level		16-bit Int.		32-bit Int.		SPFP		DPFP	
	Raw	Sustain.	Raw	Sustain.	Raw	Sustain.	Raw	Sustain.	Raw	Sustain.
Arrix FPOA	6144	6144	384	384	192	192				
ECA-64	2176	2176	13	13	6	6				
MONARCH	2048	2048	65	65	65	65	65	65		
Stratix-II S180	63181	63181	442	442	123	123	53	53	11	11
Stratix-III SL340	154422	154422	933	933	213	213	96	96	26	26
Stratix-III SE260	119539	119539	817	817	204	204	73	73	22	22
TILE64	4608	4608	240	240	144	144				
Virtex-4 LX200	89952	89952	357	116	66	42	68	46	16	16
Virtex-4 SX55	29184	29184	365	110	71	40	31	31	7	7
Virtex-5 LX330T	150163	150163	606	300	131	122	119	116	26	26
Virtex-5 SX95T	48435	48435	599	226	221	92	82	82	15	15
Cell BE	4096	4096	205	205	115	115	205	205	19	19
Tesla C870	5530	5530	346	346	216	216	346	346		
Xeon 7041	1536	1536	42	42	30	30	30	30	24	24
Xeon X3230	4095	4095	128	128	85	85	85	85	64	64

Williams et al., *Reconfigurable Systems Summer Institute, 2008*

Hardware computing

- Parallelism in hardware is at a finer granularity
- Computational Density – performance/area
  - Hardware is good for bit and integer operations
- Hardware has more opportunity for more ILP
- Synchronization between threads is less expensive in hardware

Reconfigurable Computing

- Not a new idea
  - Reconfigurable functional units
    - Chimaera, PRISC, OneChip
  - Reconfigurable coprocessor
    - Garp, PipeRench, DISC, Prism
- Hasn't left the research lab

Reconfigurable HYbrid Multicore Architecture

- What's different now?
  - 10 years ago, performance was easier to get through clock and transistor scaling
    - Hardware acceleration wasn't worth it
  - Transistors are free now
- What's the same?
  - Hardware design is still hard
    - But we have better tools
      - Matlab to HDL, C to VHDL, etc.

Reconfigurable HYbrid Multicore Architecture

- Hardware design is still hard
  - We can't expect software developers to design hardware
  - But, we don't expect software developers to write most of their software either.
    - Libraries provide common routines
      - Encryption, compression, image processing, etc.
  - Let's do the same for hardware
    - Hardware IP exists
    - Software APIs for hardware – OpenFPGA, PFIF

University of Connecticut

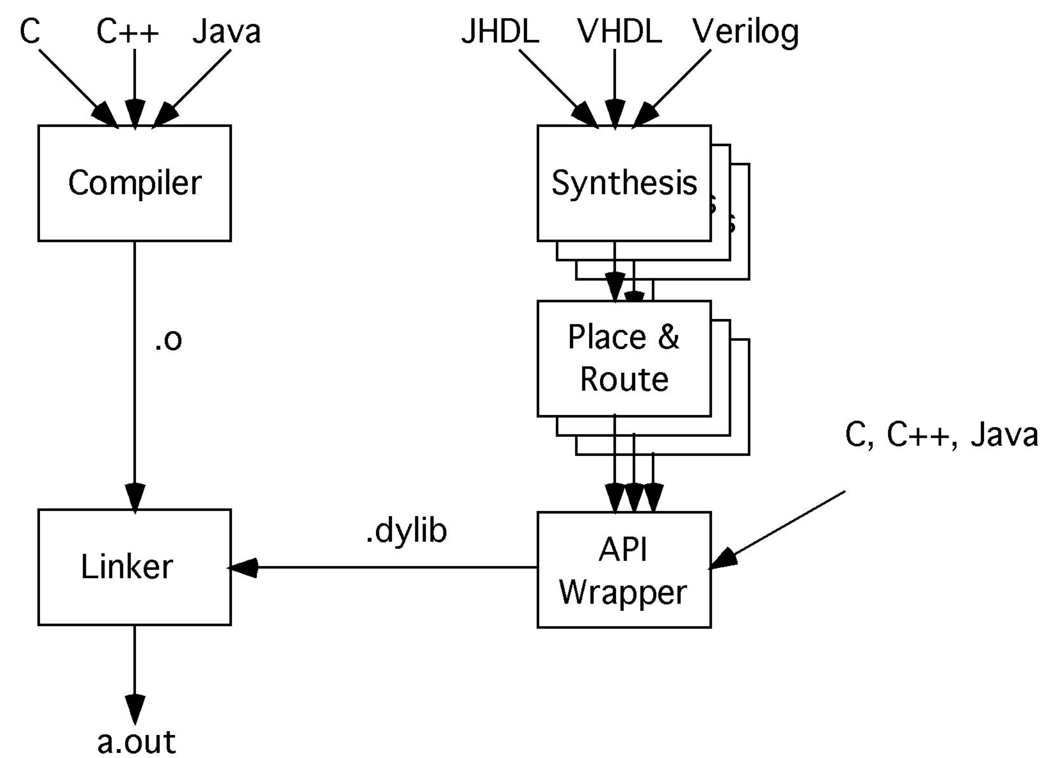
Reconfigurable HYbrid Multicore Architecture

- Hardware libraries
  - How do we create them and integrate them with software?
- Define tasks to be performed in hardware
  - Multiple implementations of a task
    - Software
    - Hardware implementations
      - Area, Performance, Power
      - Selected by OS at run-time

University of Connecticut

Hardware libraries

- Wrap multiple implementations into a single library



Task execution

- Use dynamic linker to choose between implementations
- At application startup, OS starts loading any necessary hardware components
- At initial call of function, use software implementation while hardware is being loaded
- May have to swap out implementations at run-time
  - Context switches
  - Hardware sharing

Task execution

- Hardware task execution is asynchronous
- Library call can wait for hardware to complete or give callback function
- If hardware task is switched out, any necessary state must be unloaded and loaded back in

RHYMA Computing

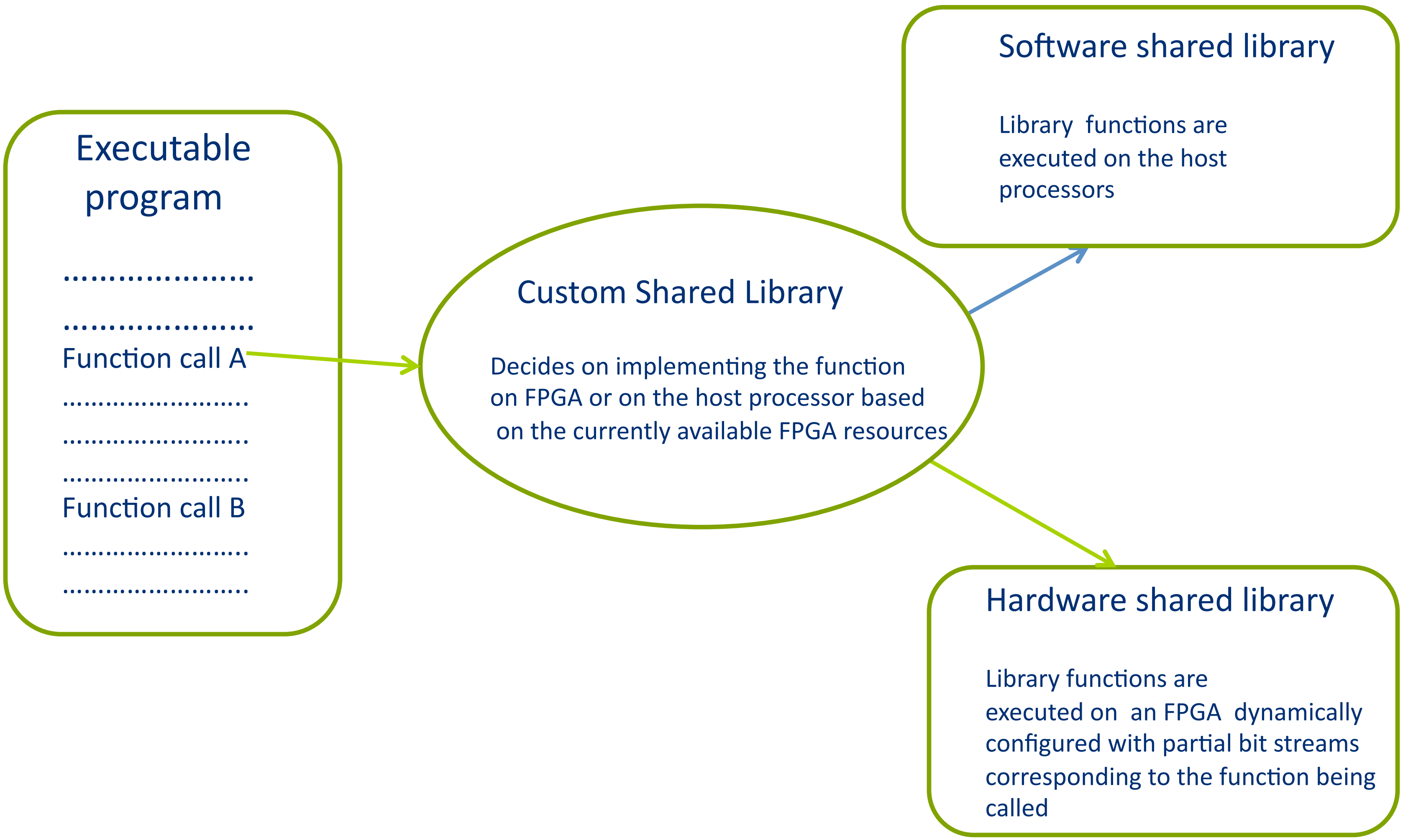
```
class encrypt_rc : rc_wrapper {
  void execute() { … }
  void wait() { … }
  void unload_state() { … }
  void load_state() { … }
  virtual void repartition() { … }
}

encrypt_rc::encrypt_rc()
{
 register_implementation(slow_encrypt_entity);
 register_implementation(fast_encrypt_entity);
 register_implementation(sw_encrypt);
}
```


University of Connecticut

Hardware sharing

- Reconfigurable resources are limited
- Multiple threads may require different hardware resources
  - How do you allocate hardware?
    - One thread gets all the hardware
    - Threads share the hardware at possible performance cost
  - If you can model the hardware area/ performance relationship, you can construct a resource allocation problem

RHYMA Computing



RHYMA Status

- Xilinx XUPV2P board
  - VirtexII Pro with 2 PowerPC cores
    - Supports partial reconfiguration
  - Linux OS
  - Dynamic library wrapper functionality

Summary

- Finding parallelism to use 1000 cores is going to be difficult
- Let's use that chip area for reconfigurable hardware cores
- Hardware can deliver parallelism and scale for a variety of applications