

CS184c: Computer Architecture [Parallel and Multithreaded]

Day 12: May 15, 2001
Interfacing Heterogeneous
Computational Blocks



CALTECH cs184c Spring2001 -- DeHon

Previously

- Homogenous model of computational array
 - single word granularity, depth, interconnect
 - all post-fabrication programmable
- Understand tradeoffs of each

CALTECH cs184c Spring2001 -- DeHon

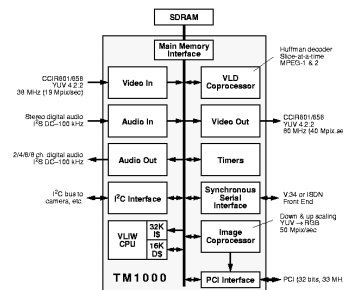
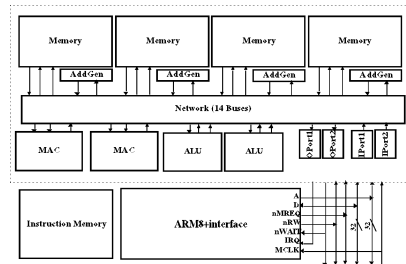
Today

- Heterogeneous architectures
 - Why?
- Focus in on Processor + Array hybrids
 - Motivation
 - Compute Models
 - Architecture
 - Examples

CALTECH cs184c Spring2001 -- DeHon

Why?

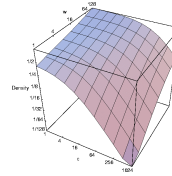
- Why would we be interested in heterogeneous architecture?
 - *E.g.*



CALTECH cs184c Spring2001 -- DeHon

Why?

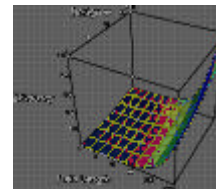
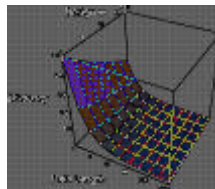
- Applications have a mix of characteristics
- Already accepted
 - seldom can afford to build most general (unstructured) array
 - bit-level, deep context, $p=1$
 - => are picking some structure to exploit
- May be beneficial to have portions of computations optimized for different structure conditions.



CALTECH cs184c Spring2001 -- DeHon

Examples

- Processor+FPGA
- Processors or FPGA add
 - multiplier or MAC unit
 - FPU
 - Motion Estimation coprocessor



CALTECH cs184c Spring2001 -- DeHon

Optimization Prospect

- Less capacity for composite than either pure
 - $(A_1+A_2)T_{12} < A_1T_1$
 - $(A_1+A_2)T_{12} < A_2T_2$

CALTECH cs184c Spring2001 -- DeHon

Optimization Prospect Example

- Floating Point
 - Task: I integer Ops + F FP-ADDs
 - $A_{\text{proc}}=125M\lambda^2$
 - $A_{\text{FPU}}=40M\lambda^2$
 - I cycles / FP Ops = 60
 - $125(I+60F) \square 165(I+F)$
 - $(7500-165)/40 = I/F$
 - $183 \approx I/F$

CALTECH cs184c Spring2001 -- DeHon

Motivational: Other Viewpoints

- Replace interface glue logic
- IO pre/post processing
- Handle real-time responsiveness
- Provide powerful, application-specific operations
 - possible because of previous observation

CALTECH cs184c Spring2001 -- DeHon

Wide Interest

- PRISM (Brown)
- PRISC (Harvard)
- DPGA-coupled uP (MIT)
- GARP, Pleiades, ... (UCB)
- OneChip (Toronto)
- REMARC (Stanford)
- NAPA ~~(NSC)~~
- E5 etc. (Triscend)
- Chameleon
- Quicksilver
- Excalibur (Altera)
- Virtex+PowerPC (Xilinx)

CALTECH cs184c Spring2001 -- DeHon

Pragmatics

- Tight coupling important
 - numerous (anecdotal) results
 - we got 10x speedup...but were bus limited
 - would have gotten 100x if removed bus bottleneck
- Speed Up = $T_{seq} / (T_{accel} + T_{data})$
 - e.g. $T_{accel} = 0.01 T_{seq}$
 - $T_{data} = 0.10 T_{seq}$

CALTECH cs184c Spring2001 -- DeHon

Key Questions

- How do we co-architect these devices?
- What is the compute model for the hybrid device?

CALTECH cs184c Spring2001 -- DeHon

Compute Models

- Unaffected by array logic (interfacing)
- Dedicated IO Processor
- Instruction Augmentation
 - Special Instructions / Coprocessor Ops
 - VLIW/microcoded extension to processor
 - Configurable Vector unit
- Autonomous co/stream processor

CALTECH cs184c Spring2001 -- DeHon

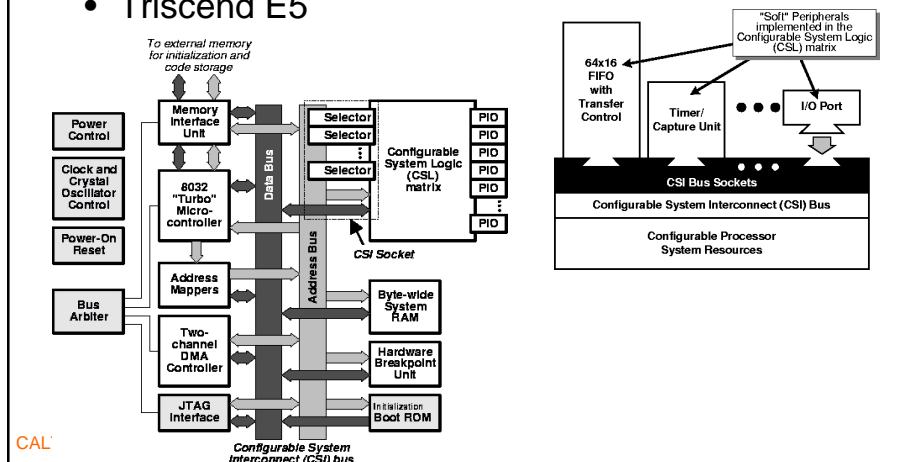
Model: Interfacing

- Logic used in place of
 - ASIC environment customization
 - external FPGA/PLD devices
- Example
 - bus protocols
 - peripherals
 - sensors, actuators
- Case for:
 - Always have some system adaptation to do
 - Modern chips have capacity to hold processor + glue logic
 - reduce part count
 - Glue logic vary
 - value added must now be accommodated on chip (formerly board level)

CALTECH cs184c Spring2001 -- DeHon

Example: Interface/Peripherals

- Triscend E5



Model: IO Processor

- Array dedicated to servicing IO channel
 - sensor, lan, wan, peripheral
- Provides
 - protocol handling
 - stream computation
 - compression, encrypt
- Looks like IO peripheral to processor
- Maybe processor can map in
 - as needed
 - physical space permitting
- Case for:
 - many protocols, services
 - only need few at a time
 - dedicate attention, offload processor

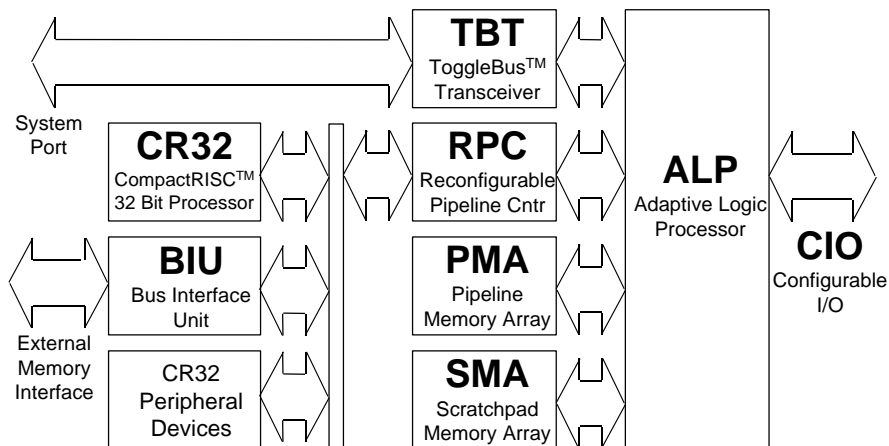
CALTECH/UCSD/UCR -- DeHon

IO Processing

- Single threaded processor
 - cannot continuously monitor multiple data pipes (src, sink)
 - need some minimal, local control to handle events
 - for performance or real-time guarantees , may need to service event rapidly
 - *E.g.* checksum (decode) and acknowledge packet

CALTECH cs184c Spring2001 -- DeHon

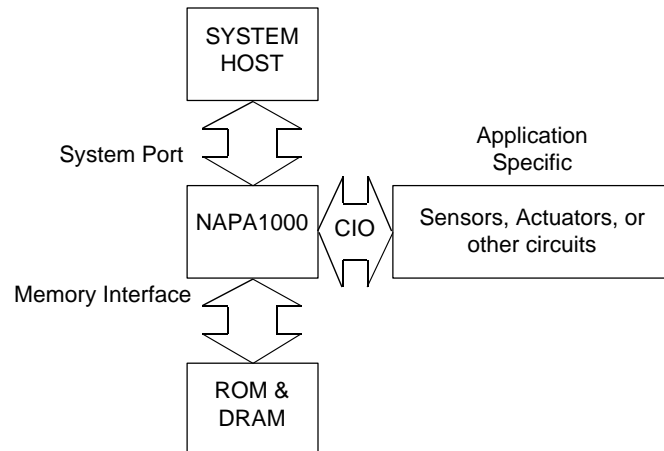
NAPA 1000 Block Diagram



Source: [National Semiconductor](#)

CALTECH cs184c Spring2001 -- DeHon

NAPA 1000 as IO Processor



Source: *National Semiconductor*

CALTECH cs184c Spring2001 -- DeHon

Model: Instruction Augmentation

- Observation: Instruction Bandwidth
 - Processor can only describe a small number of basic computations in a cycle
 - l bits $\rightarrow 2^l$ operations
 - This is a small fraction of the operations one could do even in terms of $w \otimes w \rightarrow w$ Ops
 - $w 2^{2^{(2^w)}}$ operations

CALTECH cs184c Spring2001 -- DeHon

Model: Instruction Augmentation (cont.)

- Observation: Instruction Bandwidth
 - Processor could have to issue $w2^{(2^w - 1)}$ operations just to describe some computations
 - An *a priori* selected base set of functions could be very bad for some applications

CALTECH cs184c Spring2001 -- DeHon

Instruction Augmentation

- Idea:
 - provide a way to augment the processor's instruction set
 - with operations needed by a particular application
 - close semantic gap / avoid mismatch

CALTECH cs184c Spring2001 -- DeHon

Instruction Augmentation

- What's required:
 - some way to fit augmented instructions into stream
 - execution engine for augmented instructions
 - if programmable, has own instructions
 - interconnect to augmented instructions

CALTECH cs184c Spring2001 -- DeHon

“First” Instruction Augmentation

- PRISM
 - Processor Reconfiguration through Instruction Set Metamorphosis
- PRISM-I
 - 68010 (10MHz) + XC3090
 - can reconfigure FPGA in one second!
 - 50-75 clocks for operations

[Athanas+Silverman: Brown]

CALTECH cs184c Spring2001 -- DeHon

PRISM-1 Results

| Function Name | Description (input bytes / output bytes) | Compilation Time (mins) | % Utilization of a XC9090 FPGA | Speed-up Factor |
|----------------|---------------------------------------------|-------------------------|--------------------------------|-----------------|
| Hamming(x,y) | Calculates the hamming metric. (4/2) | 6 | 38% | 24 |
| Bitrev(x) | Bit-reversal function. (4/4) | 2 | 0% | 26 |
| Neuron(x,y) | Cascadable 4-input N-Net function. (4/4) | 12 | 52% | 12 |
| MultiAccm(x,y) | Multiply/accumulate function. (4/4) | 11 | 58% | 2.9 |
| LogicEv(x) | Logic simulation engine function. (4/4) | 12 | 40% | 18 |
| ECC(x,y) | Error correction coder/decoder. (3/2) | 6 | 14% | 24 |
| FindFirst1(x) | Find first '1' in input. (4/1) | 3 | 11% | 42 |
| Piecewise(x) | 5-section piecewise linear seg. (4/4) | 24 | 77% | 5.1 |
| ALog2(x) | Computes base-2 A*log(x). (4/4) | 16 | 74% | 54 |

Raw kernel speedups

CALTECH cs184c Spring2001 -- DeHon

PRISM

- FPGA on bus
- access as memory mapped peripheral
- explicit context management
- some software discipline for use
- ...not much of an “architecture” presented to user

CALTECH cs184c Spring2001 -- DeHon

PRISC

- Takes next step
 - what look like if we put it on chip?
 - how integrate into processor ISA?

[Razdan+Smith: Harvard]

CALTECH cs184c Spring2001 -- DeHon

PRISC

- Architecture:
 - couple into register file as “superscalar” functional unit
 - flow-through array (no state)

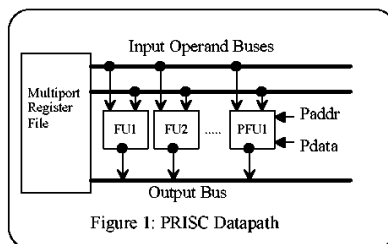
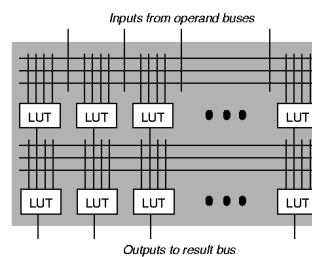


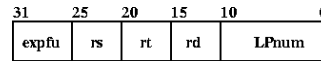
Figure 1: PRISC Datapath



CALTECH cs184c Spring2001 -- DeHon

PRISC

- ISA Integration



- add expfu instruction
- 11 bit address space for user defined expfu instructions
- fault on pfu instruction mismatch
 - trap code to service instruction miss
- all operations occur in clock cycle
- easily works with processor context switch
 - no state + fault on mismatch pfu instr

CALTECH cs184c Spring2001 -- DeHon

PRISC Results

- All compiled
- working from MIPS binary
- <200 4LUTs ?
 - 64x3
- 200MHz MIPS base

| Optimization | CPS | EQN | EXP | GCC | L1 | SC |
|------------------|-----|-----|-----|-----|----|----|
| PFU-expression | 9 | 0 | 48 | 13 | 4 | 12 |
| PFU-table-lookup | 0 | 0 | 0 | 0 | 0 | 0 |
| PFU-predication | 0 | 1 | 0 | 13 | 0 | 0 |
| PFU-jump | 10 | 0 | 47 | 103 | 0 | 35 |
| PFU-loop | 0 | 3 | 0 | 4 | 0 | 0 |
| TOTAL | 19 | 4 | 95 | 133 | 4 | 47 |

Table 1: Static PFU optimization instances in SPECint92.

| | CPS | EQN | EXP | GCC | L1 | SC |
|---------|------|------|------|------|------|------|
| Speedup | 1.15 | 1.91 | 1.16 | 1.10 | 1.06 | 1.12 |

Table 2: Cycle count speedup for a PRISC-1 microarchitecture with a single PFU resource. The speedup for each application is an arithmetic average (as defined by SPEC) of all of the data sets for that application.

Razdan/Micro27

CALTECH cs184c Spring2001 -- DeHon

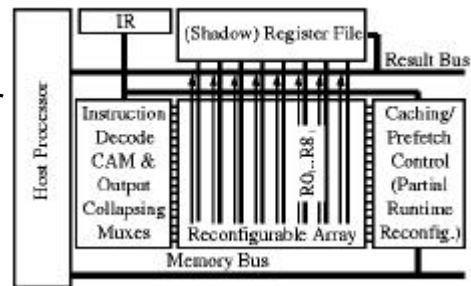
Chimaera

- Start from PRISC idea
 - integrate as functional unit
 - no state
 - RFUOPs (like expfu)
 - stall processor on instruction miss, reload
- Add
 - manage multiple instructions loaded
 - more than 2 inputs possible

[Hauck: Northwestern]
CALTECH cs184c Spring2001 -- DeHon

Chimaera Architecture

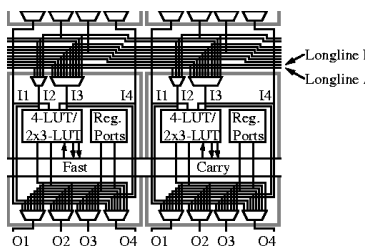
- “Live” copy of register file values feed into array
- Each row of array may compute from register values or intermediates (other rows)
- Tag on array to indicate RFUOP



CALTECH cs184c Spring2001 -- DeHon

Chimaera Architecture

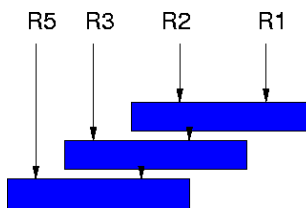
- Array can compute on values as soon as placed in register file
- Logic is combinational
- When RFUOP matches
 - stall until result ready
 - critical path
 - only from late inputs
 - drive result from matching row



CALTECH cs184c Spring2001 -- DeHon

Chimaera Timing

- If presented
 - R1, R2
 - R3
 - R5
 - can complete in one cycle
- If R1 presented last
 - will take more than one cycle for operation



CALTECH cs184c Spring2001 -- DeHon

Chimaera Results

Speedup

- Compress 1.11
- Eqntott 1.8
- Life 2.06 (160 hand parallelization)

[Hauck/FCCM97]

CALTECH cs184c Spring2001 -- DeHon

Instruction Augmentation

- Small arrays with limited state
 - so far, for automatic compilation
 - reported speedups have been small
 - open
 - discover less-local recodings which extract greater benefit

CALTECH cs184c Spring2001 -- DeHon

Big Ideas

- Exploit structure
 - area benefit to
 - tasks are heterogeneous
 - mixed device to exploit
- Instruction description
 - potential bottleneck
 - custom “instructions” to exploit

CALTECH cs184c Spring2001 -- DeHon

Big Ideas

- Model
 - for heterogeneous composition
 - limits of sequential control flow

CALTECH cs184c Spring2001 -- DeHon