Alpha Architecture
and
EV4 - The First Implementation

Rich Witek and Dick Sites

Alpha Architecture
and
EV4 - The First Implementation

Presentation Outline

- Alpha Architecture
  - Goals
  - Overview
  - Differences from other RISC architectures
  - Problems addressed -- solutions used and rejected
- First Implementation
  - Overview
  - Functional units, pipelines, dual issue
  - Interesting features in EV4
  - Pin bus

Alpha Architecture Goals

Performance

Provide Digital with leadership performance systems for both OSF/1 and VMS in a variety of implementation media and styles

Support multiple issue

Longevity

Provide architecture that will last 25 years by:

Supporting 64 bit address space

Supporting multiple issue

Supporting multiprocessor systems

Alpha Goals cont.

Scalability

- Of processor implementations (cycle time, width of issue, depth of pipelines)
- Of system via multiprocessor support (both small and large scale)

Generality

- Support VMS without penalty to OSF/1
- Support wide variety of languages without penalty
- Support binary migration of software
  - (VAX and MIPS on day one)

Architecture Overview

General

- 64 bit Load/Store machine
- 32 bit fixed length instructions

Data Types

- 8b, 16b, 32b, and 64b integers
- VAX 32b and 64b floating point formats
- IEEE 32b and 64b floating point formats

Support for MP

- Hardware synchronization primitive (ldx_l/stx_c)
- Read/Write ordering & memory barrier
- Processor number (whami) and per processor 64b value/ptr (sysval)

Architecture Overview cont.

32 - 64 bit integer registers (r0..r31)

- r30 used as stack pointer (sp)
- r31 hardwired as zero

32 - 64 bit floating point registers (fr0..fr31)

- fr31 hardwired as zero

64 bit program counter (pc)

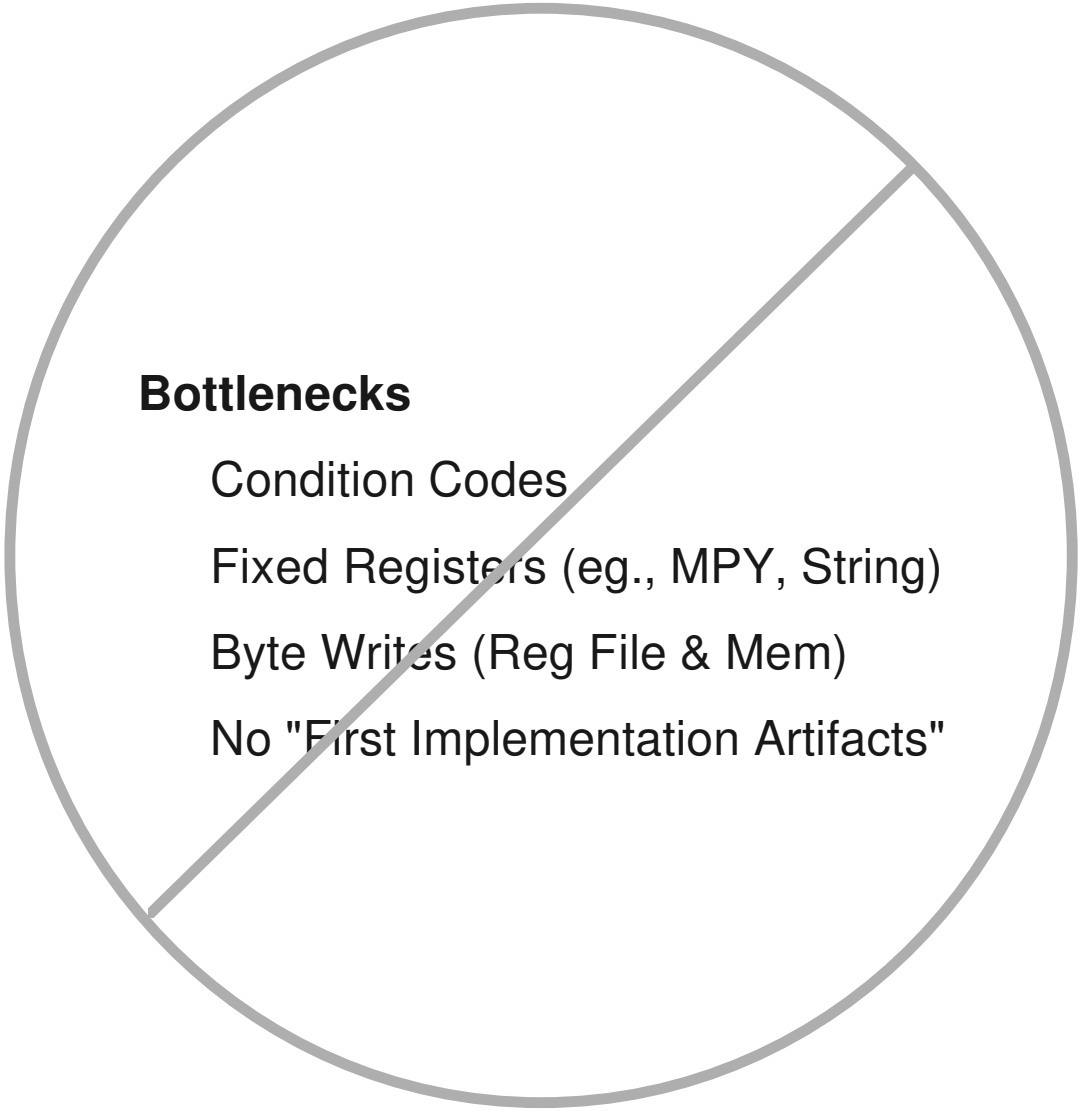
32 bit cycle counter (pcc)

FPCSR - 7 status bits, 2 rounding control bits

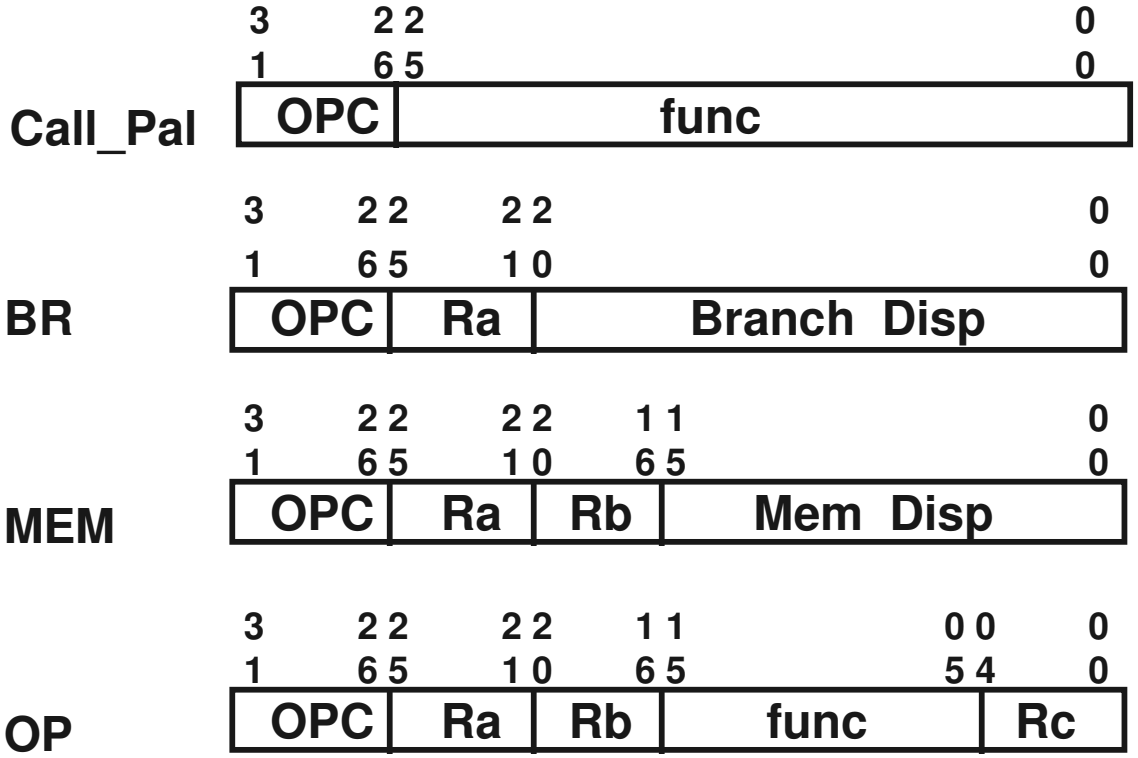
64 bit virtual addresses (64 bits checked; subsets implemented)

64 bit thread unique register

Architecture Overview cont.



Instruction Formats



Now what is different!

"Clean" Features vs "First Implementation Artifacts"

- Barrier vs Strict read/write ordering
- Branch Predict, Prefetch, and Conditional Move vs Delayed Branch
- Multiple Issue vs special operations

Hazard-free Architecture

Minimal global state

No mode bits

Designed for Concurrency, Pipelines, Memory Latency

- Imprecise Exceptions
- No Side-Effects
- TRAPB, FETCH, MB

Architected hardware/software compact

What is different cont.
PAL

Provides OS and applications with insulation from hardware details if wanted

Privileged Mode

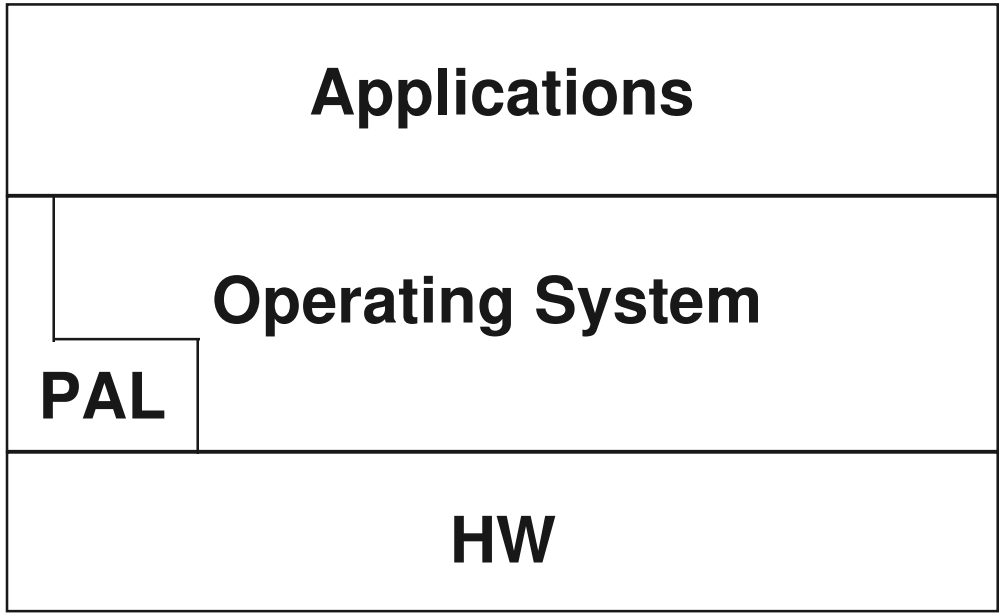
- Interrupts disabled

Special Instructions

- Access to complete machine state, Private GPRs
- Physical addressing
- Privileged jump

Strict Coding Rules

- Software timing



Problem - Branches

Solution

- **Architect branch prediction hints and rules for conditional branches**
  - **Sign of displacement is initial prediction**
- **Architect subroutine return prediction rules so hardware can correctly predict subroutine returns**
- **Conditional moves to remove small branch blocks**

```
; r2 <- min(r1, r2)
cmplt     r1, r2, r0    ; see if r1 < r2
cmovne  r0,  r1, r2     ; if so move r1 to r2
```

Rejected

- **Branch delay slots**
  - **Cannot pick right size for different issue widths and pipelines**

Problem - Arithmetic Exceptions

Solution

- **Imprecise exceptions**
- **Out of order completion**
- **Provide TRAPB instruction to make exceptions precise when needed**
- **Provide combination of hardware and software for recovery**
- **Pay for precise exceptions when wanted, rather than all the time**
- **Provide underflow to zero in place of denormals**

Rejected

- **Hardware rollback when exception happens**
- **Checking of input arguments and restricting issue until you know there is no exception**

Problem - Achieving multiple issue

Solution

- **No inter-instruction dependencies**
- **No hazards**
- **No special registers**
- **No precise arithmetic exceptions**
- **Minimize global state - IEEE controls in instructions**

Rejected

- **Fixed width of issue (VLIW)**
- **Branch delay or load delay slots**
- **Mode bits**

Problem - Increasing Memory Latency Relative to CPU cycle time

Solution

- **Prefetch instructions (bring block of memory into cache before needed)**
- **No implied READ/WRITE ordering in MP systems, provide barrier (mb) when needed (allow hits under misses)**

Rejected

- **Implied read/write ordering**
- **User-visible cache effects**

Problem - Software

Solution

- **Support data types and OS interfaces for both VMS and OSF/1**
- **Support binary migration of applications via software translator as safety net for source migration. Allows staging of source migration**
- **Source language compatibility by reusing VMS and OSF/1 compiler frontends**
- **Provide 32 bit "mode" via complier and OS options**

Rejected

- **Hardware "compatibility mode" for VAX or MIPS**
- **Pure software simulators for VAX or MIPS**
- **Hardware modes for 32 bits**
- **Mode for IEEE vs VAX floating point formats**

EV4

The First Alpha Implementation

Chip Features

Cycle time	5.0/6.6 nS bins
Process technology	CMOS-4 (0.75μm), 0.5μm Leff, 3 metal layers
Die size	13.9mm X 16.8mm
Transistor count	1.68 million
Package	431 pin PGA (291 signals)
Power dissipation	30/23W (3.3V Vdd)
Virtual address size	2^43 byte addresses (linear)
Phys address size	2^34 byte addresses
Issue rate	Two instructions per CPU cycle
Functional units	Integer Op, FP Op, Ld/St, Branch
Translation buffers	32 data, 12 instruction (fully associative)
Write buffer	4 entries X 32 bytes/entry
Primary caches	8kB Icache, 8kB Dcache (direct mapped)
Bus interface	128/64b data + 34b address

Chip Features

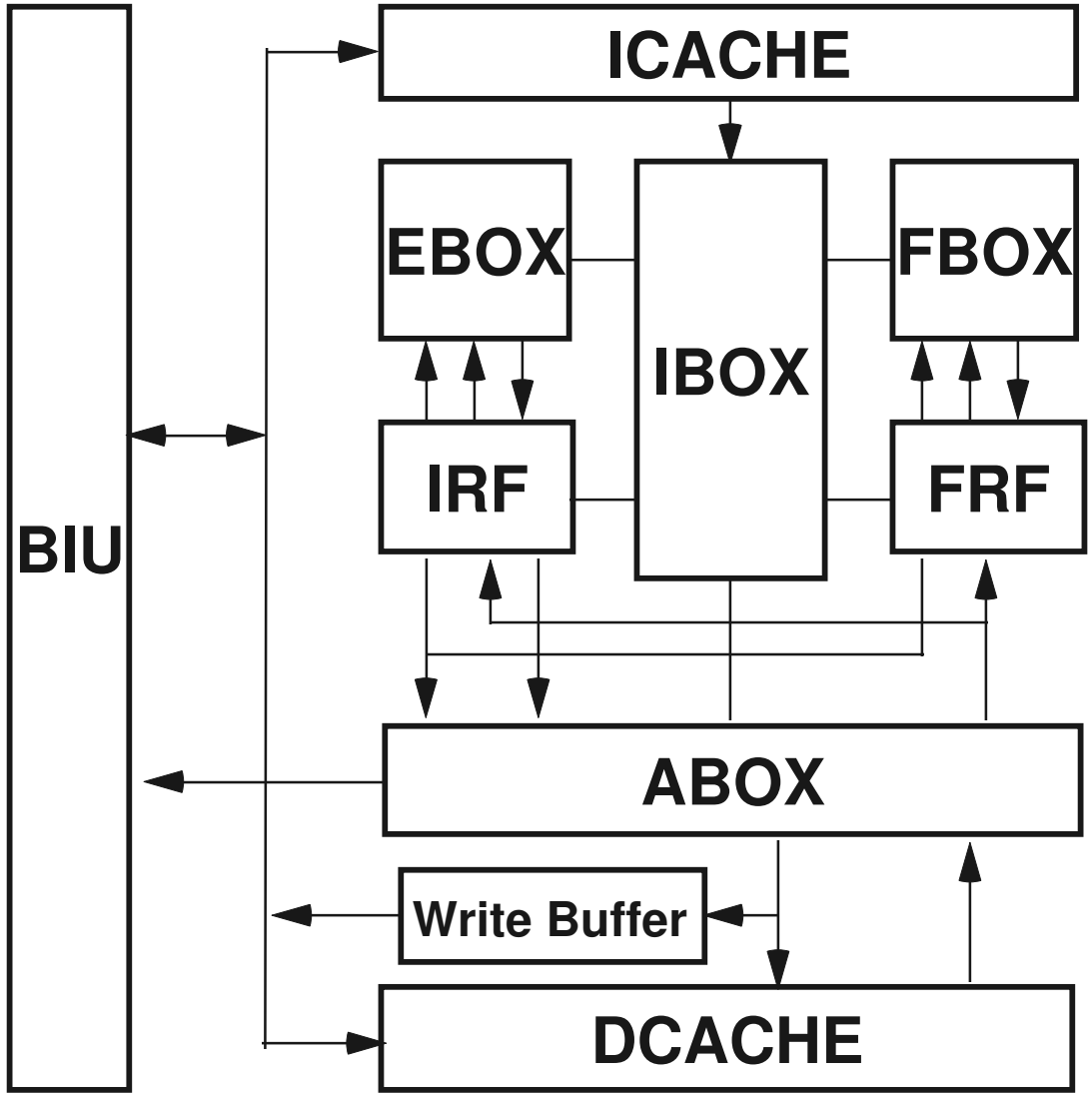
Cycle time	**5.0/6.6 nS Bins**
Process technology	**CMOS-4 (0.75μm), 0.5μm Leff, 3 metal layers**
Die Size	**13.9mm X 16.8mm**
Transistor count	**1.68 million**
Package	**431 pin PGA (291 signals)**
Power dissipation	**30/23W (3.3V Vdd)**

Chip Features cont.

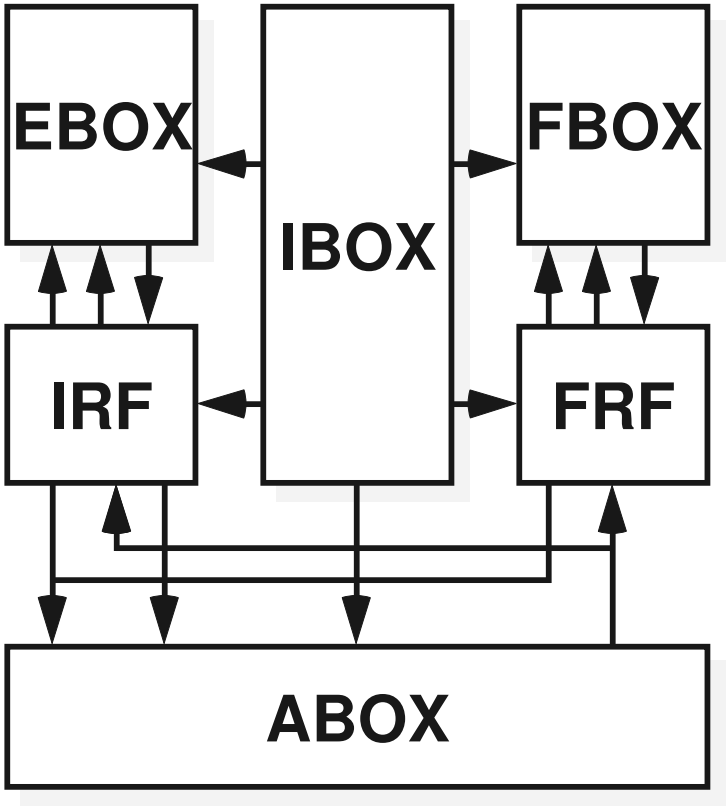
Virtual address size	2^43 byte addresses (linear)
Phys address size	2^34 byte addresses
Issue rate	Two instructions per CPU cycle
Functional units	Integer Op, FP Op, Ld/St, Branch
Translation buffers	32 data, 12 instruction (fully associative)
Write buffer	4 entries X 32 bytes/entry
Primary caches	8kB Icache, 8kB Dcache (direct mapped)
Bus interface	128/64b data + 34b address

Chip Microphotograph

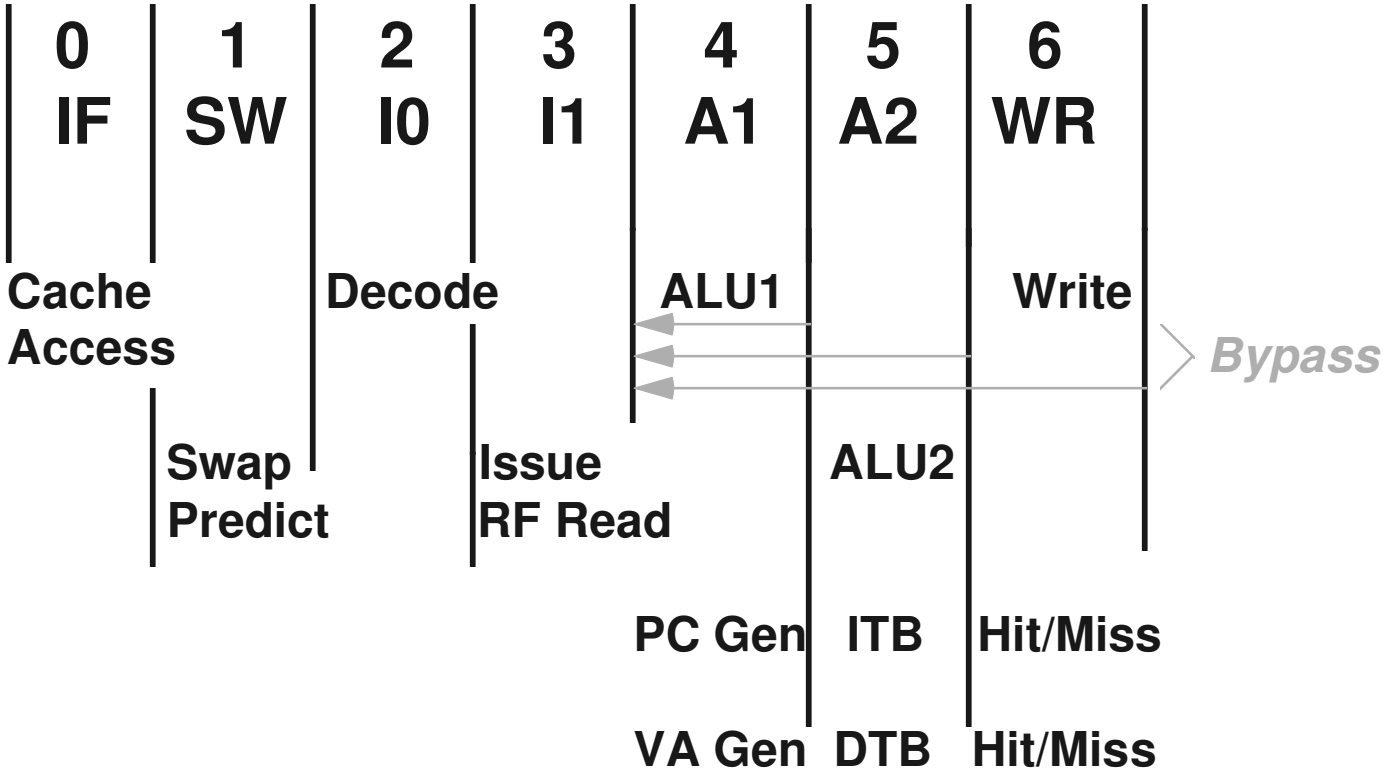
Chip Block Diagram



EV4 Core



Integer Pipeline



5ns cycles

Abox Load Pipeline

LOADS eg. LDQ R1, 100 (R2)

[4]: displacement add: VA <-- 100 + (R2)

[5]: read DCache, TLB

[6]: DCache hit? --> write register file

miss? --> send request to BIU

5ns cycles

Abox Store Pipeline

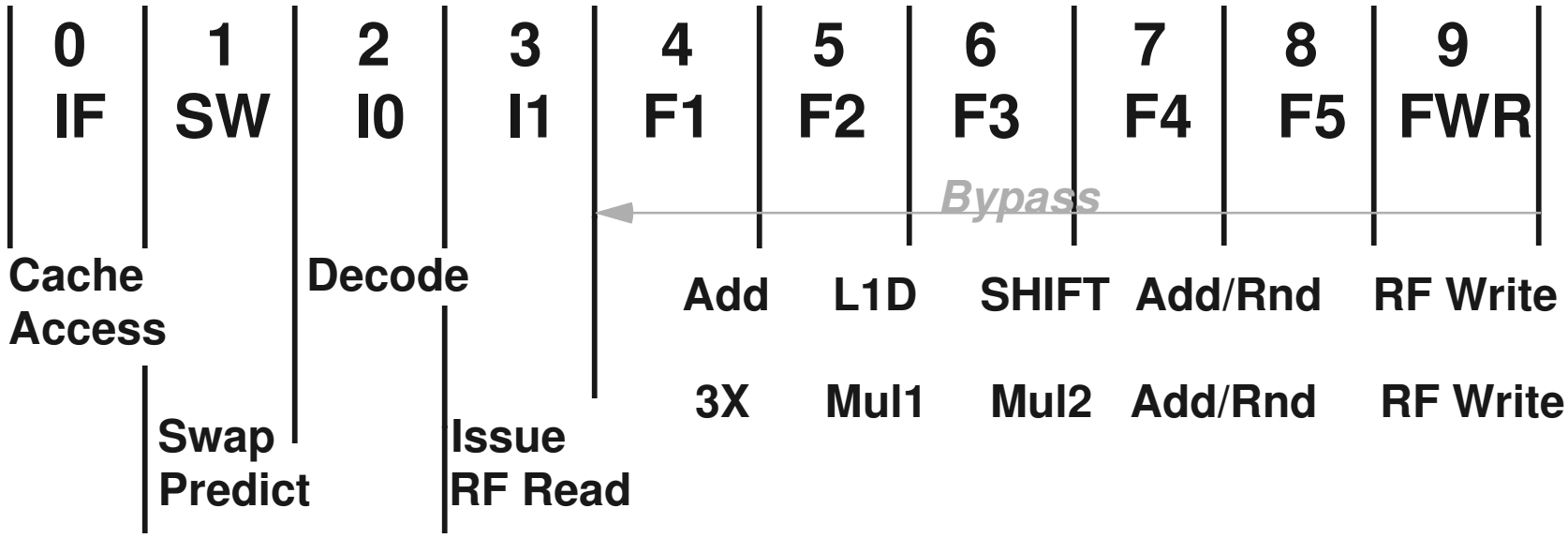
STORES eg. STQ R1, 100 (R2)

[6]: DCache hit? --> write data into cache next time it's free

ALWAYS: Put data into write buffer. It'll get off-chip when the BIU has nothing better to do.

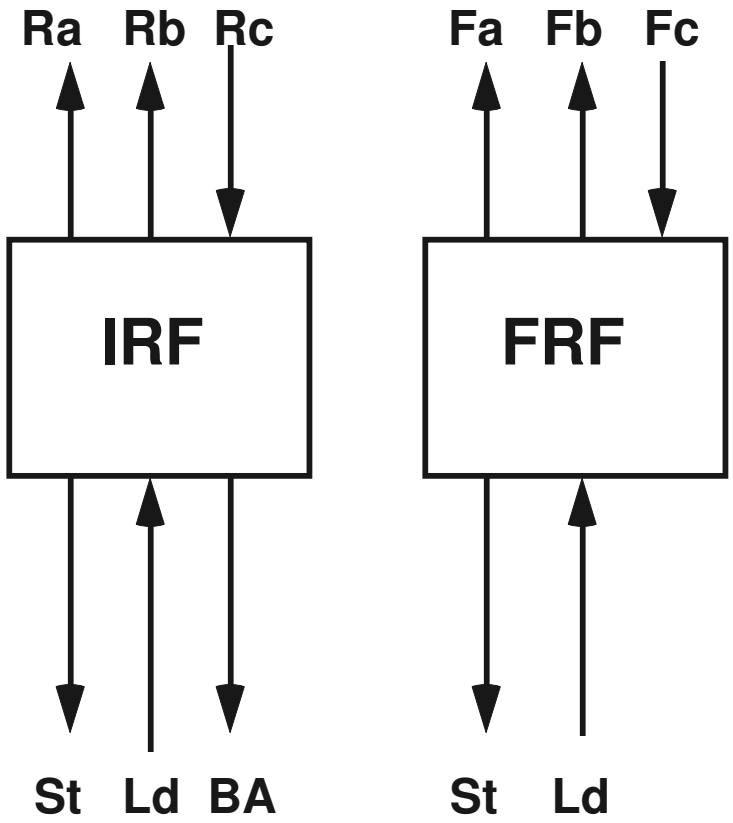
5ns cycles

Floating Point Pipe



5ns cycles

Register File



EV4 Dual Issue

LDx/STx <---> xOP

IOP <---> FOP

FOP <---> FBR

IOP <---> IBR

NO STI <---> FOP
STF <---> IOP

Interesting Features In EV4

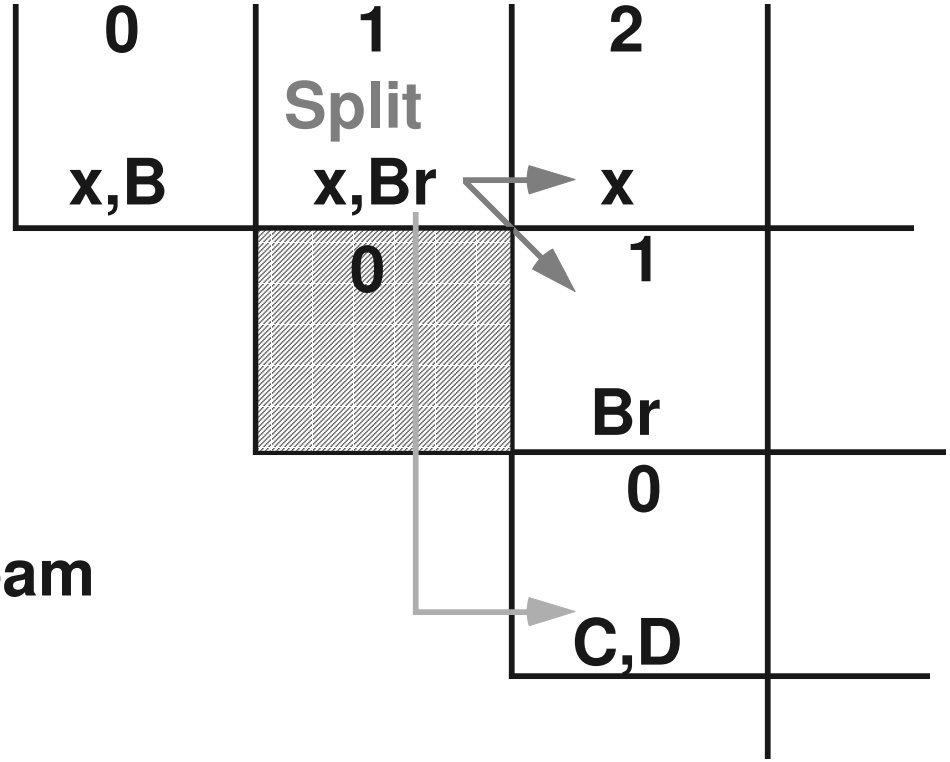
Bubble squashing

Predicting subroutine returns

TB granularity hints

Performance monitoring hardware

Bubble Squash



I-Stream
- x
- Br
- C
- D

JSR Stack

Accurate through pipe flush

Depth of 4

Utilized by CBR, HW_REI

Requires use of hint bits

Granularity Hints in DTB

DTB

- **Each DTB entry can map 8KB, 64KB, 512KB, or 4MB pages**
- **Size of page mapped by entry is set at TB load time**
- **Allows small on-chip TBs to map large structures (frame buffers, large arrays, memory pools) with a small number of TB entries**

ITB

- **Contains four 4MB page entries and eight 8KB entries. Entry type is selected at TB fill time**
- **The 4MB entries can be used by the OS to map the OS or large code pools (shared libraries)**

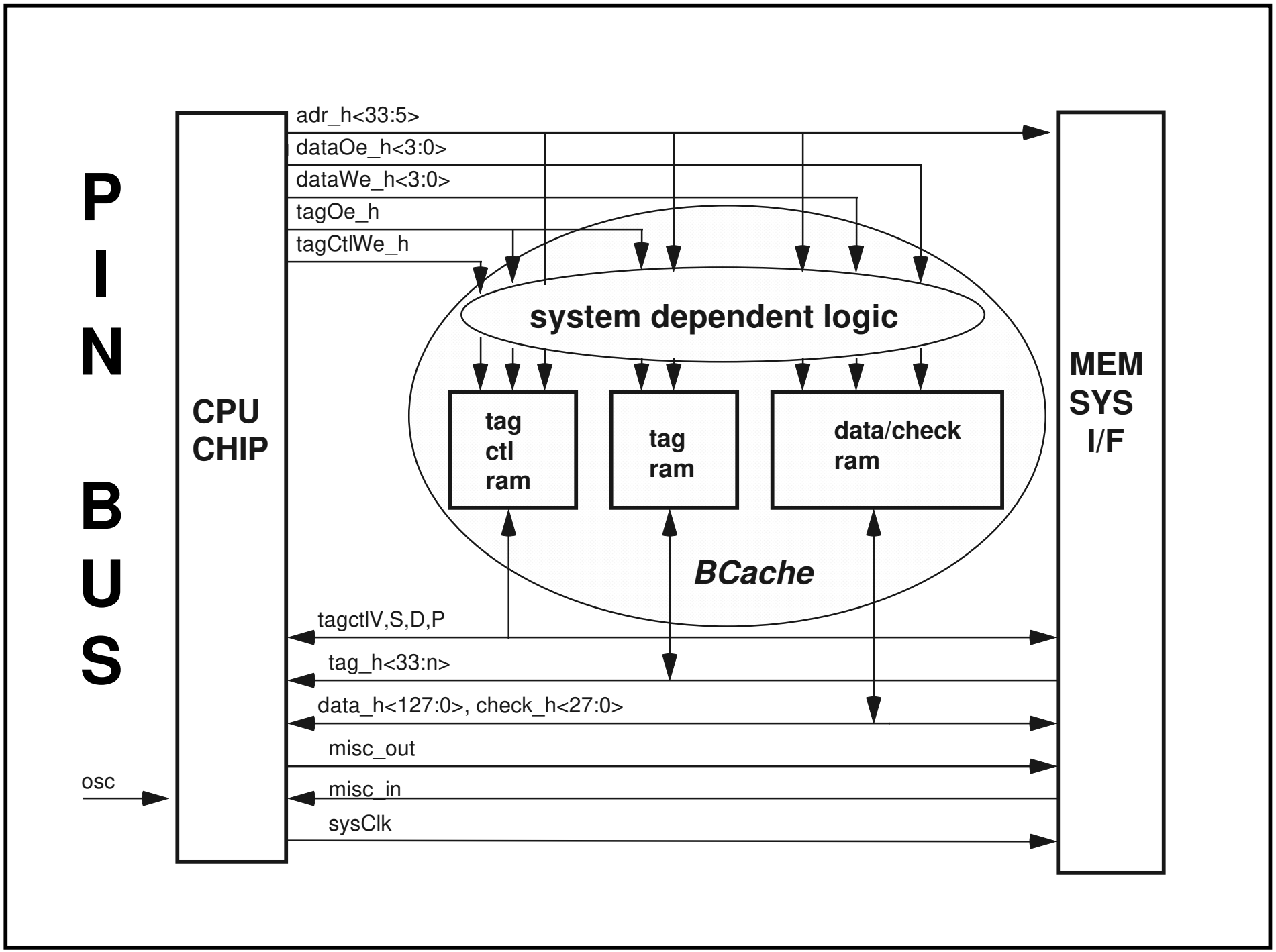
Performance Monitoring Hardware

- Provides access to critical statistics: cycle count, cache miss rates, different stall reasons, etc
- Implemented as combination of hardware and PALcode
- Access and control provided by privileged PAL calls
- Cycle counter provides both small grain real time and process virtual time

EV4 External Interface

Goals

- **Support range of systems from desktops to large MP's**
- **Allow system to make cost/performance tradeoffs (size, speed and interface levels to cache RAMs)**
- **Provide usable test features**
- **Interface to industry standard components**


PIN BUS
CPU CHIP
adr_h<33:5>
dataOe_h<3:0>
dataWe_h<3:0>
tagOe_h
tagCtlWe_h
system dependent logic
tag ctl ram
tag ram
data/check ram
BCache
MEM SYS I/F
tagctlV,S,D,P
tag_h<33:n>
data_h<127:0>, check_h<27:0>
misc_out
misc_in
sysClk
osc

Pin Bus Features

Balance performance / flexibility / simplicity

- **Cache sizes from 128KB -> 8MB (or none)**
- **Cache speeds from 3 to 16 cycles**
- **External interface runs synchronously to system clock supplied by CPU (system clock is CPU clk divided by 2...8)**
- **No cache policy decision made by CPU, left to system designer**
- **128 or 64-bit data bus, chosen at reset**
- **TTL or ECL levels**
- **Longword ECC or parity**
- **Simple diagnostic interface via off-chip serial ROM**

Bcache and Memory Interface

- Simple access done by EV4 chip
  - Simple accesses are READ HIT and WRITE HIT NOT SHARED
  - Timing in increments of CPU cycles
  - Cache loop is combinatorial -- board logic not needed to run at CPU cycle time
  - Chip supports maintenance of external backmap for internal Dcache invalidate filtering and cache update policy decisions
- More complicated access handled by board logic
  - Cache policy set by system designer -- not by CPU chip
  - Speed of external logic is cost/performance trade off for system designer (e.g. logic for 200Mhz CPU with 4x divider on sysclock runs at 50Mhz)

SROM
Boot, Debug, and Diagnostics

- SROM uses low cost serial ROM to fill Icache on reset
- SROM code is used to program chip interface
- SROM code can be used to test system from CPU out for either debug or production diagnostics
- SCAN only works well when most chips on module have SCAN
  - Highest speed SRAMs do not have SCAN
  - High speed MSI logic parts do not have SCAN

Functional Results

- Virtually no bugs in pass 1 parts: Fab to booting multiple operating systems in 3 days
- PAL code and SROM used as debug, and to fix minor bug that did exist in pass 1
- Pass 1 parts ran at full speed
- Pass 1 parts are now being used for system and software debug

Conclusion

Alpha - a 64 bit architecture positioned for the future

- **Promotes multiple issue**
- **Promotes multiprocessors**
- **Promotes high speed implementation**
- **Supports wide range of operating systems and languages**

EV4 is:

- **A 200 MHz dual issue implementation of the Alpha architecture**
- **Supported by OSF and VMS**
- **First in a family, with more to come**

Conclusion

Alpha - a 64 bit architecture positioned for the next 25 years

- **Promotes multiple issue**
- **Promotes multiprocessors**
- **Promotes high speed implementation**
- **Supports wide range of operating systems and languages**

EV4 is:

- **A 200 MHz dual issue implementation of the Alpha architecture**
- **Supported by OSF and VMS**
- **First in a family, with more to come**

Title