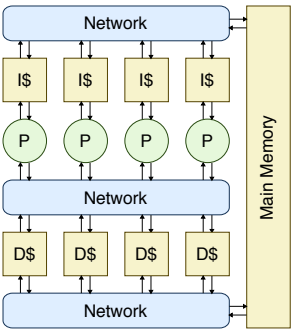
ECE 4750 Computer Architecture, Fall 2015

T05 Integrating Processors and Memories

School of Electrical and Computer Engineering
Cornell University

revision: 2015-10-14-14-11

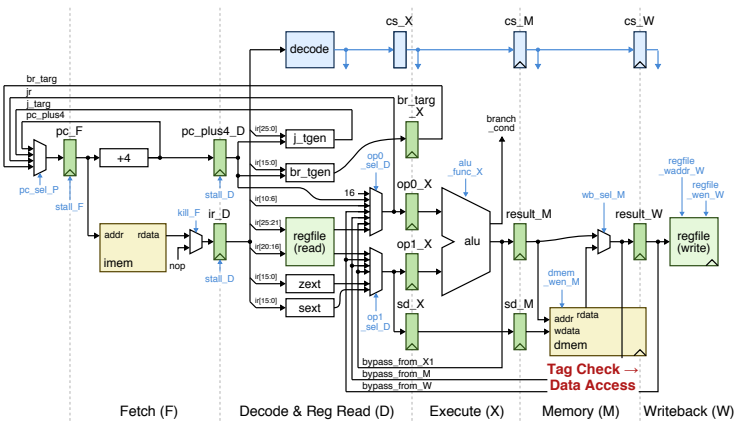


- Processors for computation
- Memories for storage
- Networks for communication

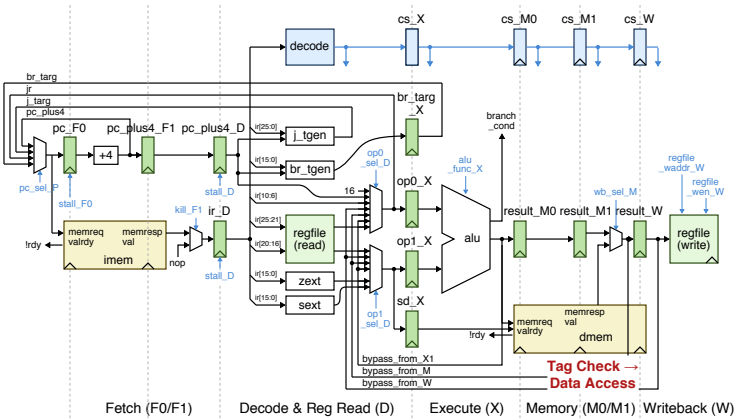
1. Processor and L1 Cache Interface

Approaches to integrate L1 caches into a processor pipeline vary based on how the L1 memory system timing is encapsulated.

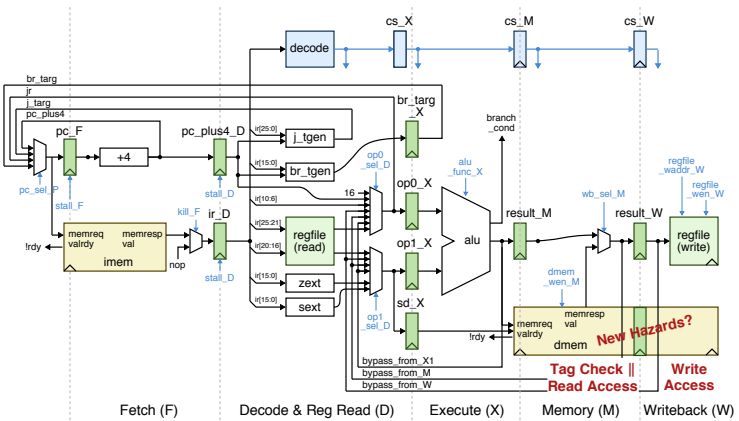
Zero-Cycle Hit Latency with Tightly Coupled Interface



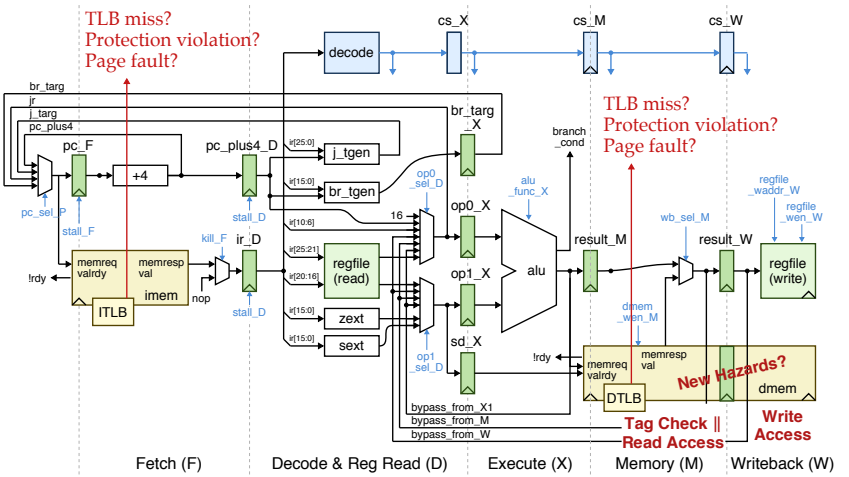
Two-Cycle Hit Latency with Val/Rdy Interface



Parallel Read, Pipelined Write Hit Path



Integrating Instruction and Data TLBs

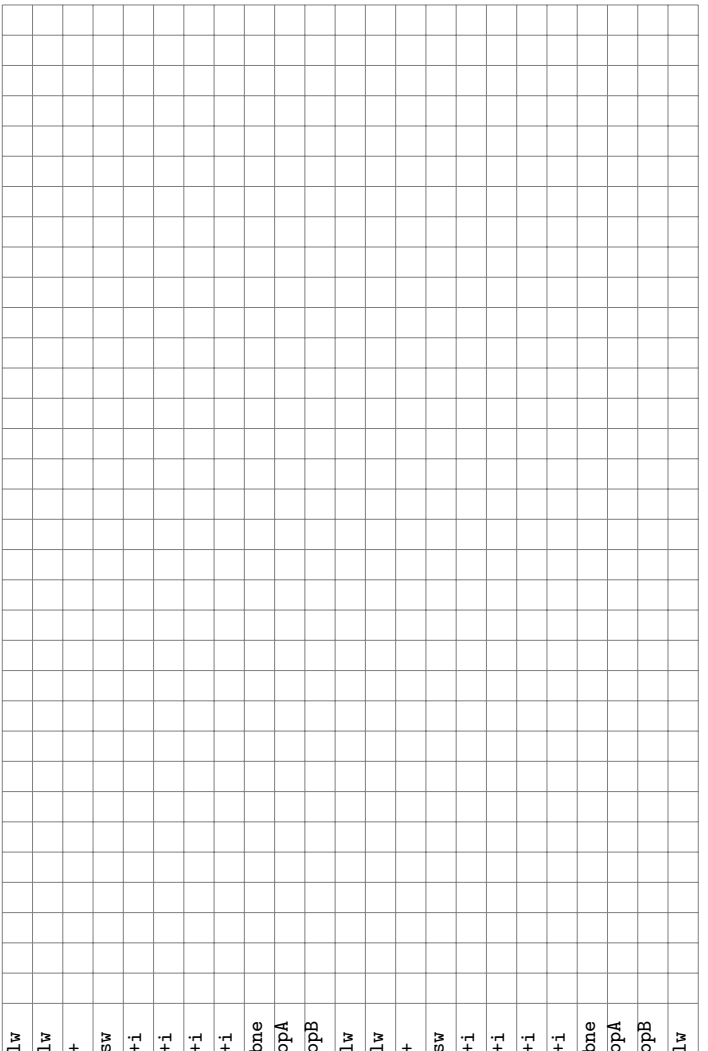


- TLB miss needs a hardware or software mechanism to refill TLB
- Software handlers need restartable exceptions on page fault
- Need mechanism to cope with the additional latency of a TLB
  - Increase the cycle time
  - Pipeline the TLB and cache access
  - Use virtually addressed caches
  - Access TLB and cache in parallel

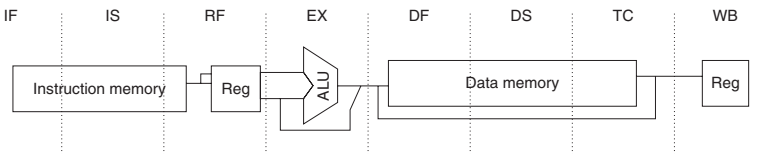
2. Analyzing Processor + Cache Performance

How long in cycles will it take to execute the vvadd example assuming `n` is 64? Assume cache is initially empty, parallel-read/pipelined-write, four-way set-associative, write-back/write-allocate, and miss penalty is two cycles.

```
loop:
 lw    r12, 0(r4)
 lw    r13, 0(r5)
 addu  r14, r12, r13
 sw    r14, 0(r6)
 addiu r4,  r4, 4
 addiu r5,  r5, 4
 addiu r6,  r6, 4
 addiu r7,  r7, -1
 bne   r7,  r0, loop
 jr    r31
```

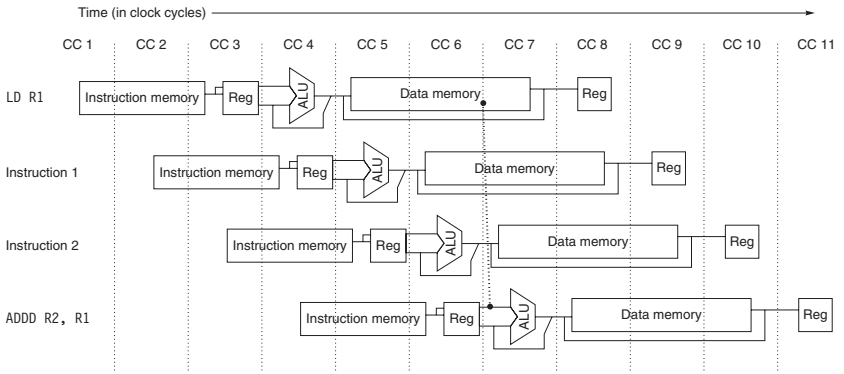

lw
lw
+
sw
+i
+i
+i
+i
bne
opA
opB
lw
lw
+
sw
+i
+i
+i
+i
bne
opA
opB
lw

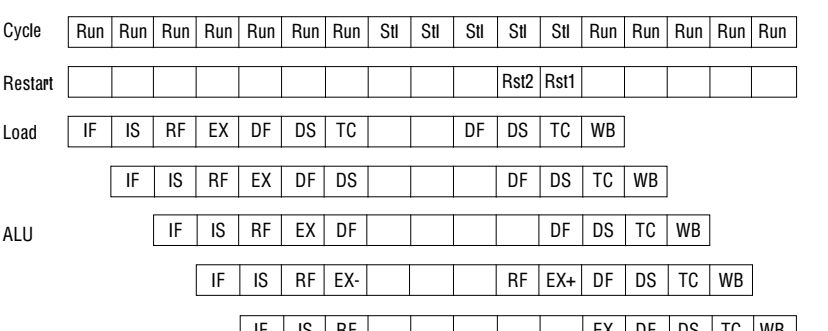
3. Case Study: MIPS R4000



- 8-stage pipeline with extra stages for instruction/data mem access
  - IF: First-half of inst fetch: PC selection and start icache access
  - IS: Second half of inst fetch, complete icache access
  - RF: Instruction decode, register read, stall logic, icache hit detection
  - EX: Execution (including effecive address calculation)
  - DF: First-half of data fetch, start dcache access
  - DS: Second half of data fetch. complete dcache access
  - TC: Tag check, dcache hit detection
  - WB: Write-back for loads and reg-reg operations
- Longer pipeline results in
  - Decreased cycle time
  - Increased load-use delay latency and branch resolution latency
  - More bypass paths

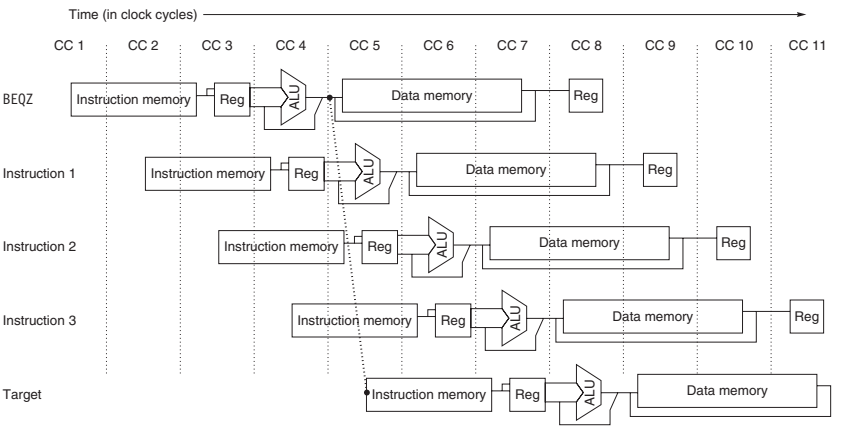
Load-Use Delay Latency





- Load-use delay latency increased by one cycle
- Data is forwarded from end of DS stage to end of RF stage
- Tag check does not happen until TC!
- On miss, instruction behind load may have bypassed incorrect data
- EX stage of dependent instruction needs to be *re-executed*

Branch Resolution Latency



- Branches are resolved in EX stage
- Instruction 1 is in the branch delay slot
- Use predicted not-taken for instruction 2 and 3