CS 451 / 551 / ECE 541

ADVANCED COMPUTER ARCHITECTURE

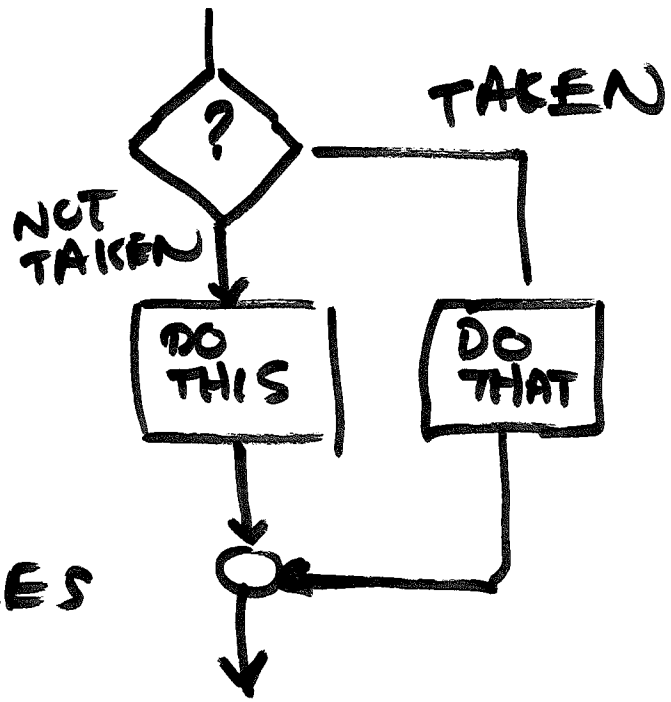
SESSION no. 9

University of Idaho

HAZARDS

- STRUCTURAL
- DATA
- CONTROL

COND. BR. BEQ



PREDICTION - STATIC

1. NEITHER - WAIT UNTIL KNOWN — COUPLE CLOCK CYCLES
2. NOT TAKEN
3. TAKEN

STATIC ANALYSIS OF CODE
COMPILE & OPTIMIZATION

EASY CASES

```
for (i=0; i<1000; i++)
```

1:100000 WRONG GUESSES

HARD CASES FOR STATIC ANALYSIS

```
while (_____)
```

↑ some calculation

University of Idaho

DYNAMIC BRANCH PREDICTION

- RUN TIME
- PREDICTION BIT
  - ASSUME - IF NOT TAKEN LAST BR INSTR., IT WON'T BE TAKEN THIS TIME.
- ONE STEP PREDICTION - NOT VERY GOOD
- TWO STEP PREDICTION
  "IF NOT TAKE LAST 2 TIMES, ASSUME IT WONT BE THIS TIME."

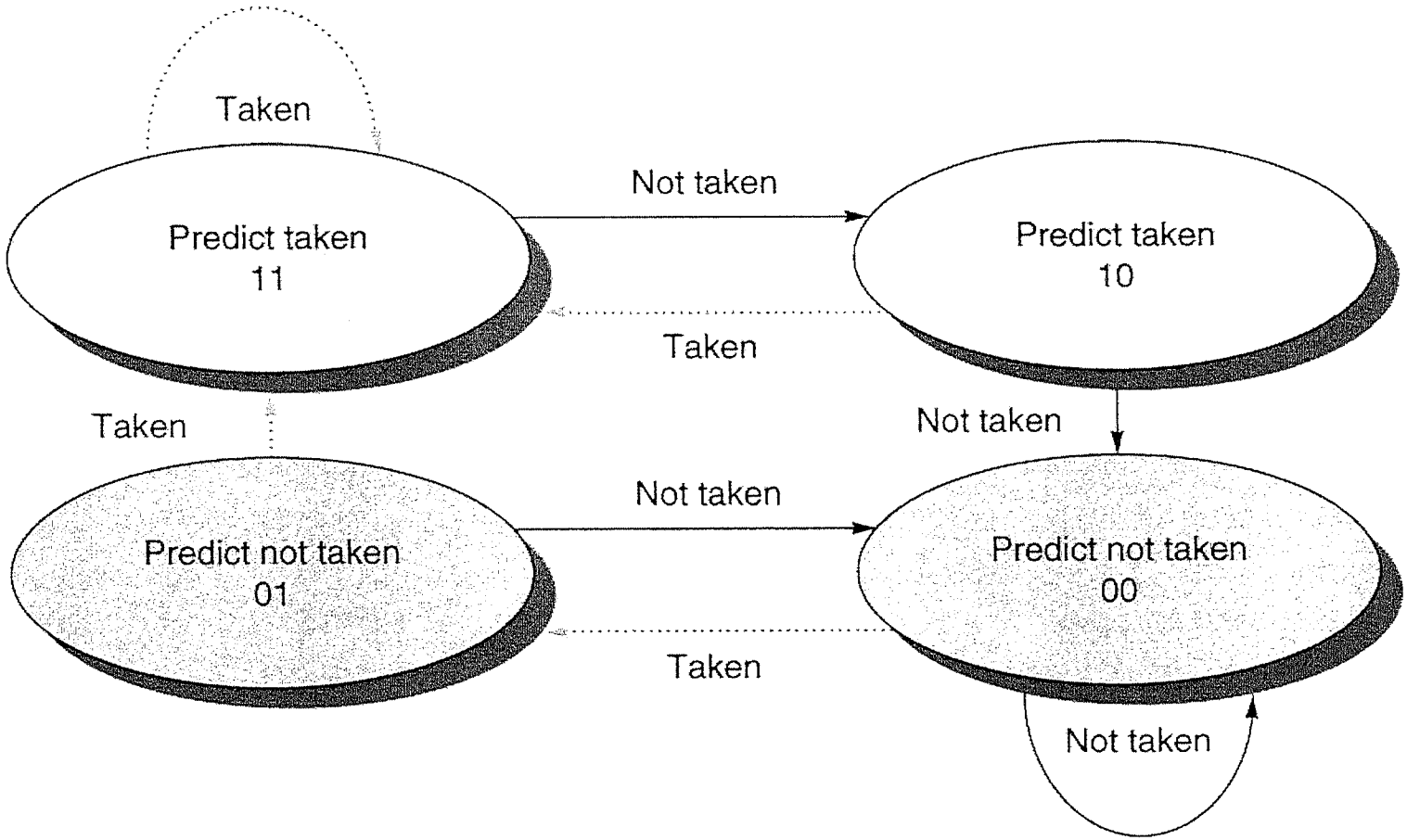


Figure C.18 The states in a 2-bit prediction scheme. By using 2 bits rather than 1, a branch that strongly favors taken or not taken —as many branches do—will be mispredicted less often than with a 1-bit predictor. The 2 bits are used to encode the four states in the system. The 2-bit scheme is actually a specialization of a more general scheme that has an *n*-bit saturating counter for each entry in the prediction buffer. With an *n*-bit counter, the counter can take on values between 0 and 2n – 1: When the counter is greater than or equal to one-half of its maximum value (2n – 1), the branch is predicted as taken; otherwise, it is predicted as untaken. Studies of *n*-bit predictors have shown that the 2-bit predictors do almost as well, thus most systems rely on 2-bit branch predictors rather than the more general *n*-bit predictors.

STATIC PREDICTION:

MIS-PREDICT	FLOAT	9%
	INT	15%
← DYNAMIC MIS. PREDICT	FLOAT	4%
	INT	11%

IMPLEMENTING PIPELINES

F.g C.21 & 22

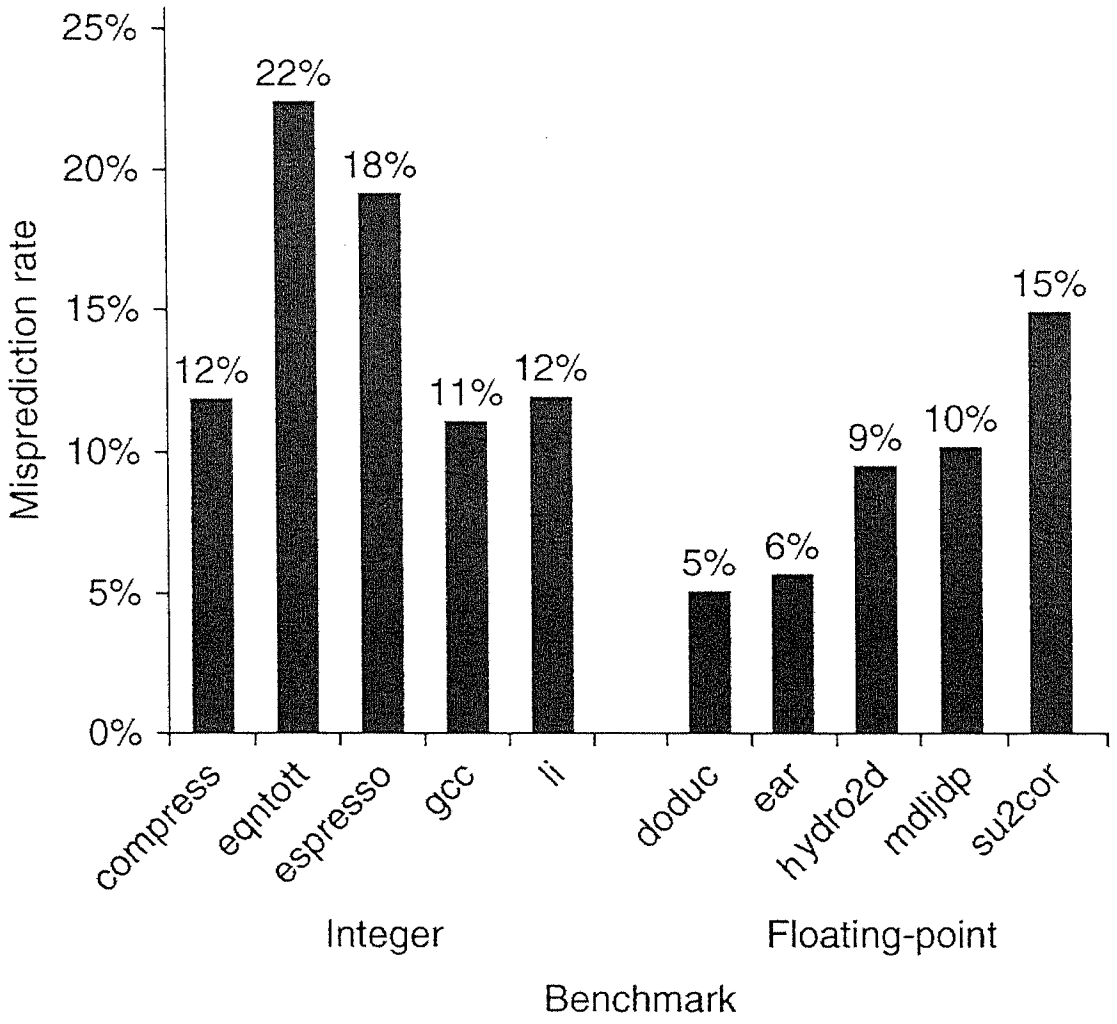


Figure C.17 Misprediction rate on SPEC92 for a profile-based predictor varies widely but is generally better for the floating-point programs, which have an average misprediction rate of 9% with a standard deviation of 4%, than for the integer programs, which have an average misprediction rate of 15% with a standard deviation of 5%. The actual performance depends on both the prediction accuracy and the branch frequency, which vary from 3% to 24%.

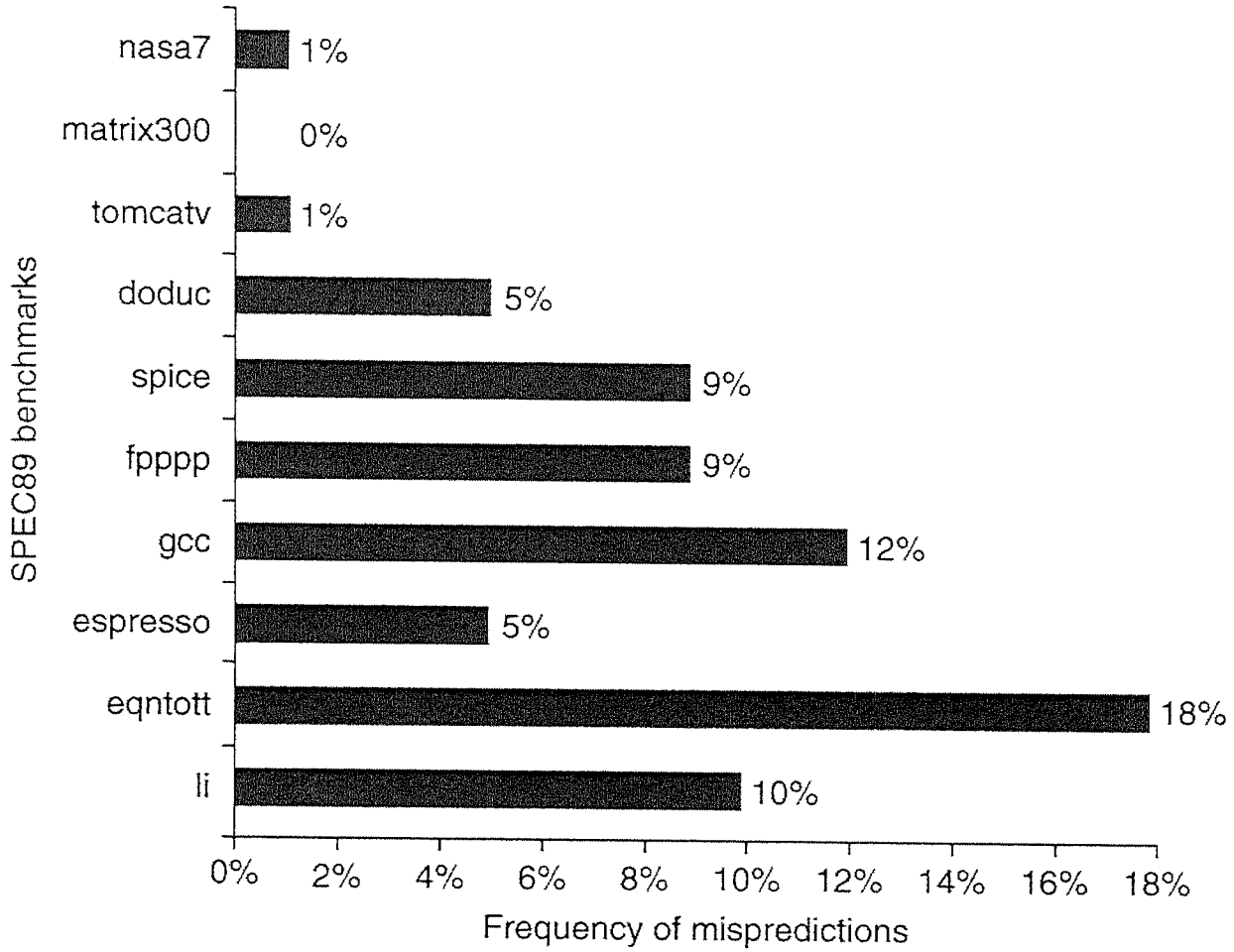


Figure C.19 Prediction accuracy of a 4096-entry 2-bit prediction buffer for the SPEC89 benchmarks. The misprediction rate for the integer benchmarks (gcc, espresso, eqntott, and li) is substantially higher (average of 11%) than that for the floating-point programs (average of 4%). Omitting the floating-point kernels (nasa7, matrix300, and tomcatv) still yields a higher accuracy for the FP benchmarks than for the integer benchmarks. These data, as well as the rest of the data in this section, are taken from a branch-prediction study done using the IBM Power architecture and optimized code for that system. See Pan, So, and Rameh [1992]. Although these data are for an older version of a subset of the SPEC benchmarks, the newer benchmarks are larger and would show slightly worse behavior, especially for the integer benchmarks.

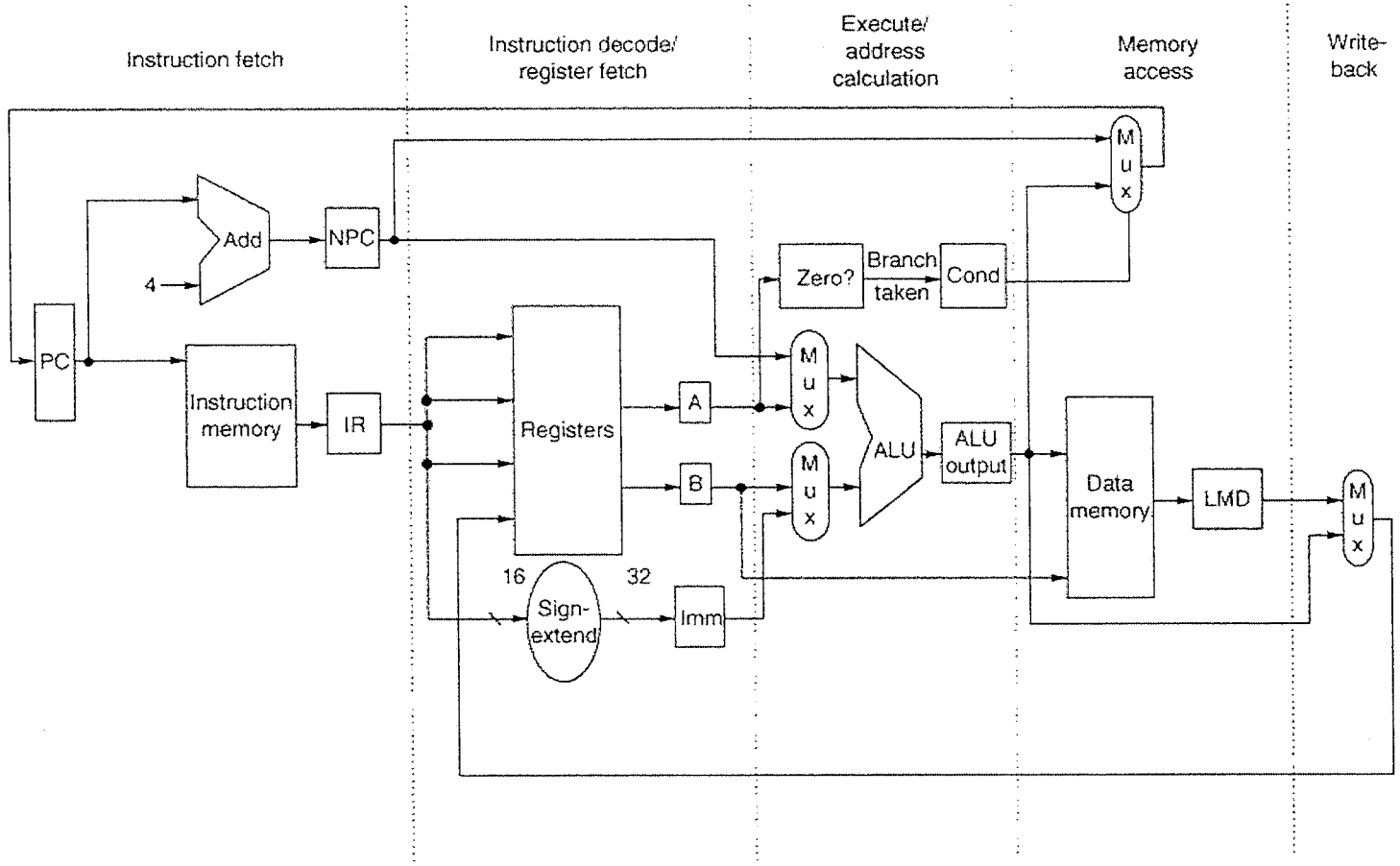


Figure C.21 The implementation of the MIPS data path allows every instruction to be executed in 4 or 5 clock cycles. Although the PC is shown in the portion of the data path that is used in instruction fetch and the registers are shown in the portion of the data path that is used in instruction decode/register fetch, both of these functional units are read as well as written by an instruction. Although we show these functional units in the cycle corresponding to where they are read, the PC is written during the memory access clock cycle and the registers are written during the write-back clock cycle. In both cases, the writes in later pipe stages are indicated by the multiplexer output (in memory access or write-back), which carries a value back to the PC or registers. These backward-flowing signals introduce much of the complexity of pipelining, since they indicate the possibility of hazards.

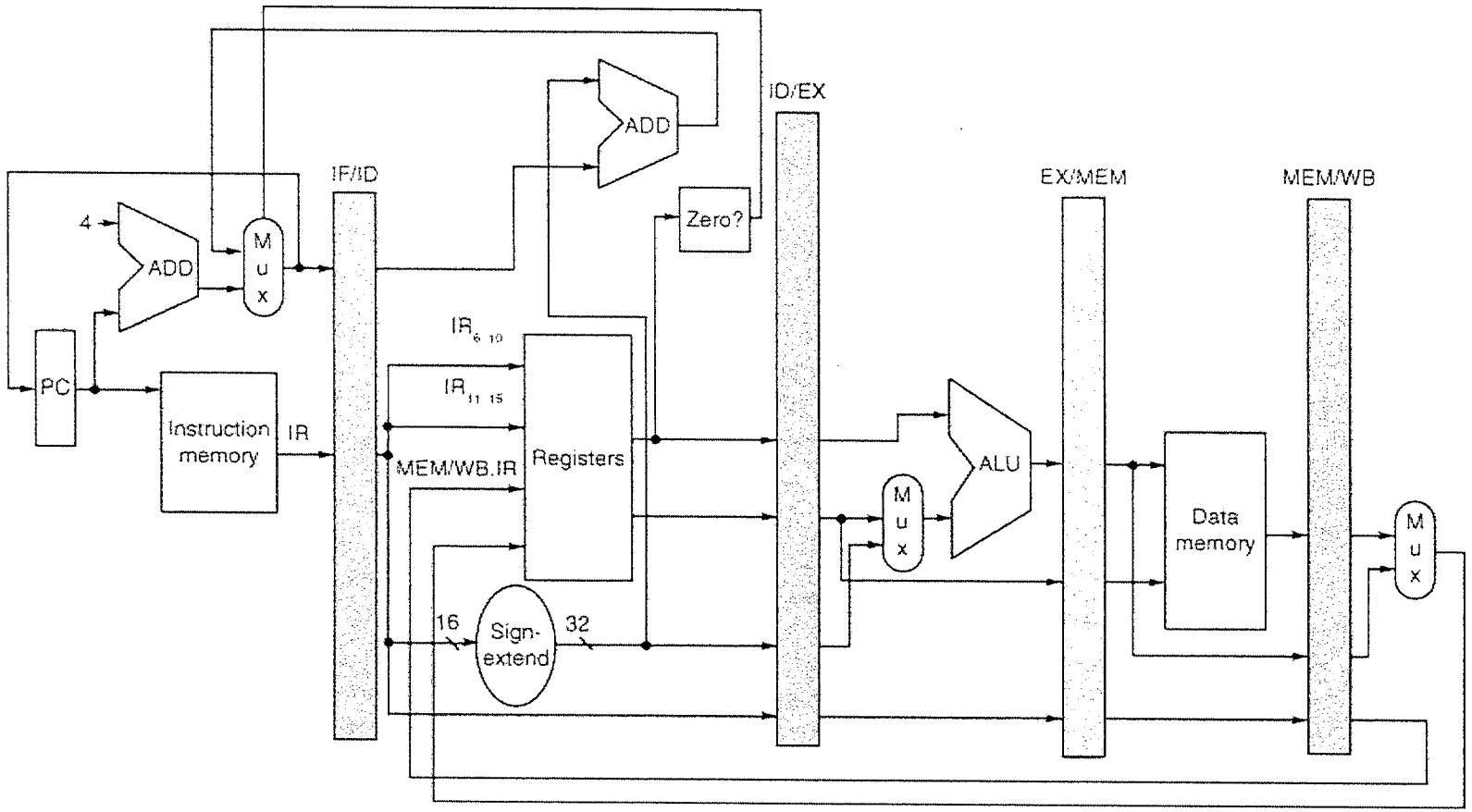


Figure C.28 The stall from branch hazards can be reduced by moving the zero test and branch-target calculation into the ID phase of the pipeline. Notice that we have made two important changes, each of which removes 1 cycle from the 3-cycle stall for branches. The first change is to move both the branch-target address calculation and the branch condition decision to the ID cycle. The second change is to write the PC of the instruction in the IF phase, using either the branch-target address computed during ID or the incremented PC computed during IF. In comparison, Figure C.22 obtained the branch-target address from the EX/MEM register and wrote the result during the MEM clock cycle. As mentioned in Figure C.22, the PC can be thought of as a pipeline register (e.g., as part of ID/IF), which is written with the address of the next instruction at the end of each IF cycle.

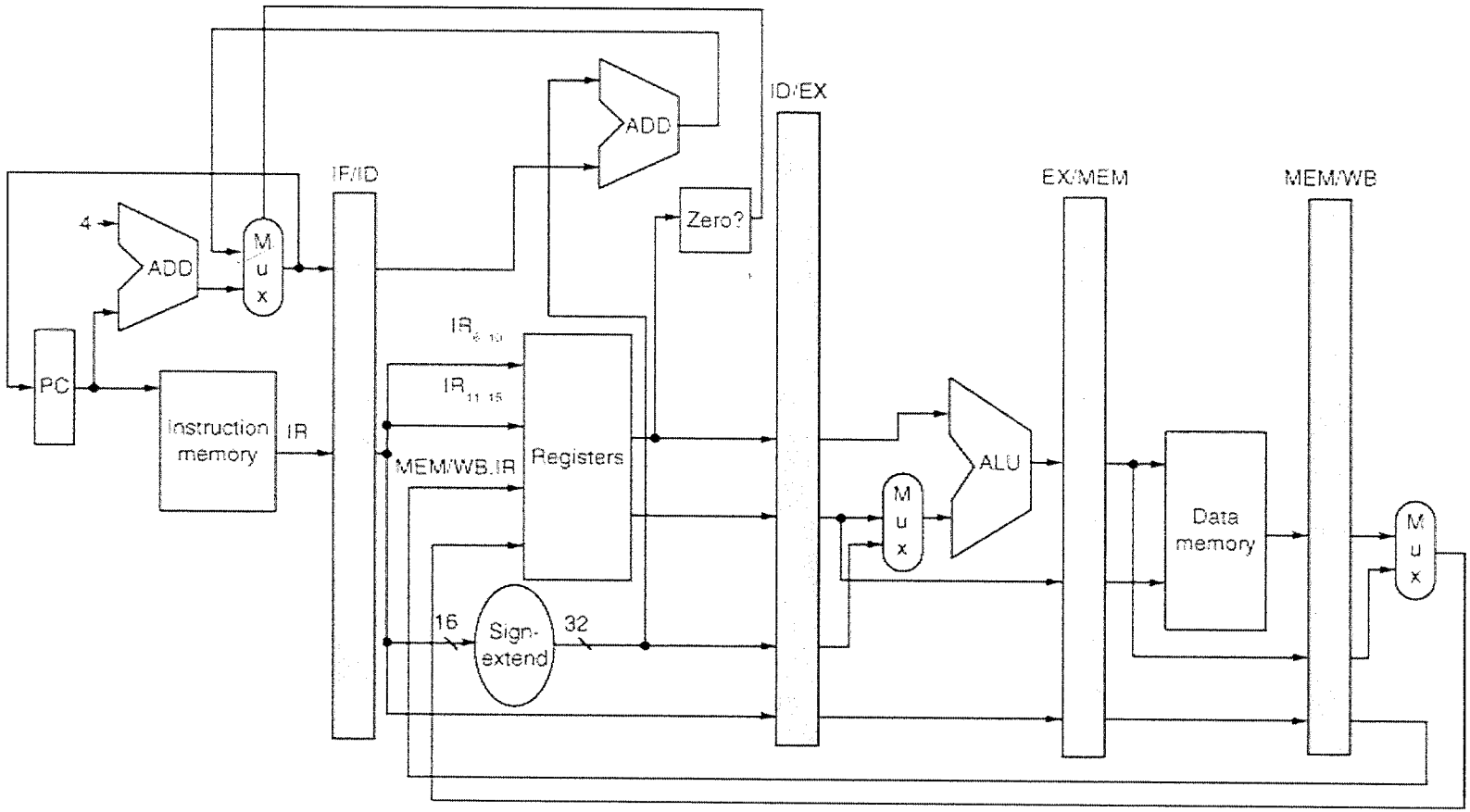


Figure C.28 The stall from branch hazards can be reduced by moving the zero test and branch-target calculation into the ID phase of the pipeline. Notice that we have made two important changes, each of which removes 1 cycle from the 3-cycle stall for branches. The first change is to move both the branch-target address calculation and the branch condition decision to the ID cycle. The second change is to write the PC of the instruction in the IF phase, using either the branch-target address computed during ID or the incremented PC computed during IF. In comparison, Figure C.22 obtained the branch-target address from the EX/MEM register and wrote the result during the MEM clock cycle. As mentioned in Figure C.22, the PC can be thought of as a pipeline register (e.g., as part of ID/IF), which is written with the address of the next instruction at the end of each IF cycle.

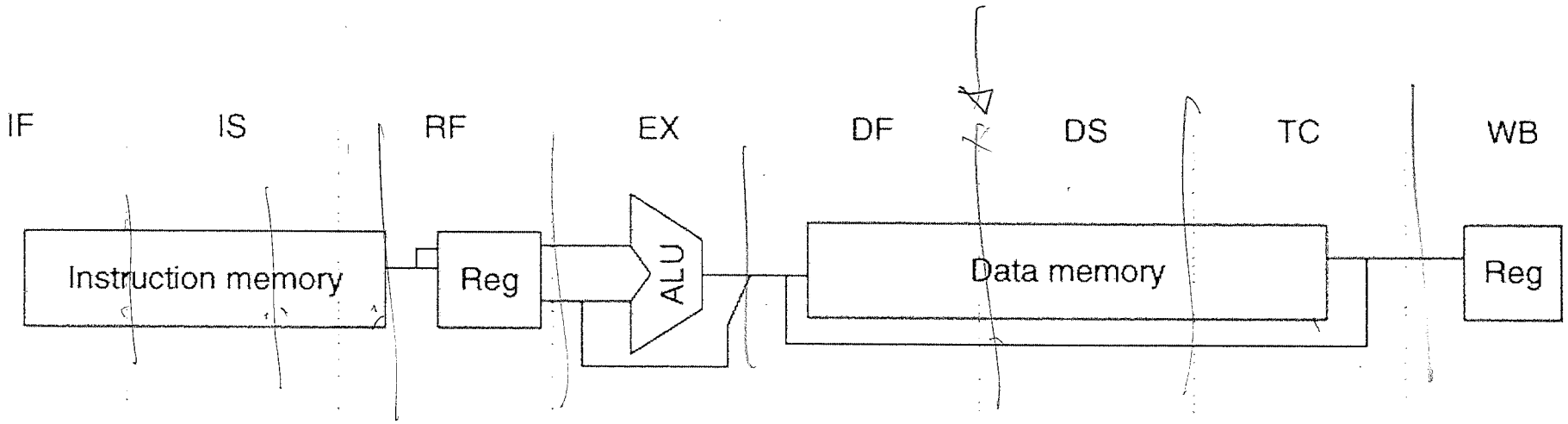


Figure C.41 The eight-stage pipeline structure of the R4000 uses pipelined instruction and data caches. The pipe stages are labeled and their detailed function is described in the text. The vertical dashed lines represent the stage boundaries as well as the location of pipeline latches. The instruction is actually available at the end of IS, but the tag check is done in RF, while the registers are fetched. Thus, we show the instruction memory as operating through RF. The TC stage is needed for data memory access, since we cannot write the data into the register until we know whether the cache access was a hit or not.

INSTRUCTION "ISSUE" -

START - FETCH, ...

ABORT - DON'T LET IT FINISH

COMMIT - LET IT FINISH.

C.4. EXCEPTIONS

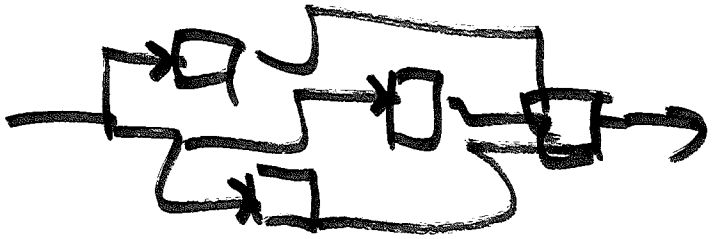
- CACHE MISS
- INTERRUPT
- INTERNAL ERROR: $\div 0$, ACCESS VIOLATION
- HARDWARE ERROR

ADVANCED TOPICS

OUT-OF-ORDER EXECUTION

- MODEL - ATOMIC INSTRUCTION STREAM

OUT OF ORDER



SUPERSCALAR 2

University of Idaho

SUPER SCALAR

SIMPLE, NON-PIPELINED: n CPI
CLOCKS/INSTRUCTION
eg. n = 5

PIPELINED: MIN 1 CPI

SUPER SCALAR: MULTIPLE INSTR/ CLOCK.

How?

~~MULTIPLE~~

MULTIPLE EXECUTION UNITS

- 2 INT ALU'S
- 1 F.P. ALU

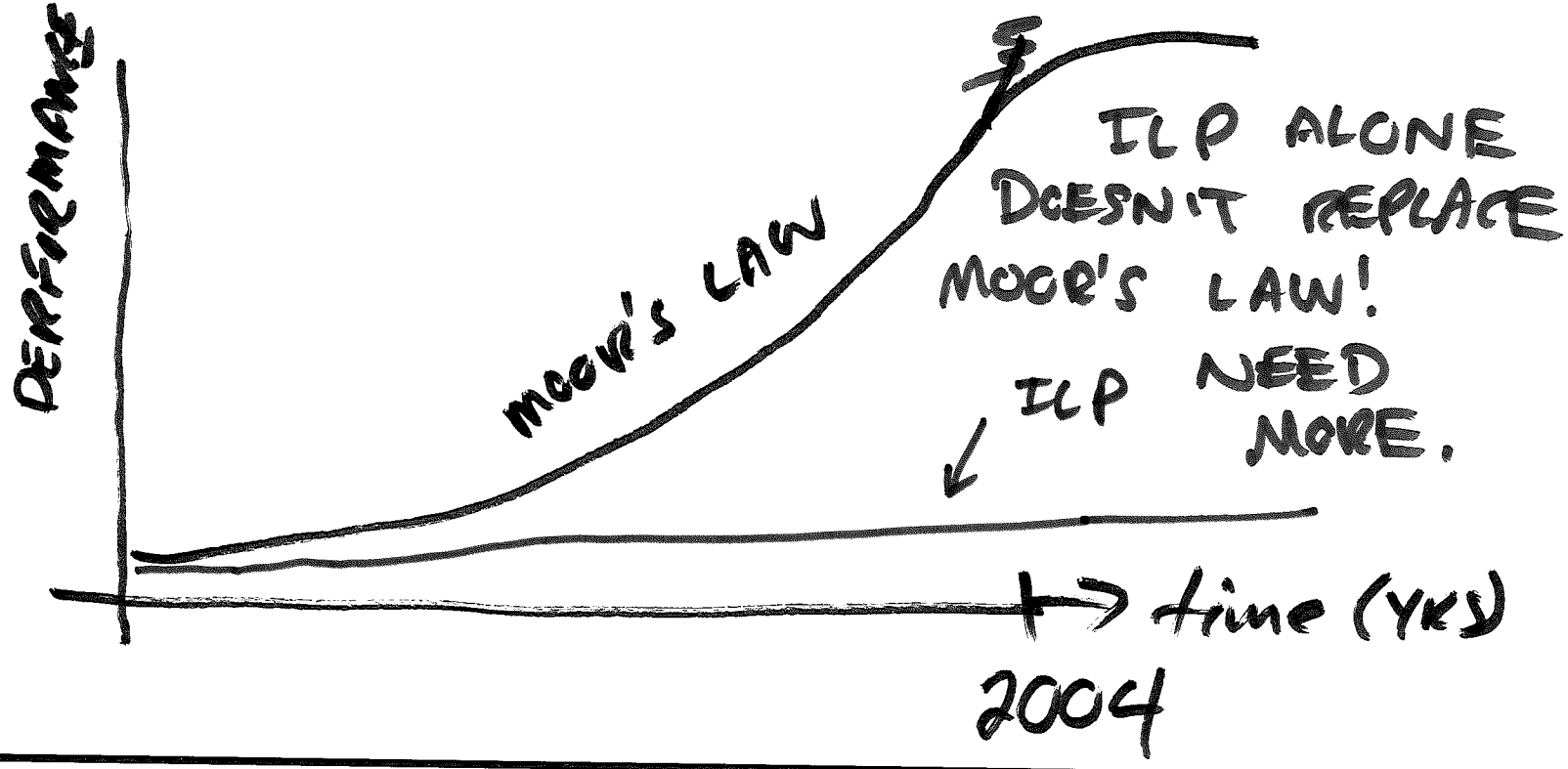
MULTIPLE PARALLEL STREAMS

University of Idaho

IN CONTEXT

PARALLELISM (ILP)

- 5X - 10X IMPROVEMENT MAX
- LESS, DUE TO HAZARDS
- FIGHT OVER 5-10% IMPRUVEMENT



University of Idaho

MEMORY SYSTEMS

CHURCH-TURING THESIS: a stored program computer can compute anything that is computable, given enough memory & time.

Model

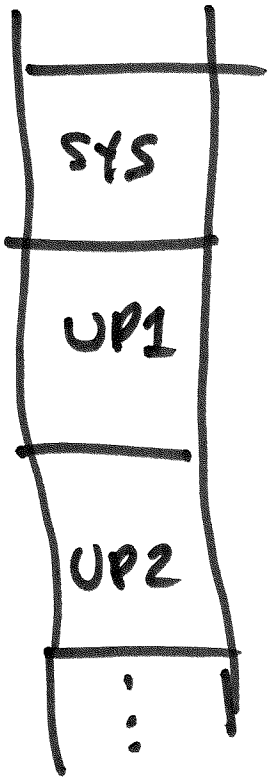
(1) Infinite amount RAM (more than the program needs). Fast, affordable.

(2) Memory protection

University of Idaho

PROTECTION

- EXECUTION IN PROCESSES OR TASKS
  - USER, APPLICATION CODE
  - SYSTEM PROCESSES
- PROCESS HAS OWN MEMORY SPACE
- ACCESS OUTSIDE MEMORY SPACE GENERATES AN EXCEPTION



- EXPLICIT MECHANISMS FOR INTER-PROCESS COMMUNICATION

University of Idaho

MEMORY HIERARCHY

ILLUSION: LARGE, FAST, CHEAP MEMORY

CACHE - "A safe place to store something."

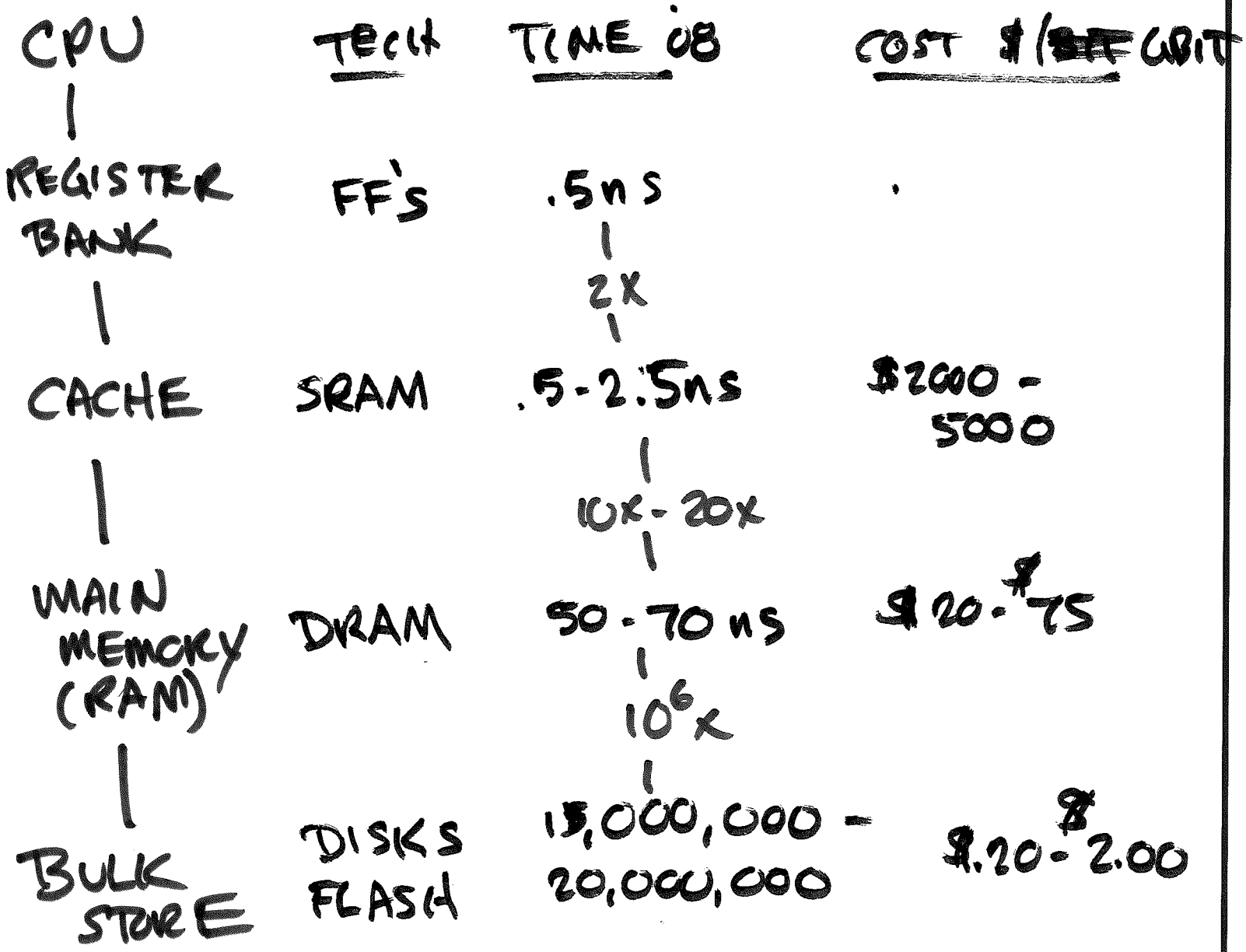
- BOOKS ON DESK! CACHE { SMALL FAST
- BOOKS IN STACKS: MAIN MEMORY LARGE, SLOW.

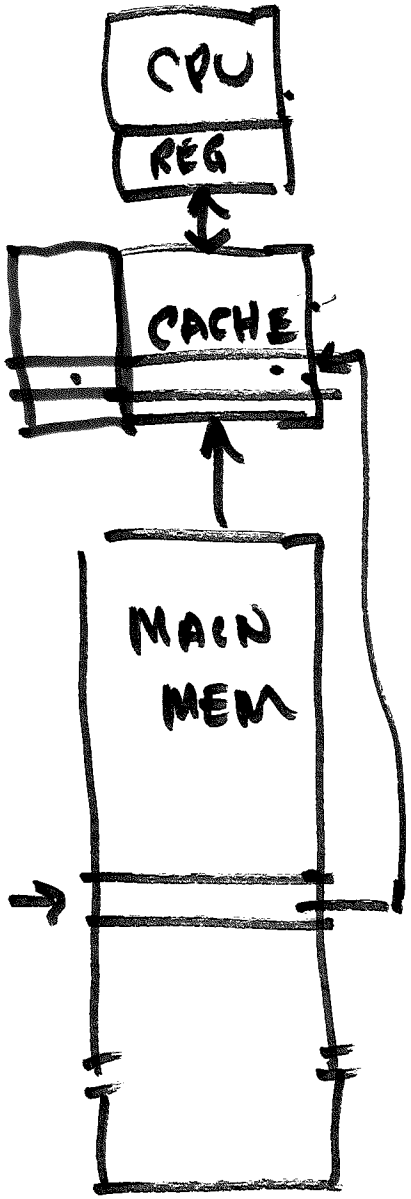
LOCALITY

TEMPORAL - An item that was referenced recently is likely to be referenced again soon.

University of Idaho

SPATIAL – The next item to be accessed is likely to be physically close to the last one.


CPU
TECH
TIME 08
COST $/BIT
REGISTER BANK
FF's
.5 ns
2x
CACHE
SRAM
.5 - 2.5 ns
$2000 - 5000
10x - 20x
MAIN MEMORY (RAM)
DRAM
50 - 70 ns
$20 - $75
10^6 x
BULK STORE
DISKS FLASH
15,000,000 - 20,000,000
$.20 - 2.00



LD R0, 0x0013 F724 f94f 0021

- CACHE EMPTY
- CPU LOADS FROM MAIN MEM TO CACHE
- READS & WRITES TO CACHE

CACHE FULL - CAN'T LOAD

- RE-USE CACHE LOCATION
- IF CACHE LOCATION WAS MODIFIED (WRITTEN BY CPU), WRITE BACK TO MAIN MEM, THEN RE-USE

- "DIRTY BIT" - SET WHEN CACHE WRITTEN

CACHE "HIT" – WHAT WE WANT IS IN CACHE

CACHE "MISS" – NOT CACHE

HIT RATE vs MISS RATE

NORMALLY – LOAD A BLOCK OF MEMORY
"CACHE LINE"
- SPATIAL LOCALITY

TRANSFERRING BLOCKS OF DATA MORE EFFICIENT THAN INDIVIDUAL ITEMS