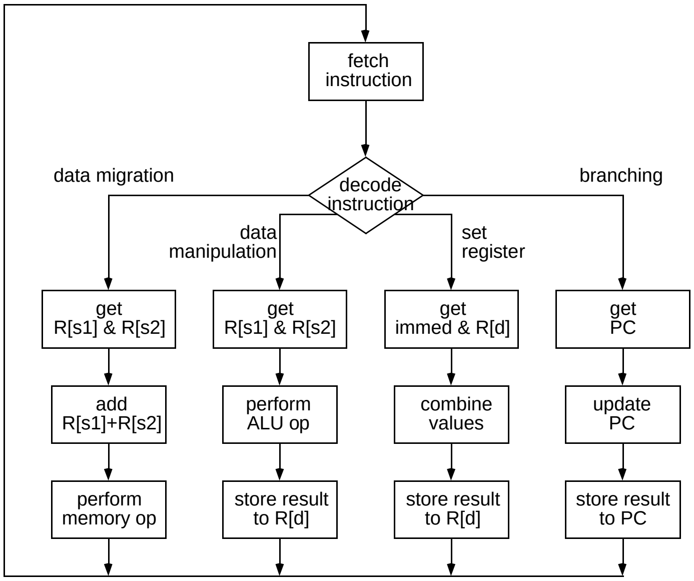
Instruction Decode / Execute Loop (Review)



fetch instruction

decode instruction

data migration

data manipulation

set register

branching

get R[s1] & R[s2]

add R[s1]+R[s2]

perform memory op

get R[s1] & R[s2]

perform ALU op

store result to R[d]

get immed & R[d]

combine values

store result to R[d]

get PC

update PC

store result to PC

Implementation of Simple Machine (Review)

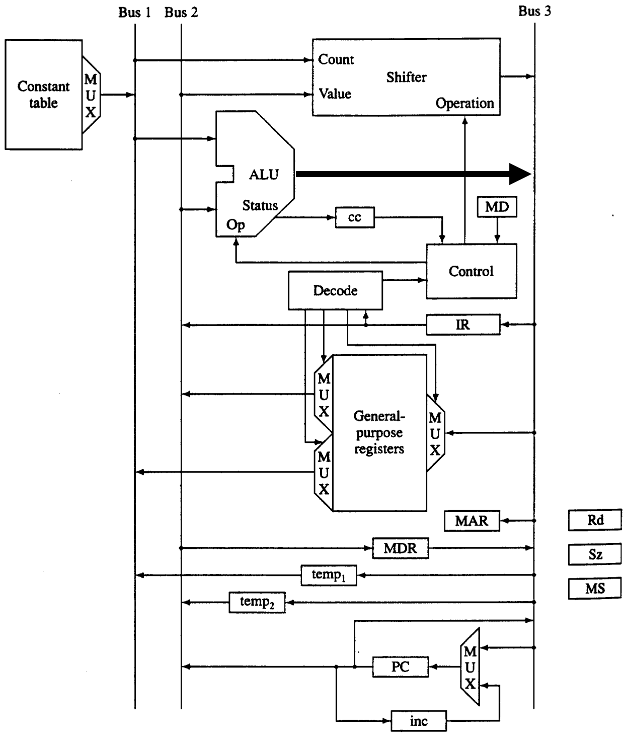


Diagram from *Computer Systems*, Maccabe, Irwin 1993

Data Migration & Manipulation Instructions (Review)

op	d	s1	s2
←4→	←4→	←4→	←4→

- Data Manipulation
  - ADD 0100 R[d] = R[s1] + R[s2]
  - SUB 0101 R[d] = R[s1] – R[s2]
  - AND 0110 R[d] = R[s1] & R[s2]
  - OR 0111 R[d] = R[s1] | R[s2]
  - XOR 1000 R[d] = R[s1] ^ R[s2]
  - XORN 1001 R[d] = R[s1] ^! R[s2]
- Data Migration
  - LDW 0000 R[d] = M[R[s1]+R[s2]]$_{16}$
  - LDB 0001 R[d] = M[R[s1]+R[s2]]$_{8}$
  - STW 0010 M[R[s1]+R[s2]] = R[d]$_{16}$
  - STB 0011 M[R[s1]+R[s2]] = R[d]$_{8}$

Implementation of Simple Machine (cont.)

- Data manipulation instructions require:
  - Two source buses, one for each ALU input
  - A result bus for the ALU output (missing line on diagram in text!)
  - Result can go through MUX to any one of 16 general purpose registers, and can come back to any one of the ALU inputs
- Data migration instructions require:
  - MDR and MAR to access memory
  - ALU must add R[s1] and R[s2] to produces address for MAR
  - MDR
    - For load, data from memory goes into MDR, then through bus 3 into register
    - For store, data goes to MDR from register (uses bus 2 —backwards from other registers!

Set Register & Branching Instructions (Review)

op	d	immed $_8$
4	4	8

- Set Register
  - SETHI 1010 $R[d]_{\text{high byte}}$ = immed_8
  - SETLO 1011 $R[d]_{\text{low byte}}$ = immed_8

op	offset $_{11}$
5	11

- Branching
  - BRA 11000 PC = PC+offset_{11}
  - BREQ 11001 if eq, PC = PC+offset_{11}x2
  - BRNE 11010 if ne, PC = PC+offset_{11}x2
  - BRLT 11011 BRLE 11100
  - BRGT 11101 BRGE 11110

Implementation of Simple Machine (cont.)

- Branches require:
  - Conditionals use shifter to multiply by 2 (same source & result buses)
    - Input comes from PC, result is stored in PC (program counter)
  - Other PC connections:
    - Incrementer increments PC when there is no branch instruction
    - PC goes to result bus and into MAR during instruction fetch
- Registers:
  - Temporary registers
    - Used during instruction interpretation
    - Specific buses for each
  - Constant table supplies constants
  - Other
    - IR instruction reg. cc condition codes
    - MD memory done MS memory select
      Sz memory size Rd read

Data Manipulation Instructions (“add” Shown Here)

- Cycle 1

  PC → bus3, load MAR
  clear MD, set MS, set Rd, set Sz

- Cycle 2…n-1

  while (MD == 0) do nothing

- Cycle n

  MDR → bus3, load IR
  inc → PC, load PC
  clear MS

fetch instruction

---

decode and execute

- Cycle n+1

  Reg[src1] → bus1
  Reg[src2] → bus2
  select ALU add operation
  ALU → bus3
  load Reg[dest]

Implementation of Simple Machine (Review)

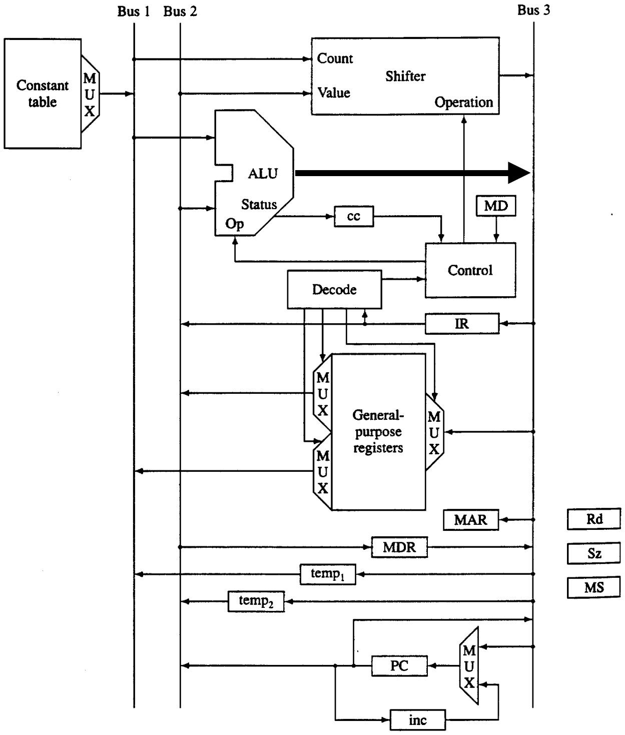


Diagram from *Computer Systems*, Maccabe, Irwin 1993

Data Migration Instructions (“load” Shown Here)

- Cycle n+1

  Reg[src1] → bus1
  Reg[src2] → bus2
  select ALU add op
  ALU → bus3
  load MAR
  clear MD, set MS, set Rd,
  set Sz to (lsb of op)’

- Cycle n+2… m

  while (MD == 0) do nothing

- Cycle m+1

  MDR → bus3
  load R[d]
  clear MS

Branching Instructions

- Cycle n+1

  IR → bus2
  #5 → bus1
  select left shift with zero fill
  shifter → bus3
  load temp2

- Cycle n+2

  temp2 → bus2
  #4 → bus1
  select right shift with sign extend
  shifter → bus3
  load temp1

- Cycle n+3

  temp1 → bus1
  PC → bus2
  select ALU add operation
  ALU → bus3
  load PC if branch condition is met

Homework #6 — Due 11/30/98

1. What problems might occur if one interrupt is allowed to interrupt the interrupt handler of another interrupt?

2. Compare the following three methods for loading immediate values into a register: (i) the MOVE instruction discussed in class, (ii) the SPARC MOV instruction, and the Chapter 9 Simple Machine’s SETLO instruction. (This question counts double.)

3. In the datapath of the Chapter 9 Simple Machine, why does the PC value need to go to both Bus 2 and Bus 3?

4. How does the μPC and μIR compare to the “real” PC and IR?

(This is the last question on Homework #6)