Chapter 9's Simple Machine

- Uses 16-bit word
- 16 general-purpose registers named R0 through R15
  - R0 is hardwired to contain the value 0
- 4 types of instructions
  - Data migration (load, store, …)
  - Data manipulation (add, sub, …)
  - Loading immediate values into registers
  - Unconditional and conditional branching
- 3 machine language instruction formats
  - 3-register format — used by data transfer and data manipulation instructions
  - Immediate format
  - Branching format

Data Migration & Manipulation Instructions

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	$R[d] = R[s1] \mid R[s2]$
● XOR	1000	$R[d] = R[s1] \wedge R[s2]$
● XORN	1001	$R[d] = R[s1]\ \wedge! R[s2]$

- 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	$M[\ R[s1]+R[s2]\ ] = R[d]_{8}$

Data Migration & Manipulation Instructions (cont.)

- The format for these instructions uses
  - 4 bits to specify the opcode
  - 4 bits to specify each register
- Data manipulation instructions
  - Each line here lists:
    - assembler mnemonic,
    - the opcode, and
    - pseudocode specifying the operation
  - Op dest, s1, s2
    - **<u>Same</u> order as text so far, <u>backwards</u> from SPARC assembler on nimitz**
- Data migration (load & store) instructions
  - Think of R[s1] as the base register,and R[s2] as the displacement
  - Word version and byte version of each
    - LDB does sign extension, STB stores least significant byte

Set Register & Branching Instructions

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 →

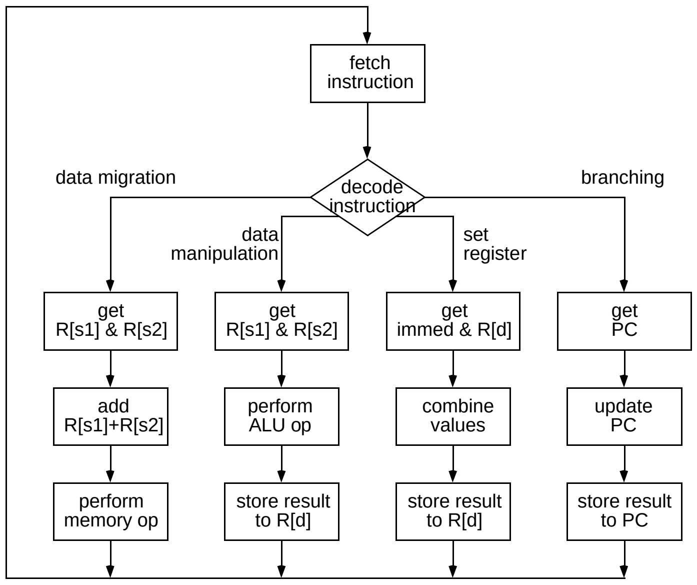
- Branching

● 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

Set Register & Branching Instructions (cont.)

- "set register" instructions
  - 4 bits for opcode, 4 bits to specify register
  - 8 bits for immediate value
  - Set the high or low byte of the register with the immediate value
    - The other byte is not affected
- Unconditional and conditional branch instructions
  - BRA (branch always = unconditional jump)
  - BREQ, etc. (conditional branch)
  - These instructions specify an offset, rather than an absolute address
    - This is the PC-relative addressing mode
    - Note that 2x that offset is used since branch will always go to a word boundary

Instruction Decode / Execute Loop



Instruction Decode / Execute Loop (cont.)

- Fetch instruction
  - Get the next instruction from memory, store it in the Instruction Register (IR), and increment the PC
- Decode instruction
  - Decode the opcode field of the IR to determine what kind of instruction is being executed
- Execute instruction
  - Data manipulation
    - Use ALU to perform operation
  - Data migration
    - Combine R[s1] and R[s2] using ALU
    - Load from or store to memory
  - Set register
    - Combine immed with specified byte of R[d]

Implementation of Simple Machine

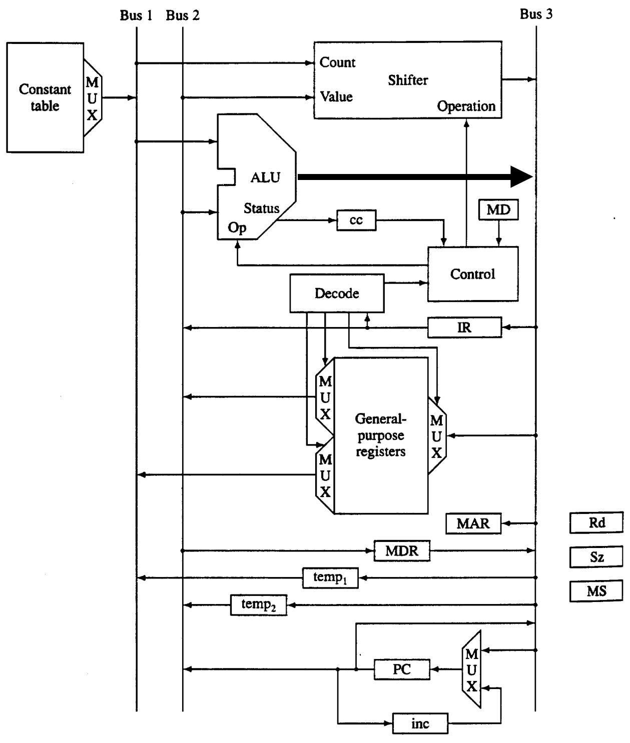


Diagram from *Computer Systems*, Maccabe, Irwin 1993