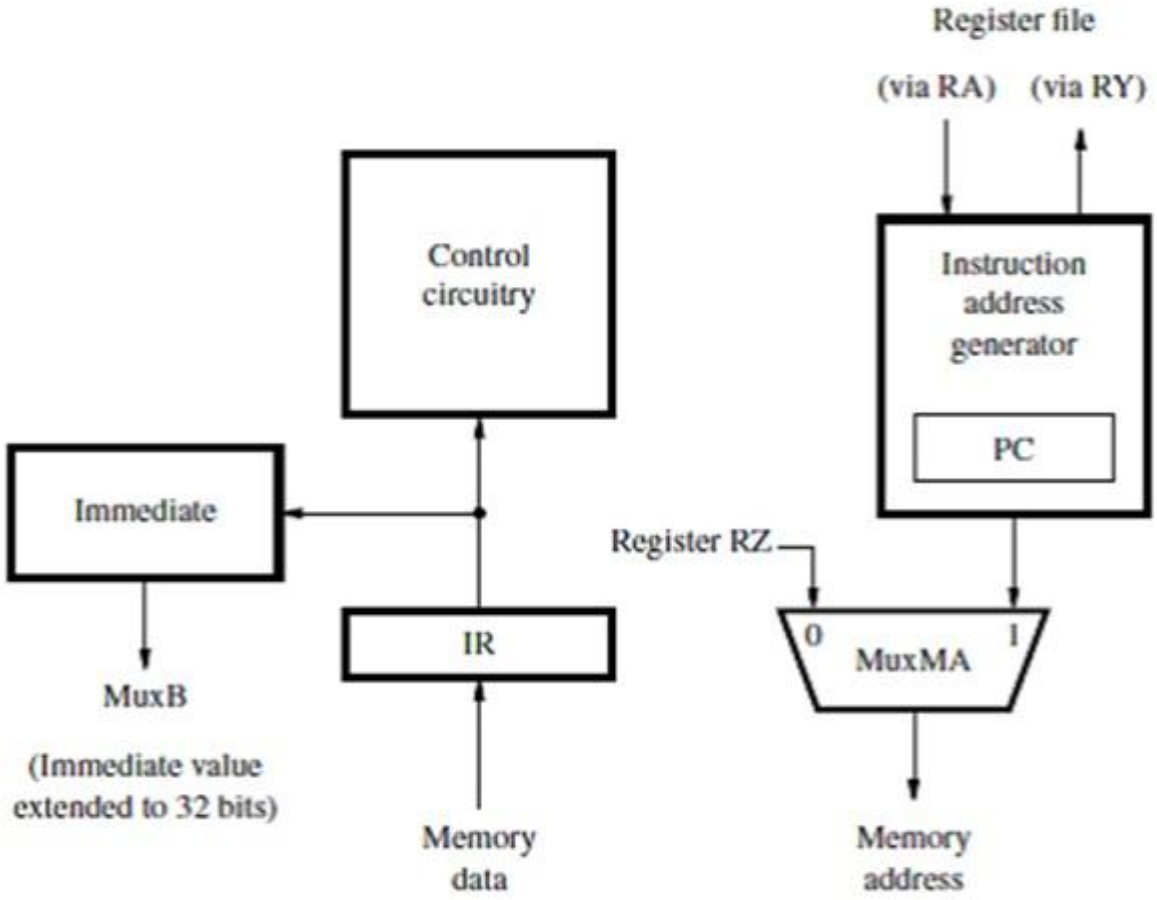
Computer Organization & Architecture

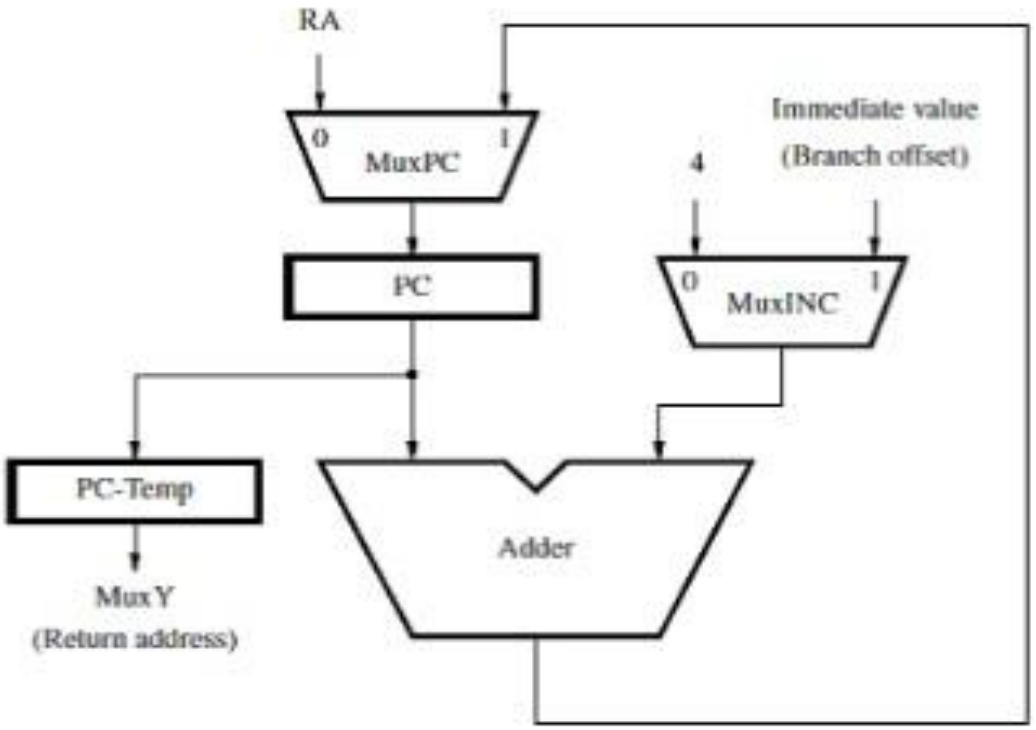
"Computer Organization and Embedded Systems"

Carl Hamacher, Zvonko Vranesic, Safwat Zaky, Naraig Manjikian

Instruction fetch



Instruction fetch



Waiting for memory

- Mostly, instruction or data referenced in memory Read and Write operations are found in cache
  - In which case the operation is completed in one clock cycle
- When the requested information is not in the cache
  - Has to be fetched from the main memory
  - Several clock cycles may be needed
- Processor-memory interface circuit must inform processor's control circuitry about such situations
  - To delay subsequent execution steps until memory operation is completed

Waiting for memory

- Assume interface circuit generates a signal called Memory Function Completed (MFC)
  - Asserts signal when a requested memory Read or Write operation has been completed
- Step 1 of execution sequence of instruction involves fetching instruction from memory
  - It must include a Wait for MFC command:
- Wait for MFC command is also needed in step 4 of Load and Store instructions

Instruction fetch-decode-execute

- Add R3, R4, R5

  1 MAR ← [PC], Read memory, Wait for MFC,

  IR ←[MDR], PC← [PC] + 4

  2 Decode instruction, RA← [R4], RB← [R5]

  3 RZ ← [RA] + [RB]

  4 RY ← [RZ]

  5 R3 ← [RY]

Instruction fetch-decode-execute

- Load R5, X(R7)

  1 MAR ← [PC], Read memory, Wait for MFC,

  IR ← [MDR], PC ← [PC] + 4

  2 Decode instruction, RA ← [R7]

  3 RZ ← [RA] + Immediate value X

  4 MAR ← [RZ], Read memory, Wait for MFC,

  RY ← [MDR]

  5 R5 ← [RY]

Instruction fetch-decode-execute

- Store R6, X(R8)
  1. MAR ← [PC], Read memory, Wait for MFC,
     IR ←[MDR], PC← [PC] + 4
  2. Decode instruction, RA← [R8], RB← [R6]
  3. RZ ← [RA] + Immediate value X, RM ← [RB]
  4. MAR ← [RZ], MDR ← [RM], Write memory, Wait for MFC
  5. No action

Branching

- Branch/subroutine call loads a new address into PC
- Subroutine call instructions also save return address
  - To be used when returning to the calling program
  - Interrupts from I/O devices and software interrupt instructions are handled in a similar manner

Branching

- Branch instructions specify branch target address relative to PC
  - Branch offset given as an immediate value in the instruction
  - Added to the current contents of the PC
- No. of bits used for offset is considerably less than word length of the computer
  - Space needed within instruction to specify opcode and branch condition
  - Hence, range of addresses that can be reached by a branch instruction is limited

Branching

- Subroutine call instructions can reach a larger range of addresses
  - Do not include a condition, so more bits are available to specify target address
- Most RISC-style computers have Jump and Call instructions that use a general-purpose register to specify a full 32-bit address

Instruction fetch-decode-execute

- Unconditional branch
  1 MAR ← [PC], Read memory, Wait for MFC,
    IR ←[MDR], PC ← [PC] + 4
  2 Decode instruction
  3 PC ← [PC] + Branch offset
  4 No action
  5 No action

Instruction fetch-decode-execute

- Conditional branch

  Ex. Branch_if_[R5]=[R6] LOOP

  1 MAR ← [PC], Read memory, Wait for MFC,
  IR ← [MDR], PC ← [PC] + 4

  2 Decode instruction, RA ← [R5], RB ← [R6]

  3 Compare [RA] to [RB], If [RA] = [RB], then
  PC ← [PC] + Branch offset

  4 No action

  5 No action

Instruction fetch-decode-execute

- Subroutine call

  Ex. Call_Register R9

  1 MAR ← [PC], Read memory, Wait for MFC,

  IR ← [MDR], PC ← [PC] + 4

  2 Decode instruction, RA ← [R9]

  3 PC-Temp ← [PC], PC ← [RA]

  4 RY ← [PC-Temp]

  5 Register LINK ← [RY]