ECE 341

Lecture # 11

Instructor: Zeshan Chishti
zeshan@ece.pdx.edu

November 3, 2014

Portland State University

Lecture Topics

- Basic Processing Unit
  - Instruction Fetch and Execution Steps
    - Branching
    - Waiting for Memory
  - Control Signals

- Reference:
  - Chapter 5: Sections 5.4.1, 5.4.2 and 5.5 (Pages 168-175 of textbook)

Branching

- Branches and subroutine calls change the control flow of program by loading a new address in the PC
- **Branch** instruction specifies the target address **relative** to the PC
  - Branch offset given as a 16-bit immediate field in IR
  - Branch offset added to the current contents of PC
  - Branches are of two types, conditional and unconditional
- **Subroutine call** instruction specifies the **absolute** address to be loaded into PC
  - Two types: *Call* and *Call_register*
  - Can reach a larger range of addresses than a branch instruction
    - *Call* uses a 26-bit immediate field in IR to specify the target address
    - *Call_register* uses a general-purpose register to specify a full 32-bit address

Unconditional Branch Execution Steps

- **Stage 1:** Memory address ← [PC], Read memory, IR ← Memory data, PC ← [PC] + 4
- **Stage 2:** Decode instruction
- **Stage 3:** PC ← [PC] + Branch offset (MuxINC selects its “1" input)
- **Stage 4:** No action
- **Stage 5:** No action

- PC is first incremented by 4 during stage 1 and then by the branch offset during stage 3
  - [PC]+4 points to the location following the branch instruction. Therefore Branch offset is the distance between the branch target and the memory location following the branch instruction

Conditional Branch Execution Steps

Example: **Branch_if_[R5]=[R6] Offset**

- **Stage 1:** Memory address ← [PC], Read memory, IR ← Memory data, PC ← [PC] + 4
- **Stage 2:** Decode instruction, RA ← [R5], RB ← [R6]
- **Stage 3:** Compare [RA] to [RB]. If [RA] == [RB], then PC ← [PC] + Branch offset (MuxINC selects its "1" input)
- **Stage 4:** No action
- **Stage 5:** No action

- Comparison of register values in stage 3 can be performed either by ALU (as a subtraction operation) or by a faster comparator circuit (Example 5.3 in textbook)

Subroutine Call Execution Steps

Example: **Call_Register R9** (Call a subroutine whose starting address is in register R9)

- **Stage 1:** Memory address ← [PC], Read memory, IR ← Memory data, PC ← [PC] + 4
- **Stage 2:** Decode instruction, RA ← [R9]
- **Stage 3:** PC-Temp ← [PC], PC ← [RA] (MuxPC selects its "0" input)
- **Stage 4:** RY ← [PC-Temp]
- **Stage 5:** Register LINK ← [RY]

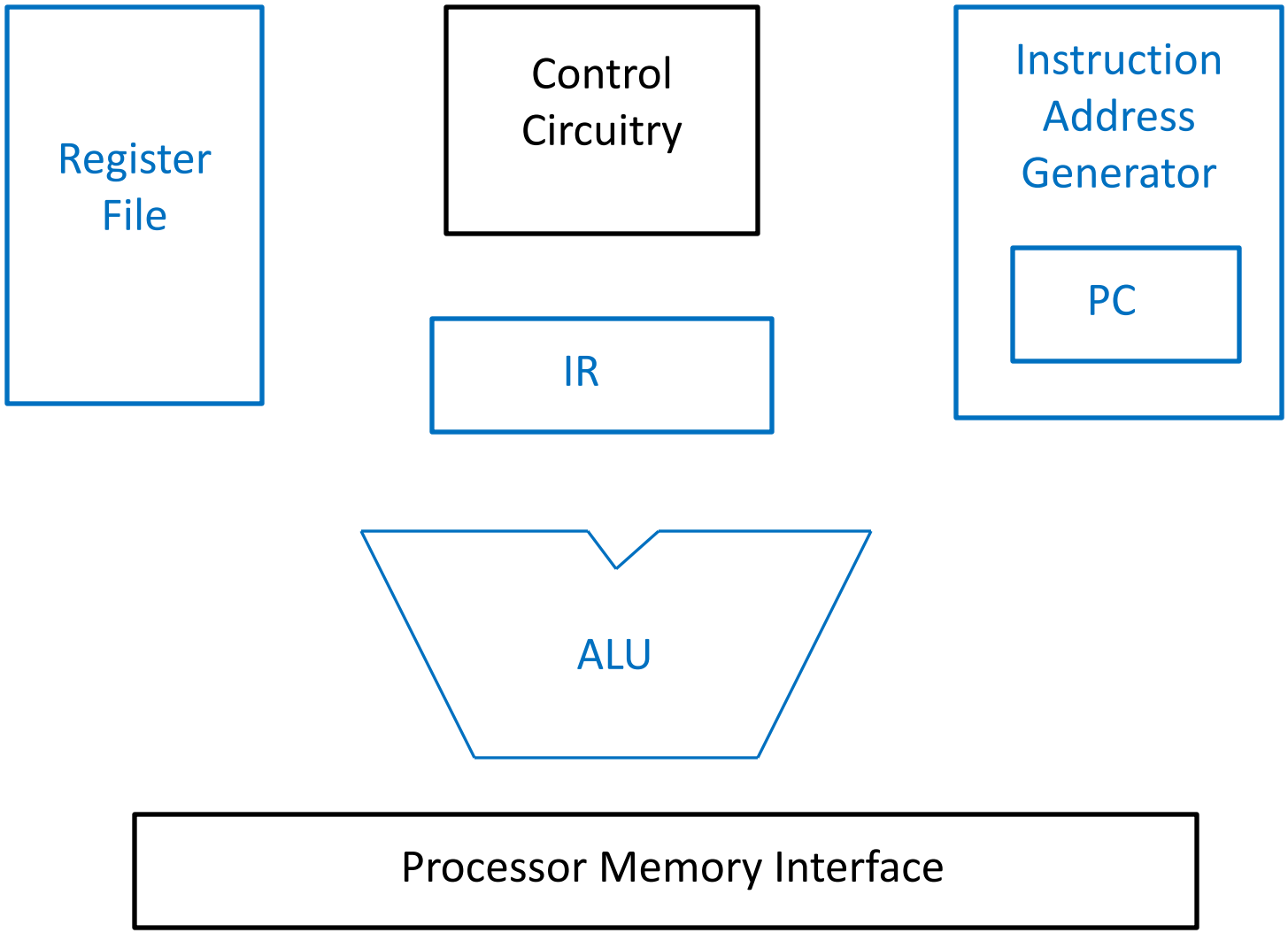
- Return address of the subroutine is saved in a general-purpose register called LINK in the register file

Return-from-Subroutine Execution Steps

- Sub-routine return instruction transfers the value saved in register LINK back to the PC
- This instruction is encoded such that the address of LINK register appears in IR[31-27]

- **Stage 1:** Memory address ← [PC], Read memory, IR ← Memory data, PC ← [PC] + 4
- **Stage 2:** Decode instruction, RA ← [Register LINK]
- **Stage 3:** PC ← [RA] (MuxPC selects its "0" input)
- **Stage 4:** No Action
- **Stage 5:** No Action

Hardware Components of a Processor



Processor Memory Interface

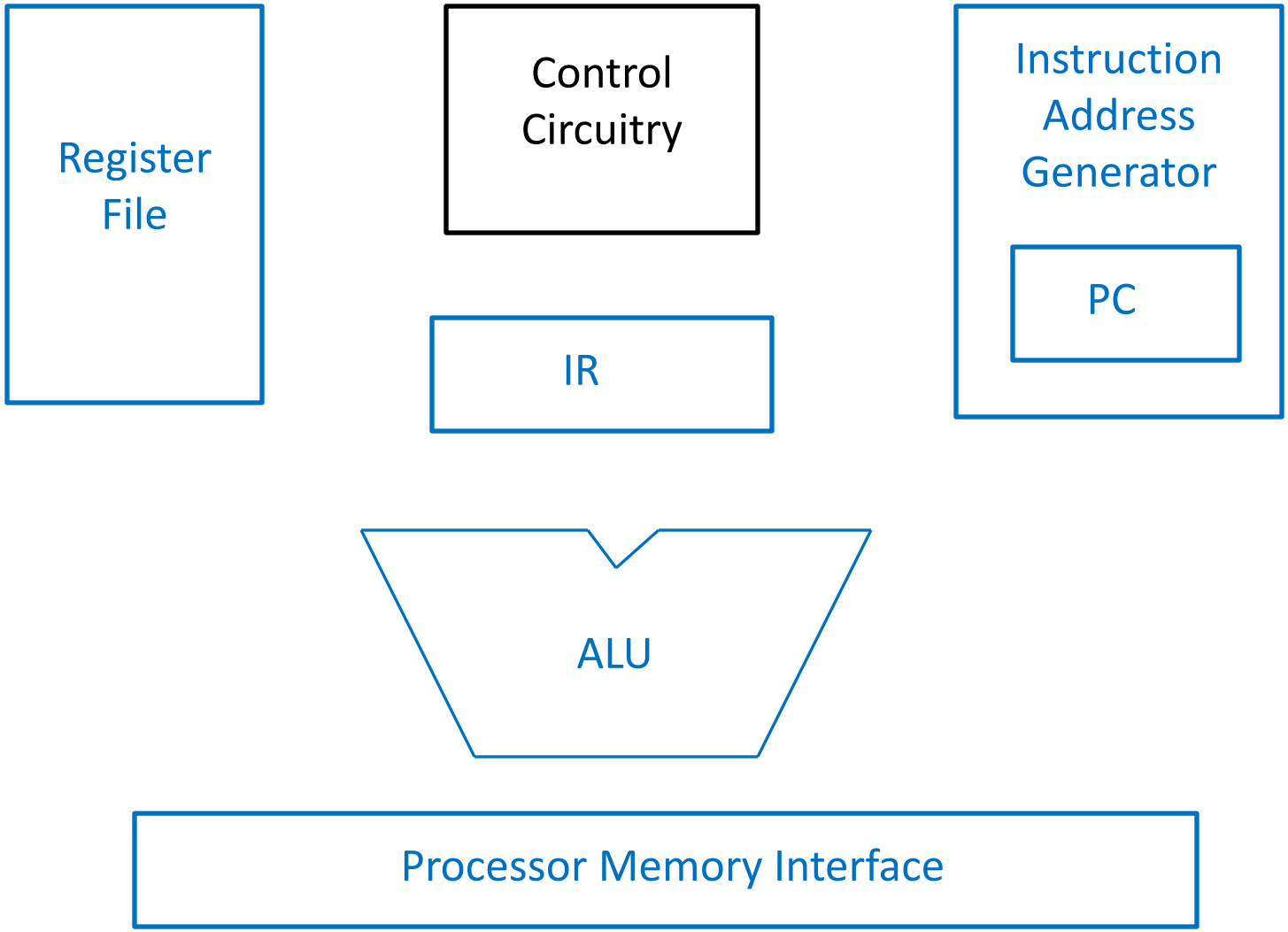
- Processor memory interface controls data transfers between processor and memory
- We have assumed so far that the memory access stage (stage 4) of the multi-stage hardware completes in one cycle
- However, main memory is typically much slower than processor
  - Memory access requires tens of processor clock cycles
- Fast on-chip caches enable single cycle reads and writes
- But, required instruction/data is sometimes unavailable in the cache (cache miss) and has to be fetched from main memory
- Memory interface must inform the processor about such situations, to delay subsequent steps until the memory operation is completed

Waiting for Memory

- Memory interface generates a signal **Memory Function Completed (MFC)**
- When a memory read/write completes, memory interface asserts MFC
- Processor's control circuitry monitors MFC during any stage in which it issues a memory read or write request, to determine when it can proceed to the next processing stage
  - If data is found in cache, MFC is asserted before the end of clock cycle in which memory request is sent => execution proceeds uninterrupted
  - If data is not found in cache, MFC is not asserted (for several cycles) until access to main memory is completed => execution stalls until MFC is asserted
- For example, stage 1 involves fetching instruction from the memory. Therefore stage 1 must include a "Wait for MFC" command as follows:

Memory Address ← [PC], Read memory, **Wait for MFC**, IR ← Memory data, PC ← [PC] + 4

Hardware Components of a Processor



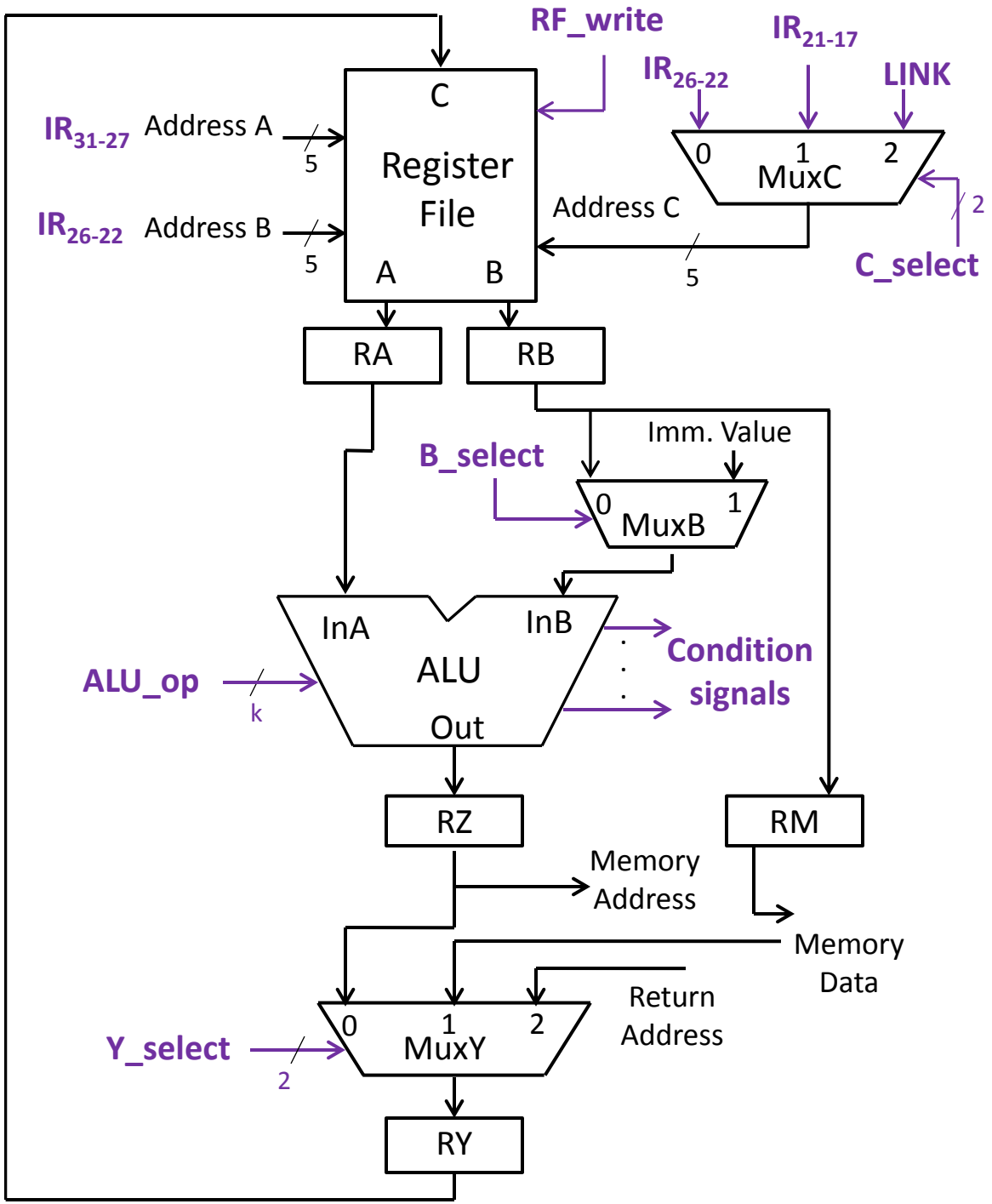
Control Signals

- Control signals govern the operation of a processor's components
- Control circuitry examines the instruction in IR and generates the control signals needed to execute the instruction
- Examples of decisions made by control signals:
  - Which registers (if any) are enabled for writing?
  - Which input is selected by a multiplexer?
  - What operation is performed by the ALU?
- Some control signals depend only on instruction type, while others depend on both the instruction type and current processing step

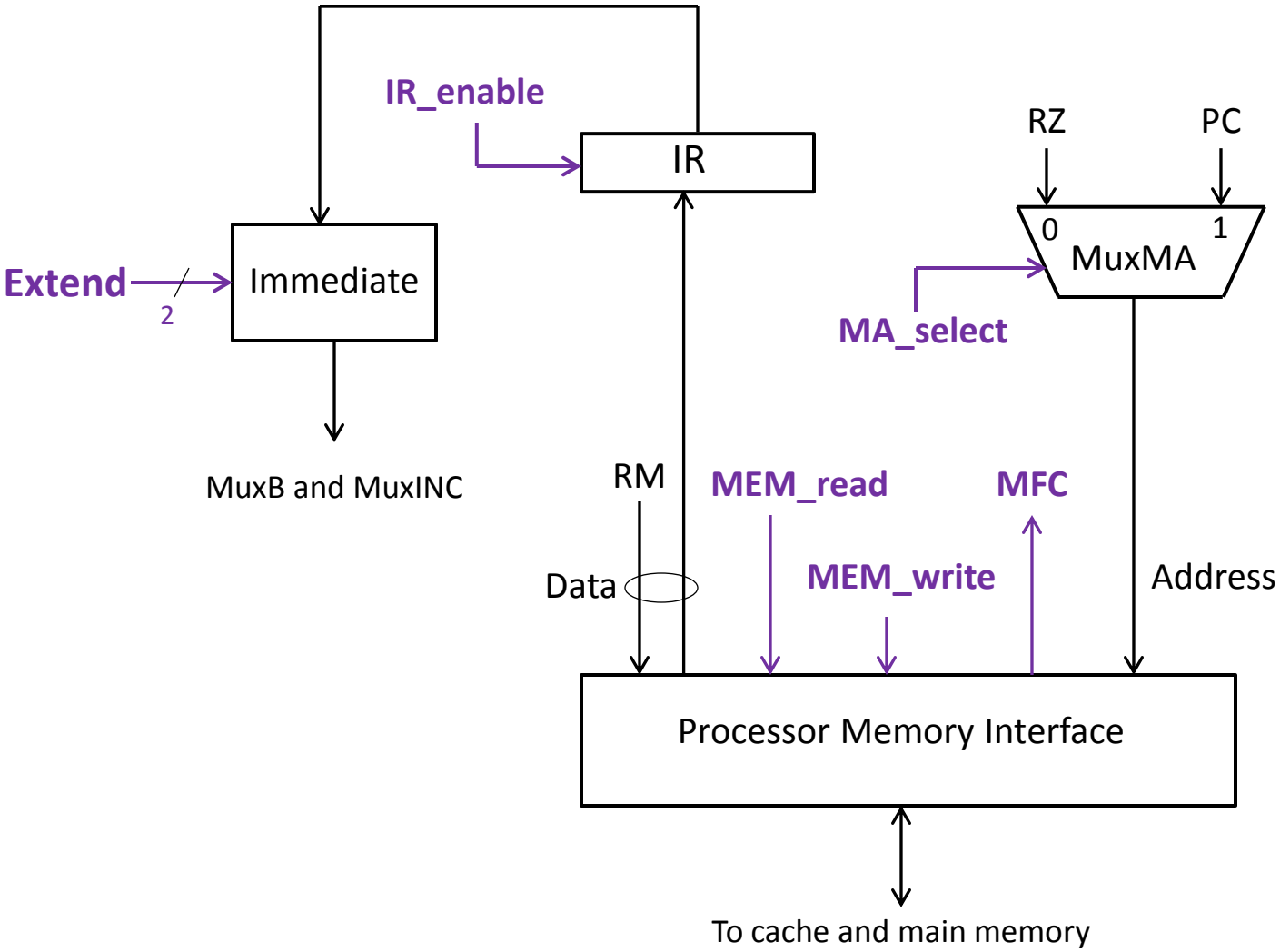
Control Signals for Datapath

Need control signals for register file, multiplexers and ALU

Inter-stage registers transfer data from one stage to the next in ***every*** cycle => **no need** for a control signal, since these registers are always enabled



Control Signals for Memory Interface and IR



Control Signals for Instruction Address Generator