Exceptions and Interrupts

- Unexpected events that change the control flow
    - Exceptions: events that occur within the CPU
        - Arithmetic overflow
        - Invalid instruction
    - Interrupts: events caused by external sources
        - I/O device communication mechanism
        - Watchdog timer
- CPU must provide OS with
    - An indication what type of event occurred
    - An indication where the event occurred

Handling Exceptions

- CPU provides the address of the instruction where the event occurred

  - The Exception Program Counter (EPC)

  - CPU might undo addition of 4 from fetch cycle

- Two ways to indicate the type of event

  - Cause Register: CPU provides the OS with a value in a register that indicates what caused the event

  - Vectored: CPU starts executing at an address that depends on the event type

Handling Exceptions – Cause Register

- The EPC contains the address of the instruction
- The Cause register contains a value that indicates what type of event occurred
  - For example:

    Invalid Instruction: Cause = 0x0000000A

    Arithmetic Overflow: Cause = 0x0000000C

- When an exception or interrupt occurs:
  - The CPU sets the EPC and Cause registers
  - Starts executing at a defined address
    - 0x80000180 in MIPS
  - The OS determines how to handle the event
- MIPS handles exceptions and interrupts this way

3

Handling Exceptions – Vectored

- EPC contains instruction address

- No Cause register

- CPU goes to an address based on the event type
  - Looks at the interrupt vector (or description) table
  - For example:

    | Arithmetic Overflow: | PC = 0xC0000000 |
    | Undefined Instruction: | PC = 0xC0000020 |

- When an exception or interrupt occurs:
  - The CPU sets the EPC and looks up interrupt handler address
  - Starts executing the interrupt handler
  - The handler returns to the program when done

Interrupt Classification

- Internal or external
  - Internal interrupts caused by instruction
    - Overflow
    - Invalid instruction
  - External interrupts caused by sources outside CPU
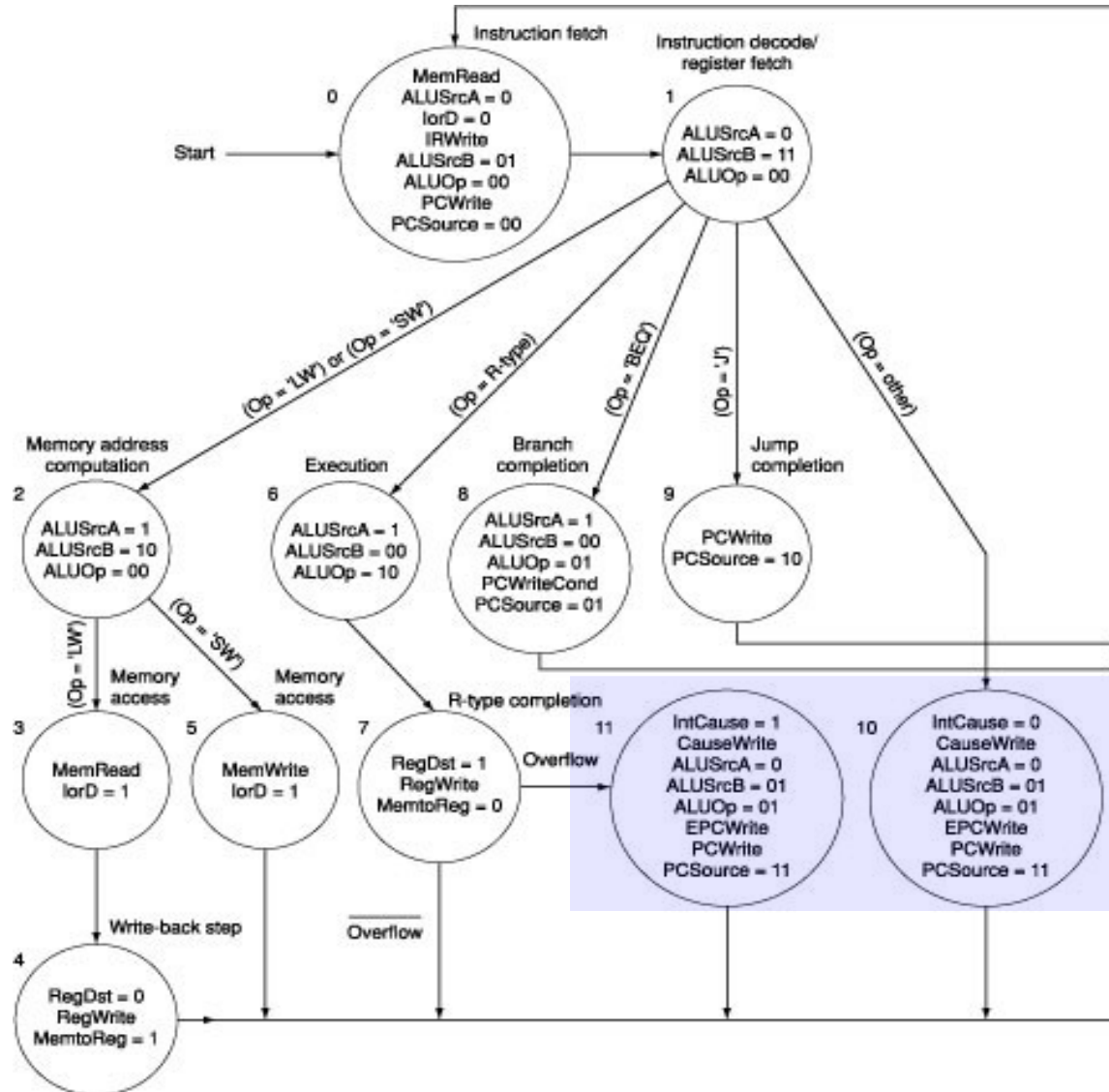   - Device request
    - Bus error
- Precision
  - Precise interrupts
    - Instructions before interrupt completed
    - Instruction that caused the interrupt and those after have not changed the CPU state
  - Imprecise interrupts cannot guarantee these conditions

Control Unit Adaptation

• Control Unit of CPU must be modified to detect and handle exceptions and interrupts

- Logic necessary to detect exceptions
  - Check for invalid opcode/function field values
  - ALU modified to detect overflow
- Exception handling address input to PC multiplexer
- Control signals for Cause and EPC registers
- Use ALU to compute PC of current instruction
  - PC updated to PC+4 during fetch cycle
  - Compute PC+4-4=PC during exception cycle

State Diagram with Exceptions

From Book: