4.9 Instruction Processing

- The fetch-decode-execute cycle is the series of steps that a computer carries out when it runs a program.
- We first have to fetch an instruction from memory, and place it into the IR.
- Once in the IR, it is decoded to determine what needs to be done next.
- If a memory value (operand) is involved in the operation, it is retrieved and placed into the MBR.
- With everything in place, the instruction is executed.

The next slide shows a flowchart of this process.

---

4.9 Instruction Processing

- All computers provide a way of interrupting the fetch-decode-execute cycle.
- Interrupts occur when:
  - A user break (e.g., Control+C) is issued
  - I/O is requested by the user or a program
  - A critical error occurs
- Interrupts can be caused by hardware or software.
  - Software interrupts are also called traps.

---

4.9 Instruction Processing

- Interrupt processing involves adding another step to the fetch-decode-execute cycle as shown below.

The next slide shows a flowchart of "Process the interrupt."
4.9 Instruction Processing

For general-purpose systems, it is common to disable all interrupts during the time in which an interrupt is being processed.

- Typically, this is achieved by setting a bit in the flags register.
- Interrupts that are ignored in this case are called **maskable**.
- **Nonmaskable** interrupts are those interrupts that must be processed in order to keep the system in a stable condition.

Interrupts are very useful in processing I/O.
- However, interrupt-driven I/O is complicated, and is beyond the scope of our present discussion.
  - We will look into this idea in greater detail in Chapter 7.
- MARIE, being the simplest of simple systems, uses a modified form of programmed I/O.
- All output is placed in an output register, OutREG, and the CPU polls the input register, InREG, until input is sensed, at which time the value is copied into the accumulator.