MODULE 6: Instruction Set Architecture

Lecture 6.2
Instruction Types

Prepared By:
- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering
Virginia Tech

Lecture 6.2 Objectives

- Describe the different types of instructions
- List a few example operations for each different type of instruction

Instruction Types

- Seven categories of instructions are commonly specified
  - Data movement instructions
  - Arithmetic operations instructions
  - Boolean logic instructions
  - Bit manipulation instructions
  - Input/output (I/O) instructions
  - Transfer control instructions
  - Special purpose instructions

Data Movement Instructions

- Move data without altering the data
- Examples (fictitious)
  - MOVE R1,R2 – Move the contents of register R1 into R2
  - XCHG R1,R2 – Exchange the contents of R1 with the contents of R2
  - LOAD #52H,R1 – Move the hexadecimal value 52 into register R1

Arithmetic Instructions

- Perform an arithmetic operation (add, subtract, negate, increment, multiply, etc.) on one or more operands
- Need to assume a particular data representation
  - Fixed integer versus floating point
  - Signed two's complement, signed one's complement, or sign-magnitude
- May have "side effects" such as setting a flag to indicate…
  - Carry out
  - Negative result
  - Overflow
  - Loss of accuracy (for floating point values)

Arithmetic Instructions (continued)

- Examples (fictitious)
  - ADD R1,R2,R3 – Add contents of R1 to the contents of R2 and put the result in R3
  - DEC R1 – Decrement R1 and put the result in R1
  - MULT R1,R2 – Multiply the contents of R1 by the contents of R2 and put the results in R2
  - FMULT R1,R2 – Floating point multiply of the contents of R1 by the contents of R2 with results to R2

Boolean Logic Instructions

- Perform bit-wise Boolean operations on one or more operands
- Examples (fictitious)
  - AND R1,R2 – Perform a logical AND of the bits in R1 with those in R2 and put the result in R2
  - XOR R1,R2 – Perform a logical exclusive-or of the bits in R1 with those in R2 and put the result in R2
  - NOT R1 – Complement (invert) the bits in R1 and leave the result in R1

CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can do the following:

- Describe and state a few examples of Data Movement, Arithmetic, and Boolean Logic instructions

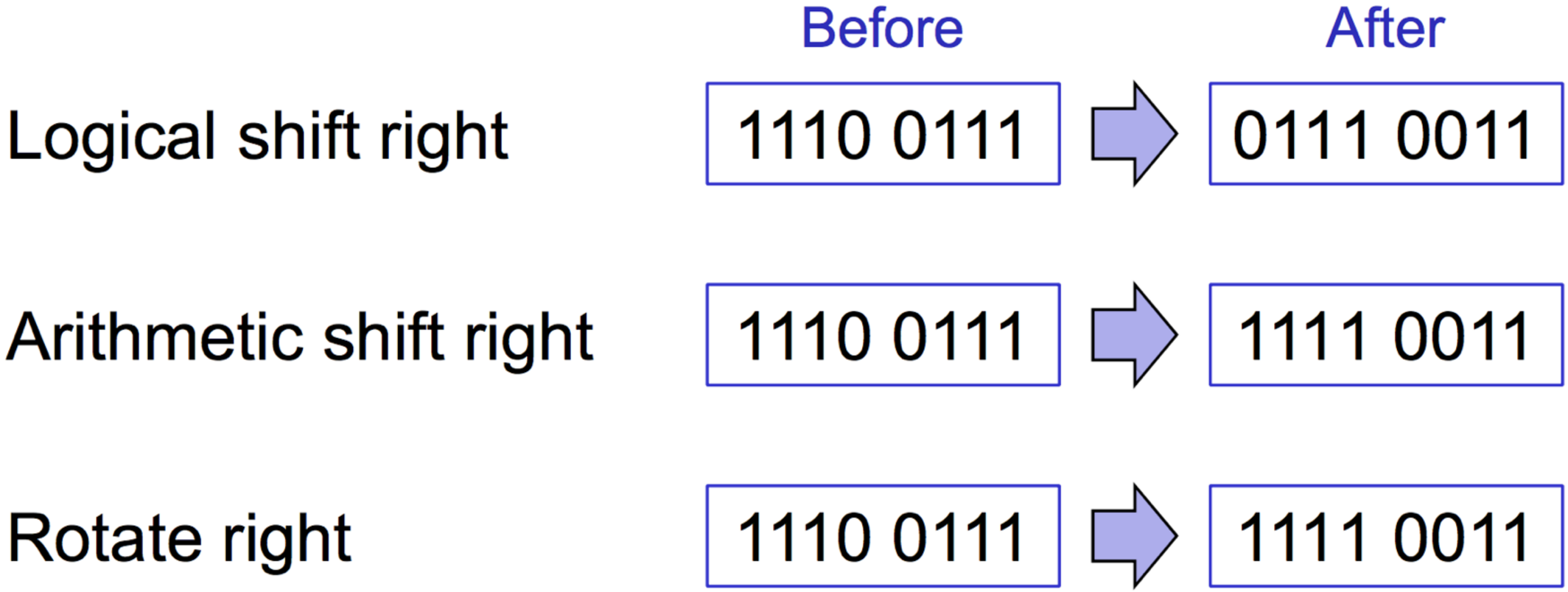
If you have any difficulties, please review the lecture video before continuing.

Bit Manipulation Instructions

- Manipulate individual bits or operate on the full register as a set of bits (e.g., rotation or shift)
- Note that shifts may be:
  - Logical, where 0 is shifted in
  - Arithmetic (or divide or multiply by 2), where the sign is replicated for an arithmetic right shift
  - Rotate, where bits wrap around
- Bit operation example (fictitious)
  - SET R1,3 – Set (make 1) bit 3 of the contents of R1 and put the result back in R1

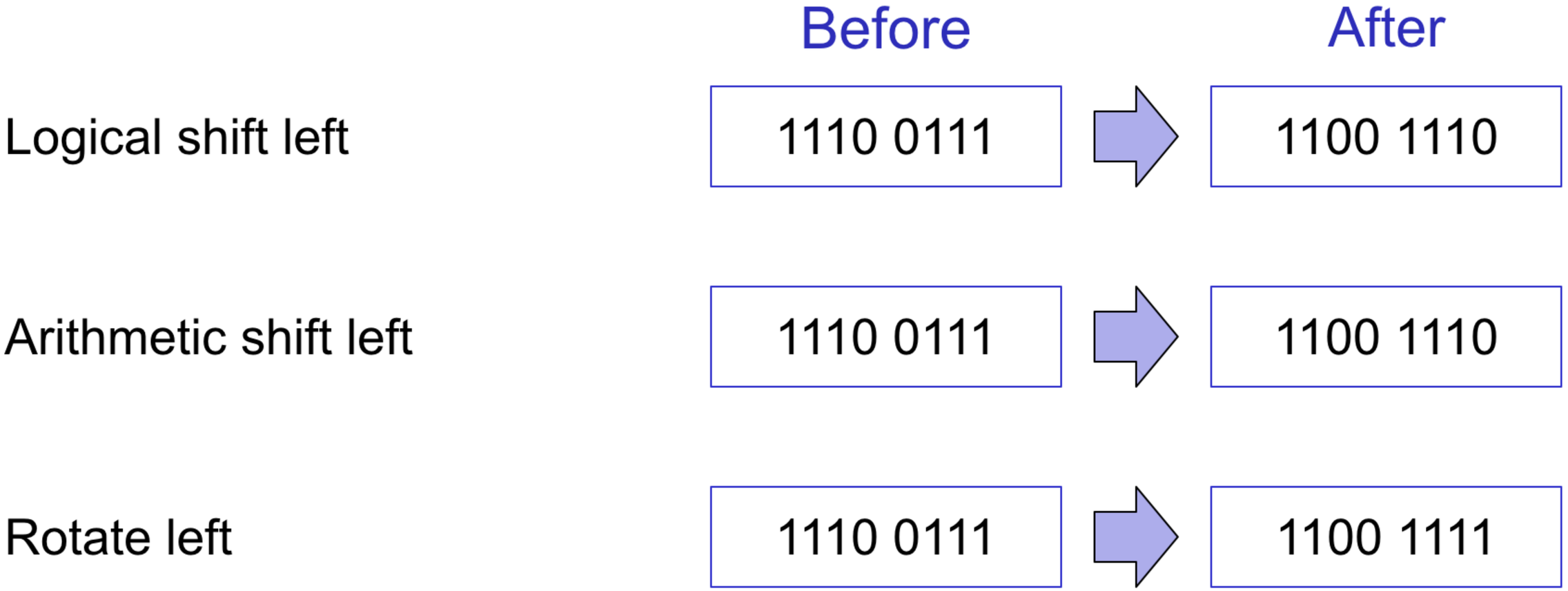
Bit Manipulation Instructions (2)

- Shift examples (to the right)



Bit Manipulation Instructions (3)

- Shift examples (to the left)



Input/Output Instructions

- Perform input and output operations
  - Transfer data from a register to an output device
  - Transfer data from an input device to a register
  - Control I/O, interrupts, etc.
- Addressing may vary
  - Direct-mapped or I/O-mapped I/O uses dedicated I/O ports to specify I/O interface locations
  - Memory-mapped I/O puts the I/O interface into the normal memory address space (like writing to or reading from memory)

Input/Output Instructions (2)

- Examples (fictitious)
  - IN PORT3,R1 – Move data value from I/O Port 3 to R1 (this is direct-mapped I/O)
  - STORE R1,A4 – Move contents of register R1 to the I/O location mapped to the memory address stored in register A4 (this is memory-mapped I/O)
  - ENBL INT – Enable interrupts for interrupt-driven I/O

CHECK POINT

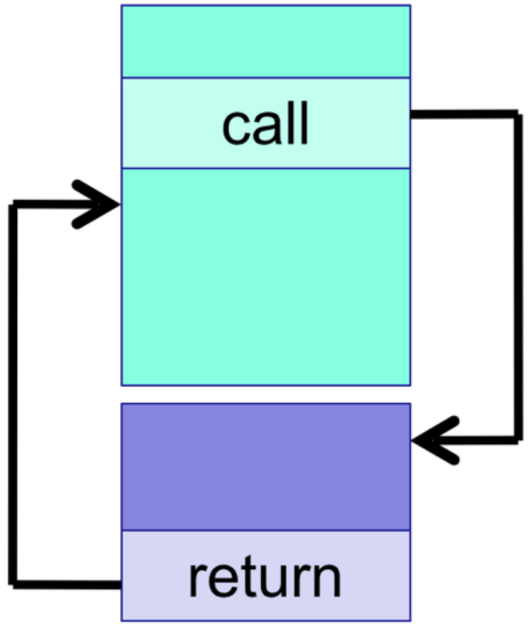
As a checkpoint of your understanding, please pause the video and make sure you can do the following:

- Describe and state a few examples of Bit Manipulation and I/O instructions

If you have any difficulties, please review the lecture video before continuing.

Transfer of Control Instructions

- Alter the instruction execution sequence
- May jump unconditionally or conditionally
- May jump or call a subroutine in a way that allows a return
  - Save return address on the stack



Transfer of Control Instructions (2)

- Examples (fictitious):
  - JMP +8 – Jump ahead 8 bytes in the instruction memory
  - JPOS A1 – Jump to the instruction location stored in register A1 if the result of the last arithmetic operation was positive
  - CALL PROC – Jump to the subroutine at location PROC and save a return address on the stack
  - RET – Return to the calling routine by using the return address saved on the stack

Special Purpose Instructions

- A “catch all” category for special instructions
  - May be tied to the specifics of a processor, memory system, or control structure
  - May be specialized
- Examples (fictitious)
  - UPPER A1 – Convert the character string pointed to by register A1 to all upper case
  - BASE #7F000000 – Set an addressing base register to hexadecimal 7F000000

Instruction Set Orthogonality

- Orthogonality with respect to instruction operations
  - Each instruction performs a unique operation without overlapping the operations performed by other instructions
- Orthogonality with respect to operations and operands
  - The same addressing modes and operand types are available for all operations, regardless of the particular instruction

CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can do the following:

- Describe and state a few examples of Transfer of Control and Special Purpose instructions
- Describe what we mean by “instruction set orthogonality”

If you have any difficulties, please review the lecture video before continuing.

Summary

- There are seven commonly used types of instructions
  - Data movement instructions
  - Arithmetic operations instructions
  - Boolean logic instructions
  - Bit manipulation instructions
  - Input/output (I/O) instructions
  - Transfer control instructions
  - Special purpose instructions
- An orthogonal instruction set is efficient (lacks redundancy) and flexible (uniformity in addressing modes available)

MODULE 6: Instruction Set Architecture

Lecture 6.2
Instruction Types

Prepared By:
- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering
Virginia Tech