Chapter 5: Two Computers and Their Programs

How do programs, written by programmers, work with the computer's hardware components to manipulate the information in the computer's memory, producing a desired outcome?

Two Computers and Their Programs

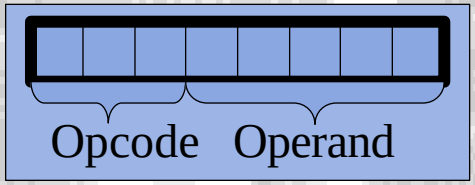
- In this chapter:
  - How do computer programs enable computer hardware to perform tasks?
  - How does the stored-program computer store both programs and the information the programs need?
  - How does the instruction set define the computer's uses and capabilities?
  - How does the CPU function?
  - How does the CPU fetch and execute programs?
  - What are some special features of electronic spreadsheet software?

Reviewing Key Concepts

- Review of a few important concepts from earlier chapters:
  - Computers are not able to do anything with the data they have stored until they are given instructions for processing it.
  - All data that computers contain and manipulate must be in binary form.
  - Computer instructions must be performed sequentially, in the order presented.
  - Every instruction in a program must have one meaning.

Reviewing Key Concepts

- Review continued:
  - Computer instructions are divided into two parts: operation code (opcode) and operand.



  - In every language, there are commands that make no use of the operand part of an instruction. (Write, Stop in Word Hunt)
  - Programs, written in any language, are translated into binary form by assigning a numeric form to each instruction, then converting each numeric value to its binary equivalent.
  - Once a program is expressed in binary form, the computer can use it directly. (Machine language program)

Two Conceptual Computers

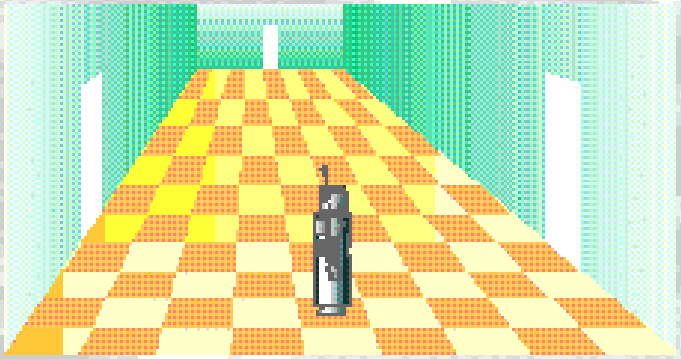
- In this chapter, we will examine two conceptual computers:
  - A robot.
  - A simplified mathematical computer.
- Why choose imaginary computers to examine how a computer works?
  - Today's real computers are incredibly complex machines.
    - Huge memories, Large instruction sets, Nearly limitless options and flexibility.
  - Using conceptual computers allows us to strip away the complexity, revealing the underlying simplicity of operation.

Two Conceptual Computers

- How is each conceptual computer similar?
  - Each is a **stored-program computer** (von Neumann computer)
    - Each contains a minimal configuration of:
      - Input units
      - Memory
      - Central processing unit
      - Output units
  - Each stores a program and the data it needs in its own memory.
  - Each executes instructions sequentially.
  - Each is designed with very limited capabilities. (small memory and instruction set.)

The ROBOT Computer: Programs and Algorithms

- Our first example of the computer: The ROBOT computer.



- The ROBOT's domain
  - The room is empty.
  - The room is rectangular.
  - There may be one or more open doorways in the walls.
  - The floor is paved with square tiles with lines between them. The lines are easy to see.
  - The size of the room is unknown to us at any given time.
  - The size of the room does not change during the execution of a program.
  - Doorways will never be located in corners.

The ROBOT Computer: Programs and Algorithms

- Hardware: Defining ROBOT Capabilities
  - ROBOT's capabilities are defined by its hardware:
    - Limitations:
      - The ROBOT has no eyes: Can't see its surroundings.
      - The ROBOT has no numerical capacity: Can't count the number of squares before it reaches a wall.

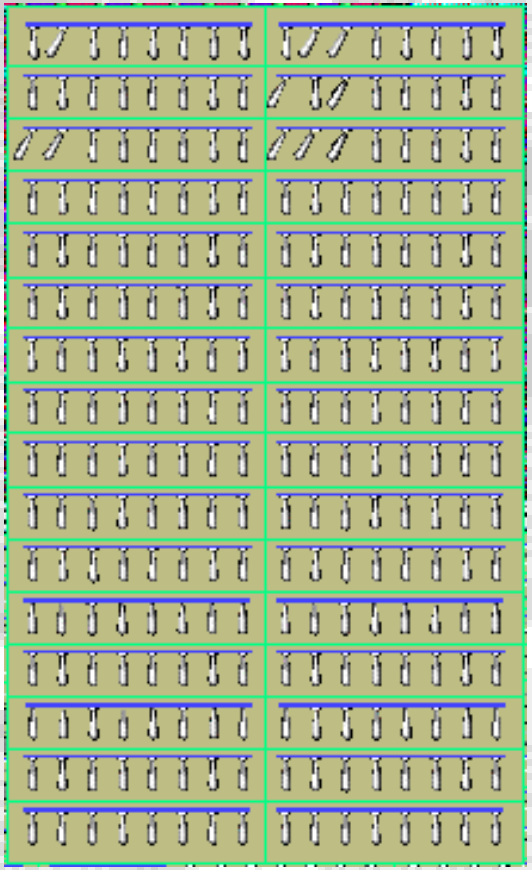
The ROBOT Computer: Programs and Algorithms

- Hardware Features:
  - Locomotion:
    - Forward motion from one square to an adjacent square within its domain (**STEP**).
    - Pivot: Only to the right, 90 degrees (**TURN**).
  - Arms (two - one at each side)
    - Can be raised (**RAISE**) and lowered (**LOWER**).
    - Extension: arms reach to the far side of the next square.
    - Sensors:
      - At tips of arms: Can sense if a wall is in front of it if arms are raised and a **SENSE** command is given.
      - ROBOT also senses a wall if instructed to RAISE its arms and a wall is directly in front of the ROBOT.

The ROBOT Computer: Programs and Algorithms

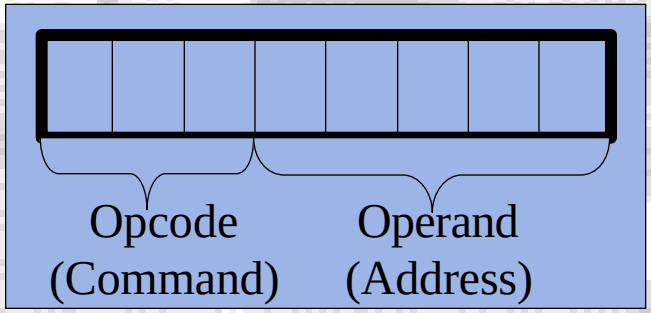
- Hardware: ROBOT's Memory:
  - Located on the ROBOT's torso.
  - 32 memory locations numbered 0 to 31.
  - Each memory location is a set of 8 toggle switches.

    On = 1 Off = 0
  - Each location is capable of storing one ROBOT instruction.
  - **Loading** a program: setting the switches.



The ROBOT Computer: Programs and Algorithms

- ROBOT Hardware:
  - **Fetch**: Electronic circuits cause it to fetch (or retrieve) instructions from memory one at a time, and usually in the order in which they are stored.
  - **Decode**: An Instruction Decoder is a set of circuits which causes the appropriate actions to be taken based on the particular binary number instruction that is received as input.
    - ROBOT instructions are split into two parts.



The ROBOT Computer: Programs and Algorithms

- Software: The ROBOT's Language.
  - The ROBOT has no intelligence. It cannot think on its own. Any instruction must come in the form of a program.
    - To "Program the ROBOT": Devise a sequence of instructions designed to accomplish some particular task.
  - The ROBOT's language consists of eight different commands. These make up the **ROBOT's Instruction Set**.

STEP	RAISE	SENSE	LIGHT
TURN	LOWER	GOTO	STOP

    - These commands are used to form instructions.
    - The ROBOT can store 32 instructions in its memory.

The ROBOT Computer: Programs and Algorithms

The ROBOT's Instruction Set

	Opcode	***The action taken by the ROBOT, in English:***
000	STEP	The ROBOT takes one STEP forward if possible.
001	TURN	The ROBOT pivots 90 degrees to the right.
010	RAISE	The ROBOT raises its arms if possible. If can't RAISE: There MUST be a wall directly in front of the ROBOT. The ROBOT's warning light comes on. *No other commands will be recognized until the light is turned off.*
011	LOWER	The ROBOT lowers its arms if they are raised.
100	SENSE	The ROBOT, with its arms in ***raised*** position, can detect if it is one step away from the wall it is facing. IF IT IS, the warning light will turn on. *Recognizes no other commands until the light is turned off.*
101	GOTO	The ROBOT takes the next command out of normal order. The **Operand**, the last 5 bits of the instruction, tells which memory location is to be performed next.
110	LIGHT	*IF the light is turned on, this command turns it off.* The ROBOT will again recognize instructions in the program.
111	STOP	The ROBOT shuts off its own power.

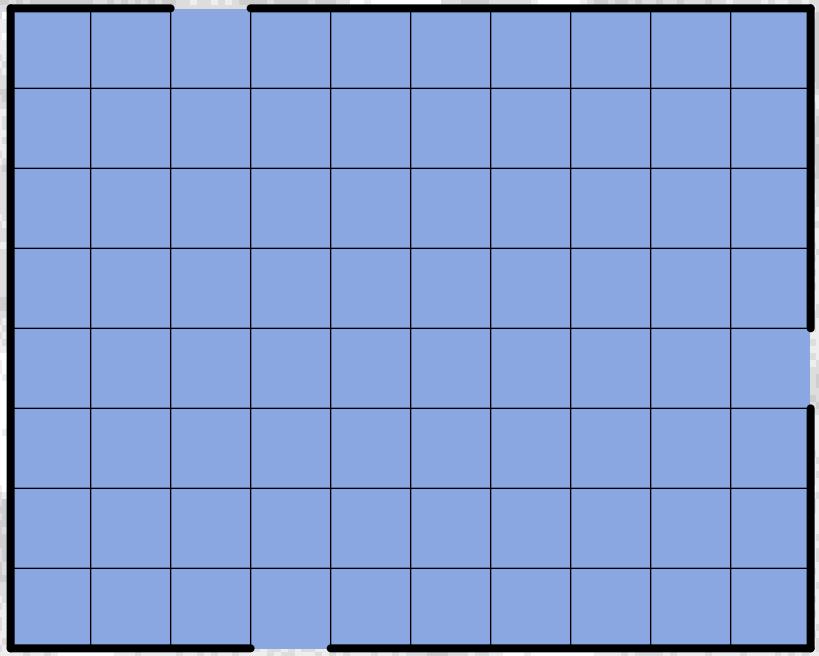
The ROBOT Computer: Programs and Algorithms

- **Problem:** Cause the ROBOT to walk to the wall it is initially facing and then stop with its arms lowered and facing against the wall. Assume the ROBOT is not initially facing an open doorway.
  - Remember: We have NO IDEA how big the room is!
    - We CAN'T just tell it to STEP X-number of times!
- First, find the Algorithm:
  - An **algorithm** is a step-by-step process used to solve a problem.
  - The solution to the problem must be general.
    - Raise arms
    - See if a wall is in front of the ROBOT.
    - Take a step.
    - Repeat until a wall is found.

The ROBOT Computer: Programs and Algorithms

- Why isn't this a "good enough" solution to the problem of finding the wall in front of the ROBOT?

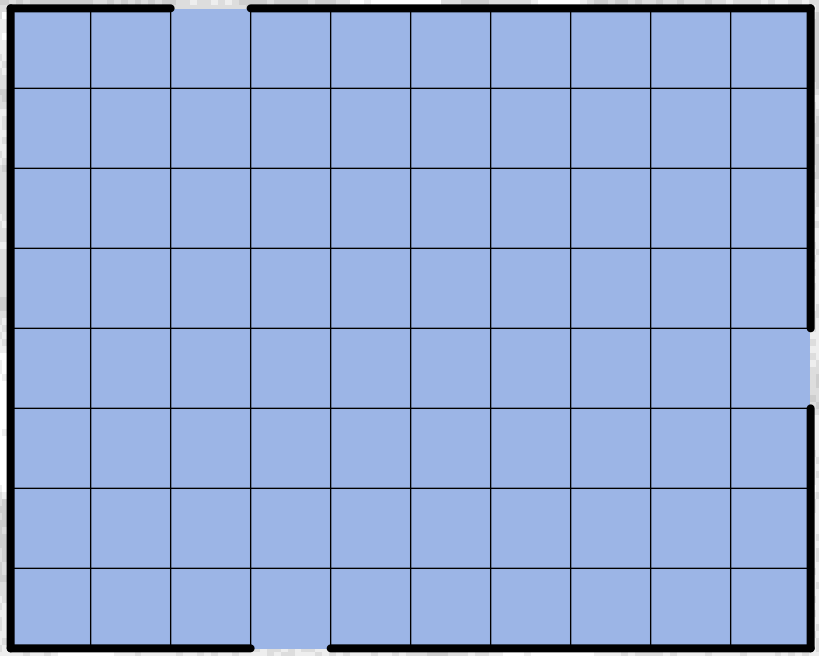
```
0  RAISE
1  SENSE
2  STEP
3  GOTO 1
4  LIGHT
5  STOP
```



The ROBOT Computer: Programs and Algorithms

- Why is this a better solution to the problem of finding the wall in front of the ROBOT?

```
0  RAISE
1 LOWER
2  STEP
3  GOTO  0
4  LIGHT
5  STOP
```



The ROBOT Computer: Programs and Algorithms

- Programming the ROBOT - Taking the “English” steps and writing them in the language the ROBOT understands (Machine Language).
  - **Machine Language** - Written in binary code, the program is in the form the computer understands.

“English” Version	Machine Language Version
0 RAISE	01000000
1 LOWER	01100000
2 STEP	00000000
3 GOTO 0	10100000
4 LIGHT	11000000
5 STOP	11100000

The ROBOT Computer: Programs and Algorithms

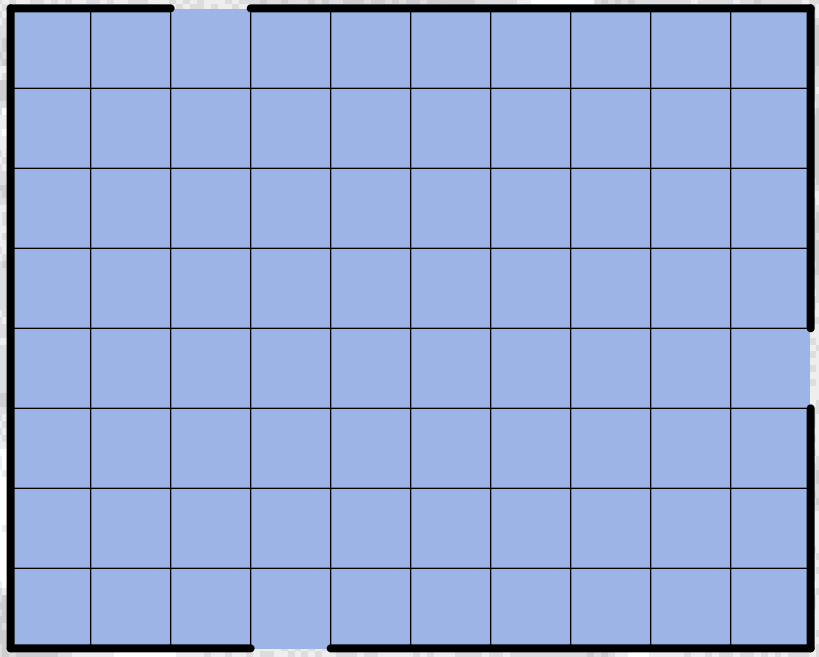
- **Loop** - A sequence of instructions which is repeated one or more times when a program is executed.

- **Infinite loop** - A set of instructions which causes the program to repeat the same commands over and over with no possible way of stopping.

The ROBOT Computer: Programs and Algorithms

- Cause the ROBOT to walk around the perimeter of the room.

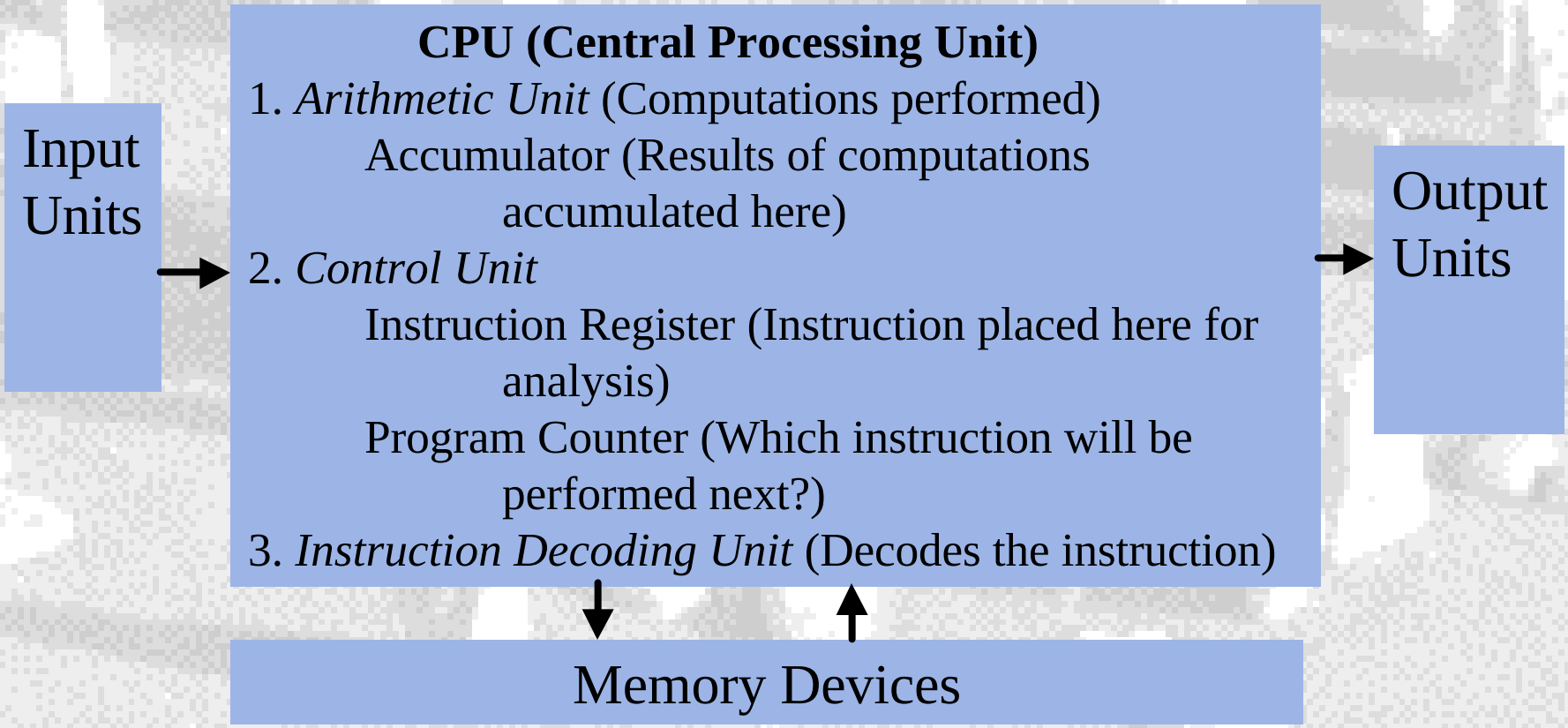
```
0  RAISE
1 LOWER
2  STEP
3  GOTO  0
4  LIGHT
5  TURN
6  GOTO  0
7  STOP
```



Does the program ever stop? What kind of loop does this program contain?

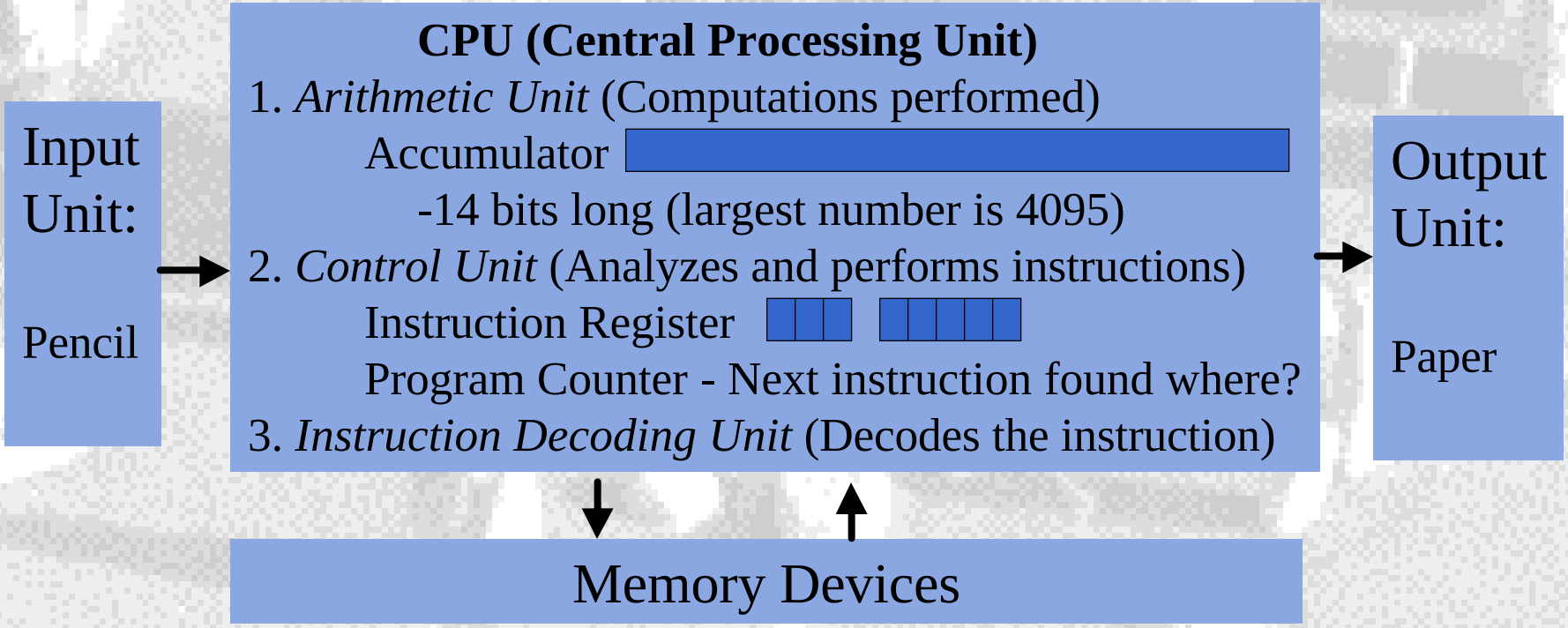
The Pencil and Paper Computer

- Parts of a typical **General-purpose Computer:**



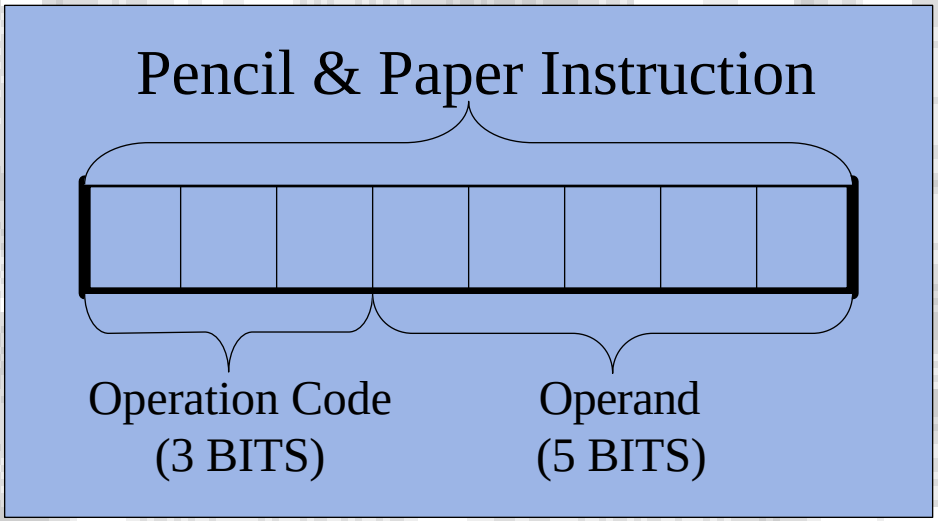
The Pencil and Paper Computer

- Parts of the **Pencil & Paper Computer** (a conceptual computer):



The Pencil and Paper Computer

- The Pencil & Paper Instruction Format:
  - **Operation Code** (Opcode) - Dictates action by Pencil & Paper Computer.
  - **Operand** (Argument) - The address of a position in memory.



The Pencil and Paper Computer

- The Pencil & Paper Instruction Set:
  - A unique set of commands to be used only when programming the Pencil & Paper Computer.

Opcode	***English***	***Action taken by command***
001	ADD	Add the contents of the referenced memory location to the value in the accumulator.
010	SUB	Subtract the contents of the referenced memory location from the value found in the accumulator.
011	LOAD	Load a copy of the value of the referenced memory location into the accumulator.
100	STORE	Store a copy of the contents of the accumulator into the referenced memory location.
101	READ	Read a value from the keyboard and store it at the referenced memory location.
110	PRINT	Print the value found at the referenced memory location.
111	PJUMP	Jump to the referenced memory location if the value of the accumulator is a positive nonzero number.
000	STOP	This causes the computer to stop execution of the program.

The Pencil and Paper Computer

- Process for writing Pencil & Paper Computer programs:
  1. Read the problem.
  2. Determine the algorithm.
     - Break the problem down into steps.
     - What will be needed as input?
     - What will be needed to be computed?
     - What is the desired output?
  3. Write an outline of the steps.
  4. Write the program.
  5. Trace through the program.

The Pencil and Paper Computer

- Problem: Write a program that sums 2 numbers read in from the "user."
- 2. Determine the algorithm.
  - Input: Read in 2 numbers.
  - Computations: Add the two numbers together.
  - Output: Print out the sum of the 2 numbers.
- 3. Outline:
  - Read in 2 numbers.
  - Add the 2 numbers.
  - Print the Sum.
- 4. Write the program.

The Pencil and Paper Computer

- Problem: Write a program that sums 2 numbers read in from the "user."
- Program:

```
READ NUM1
READ NUM2
LOAD NUM1
ADD NUM2
STORE SUM
PRINT SUM
STOP
```

Memory:	Num1	Num2	Sum

Accumulator:

Output:

Trace the program.

The Pencil and Paper Computer

- Problem: Write a program that prints the counting numbers from 5 down to 1.

```
ONE:      1
COUNT:    5
START:    PRINT    COUNT
          LOAD     COUNT
          SUB      ONE
          STORE    COUNT
          PJUMP    START
          STOP
```

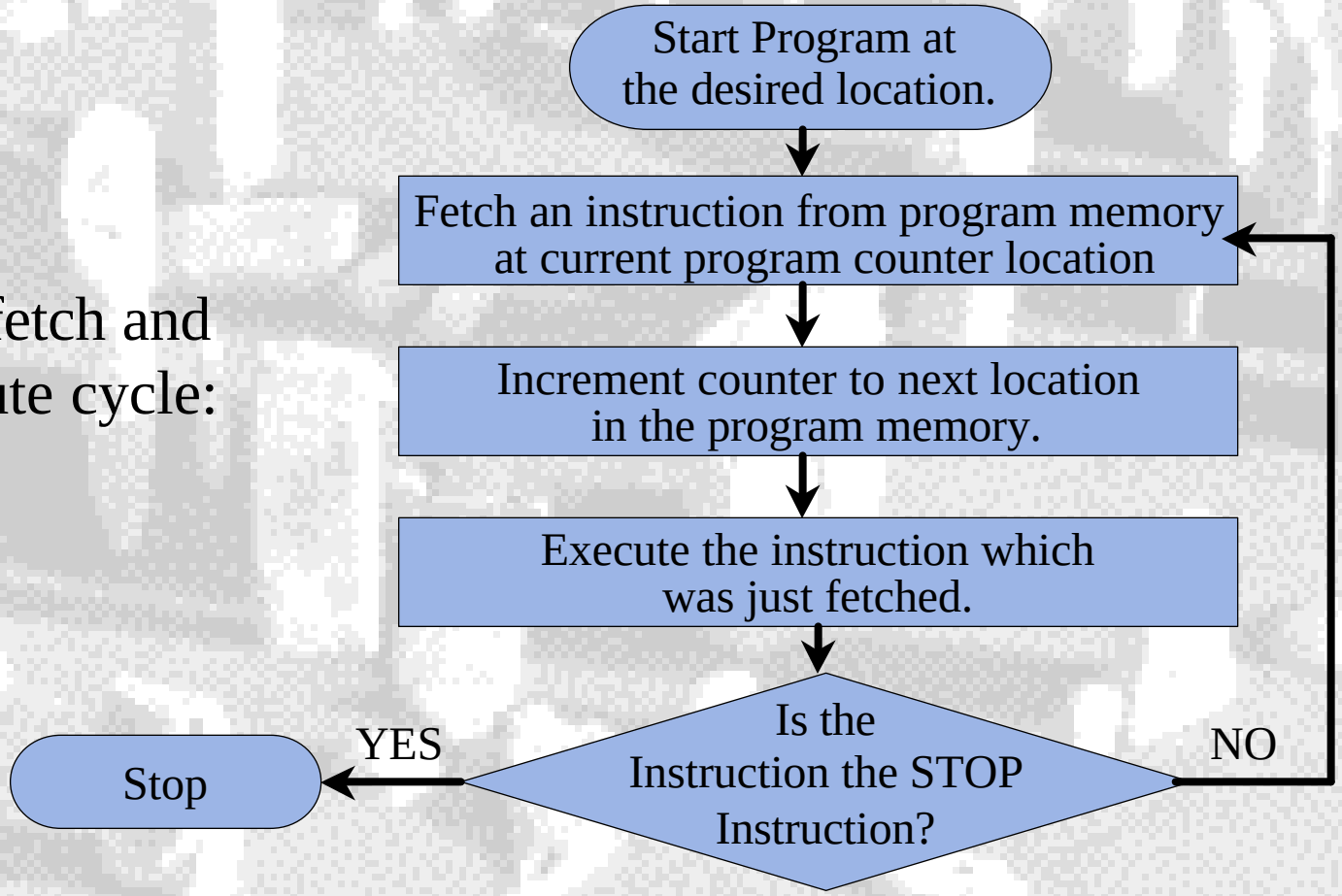
Memory: Count One

Accumulator:

Output:

The Pencil and Paper Computer

The fetch and execute cycle:



Using an Electronic Spreadsheet - Microsoft Excel

- Electronic spreadsheet:
  - A computerized ledger page (larger and more flexible).
  - Arranged in columns and rows.
  - Functions:
    - Performs calculations using any values entered on the spreadsheet.
    - Performs **instant recalculation** if any values are changed throughout the spreadsheet.
    - Shows numeric relationships in visual form.
- ***Visicalc*** - by Bob Frankston and Dan Brickland (1979)
  - First electronic spreadsheet.
  - Did pretty much everything but graph relationships.

Using an Electronic Spreadsheet - Microsoft Excel

- Why are electronic spreadsheets needed?
  - **Number crunching** -
    - The manipulation of numeric data into useful calculated results.
    - Includes mathematical operations (addition, subtraction, multiplication, and division).
    - Also includes statistical and scientific calculations of complex formulae.
  - **What-if-forecasting** -
    - Examines the effects of current policies and events on the future.

Using an Electronic Spreadsheet - Microsoft Excel

- What minimal functions must an electronic spreadsheet have?
  - Allow the user to:
    - Enter numeric values, labels (such as column headings), and formulas into specific locations on the page.
    - Perform mathematical calculations on demand.
    - Make changes (instant recalculation).
    - Print out all or part of the spreadsheet.

Using an Electronic Spreadsheet - Microsoft Excel

- What types of support programs are available to enhance the use of electronic spreadsheets?
  - Presentation programs -
    - Allows spreadsheet to be converted into a variety of different graphic forms.
    - Can present the results of the spreadsheet in color with enhanced visual design.

Using an Electronic Spreadsheet - Microsoft Excel

- Is special support hardware available for electronic spreadsheets?
  - Math co-processor (accelerator board)-
    - A second microchip that speeds up intricate calculations, data searches, and printing operations.