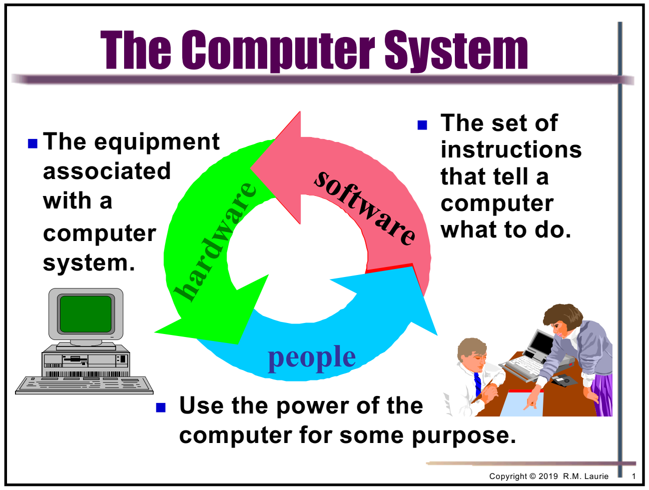
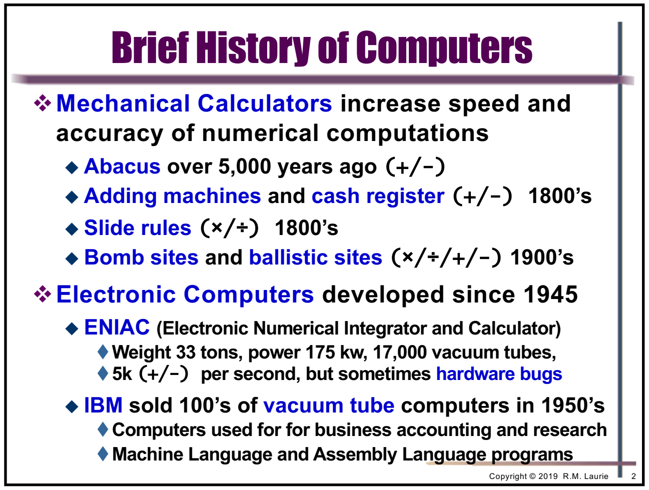


The Computer System

- The equipment associated with a computer system.
- The set of instructions that tell a computer what to do.
- Use the power of the computer for some purpose.

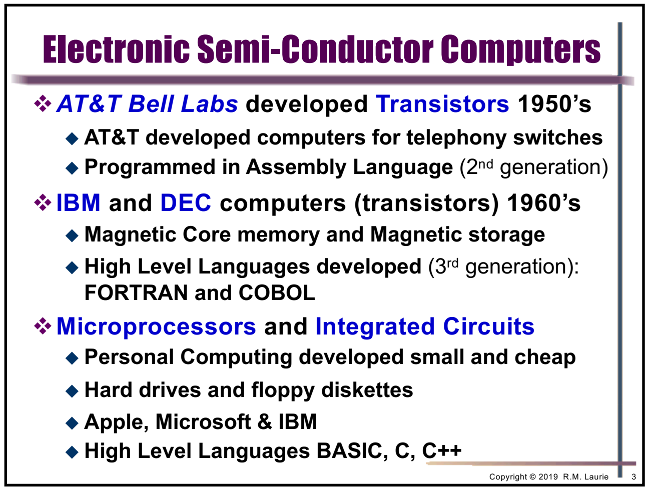
1



Brief History of Computers

- Mechanical Calculators increase speed and accuracy of numerical computations
  - Abacus over 5,000 years ago (+/-)
  - Adding machines and cash register (+/-) 1800's
  - Slide rules (×/÷) 1800's
  - Bomb sites and ballistic sites (×/÷/+/-) 1900's
- Electronic Computers developed since 1945
  - ENIAC (Electronic Numerical Integrator and Calculator)
    - Weight 33 tons, power 175 kw, 17,000 vacuum tubes,
    - 5k (+/-) per second, but sometimes hardware bugs
  - IBM sold 100's of vacuum tube computers in 1950's
    - Computers used for for business accounting and research
    - Machine Language and Assembly Language programs

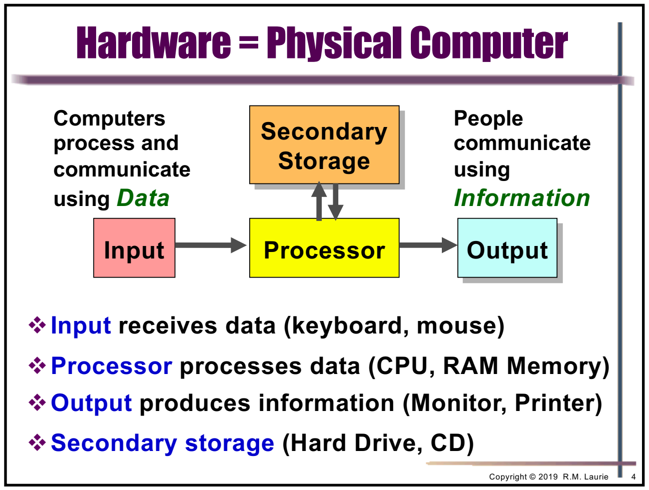
2



Electronic Semi-Conductor Computers

- *AT&T Bell Labs* developed Transistors 1950's
  - AT&T developed computers for telephony switches
  - Programmed in Assembly Language (2nd generation)
- IBM and DEC computers (transistors) 1960's
  - Magnetic Core memory and Magnetic storage
  - High Level Languages developed (3rd generation): FORTRAN and COBOL
- Microprocessors and Integrated Circuits
  - Personal Computing developed small and cheap
  - Hard drives and floppy diskettes
  - Apple, Microsoft & IBM
  - High Level Languages BASIC, C, C++

3



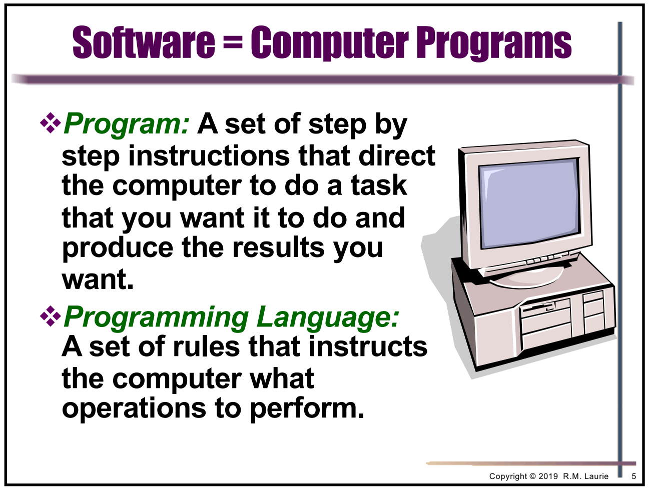
Hardware = Physical Computer

Computers process and communicate using *Data*

People communicate using *Information*

- Input receives data (keyboard, mouse)
- Processor processes data (CPU, RAM Memory)
- Output produces information (Monitor, Printer)
- Secondary storage (Hard Drive, CD)

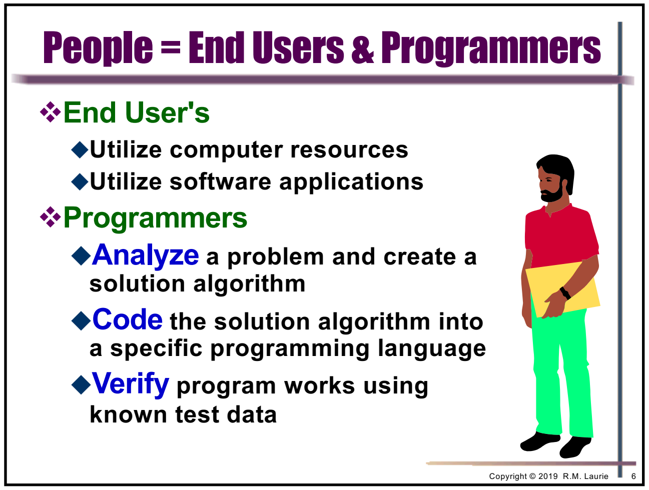
4



Software = Computer Programs

- ❖ *Program:* A set of step by step instructions that direct the computer to do a task that you want it to do and produce the results you want.
- ❖ *Programming Language:* A set of rules that instructs the computer what operations to perform.

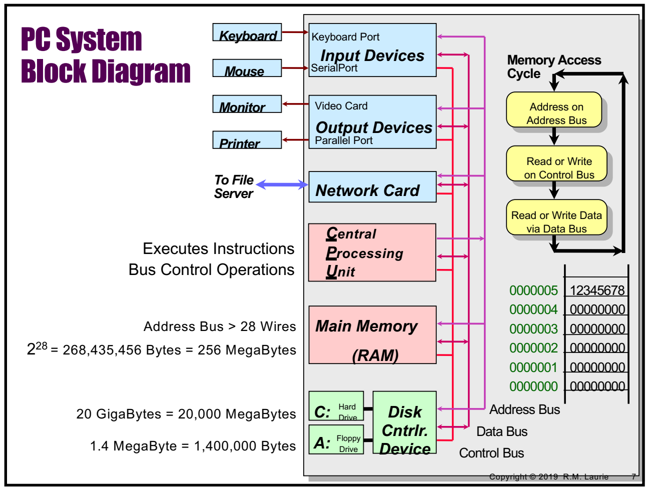
Copyright © 2019 R.M. Laurie 5



People = End Users & Programmers

- ❖ End User's
  - ◆ Utilize computer resources
  - ◆ Utilize software applications
- ❖ Programmers
  - ◆ **Analyze** a problem and create a solution algorithm
  - ◆ **Code** the solution algorithm into a specific programming language
  - ◆ **Verify** program works using known test data

Copyright © 2019 R.M. Laurie 6



PC System Block Diagram

Executes Instructions
Bus Control Operations

Address Bus > 28 Wires

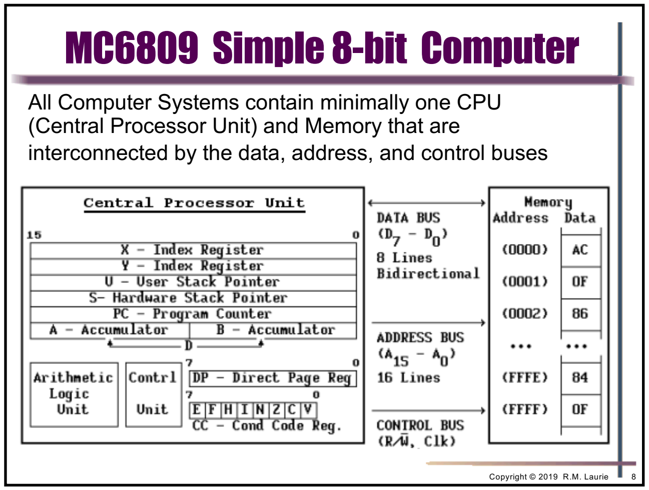
2^{28} = 268,435,456 Bytes = 256 MegaBytes

20 GigaBytes = 20,000 MegaBytes

1.4 MegaByte = 1,400,000 Bytes

0000005	12345678
0000004	00000000
0000003	00000000
0000002	00000000
0000001	00000000
0000000	00000000

Copyright © 2019 R.M. Laurie 7



MC6809 Simple 8-bit Computer

All Computer Systems contain minimally one CPU (Central Processor Unit) and Memory that are interconnected by the data, address, and control buses

DATA BUS (D_7 - D_0) 8 Lines Bidirectional

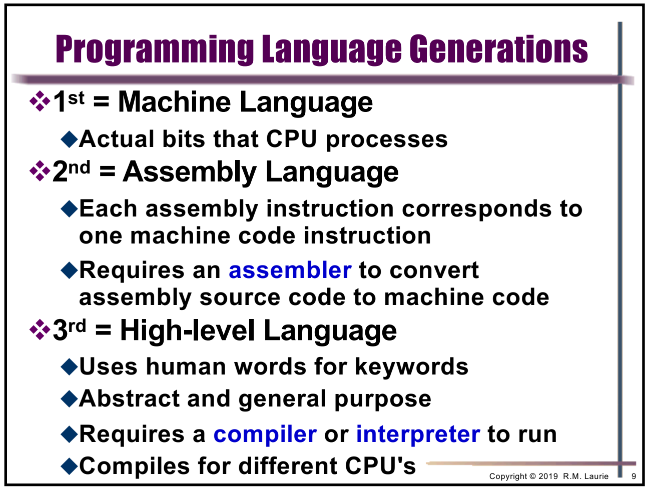
ADDRESS BUS (A_{15} - A_0) 16 Lines

CONTROL BUS ($R/\overline{W}$, Clk)

Memory

Address	Data
(0000)	AC
(0001)	0F
(0002)	86
•••	•••
(FFFE)	84
(FFFF)	0F

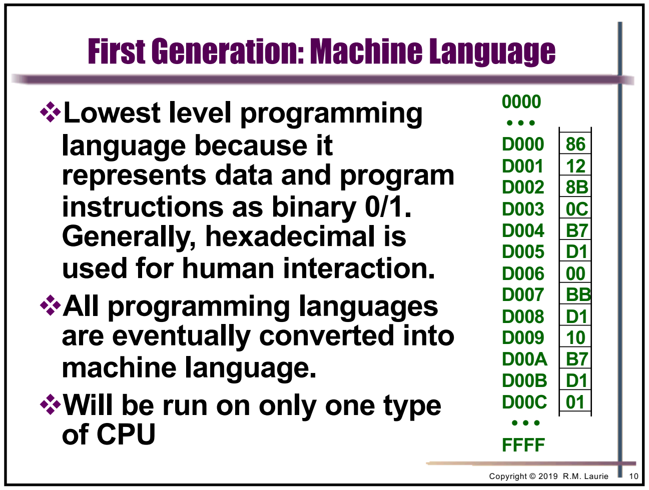
Copyright © 2019 R.M. Laurie 8



Programming Language Generations

- 1st = Machine Language
  - Actual bits that CPU processes
- 2nd = Assembly Language
  - Each assembly instruction corresponds to one machine code instruction
  - Requires an assembler to convert assembly source code to machine code
- 3rd = High-level Language
  - Uses human words for keywords
  - Abstract and general purpose
  - Requires a compiler or interpreter to run
  - Compiles for different CPU's

Copyright © 2019 R.M. Laurie 9

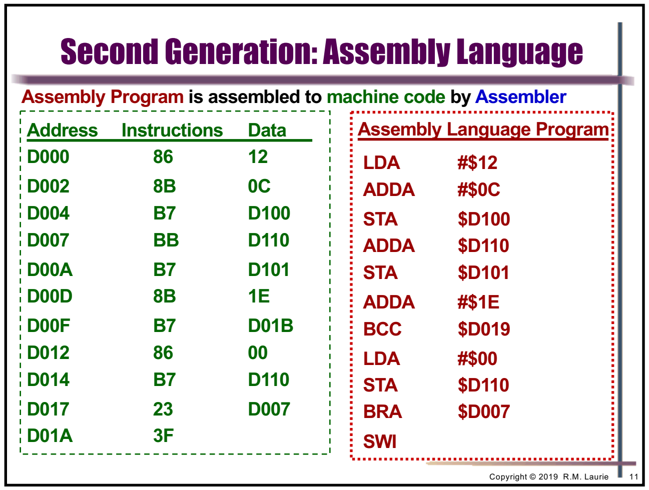


First Generation: Machine Language

- Lowest level programming language because it represents data and program instructions as binary 0/1. Generally, hexadecimal is used for human interaction.
- All programming languages are eventually converted into machine language.
- Will be run on only one type of CPU

0000	
• • •	
D000	86
D001	12
D002	8B
D003	0C
D004	B7
D005	D1
D006	00
D007	BB
D008	D1
D009	10
D00A	B7
D00B	D1
D00C	01
• • •	
FFFF	

Copyright © 2019 R.M. Laurie 10



Second Generation: Assembly Language

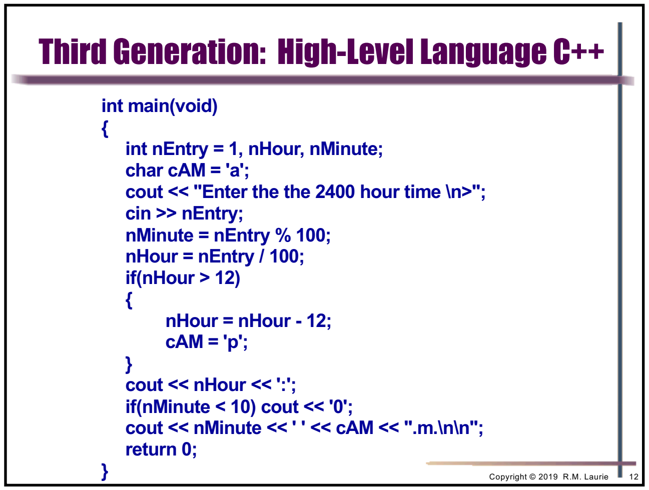
Assembly Program is assembled to machine code by Assembler

Address	Instructions	Data
D000	86	12
D002	8B	0C
D004	B7	D100
D007	BB	D110
D00A	B7	D101
D00D	8B	1E
D00F	B7	D01B
D012	86	00
D014	B7	D110
D017	23	D007
D01A	3F	

Assembly Language Program

```
LDA    #$12
ADDA   #$0C
STA    $D100
ADDA   $D110
STA    $D101
ADDA   #$1E
BCC    $D019
LDA    #$00
STA    $D110
BRA    $D007
SWI
```

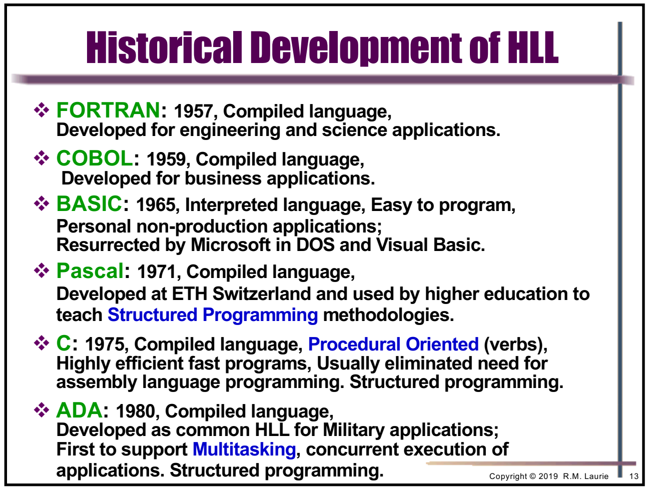
Copyright © 2019 R.M. Laurie 11



Third Generation: High-Level Language C++

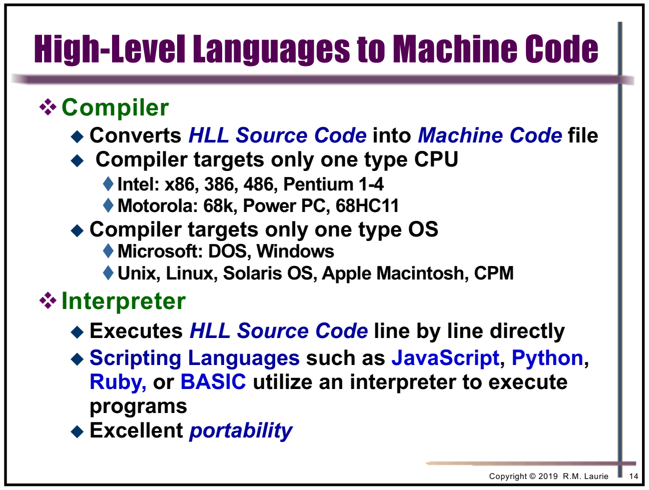
```
int main(void)
{
    int nEntry = 1, nHour, nMinute;
    char cAM = 'a';
    cout << "Enter the the 2400 hour time \n>";
    cin >> nEntry;
    nMinute = nEntry % 100;
    nHour = nEntry / 100;
    if(nHour > 12)
    {
        nHour = nHour - 12;
        cAM = 'p';
    }
    cout << nHour << ':';
    if(nMinute < 10) cout << '0';
    cout << nMinute << ' ' << cAM << ".m.\n\n";
    return 0;
}
```

Copyright © 2019 R.M. Laurie 12



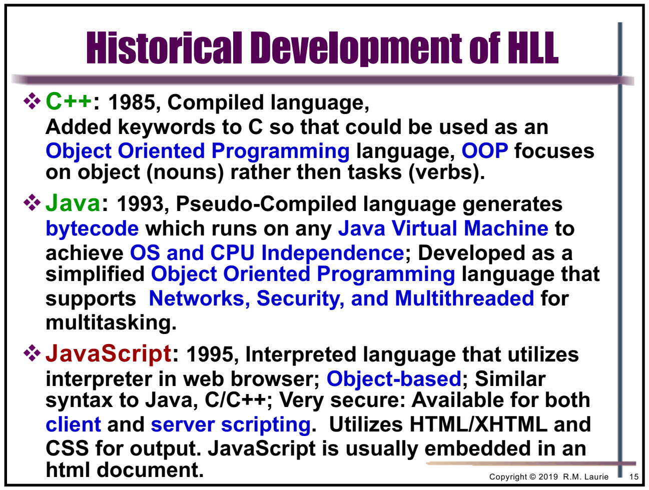
Historical Development of HLL

- **FORTRAN**: 1957, Compiled language,
  Developed for engineering and science applications.
- **COBOL**: 1959, Compiled language,
  Developed for business applications.
- **BASIC**: 1965, Interpreted language, Easy to program,
  Personal non-production applications;
  Resurrected by Microsoft in DOS and Visual Basic.
- **Pascal**: 1971, Compiled language,
  Developed at ETH Switzerland and used by higher education to teach Structured Programming methodologies.
- **C**: 1975, Compiled language, Procedural Oriented (verbs),
  Highly efficient fast programs, Usually eliminated need for assembly language programming. Structured programming.
- **ADA**: 1980, Compiled language,
  Developed as common HLL for Military applications;
  First to support Multitasking, concurrent execution of applications. Structured programming.



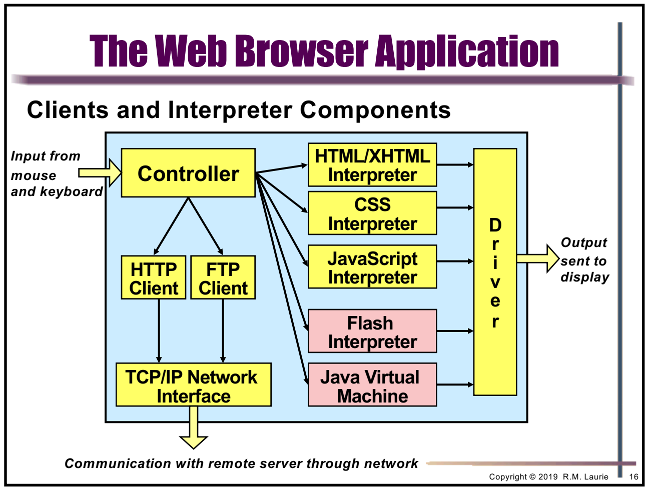
High-Level Languages to Machine Code

- **Compiler**
  - Converts *HLL Source Code* into *Machine Code* file
  - Compiler targets only one type CPU
    - Intel: x86, 386, 486, Pentium 1-4
    - Motorola: 68k, Power PC, 68HC11
  - Compiler targets only one type OS
    - Microsoft: DOS, Windows
    - Unix, Linux, Solaris OS, Apple Macintosh, CPM
- **Interpreter**
  - Executes *HLL Source Code* line by line directly
  - Scripting Languages such as JavaScript, Python, Ruby, or BASIC utilize an interpreter to execute programs
  - Excellent *portability*



Historical Development of HLL

- **C++**: 1985, Compiled language,
  Added keywords to C so that could be used as an Object Oriented Programming language, OOP focuses on object (nouns) rather then tasks (verbs).
- **Java**: 1993, Pseudo-Compiled language generates bytecode which runs on any Java Virtual Machine to achieve OS and CPU Independence; Developed as a simplified Object Oriented Programming language that supports Networks, Security, and Multithreaded for multitasking.
- **JavaScript**: 1995, Interpreted language that utilizes interpreter in web browser; Object-based; Similar syntax to Java, C/C++; Very secure: Available for both client and server scripting. Utilizes HTML/XHTML and CSS for output. JavaScript is usually embedded in an html document.



The Web Browser Application

Clients and Interpreter Components