What’s inside your computer?
Session 3

Peter Henderson
phenders@butler.edu
• **Time** – How long does it take to do something? (retrieve data from memory, execute a computer instruction, send msg NY to LA, etc)

• **Space/Size** – How much space does something take? *(Moore’s Law)* How big? (500 megabytes DRAM memory, 1 TB hard drive)

• **Speed** – How fast can something be achieved? (1 gigahertz CPU, speed of DRAM or hard drive, execute an algorithm)
VIDEO Computer Memory

1. DRAM & SRAM

2. Hard drive

3. Optical (CD/DVD)

4. Flash Drive & SSD
Time

Millisecond 1/1,000 (stopwatch accuracy)
Microsecond 1/1,000,000
Nanosecond 1/1,000,000,000
Picosecond 1/1,000,000,000,000

- Sound travels 1.3 milliseconds per mile
- Light travels 5.38 microseconds per mile
- Electric pulse thru wire 3.5 micro sec per mile
- Hard drive average seek time 5 to 10 milli sec
- DRAM (modern) access time is ~15 nano sec
- SRAM read or write time is ~1 to 5 nano sec

Latency - time delay between the cause and the effect
Space/Size

- Physical space – How many transistors in 1 square centimeter on a chip? Density!! (Moore’s Law)

- Conceptual Space/Size
  - How many bits in a byte?
  - How many data values can one byte represent?
    0000 0000, 0000 0001, ..., 1111 1111
  - How many bytes are there in 1 gigabyte of RAM memory?
Space/size

One ‘thing’

Kilo - One thousand ‘things’

Mega - One million ‘things’

Giga – One billion ‘things’

Tera – One thousand billion ‘things’

Peta – One thousand thousand billion ‘things’

The fastest supercomputer in 2016 is China’s Sunway TaihuLight running at 93 Peta FLOPS
Computer Memory – a simple visual model

Memory Addresses: 0, 1, 2, 3, 4, etc

In modern computers each memory address identifies one Byte (8 bits) of memory storage.

Here the value 7 is stored in memory address 13 (in binary 7 is 0000 0111).

Random access means the CPU can ‘access’ any byte directly, retrieve (read) or store (write).

A 32/64 bit computer reads/writes 4/8 bytes at a time – # bits used to represent integer data values.

A Gigabyte is 1,073,741,824 ($2^{30}$) bytes.
Data and Instructions are stored in memory as sequences of bits

Integers: 0, +235, -19087 [ 32 or 64 bits ]
Characters: ASCII or Unicode [ 8 or 16 bits ]
Strings: Sequence of characters (e.g., Hello!)
Decimal: 23.467, -876.3453 [ 32 or 64 bits ]
Floating Point: scientific notation

\[ 6.563092 \times 10^{15} \]

32 bits single precision - \(~7\) decimal place accuracy
64 bits double precision - \(~16\) decimal place accuracy
2016 China's 'Sunway-TaihuLight' is world's fastest supercomputer 7th year in a row

The fastest supercomputer running at 93 **Peta FLOPS**.

One thousand billion Floating Point Operations Per Second
Hierarchy of memory in a modern stored program computer

‘Registers’ – temporary storage
Static RAM – SRAM
Super high speed transistors

Cache Memory – SRAM
Very high speed transistors
(Associative Memory)

A 32 bit computer might have 32, 32 bit registers R0, R1, …, R31

Instructions & Data Locality
Several kilobytes to several megabytes

Dynamic Random Access Memory (DRAM)

Up to 32 Giga Bytes

Faster Memory

SSD, Flash Drive

Up to one Terabyte

Mechanical Hard Drives

Up to 10 Terabytes

Handout full page
Microprocessor CPU – Intel Pentium
Main Memory – DRAM

- 8 one bit DRAM chips
- 9 one bit DRAM chips - error detecting using parity check
- 12 one bit DRAM chips - Error Correcting Code [ ECC ]
Cache - very high speed memory

Registers – super high speed local temporary memory

Instruction Decoder – logic circuits

Von Neumann architecture scheme.
Expanded view of CPU – Cache memory not shown

- Control Unit
- Arithmetic Logic Unit
- Instruction Decoder
- Registers

Internal data bus
Machine Cycle (CPU Cycle)

1. Retrieve an instruction from RAM
2. Decode the instruction
3. Execute the instruction
4. Store the result in RAM

Machine cycle consists of:
Example of the operation of the CPU (sequence of computer instructions)

Compute:

\[ \text{TemporaryValue1} \leftarrow F - 32 \]

The value of \( \text{TemporaryValue1} \) becomes the value of \( F - 32 \)

FORTRAN – FORmula TRANslation Programming Language

\[ \text{TemporaryValue1} = F - 32 \]
Sequence of instructions the CPU ‘executes’ for this subtraction

1. Get the value of F from memory and write it down
2. Get the value 32 from memory and write it down
3. Subtract the value 32 from F, write down the result
4. Update the value of TemporaryValue1 in memory with the value of the result

Here, “write it down” & “write down result” means put the value into one of the registers in the CPU
Sequence of CPU instructions the CPU ‘Executes’ using the model of memory shown on the handout – RAM & registers

1. Load value from memory address 53 into R3
   LOAD R3 with value in MA 53

2. Load value from memory address 86 into R5
   LOAD R5 with value of MA 86

3. Subtract R5 from R3 storing result in R3
   SUBTRACT R3, R5

4. Store R3 into memory address 55
   STORE R3 in MA 55
Subtraction Operation

R3 <- R3-R5 in the CPU
So far the model shown includes data stored in memory; what is missing? Hint: Stored Program Computer
Store this sequence of CPU instructions in main memory (RAM) starting at byte address 11

<table>
<thead>
<tr>
<th>Addresses</th>
<th>Instruction</th>
<th># bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>11 &amp; 12</td>
<td>LOAD R3 with value of MA 53</td>
<td>2</td>
</tr>
<tr>
<td>13 &amp; 14</td>
<td>LOAD R5 with value of MA 86</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>SUBTRACT R3, R5</td>
<td>1</td>
</tr>
<tr>
<td>16 &amp; 17</td>
<td>STORE R3 value into MA 53</td>
<td>2</td>
</tr>
<tr>
<td>18</td>
<td>STOP or HALT</td>
<td></td>
</tr>
</tbody>
</table>
# RAM (256 bytes)

<table>
<thead>
<tr>
<th>Address</th>
<th>Code (hex)</th>
<th>Binary (2's complement)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>LOAD R3</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>from MA 53</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>LOAD R5</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>from MA 86</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>SUB R3, R5</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>STORE R3</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>into MA 53</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>STOP</td>
<td></td>
</tr>
<tr>
<td>53</td>
<td>50 {0011 0010}</td>
<td>0011 0101₂</td>
</tr>
<tr>
<td>86</td>
<td>32 {0010 0000}</td>
<td>0101 0110₂</td>
</tr>
</tbody>
</table>
VIDEO CPU

Fetch/Decode/Execute cycle

for a simple addition \( X \leftarrow a + b \)
History of Processor Performance