The Control Data 6600 and The IBM System/360 versus Computing in the 1960s
Significance of These Machines

- Many of the ideas in these papers initially seem trivial to computer scientists raised in the twenty-first century.
- To understand the usefulness and significance of these two architectures, we need first to put ourselves into the world for which they were created.
Discussion Outline

• Computing in the 1960s
  – Uses of Computers
  – Obstacles to Efficient Computation

• The Control Data 6600
  – Design Goals
  – Implementation of Solutions

• The IBM System/360 Solution
  – Design Goals
  – Implementation of Solutions
Computing in the 1960s

- Uses of Computers
  - Scientific computing
    - Solving increasingly difficult scientific problems as computers got faster
    - There were few large computer systems; timesharing was implemented on these systems to allow multiple users to solve their own independent problems on single machines
    - No realtime applications
  - Some extraneous military applications
    - Rarely encountered
    - Realtime processing required
Computing in the 1960s

- Obstacles to Efficient Computation
  - Transistor Technology
  - Few Standards
  - Long Wires
  - Slow Memories
  - Waiting for I/O Delayed Program Execution
  - Sequential Execution of Instructions
Computing in the 1960s

- Obstacles to Computation
  - Transistor Technology
    - Germainium transistors were the standard. This was a slow and unreliable technology, which could not withstand high temperatures.
    - Silicon transistors were a new, largely unused technology. They promised greater speed and reliability, but had not yet been proven in use.
Computing in the 1960s

- Obstacles to Computation
  - Few Standards
    - It was rare for any machine code to be able to run on multiple types of machines
    - Different machines were designed for different purposes and thus had different hardware
      - number of floating point units
      - width of a word
      - other specialized hardware
      - etc.
    - Machine code had to be tailored to the capabilities of the machine it would be run on, as well as to that machine's interface
Computing in the 1960s

- Obstacles to Computation
  - Long Wires
    - Technology allowed electrical signals to travel through wires at around $\frac{3}{4}$ light speed.
    - Since wires were so long, the computation itself (time spent performing transistor operations) accounted for only 90% of the total compute time. The other 10% was due solely to the information's travel time through the wires.
Computing in the 1960s

• Obstacles to Computation
  – Long Wires

  • There existed (and still exists) a tradeoff between wiring length and logical flexibility
    – Longer wires allow for more room to create different logical operators
    – Longer wires also allow for greater heat dissipation because operators are more spread out
    – However, longer wires increase wiring delay, making computation slower
Computing in the 1960s

• Obstacles to Computation
  – Slow Memories
    • Memories were positioned far away from processing units. This caused slow access due to increased wire time delays.
    • Core memory was the fastest
    • Secondary storage was based on physical magnetic technology, resulting in slow single block access times
    • Secondary storage was quicker when delivering multiple contiguous blocks at once
    • These problems still exist today; they are being dealt with by putting more cache on chip, and by developing flash technology for secondary storage
Computing in the 1960s

- Obstacles to Computation
  - Waiting for I/O Delayed Program Execution
    - I/O was slow, and programs often had to wait for it to complete in order to perform their next instruction, since the CPU was utilized by the I/O subsystem to perform the I/O operations
    - This resulted in programs spending a lot of time waiting for the CPU, and the CPU spending a lot of time working on menial I/O tasks
Computing in the 1960s

- Obstacles to Computation
  - Sequential Execution of Instructions
    - Programs tended to be written as a series of instructions to be executed in the order in which they were defined.
    - If each instruction can only be executed after another instruction has completed, then speed is limited to the sum of the time it takes the hardware to execute each instruction.
    - If instructions could be executed in parallel, then the delays due to multiple instructions could be concurrently incurred, allowing total wall execution time to be reduced even though the same amount of compute time were used.
The Control Data 6600
The Control Data 6600

- Design Goals
  - Make use of new Silicon transistor technology
  - Decrease memory access time
  - Increase parallelism within the CPU
  - Allow utility (such as I/O) and housekeeping work to take place outside of the critical path
  - Offer specialized hardware, which can be utilized by the operating system outside of the critical path
  - Allow multiple machines to utilize the same resources, and thereby communicate easily
The Control Data 6600

• Implementation of Solutions
  – Did away with Germanium transistors in favor of newly realized Silicon transistors.
  • This increased speed, reliability, and allowable operating temperature.
The Control Data 6600

• Implementation of Solutions
  – Increased Parallelism
    • Removed utility and housekeeping functions from the critical path by allowing the CPU to delegate responsibilities to peripheral processing units (PPUs).
    • Each PPU had its own storage, in which it would store the code it executed
    • The PPUs had access to central storage
    • The CPU and each PPU could communicate by shared memory in central storage and by interrupting each other's execution with jump commands
    • The PPUs still used accumulators
    • Only the OS could control the PPUs so it was not necessary to protect memory from their programs
The Control Data 6600

- Implementation of Solutions
  - Specialized Hardware
    - By utilizing 10 peripheral processing units, each with access to the machine's 12 communication channels, the Control Data 6600 was able effectively to fabricate specialized hardware on the fly.
    - The effective operation of these units was similar to what we now do with the PCI (express) bus.
      - For example, the CPU can program and interrupt the graphics card's specialized hardware
      - Also, networking cards can send signals to the CPU which result in traps interrupting its execution
The Control Data 6600
The Control Data 6600

• Implementation of Solutions
  – Increased Parallelism
    • Removed some utility and housekeeping operations from the critical path by allowing the operating system, running on the CPU, access to multiple peripheral processing units
    • Used multiple registers in the CPU instead of a single accumulator. This allowed for memory access to be taken out of the critical path, and for multiple, often used values not to have to be swapped directly to memory per use. Since the PPU's were simpler, cheaper processors than the CPU, they continued to use accumulators instead of registers
    • Used a scoreboard to allow for ILP and time-sharing to gather more instructions for the CPU in order to make greater use of the CPU's operator hardware
The Control Data 6600

12 Peripheral Channels → 10 Peripheral Processors → Central Storage

24 Registers → 10 Functional Units

Instruction Stack → Scoreboard

Central Processor
The Control Data 6600

- Implementation of Solutions
  - Decreased Memory Access Time
    - Utilized multiple registers to parallelize access to memory and remove it, in some cases, from the critical path
    - Designated a set of registers to deal with operands, and a several sets to deal with swapping register values in and out of central memory
The Control Data 6600
The Control Data 6600

- Implementation of Solutions
  - Decreased Memory Access Time
    - Separated memory into a two level hierarchy in which the first level, Central Storage, was significantly faster than the second, Extended Core Storage
    - All processors had access to both levels of memory
  - Separated memory into 32 different banks arranged such that sequential access could be parallelized between the banks
  - Used a specialized memory address translator called the STUNT box to determine where to go to fetch a requested word
The Control Data 6600

- Implementation of Solutions
  - Made resources available to multiple machines
    - External devices such as printers could be connected to the communication channels of multiple devices
    - Extended Core Storage could be accessed by multiple machines
The Control Data 6600
The Control Data 6600

- Goals Realized
  - Made use of new Silicon transistor technology
  - Decreased memory access time
  - Increased parallelism within the CPU
  - Allowed housekeeping work to take place outside of the critical path
  - Offered specialized hardware, which could be utilized by the operating system outside of the critical path
  - Allowed multiple machines to utilize the same resources, and thereby communicate easily
The IBM System/360
The IBM System/360

- Design Goals
  - Utilize new Silicon transistor technology
  - Implement instruction level parallelism
  - Standardize machine architecture
    - Use a single machine architectural interface for different machines of differing levels of power (up to 50x performance range factor).
    - Any program that can run on any machine should be able to run on all other machines of the same class.
    - Programs may be tailored to run best on one machine or another.
    - Machines may be tailored to be more powerful for some applications, or to be less costly to construct.
**The IBM System/360**

- **Design Goals**
  - Upgrade storage (memory) to accommodate large capacities, greater speed, and different hierarchies of speed
  - Ensure that programs can be relocated in memory in a simple manner
  - Standardize I/O interface
  - Parallelize I/O when possible
The IBM System/360

- Implementation of Solutions
  - Used Silicon transistors instead of Germanium
The IBM System/360

- Implementation of Solutions
  - Standardized architecture
    - Created five different models, each with different capabilities, and a performance increase factor of 50x between the lowest and highest performers
    - Each model could run machine code generated for any model
    - Models had different levels of hardware; cheaper models used software to make up for any hardware they were missing
      - For example, some models had hardware floating point units, and some used software to emulate them
The IBM System/360

• Implementation of Solutions
  – Standards
    • Fixed 8-bit byte (no 4 or 6 bit option)
    • 8-bit character width, which allowed for ASCII or BCD
      – each CPU was designed with a "high degree of code independence [and] generalized code translation facilities" so that it could accommodate ASCII or BCD according to the user's preference
The IBM System/360

- Implementation of Solutions
  - Standards
    - Fixed 32-bit word length
      - Allowed the use of double-words and 4-bit nibbles as well; programmers could use higher precision for programs that were intended to be run on more powerful systems
      - Systems could hardware implement or software implement operations with integers and floats of lengths other than 32
      - All systems could emulate all hardware so that they could run any program for any system
    - Variable field length
      - Allowed for fields to be of variable length, but required that all fields start at a byte address which is a multiple of their length, avoiding field overlapping problems between systems in which different integer lengths were being used
The IBM System/360

- Implementation of Solutions
  - Standards
    - Byte-level addressing (as opposed to word level) allowing for increased flexibility
    - Binary addressing (as opposed to decimal)
      - Translation could occur in compiler
The IBM System/360

- Implementation of Solutions
  - Upgraded storage capacity and speed
    - Used 24-bit memory addresses
    - Implemented a hierarchy of different speeds of memory
    - Used multiple registers to allow for access to storage to occur less frequently
    - Could utilize the flexible I/O system with external storage for even greater capacity at lower speed
The IBM System/360

• Implementation of Solutions
  – Simplified Program Relocation
    • The only way to address a piece of memory was through a base register and an offset. This allowed for a simple method of relocating an entire program: just change the base register value
    • Because the displacement was given only 12-bits in which to be defined, this limited a specific program to 4096 Bytes of data
The IBM System/360

- Implementation of Solutions
  - Standardized I/O interface
    - Utilized channels of multiple speeds to connect to external devices
      - byte-multiplexer channels connected to slower devices such as printers
      - selector channels connected to faster devices such as external storage devices
    - Allowed multiple devices and device types to connect to individual channels
    - Allowed the CPU to specify some code for use in controlling a channel
The IBM System/360

• Design Goals
  – Parallelized I/O in certain models
    • Similar to the situation in the 6600, in larger versions of the System/360, the CPU was not needed to deal with I/O, since specialized processors were used to control the channels
    • Channel control processors had their own access to main memory, so they could deal with I/O without using the CPU
    • In smaller versions of the System/360, the CPU could be used in place of specialized control processors; this made channels a complete abstraction, which could be quickly or cheaply implemented on different systems
Other 6600 and System/360 Features Not Usually Seen before the 1960s

- CRT computer screens used instead of line printers
References

- J. Thornton, Considerations in Computer Design – Leading up to the Control Data 6600
- J. Thornton, Design of a Computer -- The Control Data 6600
- G. Amdadhl, et al., Architecture of the IBM System/360
- Slide 37 Image: http://upload.wikimedia.org/wikipedia/commons/9/9e/360-91-panel.jpg