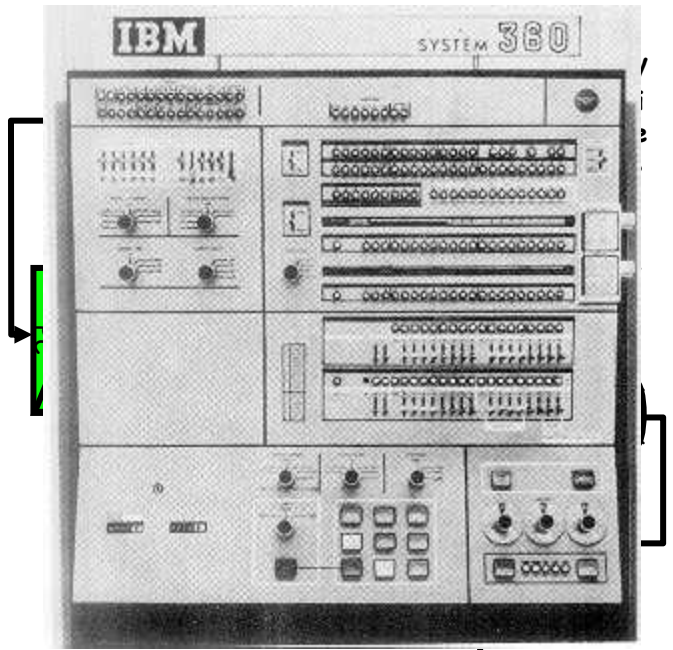


What Do Bell Bottoms, Peace Signs, and Computer Architecture Have In Common?

Jim Larus

Microsoft Research

Quiz Time



Created in the...

1960's

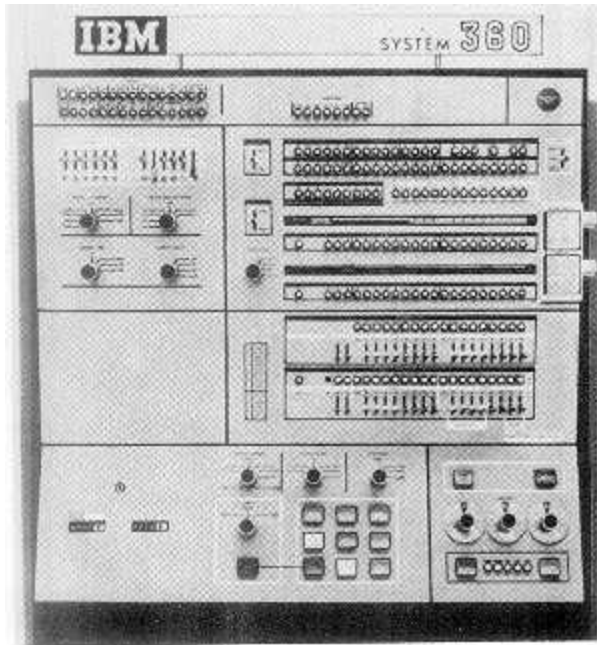
A Few Changes Since Then...

- Processor speed (300x)
 - 10 MHz (IBM 360/91) → 3 GHz (x86)
- Memory size (32,768x)
 - 32 KB → 1 GB
- Transistors count (170,000x)
 - 10K-100K (IBM 360/91) → 1.7B (Itanium Montecito)
- OS size (1,000x)
 - 100 KLOC → 100 MLOC
- Internet size (100,000,000x)
 - 4 (1969) → 439 million computers
- Internet Users (25,000,000x)
 - 40(?) → 1.1 billion people

A Few Changes Since Then...

- Security and reliability more important than performance
 - Computers ubiquitous and networked
 - Users untrained and uninterested
- Software increasingly written in modern high-level languages (not assembler!)
 - Java, C#, Perl, Python, Ruby, ...
 - C/C++ is language for 1960's architectures

Is There a Difference?



Wild & Crazy Idea

- Computer architectures should provide first class support for modern programming languages and modern software requirements (security & dependability)
- Even crazier idea:
 - No paper in ISCA '10 should contain more than one table or chart with performance measurements
 - Use the space to evaluate correctness and dependability

Fast Forward to 1980

- Dave Ditzel puts the nail in the coffin of high-level language computers
 - Clears the way for RISC
- He was right
 - Bad way to get good performance
 - Not enough transistors
 - Some truly bad ideas (compiler in hardware???)

Maybe HLL Support Isn't a Bad Idea

- George Necula (UCB) CCured project
 - Safe (memory & type) dialect of C
 - Bring C into the 1970's
 - Eliminate entire classes of (serious) bugs
 - Port existing code (don't rewrite)
- But, memory layout changes
 - Type information (for downcasts)
 - Array bounds information
- Where does this information go, and still maintain low-level compatibility?
 - Why can't HW track this information?

Another Example

- Emery Berger (UMass) & Ben Zorn (MSR)
DieHard Project
 - Infinite heap semantics
 - Objects never deallocated
 - Objects allocated infinitely far from each other
- Berger & Zorn approximated with probabilistic memory safety
 - Randomly place objects in much larger heap
- Why can't HW give us the real thing?
 - Why do we need a linear address space?

Singularity Example

Webfiles Macrobenchmark: Relative Cycles Unsafe Code Tax

