HLL Architectures: An overview
What is an HLL architecture?

• An architecture that runs a high level language.
II. An HLL architecture maps concepts present in HLLs to hardware.

• Why?
  i. Error reduction
  ii. Speed (?)
  iii. Easier assembly-language programming
  iv. Program size reduction

• Example Features: Tagged storage, array descriptors, capability-based addressing[1], hardware garbage collection[5]

• Implementations: SWARD, iAPX, HP FOCUS
III. An HLL architecture allows more-or-less direct execution of some HLL.

• Advantages
  i. Speed (optimize for common operations)
  ii. Ease of compilation (machine code is relatively similar to HLL)
  iii. Ease of debugging (HLL is relatively similar to machine code)
  iv. Ease of low-level programming (since HLL is the machine’s lowest level language)

• Disadvantages
  • Language specific
  • Either extensive microprogramming or complicated hardware
  • Having every operation in hardware can reduce performance of all operations
Imperative languages

• Examples: SYMBOL, SPL, APL microcode assists [1]
• Advantages, if any, come from speed (compilation, or APL interpretation, is simple), and otherwise as definition II.
Highly symbolic languages

• Examples: Lisp machines, Snobol machines

• Efficient operations are provided for the basic operations of these languages
  • Lisp: quick list allocation, deallocation, type checks, cons, car, cdr [2](note: quick cons, car, cdr are of course perfectly easy on traditional architectures) [3] space-efficient list allocation, and quick function calls [5]
  • Snobol: quick pattern matching [4]
Functional(single-assignment) languages

• These admit either of sequential or data-flow implementation
• Sequential implementation is not interesting enough to discuss here
Data-flow languages

• Programs can be translated into a graphical dataflow representation
• Consider quadratic formula, a, b, c inputs
• \( \text{Out} := \frac{((\sqrt{(b*b) - (4*a*c)}) - b)}{2*a} \)
Data-flow languages

This is the representation used by Myers, pp. 467-474

{address} Operator(immediate value) [destination]
1) Multiply [2]
2) Multiply-immediate(4) [5.a]
3) Multiply-immediate(2) [8.b]
4) Multiply [5.b]
5) Subtract [7]
6) Subtract [8.a]
7) Sqrt [6.a]
8) Divide [Out]
Characteristics of a dataflow architecture

• Programs can be translated into data-flow representation
  (this will be familiar: it is employed on a small scale in out-of-order execution schemes)
• A PC can be dispensed with ("Should it be?" is a different question)
• Promises very high parallelism
• Loop, branch implementation is complex
• Much hardware required to implement (any instruction may be executed if its operands become available)
Why did HLL architectures fail?

• Includes most of the same reasons CISC failed
• Software compilers got better.
• More programming languages reaches a wider audience
• Interpreted Languages
Bibliography

2. David A. Moon, Architecture of the Symbolics 3600, Proceedings of the 12th annual international symposium on Computer architecture, p.76-83, June 17-19, 1985, Boston, Massachusetts, USA