

Advanced Computer Systems Architecture

Chip-Multiprocessors: Applications and Architectures

CSE 526M

Prof. Patrick Crowley

Plan for Today

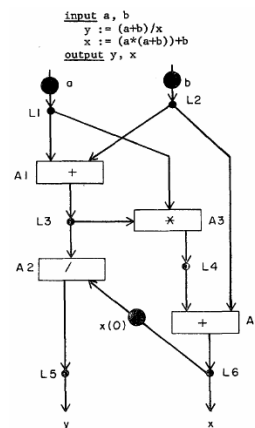
- Announcements
 - Office hours end at 3:45 today
- Questions
- Today's discussion

Data Flow

- The idea
 - Achieve concurrent program execution with a processor organization that allows an instruction to execute as soon as its operands are available
 - In contrast to the implied serial execution of traditional computers
- Most data flow proponents used new programming languages devoid of serial baggage
- A formal modeling technique, based on petri nets, inspired these ideas and can be used to derive formal properties of such systems

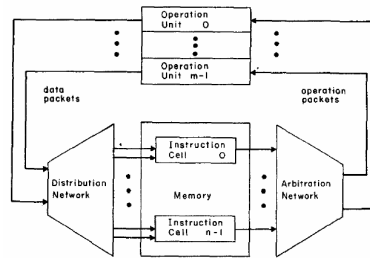
Sample Data Flow Program

- Nodes are operators or links (large dots = init vals)
- Arcs transmit values, represented by *tokens*
- A node is enabled only when all its inputs contain tokens
- A node may *fire* when it is enabled; tokens are removed from inputs, and one is placed on the output arc
- Note: no control flow



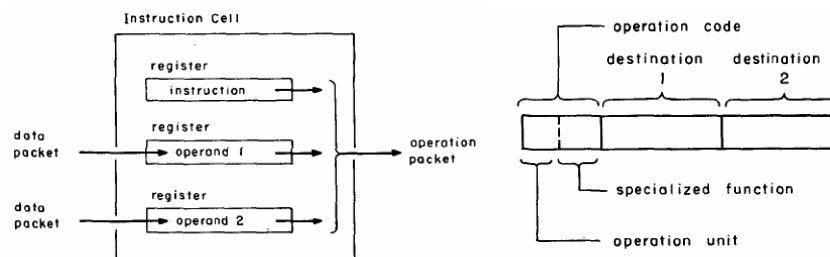
Elementary Processor

- Data flow program stored in *instruction cells*
- Arb network routes instruction cells to operation units
- Dist network routes results to memory

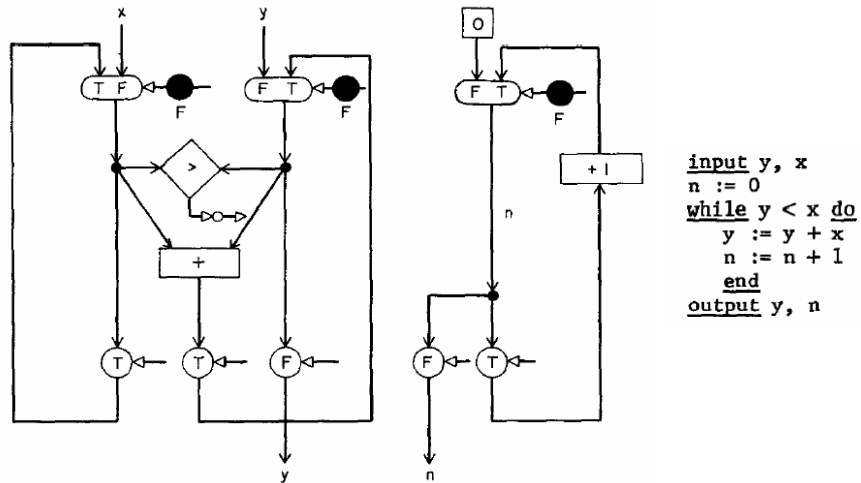


Memory Format

- An instruction cell is enabled when each of its three registers are full

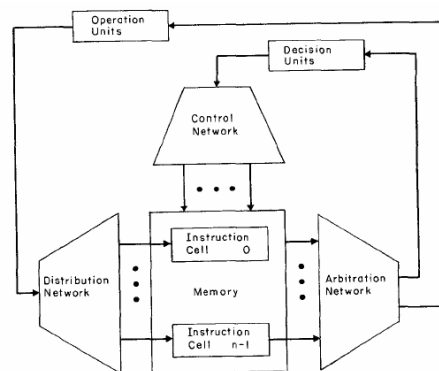


Basic Data Flow Program



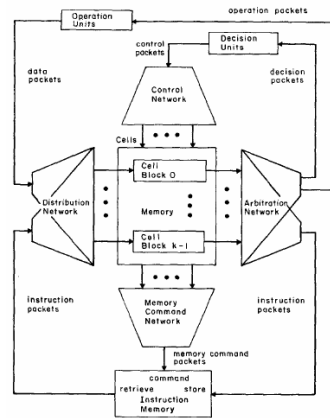
Basic Processor

- Structure is essentially the same
- Packet formats have same character
- Number of instruction cells is a limitation



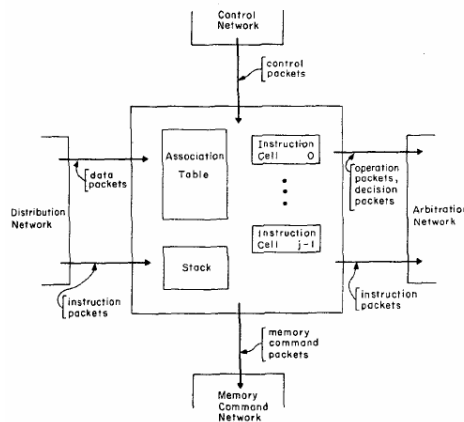
Basic Processor with Memory Hierarchy

- Auxiliary memory allows larger programs
- Cell block must now perform a sort of caching and demand fetching



Cell Block Structure

- There are more memory locations than instruction cells
- Assoc table records that status of each cell
- Stack maintains displacement order



Intellectual Digestion

- What difficulties do you see with using data flow ideas in a system?
- Most data flow examples focus on concurrent computations. Can data flow ideas help tolerate slow memory or fast I/O? If so, how?
- Does the presence of multiple threads increase or decrease the utility of a data flow organization? Why?

Difficulties

- Data flow-friendly languages never achieved critical mass (serial, control flow baggage must be tolerated)
- Serial execution lends itself to step-through debugging; how do you debug a highly concurrent parallel computation?

Data Flow vs. Slow Mem and Fast I/O

- Slow memory
 - Execution can proceed past memory ops whenever independent instructions exist (indirect benefit)
- Fast I/O
 - Perhaps, but no one has done anything yet

Threads and Data Flow

- I think there is merit in considering thread dispatch mechanisms inspired by data flow ideas, but no one has looked at it yet

Assignment

- Thursday (4/22)
 - **Commentary:** Architecture and Applications of the HEP Multiprocessor Computer System
 - 15 minute perspective: TOE
- Tuesday (4/20):
 - **Commentary:** Reflections in a Pool of Processors/An Experience Report on C.mmp/Hydra
 - 15 minute perspective: Bloom Filter