Thinking Machines Corporation's Connection Machine CM-2

- Thinking Machines Corp. founded 1983
  - CM-1, 1986 (1000 MIPS, 4K mem / proc)
  - CM-2, 1987 (2500 MFLOPS, 64K…)
- SIMD, distributed memory, bit-serial
- Programs run on one of 4 front-end processors, which issues instructions to the Parallel Processing Unit (PE array)
  - Control flow and scalar operations run on front-end processors, parallel operations run on PPU
  - A 4x4 crossbar switch (Nexus) connects the 4 front-ends to 4 sections of the PPU
  - Each PPU section is controlled by a sequencer, which receives assembly language (Paris) instructions and broadcasts micro-instructions to each processor

CM-2 Processors

- 32 processors are implemented by 2 custom processor chips, memory chips, and 2 floating-point accelerator chips
- Processor chip (contains 16 processors)
  - Contains ALU, flag registers, etc.
  - Contains NEWS interface, router interface, and I/O interface
  - 16 processors are connected in a 4x4 mesh to their N, E, W, and S neighbors
- RAM memory
  - 64Kbits, bit addressable
- FP acceleration (2 chips)
  - First chip is interface, second is FP execution unit
  - 2 chips serve 32 processors

CM-2 Processor Chip

- 16 flag registers
  - 8 bits for general purpose use
  - 8 bits predefined
    - NEWS flag — accessible by neighbors
    - 2 flags for message router data movement and handshaking
    - Memory parity flag
    - Flag for daisy-chaining processors
    - Zero flag (hardcoded)
    - 2 diagnostic flags
- Instruction
  - All processor chips receive the same instruction from the sequencer
    - Individual processors may be masked using a flag bit for processor activation
  - Produces outputs based on memory / flag lookup table (256 possible functions)
  - Arithmetic is bit-serial

CM-2 Interconnect

- The 16-processor chips are linked by a 12-dimensional hypercube
  - Data-parallel message passing
  - Message-combining hardware: receive bitwise OR, numerically largest, or sum
- Simple communication
  - Front-end system broadcasts a value to all PEs, context flag in each PE controls who receives it
- Parallel communication
  - NEWS grid —shift fixed amount
Send, Get — arbitrary processors
Spread — spread data through PEs
  - Sort
  - Reduce and broadcast — sum, etc.
Scan — collect cumulative results over sequence of processors

CM-2 Interconnect (cont.)

- Hypercube uses router for point-to-point communication between processor chips
  - Messages move across each of 12 dimensions in sequence
  - If no conflicts, a message will reach its destination within 1 cycle of the sequence
  - All processors can send a message (of any length), all messages are sent and delivered at same time
  - Throughput depends on message length and access patterns
- NEWS grid can be used for nearest-neighbor communication
  - Communication in multiple dimensions: 256x256, 1024x64, 8x8192, 64x32x32, 16x16x16x16, 8x8x4x8x8x4
  - Regular pattern avoids overhead of explicitly specifying destination address

CM-2 Nexus

- The nexus is a 4x4 crosspoint switch that connects up to 4 front-end computers to up to 4 sequencers
  - CM can be configured as up to 4 sections, each used separately
  - Any front-end can be connected to any section or combination of sections
  - Example: 64K processors, four 16K sections (1 to one FE, 1 to another FE, 2 to third FE, fourth PE for other tasks)
- Each section connects to one of 8 I/O channels (graphics display frame buffer, or I/O controller)
  - Transfers initiated by front-end computers
  - Data goes into buffers, when buffers are full it goes to a data vault (each has 39 disk drives, total capacity 10GB)

Software

- System software is based on OS used in front-end computers
  - Use familiar OS, languages
  - Front end handles all flow of control, including storing and executing program, and interaction with user and programmer
  - Languages: Paris, *LISP, CM-LISP, C*
- Paris (parallel instruction set)
  - Inter-processor communication, vector summation, matrix multiplication, sorting
  - Front-end processor sends Paris instructions to processor sequencers
    - Functions & subroutines (direct actions of processors, router, I/O, etc., including scan and spread operations), global variables (find out how many processor are available, etc.)
    - Sequencer produces low-level instructs.