Connectionist Network Supercomputer Project

A Collaboration of the University of California and the International Computer Science Institute

John Wawrzynek

March 1994
Personnel

- Faculty
  Jerry Feldman
  Nelson Morgan
  John Wawrzynek
- Post-doc
  John Lazzaro
- Staff
  James Beck
  Phil Kohn
  David Johnson
  Bertrand Iriessou
- Students
  Krste Asanović
  David Bailey
  Chris Bregler
  Tim Callahan
  Ben Gomes
  Brian Kingsbury
  Sven Meier
- Visiting Researchers
  Silvia Mueller
  Arno Formello
  Srini Narayanan
  Stelios Perissakis
  David Stoutamire
  Su-Lin Wu

Funding

- Office of Naval Research URI Grant (since May 1992)
- National Science Foundation
  Experimental Systems
  PYI award
  Graduate Fellowships
  Mammoth Infrastructure Grant
- ICSI

  Funds provided by ministries of research of Germany, Italy, and Switzerland, and cooperating companies.
- ARPA/ONR Grant
- Total approximately $2M per year.
Project Overview

Moderately-priced scalable high-performance connectionist computation

- Tool for Artificial Neural Network Research
  - Training Large Nets (up to $10^9$ parameters)
  - Experiments with Real-World I/O (new work in speech and vision)
- Research & Education in Parallel Architectures, VLSI, Connectionist Software.

Benchmark Problem

Evaluate a network with a million units and an average of a thousand connections per unit for a total of a billion connections. This should be done 100 times per second.
Low-Moderate Precision Arithmetic

- Fast Digital Multipliers require $O(N^2)$ area.
- FP makes it worse.
- High-precision arithmetic requires high operand bandwidth.

- Full precision or FP not needed for a wide class of problems
  - Example: NN for speech need only 16b values and 8b weights.

*We use 16x16 bit multiplies with 32b accumulates.*
Processor Organization

How does one organize many small arithmetic units?

Based on our experience:

1. Amdahl's Law applies
   - Need a well integrated general purpose processor.
   
   A key issue is finding the right balance of special purpose multiply/add resources and general purpose processing.

2. Hardware should support software and not vice versa.
Vector Processing

- **Vector instruction set architecture (ISA)** provides a simple abstraction of highly regular parallelism.

- Many arithmetic operations specified in a single instruction:

  \[
  \begin{array}{c}
  [a_1 a_2 a_3 \ldots a_n] \\
  \text{OP} \\
  [b_1 b_2 b_3 \ldots b_n] \\
  \downarrow \\
  [c_1 c_2 c_3 \ldots c_n]
  \end{array}
  \quad \quad
  \begin{array}{c}
  [a_1 a_2 a_3 \ldots a_n] \\
  \text{OP} \\
  [b_1 b_2 b_3 \ldots b_n] \\
  \downarrow \\
  [c_1 c_2 c_3 \ldots c_n]
  \end{array}
  \]

- Well established paradigm
  - Well understood compiler technology
  - Many existing algorithms

- A range of implementation costs and performance are possible (without changing ISA).
Vector load/store architecture

Two vector arithmetic datapaths, each 8-way parallel

Maximum vector length of 32 elements

Fixed-point and integer (FP emulation)

Scalar Unit executes MIPS-II instruction set
  - general scalar computations and
  - address generation and control for vector units

Single instruction issue

Wide memory bus with various load/store options
TO System Software

Vector Instructions are implemented as MIPS "coprocessor" instructions ⇒

- Standard MIPS programming tools directly usable:
  - C compiler
  - assembler
  - debugger (gdb)
  - system library (standard I/O routines)
- Handcrafted vector libraries for common operations
- Cycle accurate RTL simulator (1100 cycle/sec on SPARC 10/51)
- Fast ISA simulator (100–500K cycles/sec)
- FP emulation for scalar unit
- Yet to come:
  - optimized code scheduler
  - vectorizing compiler
  - FP emulation for vector units
  - Fixed point to FP analyzer
- MOSIS/HP CMOS26B (1.0\textmu m), 50MHz
- 800K transistors
- 800M arithmetic operations per second (vector units only)
- 400M operands/s (800MB/sec) memory bandwidth
T0 Performance

- Neural Network Computations (16b weights, 8b activations):
  - 360 MCPS forward pass
  - 100 MCUPS backprop training
- MPEG decoding (160 X 128 frame):
  - iDCT: 1.02ms
  - (sparc2 implementation: 21.07ms)
TO for Vision

Horizontal Gradient Calculation

Disparity Calculation

Sample Performance

<table>
<thead>
<tr>
<th>Task*</th>
<th>Sparc-10 (C++)</th>
<th>T0</th>
<th>T0 Resource Utilization</th>
<th>T0 Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convolution</td>
<td>2.6 sec (5x7 kernel)</td>
<td>0.0172 sec (8x8 kernel)</td>
<td>VP0: .80 VP1: .46 VMP: .56</td>
<td>arith: 500 MOPS/sec mem: 450 MB/sec</td>
</tr>
<tr>
<td>Disparity</td>
<td>10.08 sec (&quot;focused&quot;)</td>
<td>0.1671 sec (&quot;raw&quot;)</td>
<td>VP0: .20 VP1: .36 VMP: .83</td>
<td>arith: 224 MOPS/sec mem: 664 MB/sec</td>
</tr>
</tbody>
</table>

*. on 640 x 480 images
• Sbus (SUN workstation) board
• 33 MBytes/s input / 40 MBytes/s output to Xilinx (DMA)
• 8 MBytes/s I/O via Sbus (programmed I/O) (measured)
• 20MB/s I/O via Sbus DMA (estimate)
• 8 Mbytes fast local SRAM
**Multiple Node Systems**

- **T1** — Node in MIMD massively parallel processor
  - network interface for 2-D mesh connections
  - support for efficient message passing
  - scalable from 1 to 1K nodes
Hydrant I/O Interface

Hydrant I/O Interfaces

- T - T1  H - Hydrant  X - Xilinx FPGA

- T1 side — 8b link at 250 Mbytes/s

- Xilinx side —
  - independent address/data pins for each direction
  - 32b wide at 250 Mbytes/s

- RTL level design complete, partial VLSI layout.