HOT CHIPS III SYMPOSIUM

An 80MHz 64-Bit

Floating Point RISC Processor

with Direct DRAM Support

MICRON TECHNOLOGY, INC.

Presentation Outline

- MICRON FRISC PROJECT
- SYSTEM DESIGN / TARGET APPLICATIONS
- FRISC CHIP ARCHITECTURE AND DESIGN
- SOFTWARE SYSTEM

What is the Micron FRISC Project?

FRISC - Stands for: Floating Point Reduced Instruction Set Computer

Objectives:

1. Design and Develop the World's Fastest Single Chip Processor
2. Construct Systems From Processor Chip:
   - (SS) Scalar Supercomputer (256 MByte - 1 GByte)
   - (MP) Multi-Processor 8 - 32 Processors (256 MByte - 1 GByte)
   - (LSP) Large Scale Parallel Processor 64 - 256 Processors (32 MByte - 128 MByte per Node)
   - (RTW) Real Time Unix Workstation
   - (RTL) Real Time Unix Laptop
3. Construct Embedded Control Board Level Products

Performance Measures:

Peak Performance:

160 MFlop - 160 million Floating Point Operations per second peak rate for single precision IEEE format (32 bit)

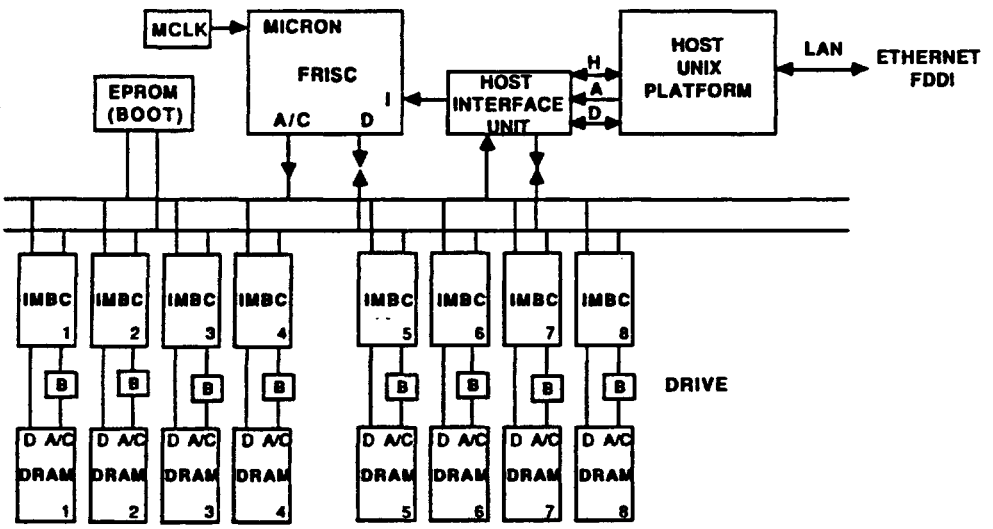
80 MFlop - 80 million Floating Point Operations per second peak rate for double precision IEEE format (64 bit)

Sustained Performance:

LINPACK Benchmark 38 - 42 MFlop, Double Precision
75 - 80 MFlop, Single Precision

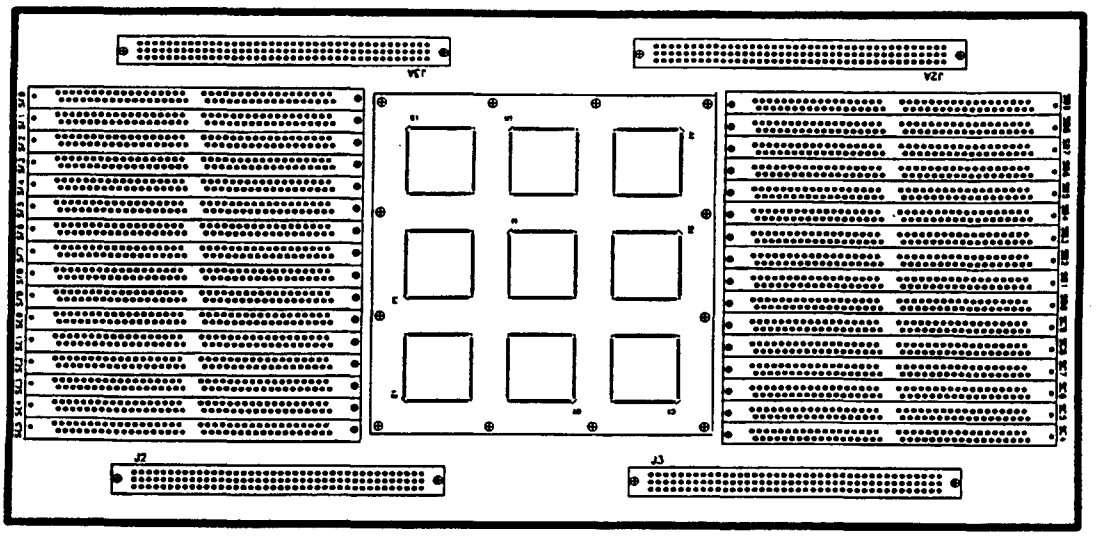
MICRON TECHNOLOGY, INC

FRISC SS - Scalar Supercomputer System Block Diagram



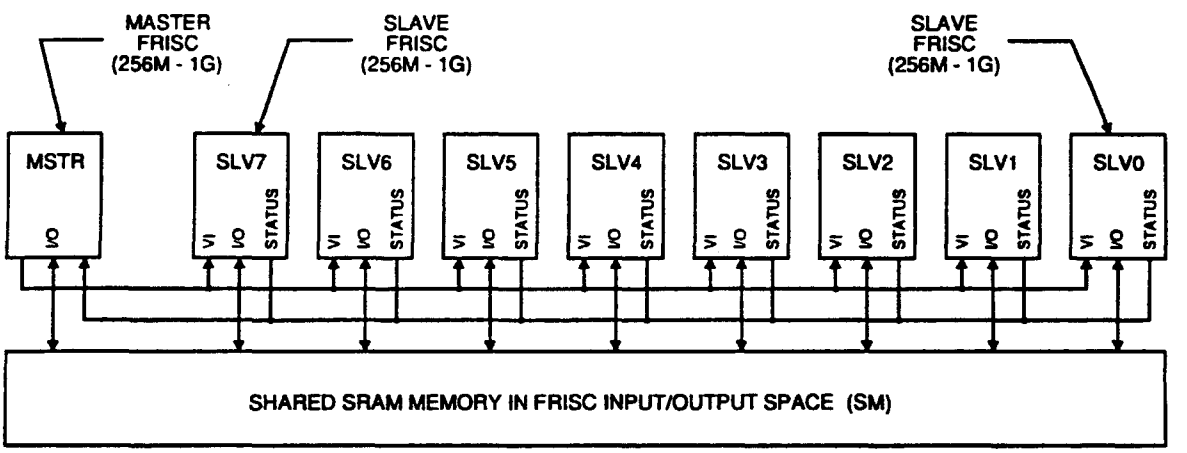
MICRON TECHNOLOGY, INC

FRISC SS - Scalar Supercomputer System Block Diagram



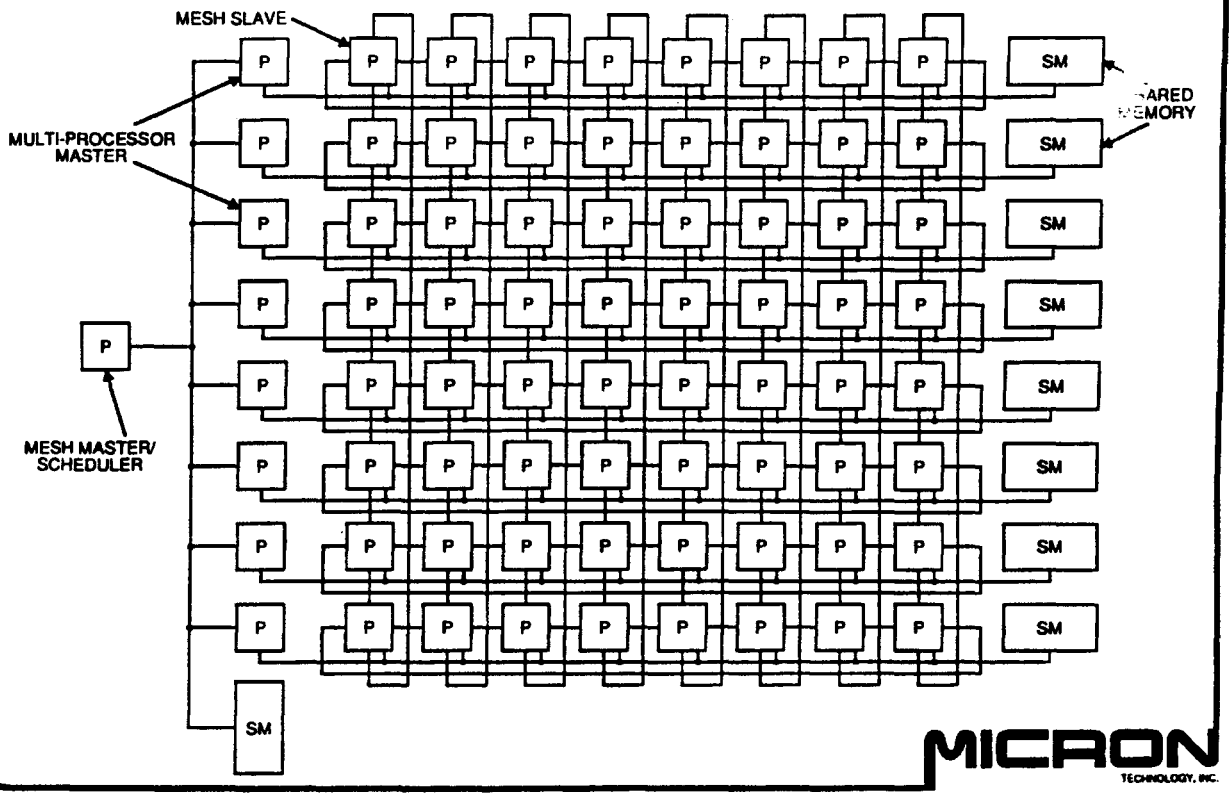
MICRON TECHNOLOGY, INC.

FRISC MP - Multi-Processor System Block Diagram



MICRON TECHNOLOGY, INC.

FRISC LSP - Multi-Mode Large Scale Parallel Processor



MICRON TECHNOLOGY, INC.

Target Applications

Scientific and Technical Users:

1. Electronic Simulation - analog and switch level simulation
   SPICE, SILOS, COSMOS, VERILOG, MSC/EMAS

2. Process Simulation -
   Supreme, Pisces

3. Mechanical Simulation - finite element and finite difference
   Nastran, Patran, ADINA, CINDA, DYNA3D

4. Fluid Dynamics - Navier Stkes Equations
   FIDAP, (ADM, ARC3D, FLO52, OCEAN, SPEC77)

5. Mathematics -
   IMSL, LINPACK, EISPACK, SPARSEPACK, Mathematica

6. Optical Design
   ACOS5

7. 3D Graphics and Rendering
   Wavefront Technologies

8. Computational Chemistry
   (BDNA, MDG, QCD, TRFD)

9. Image and Signal Signal Processing-
   Radar, Sonar, Speech processing, Image Compression

10. Medical Imaging-
    MRI Image Reconstruction

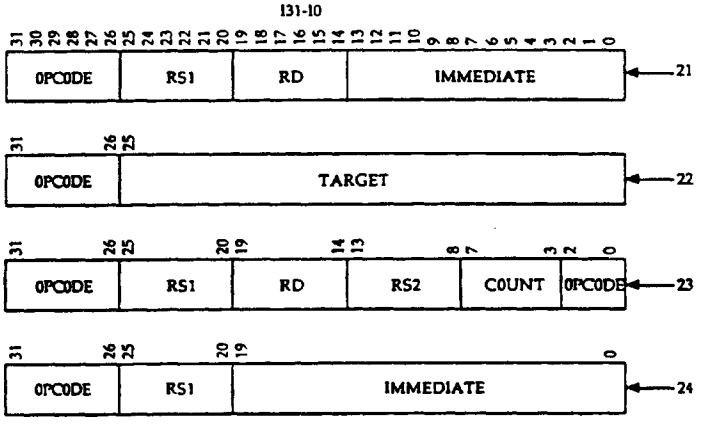
MICRON TECHNOLOGY, INC.

FRISC Chip Unique Architecture Features

- Processor bus unit directly supports a 32 way, bank interleaved, DRAM memory system and input/output memory
- Block transfer load/store operation permits concurrent execution unit and data load store unit operation
- Vector interrupt unit supports premptable and non-premptable interrupts with interrupt latency as short as 2 machine cycles
- Single cycle floating point units support 32 bit, 64 bit, fixed point and 32 bit, 64 bit IEEE floating point formats and chain mode operation in floating point
- Eight fully associative instruction buffers auto load with a least recently used replacement strategy, and permits concurrent execution and instruction buffer load operation
- Pipeline interlocks and vector interrupts supported in hardware

MICRON TECHNOLOGY, INC.

Instruction Set Summary (10F2)



MICRON TECHNOLOGY, INC.

Instruction Set Summary (20F2)

1. **Load and Store Instructions**

   The load and store instructions consist of:
   - byte, halfword, word, and doubleword load and store operations.
   - coprocessor load and store operations

   a. I-Type Load and Store Instructions
   1. Load Byte
   2. Load Byte Unsigned
   3. Load Halfword
   4. Load Halfword Unsigned
   5. Load Word
   6. Load Doubleword
   7. Store Byte
   8. Store Halfword
   9. Store Word
   10. Store Doubleword

   b. I-Type Coprocessor Load and Store Instructions
   1. Move to Coprocessor
   2. Move from Coprocessor
   3. Move Control to Coprocessor
   4. Coprocessor Operation

   c. R- Type Block Transfer Load/Store Instructions
   1. Computed Block Transfer Word Load
   2. Computed Block Transfer Word Store
   3. Computed Block Transfer Doubleword Load
   4. Computed Block Transfer Doubleword Store

2. **Computational Instructions**

   Computational instructions perform:
   - integer arithmetic operations
   - single- and double-precision floating point arithmetic
   - logical operations
   - bit field operations

   Computational instructions are both I-Type and R-Type.

   a. I- Type Computational Instructions
   1. Add Immediate
   2. Add Immediate Unsigned
   3. Subtract Immediate and Compute Conditions, signed
   4. Subtract Immediate and Compute Conditions, unsigned
   5. And Immediate
   6. Or Immediate
   7. Exclusive Or Immediate

2. **Computational Instructions**

   b. R- Type Fixed Point Computational Instructions

   Both 32-bit and 64-bit fixed point operations are supported.
   1. Add
   2. Add Unsigned
   3. Subtract
   4. Subtract Unsigned
   5. Subtract and Compute Conditions, signed
   6. Subtract and Compute Conditions, unsigned
   7. Maximum
   8. Minimum
   9. And
   10. Or
   11. Exclusive Or
   12. Nor
   13. Mask
   14. Shift Left
   15. Shift Right Arithmetic
   16. Shift Right Logical
   17. Shift Left Logical Variable
   18. Shift Right Logical Variable
   19. Shift Right Arithmetic Variable
   20. Multiply

   c. R- Type Floating Point Computational Instructions
   1. Floating Point Reciprocal (64-bit DP only)
   2. Floating Point Reciprocal Square Root
   3. Floating Point Add
   4. Floating Point Subtract
   5. Floating Point Subtract and Compute Conditions
   6. Floating Point Maximum
   7. Floating Point Minimum
   8. Floating Point Add Absolute
   9. Floating Point Multiply
   10. Floating Point Multiply-Accumulate (chained)
   11. Fixed Point to Floating Point Conversion
       32-bit fixed to 32-bit float
       64-bit fixed to 64-bit float
       32-bit fixed to 64-bit float
   12. Floating Point to Fixed Point Conversion
       32-bit float to 64-bit fixed
       64-bit float to 64-bit fixed
   13. Floating Point to Floating Point Conversion
       32-bit float to 64-bit float
       64-bit float to 32-bit float

3. **Control Transfer Instructions**

   Control transfer instructions change the instruction flow of a program. These instructions include:
   - Jump
   - Jump and link for subroutine calls
   - Conditional branch on ALU result
   - Unconditional branch

   a. J- Type Control Transfer Instructions
   1. Jump
   2. Jump And Link

   b. R- Type Control Transfer Instructions
   1. Jump Register
   2. Jump And Link Register

   c. J- Type Branch Instructions
   1. Conditional Branch on Equal
   2. Conditional Branch on Not Equal
   3. Conditional Branch on Less Than or Equal
   4. Conditional Branch on Greater Than
   5. Conditional Branch on Less Than
   6. Conditional Branch on Greater Than
   7. Conditional Branch on Equal and Link
   8. Conditional Branch on Not Equal and Link
   9. Conditional Branch on Less Than or Equal and Link
   10. Conditional Branch on Greater Than and Link
   11. Conditional Branch on Less Than and Link
   12. Conditional Branch on Greater Than or Equal and Link

4. **I-Type Load Immediate Instructions**

   a. Load immediate instructions are used to form arbitrary 32-bit values from immediate data in conjunction with the Or Immediate instruction.

   1. Load Upper Immediate

5. **Special Register Load/Store Instructions**

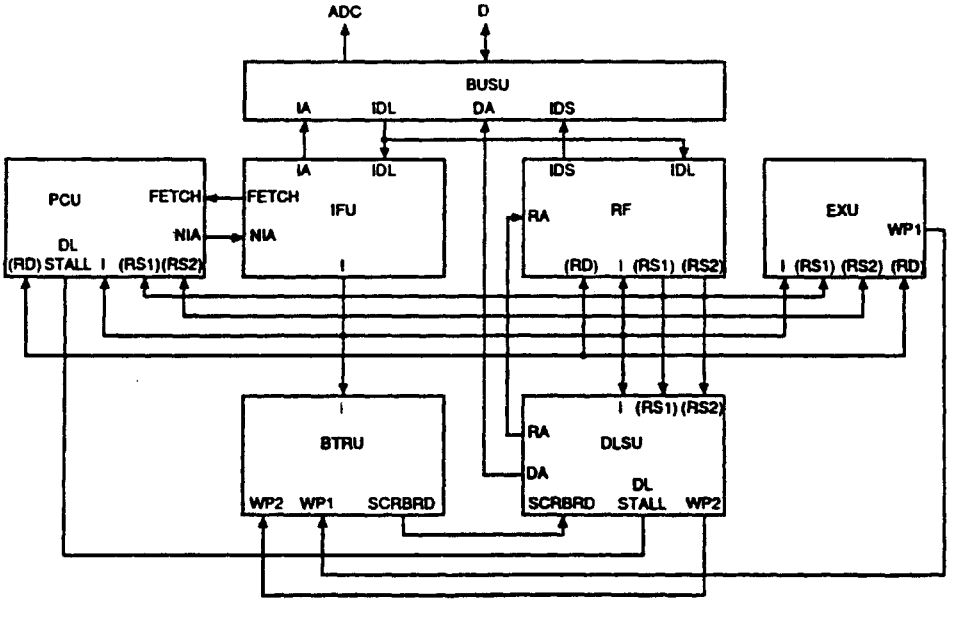
   1. Special Register Doubleword Load
   2. Special Register Doubleword Store

6. **System Control Instructions**

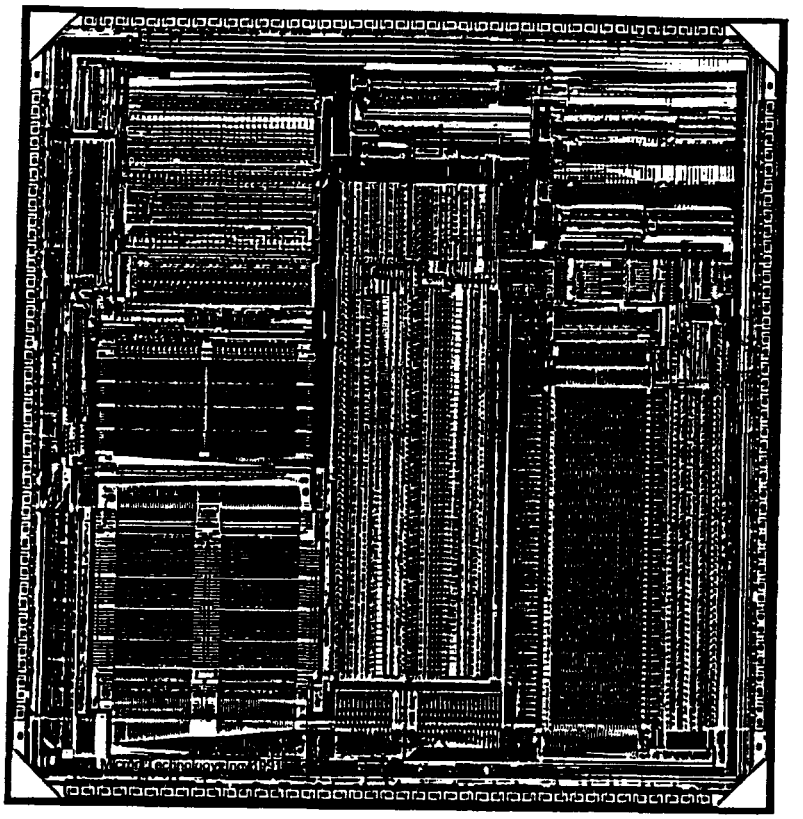
   1. NOP
   2. Vector Ext, condsional supervisor reset (rti)
   3. Normal Ext (trap)
   4. Clear Pipe
   5. Start Programmable Timers
   6. Supervisor Reset (csrr)
   7. Vector Ext, retain supervisor mode (rmp)

MICRON TECHNOLOGY, INC

Micron FRISC Supercomputer Chip Function Block Diagram

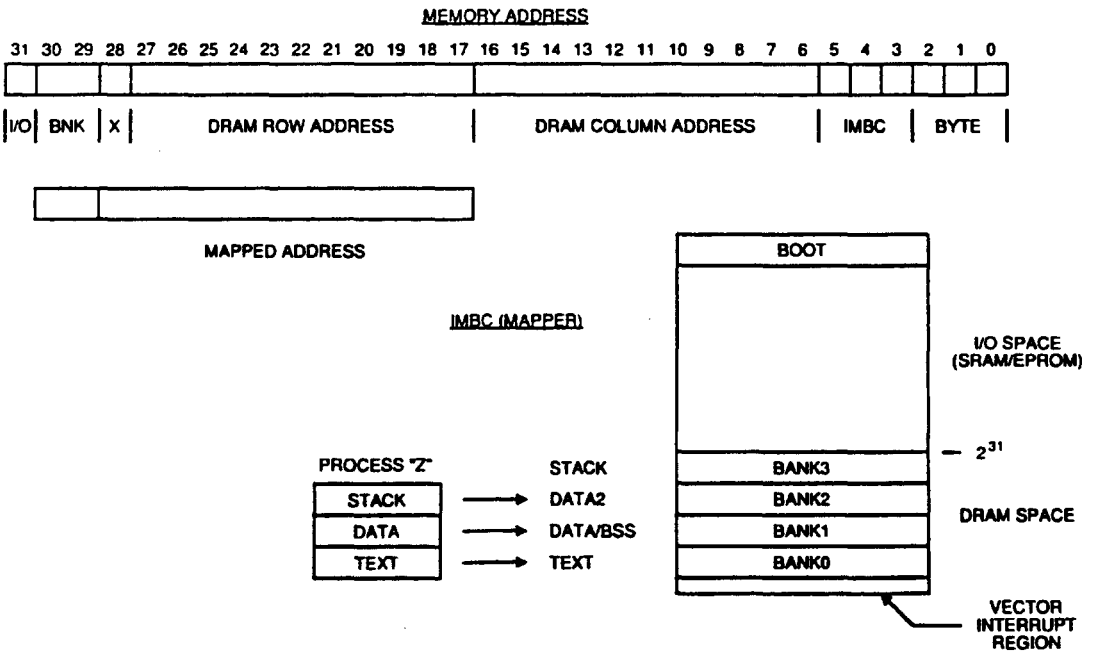


MICRON TECHNOLOGY, INC.



MICRON TECHNOLOGY, INC.

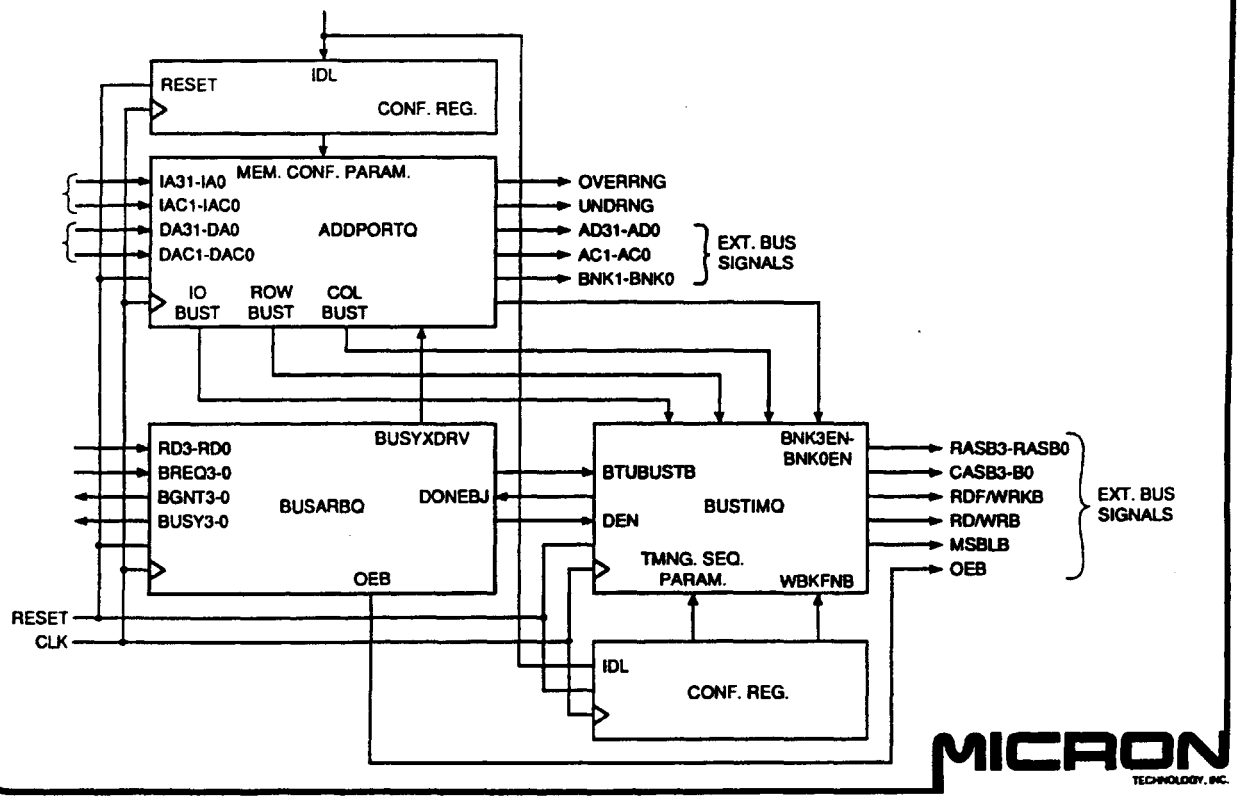
Unique Bank Interleaved Main Memory System Provides 640 MByte/Second Burst Transfer Rate



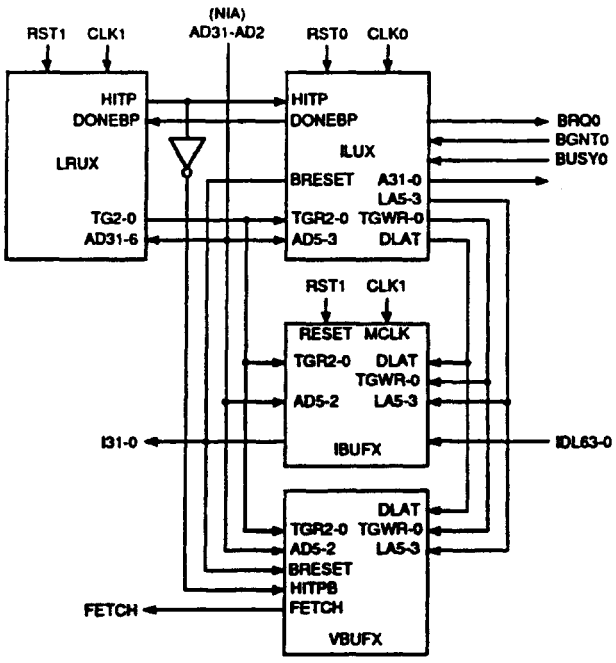
1 GIGA-BYTE MEMORY ADDRESS DEFINITION:

MICRON TECHNOLOGY, INC.

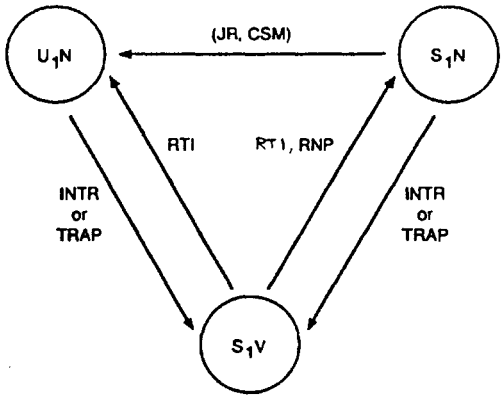
Bus Interface Unit



Instruction Fetch Unit

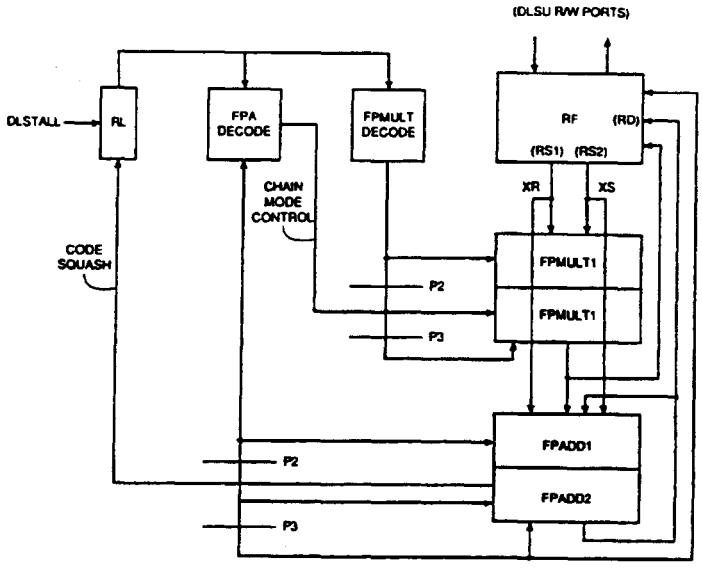


Vector Interrupt Unit State Transition Diagram



MICRON TECHNOLOGY, INC.

Floating Point Units Chained Mode Operation



Fast Newton Method for Divides and Square Root Computations

RECIPROCAL ITERATIONS

	COMPUTATION	XR OPERAND	XS OPERAND	COMMENTS
1.	x(0)	c	-----	seed lookup
2.	p(0) = cx(0)	x(0)	c	mixed precision with feedback = 0
3.	x(1) = x(0)(2 - p(0))	x(0)	p(0)	mixed precision with feedback = 0
4.	p(1) = cx(1)	x(1)	c	mixed precision with feedback = 0
5.	x(2) = x(1)(2 - p(1))	x(1)	p(1)	mixed precision with feedback = 0
6.	p(2) = cx(1)	x(1)	c	fixed point, full c LSB's of x(1)
7.	p(2) = cx(1)	x(1)	c	mixed precision with feedback from step 6.
8.	x(3) = x(2)(2 - p(2))	x(2)	p(2)	fixed point, full p(2) LSB's of x(2)
9.	x(3) = x(2)(2 - p(2))	x(2)	p(2)	mixed precision with feedback from step 8.

MICRON TECHNOLOGY, INC.

Fast Newton Method for Divides and Square Root Computations

RECIPROCAL COMPUTATION OF x(m+1):

where: $x(m+1) = x(m)(2 - p(m))$
$p(m) = cx(m)$
$p(m) = 2^{**}(alpha)(1.fp)$, $fp = p(n-2)p(n-3)\ldots p(0)$
alpha = exponent value of p(m)

$x(m) = (-1)^{**}sx\ 2^{**}(xexp - bias)(\ 1.fx)$
xexp = biased exponent of x(m)
bias = IEEE bias value 1023 for double and 127 for single precision floating point
$fx = x(n-2)x(n-3)\ldots x(0)$, x(m) fraction field

CASE 1. p(m) exponent value (alpha) = -1

$x(m+1) = (\{\ 1.0\overline{p(n-2)}\ \overline{p(n-3)}\ldots\overline{p(0)}\ \}(\ 1.x(n-2)x(n-3)\ldots x(0)\ )$

$+\ 2^{**}(-n)(\ 1.x(n-2)x(n-3)\ldots x(0)\ ))\ ((-1)^{**}sx)\ 2^{**}(xexp - bias)$

CASE 2. p(m) exponent value (alpha) = 0

$x(m+1) = (\{\ \overline{p(n-2)}.\ \overline{p(n-3)}\ldots\overline{p(0)}\ \}(\ 1.x(n-2)x(n-3)\ldots x(0)\ )$

$+\ 2^{**}(-n+2)(\ 1.x(n-2)x(n-3)\ldots x(0)\ ))\ ((-1)^{**}sx)\ 2^{**}(xexp - bias\ -1)$

MICRON TECHNOLOGY, INC.

Fast Newton Method for Divides and Square Root Computations

RECIPROCAL SQUARE ROOT ITERATIONS

	COMPUTATION	XR OPERAND	XS OPERAND	COMMENTS
1.	x(0)	c	-----	seed lookup
2.	q(0) = x(0)x(0)	x(0)	x(0)	mixed precision with feedback = 0
3.	p(0) = cq(0)	q(0)	c	mixed precision with feedback = 0
4.	x(1) = .5X(0)(3 -p(0))	x(0)	p(0)	mixed precision with feedback = 0
5.	q(1) = x(1)x(1)	x(1)	x(1)	mixed precision with feedback = 0
6.	p(1) = cq(1)	q(1)	c	mixed precision with feedback = 0
7.	x(2) = .5X(1)(3 - p(1))	x(1)	p(1)	mixed precision with feedback = 0
8.	q(2) = x(2)x(2)	x(2)	x(2)	fixed point, full x(2) LSB's of x(2)
9.	q(2) = x(2)x(2)	x(2)	x(2)	mixed precision with feedback from step 8.
10.	p(2) = cq(2)	q(2)	c	fixed point, full c LSB's of q(2)
11.	p(2) = cq(2)	q(2)	c	mixed precision with feedback from step 10.
12.	x(3) = .5x(2)(3 - p(2))	x(2)	p(2)	fixed point, full p(2) LSB's of x(2)
13.	x(3) = .5x(2)(3 - p(2))	x(2)	p(2)	mixed precision with feedback from step 12.

MICRON TECHNOLOGY, INC.

Fast Newton Method for Divides and Square Root Computations

RECIPROCAL SQUARE ROOT COMPUTATION OF x(m+1):

where: x(m+1) = .5x(m)(3 - p(m))
p(m) = cx(m)**2
p(m) = 2**(alpha)(1.fp), fp = p(n-2)p(n-3)... p(0)
alpha = exponent value of p(m)
x(m) = (-1)**sx 2**(xexp - bias)(1.fx)
xexp = biased exponent of x(m)
bias = IEEE bias value 1023 for double and 127 for single precision floating point
fx = x(n-2)x(n-3)... x(0), x(m) fraction field

CASE 1. p(m) exponent value (alpha) = 0

$x(m+1) = ((1.\overline{p(n-2)}\ \overline{p(n-3)}...\overline{p(0)})(1.x(n-2)x(n-3)...x(0))$

$+ 2^{**}(-n+1)(1.x(n-2)x(n-3)...x(0)))\ ((-1)^{**}sx)\ 2^{**}(xexp - bias -1)$

CASE 2. p(m) exponent value (alpha) = -1

SUBCASE 1. p(0) = 0

$x(m+1) = ((1.00\overline{p(n-2)}\ \overline{p(n-3)}...\overline{p(1)})(1.x(n-2)x(n-3)...x(0))$

$+ 2^{**}(-n-1)(1.x(n-2)x(n-3)...x(0)))\ ((-1)^{**}sx)\ 2^{**}(xexp - bias)$

SUBCASE 2. p(0) = 1

$x(m+1) = ((1.00\overline{p(n-2)}\ \overline{p(n-3)}...\overline{p(1)})(1.x(n-2)x(n-3)...x(0))$

$+ 2^{**}(-n)(1.x(n-2)x(n-3)...x(0)))\ ((-1)^{**}sx)\ 2^{**}(xexp - bias)$

MICRON TECHNOLOGY, INC.

Daxpy Scatter/Gather

This code overlaps arithmetic and I/O cycles by unrolling the loop and double-buffering to the register file:

```
        .
        .
        .
    addiu   r7,8(r0)
    srl     r2,r2,1             ; mpl = mpl/2
        .
        .
        .
L1: cbtl.l  r8,r3,r7,4          ; r8,r10,r12,r14 <- sx(i+4),sx(i+5)...
    fmul.l  r16,r16,r6          ; r16 <- sa*sx(i)
    fmul.l  r18,r18,r6          ; r18 <- sa*sx(i+1)
    fmul.l  r20,r20,r6          ; r20 <- sa*sx(i+2)
    fmul.l  r22,r22,r6          ; r22 <- sa*sx(i+3)
    cbtl.l  r24,r4,r7,4         ; r24,r26,r28,r30 <- sy(i+4),sy(i+5)...
    fadd.l  r32,r32,r16         ; r32 <- sy(i)+sa*sx(i)
    fadd.l  r34,r34,r18         ; r34 <- sy(i+1)+sa*sx(i+1)
    fadd.l  r36,r36,r20         ; r36 <- sy(i+2)+sa*sx(i+2)
    fadd.l  r38,r38,r22         ; r38 <- sy(i+3)+sa*sx(i+3)
    cbts.l  r32,r4,r7,4         ; sy(i),sy(i+1),...sy(i+3) <- r32,...r38
    addi    r3,32(r3)
    addi    r4,32(r4)

    cbtl.l  r16,r3,r7,4         ; ...2nd half of loop

        .
        .
        .
    addi    r1,4(r1)
    addi    r3,32(r3)
    subcc   r0,r1,r2
    ble     L1
    addi    r4,32(r4)

        .
        .
        .
    cleanup code

        .
        .
        .
```

Total is 46 machine cycles

FRISC Chip and System Patents Pending

A Flexible Microprocessor Bus Unit for the Operation of a High Speed Bank Interleaved DRAM Memory System.

Fast Newton Iterations for Reciprocal and Reciprocal Square Roots. (3)

Easily Configuraable Fully Differential Fast Logic Circuit.

Universal Multiplier Accumulator.

High Speed CMOS Driver Circuits.

Area Efficient Clock Noise Reduction in an Integrated Circuit Substrate.

Block Transfer Register Scoreboard Unit for Data Processing Systems.

High Speed, Five Port Register File Having Simultaneous Read and Write and High Tolerance to Clock Skew.

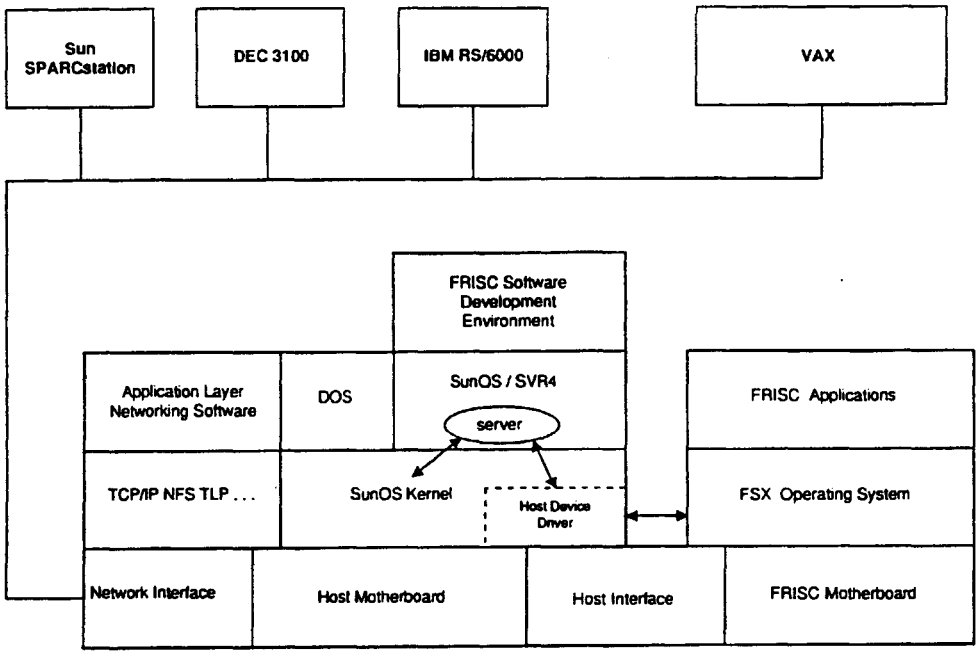
A RISC Microprocessor Priority Vector Interrupt System with Premptable and Non-Premptable Operation Modes for the Direct Support of a Real Time Operating System.

High Speed Fully Associate Instruction Buffer with Least Recently Used Replacement Strategy and Block Transfer Load Operation.

A Bank Interleaved Memory Buffer.

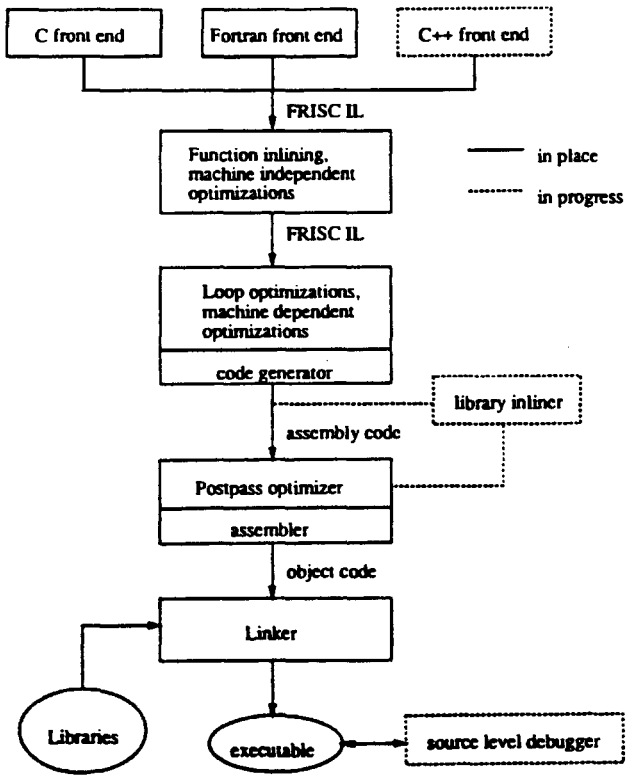
A RISC Floating Point Unit that Supports Chained Mode Instruction Execution and Hardware Pipeline Interlocking.

System Software Architecture



MICRON TECHNOLOGY, INC.

Software Development Environment



MICRON TECHNOLOGY, INC.

Acknowledgements

FRISC Software, Hardware, Layout Team:

John Mosby
Greg Perkins
Richard Schoener
Chris Unrein
Craig Carter
Larry Balsan

View Graphs By . . . Larry Cromar

MICRON TECHNOLOGY, INC