Lecture 4
Vector Registers (PPC970)

Figure 2-1. Programming Model—All Registers

SUPERVISOR MODEL - OEA

Configuration Registers

Machine State Register
USR (64/32)

Processor Version Register
PVR (32)

Memory Management Registers

Instruction BAT Registers
IBAT0U (64/32) SPR E28
IBATOL (64/32) SPR E30
IBATU (64/32) SPR E31
IBATLU (64/32) SPR E32
IBATLU1 (64/32) SPR E33
IBATL2 (64/32) SPR E34
IBATL1 (64/32) SPR E35
IBATL0 (64/32) SPR E36

Data BAT Registers
DBAT0U (64/32) SPR E36
DBATOL (64/32) SPR E37
DBATU (64/32) SPR E38
DBATLU (64/32) SPR E39
DBATLU1 (64/32) SPR E40
DBATL2 (64/32) SPR E41
DBATL1 (64/32) SPR E42
DBATL0 (64/32) SPR E43

Segment Registers 1, 2
SR0 (32)
SR1 (32)

Address Space Register
ASR (64)

Exception Handling Registers

Data Address Register
DAR (64) SPR 19

SPRs

SPR0 (64/32) SPR 272
SPR1 (64/32) SPR 273
SPR2 (64/32) SPR 274
SPR3 (64/32) SPR 275

Save and Restore Registers

ISR0 (64/32) SPR 29
ISR1 (64/32) SPR 27
ISR2 (64/32) SPR 28
ISR3 (64/32) SPR 29

Floating-Point Exception Cause Register (Optional)
FPECR SPR 1032

Miscellaneous Registers

Time Base Facility (For Writing)
TBL (32) SPR 284
TBU (32) SPR 285

Decrementer
DEC (32) SPR 22

Processor Identification Register (Optional)
PR SPR 1025

Data Address Breakpoint Register (Optional)
DBAR (64/32) SPR 1013

External Access Register (Optional)
DAR (32) SPR 282

USER MODEL - VFA

Time Base Facility (For Reading)
TBL (32) SPR 208
TBU (32) SPR 209

1 These registers are 32-bit registers only.
2 These registers are on 32-bit implementations only.
3 These registers are 04-bit implementations only.
4 In 04-bit implementations, TBP208 is read as a 04-bit value.
Complexity of CPU cores

**PPC 970 Features**

- **Instruction pipe**
  - 64KB L1 Inst cache, direct mapped
  - 32 entry I buffer
  - 8 instructions fetch / cycle
- **Branch prediction**
  - Highly accurate dynamic prediction
- **Dispatch, issue**
  - 1 group (4 + branch) / cycle
  - Up to 20 active groups
  - Up to 8 issue / cycle
  - Over 200 instructions in flight
- **Data pipe**
  - 32 KB L1 Data cache, 2-way sa
  - 32 x 64b GPR, FPR
  - 32 x 128b VRF
  - 512KB L2 cache, 8-way sa
  - 8 data prefetch streams
Complexity of execution

Figure 1-4 Execution Units and Ports in the Out-Of-Order Core

Port 0
- ALU 0 Double Speed
  - ADD/SUB
  - Logic Store Data Branches
- FP Move
  - FP Move
  - FP Store Data FXCH

Port 1
- ALU 1 Double Speed
  - ADD/SUB
- Integer Operation Normal Speed
  - Shift/Rotate
- FP Execute
  - FP_ADD
  - FP_MUL
  - FP_DIV
  - FP_MISC
  - MMX_SHFT
  - MMX_ALU
  - MMX_MISC

Port 2
- Memory Load
  - All Loads Prefetch

Port 3
- Memory Store
  - Store Address

Note:
- FP_ADD refers to x87 FP, and SIMD FP add and subtract operations
- FP_MUL refers to x87 FP, and SIMD FP multiply operations
- FP_DIV refers to x87 FP, and SIMD FP divide and square root operations
- MMX_ALU refers to SIMD integer arithmetic and logic operations
- MMX_SHFT handles Shift, Rotate, Shuffle, Pack and Unpack operations
- MMX_MISC handles SIMD reciprocal and some integer operations
points in the program, giving sense instructions (explained under "Instructions") with the addresses of the sense switches causes the calculator to follow one of two courses, depending on which sense switches are depressed. The sense switches are also effective while the calculator is on MANUAL.

**Panel Input Switches.** There are 36 panel input switches, enabling the operator to insert a word of information into various registers of the calculator while it is on MANUAL. When a panel input switch is down, it represents a 1; when up, it represents a 0.

**Index Display Keys.** The three index display keys let the operator display the contents of any of the index registers, while the calculator is on MANUAL, by pressing the key marked with the letter corresponding to the index register in question. For example, to display the contents of index register A, the operator presses the key marked DISPLAY A; the contents of index register A then appears in the index lights. The index registers are automatically displayed until the calculator is returned to automatic operation.

More than one index register can be manually displayed in sequence by pressing the Display A, Display B, and Display C keys, in that order. No return to the automatic mode is necessary.

**Load Keys.** The load keys let the operator initiate the loading of a self-loading program stored on
Programming

- direct programming of modern CPUs?
- use high level languages
- historically: FORTRAN (IBM, early 1950’s)
Fortran Automatic Coding System
## APPENDIX B. TABLE OF FORTRAN STATEMENTS

<table>
<thead>
<tr>
<th>STATEMENT</th>
<th>NORMAL SEQUENCING</th>
</tr>
</thead>
<tbody>
<tr>
<td>a = b</td>
<td>Next executable statement</td>
</tr>
<tr>
<td>GO TO n</td>
<td>Statement n</td>
</tr>
<tr>
<td>GO TO n, n₁, n₂, ..., nₙ</td>
<td>Statement list assigned</td>
</tr>
<tr>
<td>ASSIGN i TO n</td>
<td>Next executable statement</td>
</tr>
<tr>
<td>GO TO n₁, n₂, ..., nₙ, i</td>
<td>Statement n₁</td>
</tr>
<tr>
<td>IF n₁ = n₂</td>
<td>Statement n₁ as less than, =, or greater than 0</td>
</tr>
<tr>
<td>SENSE LIGHT i</td>
<td>Next executable statement</td>
</tr>
<tr>
<td>IF SENSE LIGHT IF n₃ &gt; n₄</td>
<td>Statement n₃ as Sense Light ON or OFF</td>
</tr>
<tr>
<td>IF SENSE SWITCH IF n₅ &gt; n₆</td>
<td>as Sense Switch DOWN or UP</td>
</tr>
<tr>
<td>IF ACCUMULATOR OVERFLOW n₇ &gt; n₈</td>
<td>&quot; &quot; &quot; as Accumulator Overflow trigger ON or OFF</td>
</tr>
<tr>
<td>IF QUOTIENT OVERFLOW n₉ &gt; n₁₀</td>
<td>&quot; &quot; &quot; as MQ Overflow trigger ON or OFF</td>
</tr>
<tr>
<td>IF DIVIDE CHECK n₁₁ &gt; n₁₂</td>
<td>&quot; &quot; &quot; as Divide Check trigger ON or OFF</td>
</tr>
<tr>
<td>PAUSE or PAUSE n</td>
<td>Next executable statement</td>
</tr>
<tr>
<td>STOP or STOP n</td>
<td>Terminates program</td>
</tr>
<tr>
<td>DO i = m₁ TO n₁ OR DO i = m₁,m₂,m₃...</td>
<td>Next executable statement</td>
</tr>
<tr>
<td>CONTINUE</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>FORMAT(specification)</td>
<td>Not executed</td>
</tr>
<tr>
<td>READ n.List</td>
<td>Next executable statement</td>
</tr>
<tr>
<td>READ INPUT TAPE i, n, List</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>PUNCH n.List</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>PRINT n.List</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>WRITE OUTPUT TAPE i, n, List</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>READ TAPE i, List</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>READ DRUM i, j, List</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>WRITE TAPE i, List</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>WRITE DRUM i, j, List</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>END FILE i</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>REWIND i</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>BACKSPACE i</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>DIMENSION i, j, k, ...,</td>
<td>Not executed</td>
</tr>
<tr>
<td>EQUIVALENCE i, j, k, ..., I, ...,</td>
<td>&quot; &quot; &quot;</td>
</tr>
<tr>
<td>FREQUENCY i, j, k, m, l, I, ...,</td>
<td>&quot; &quot; &quot;</td>
</tr>
</tbody>
</table>

50
FORTRAN II (IBM 704)
IBM System/360
and System/370
FORTRAN IV Language

This publication describes and illustrates the use of the FORTRAN IV language for IBM System/360 and System/370. It is primarily a reference manual for programmers who are familiar with the elements of the language.

FORTRAN IV is a mathematically-oriented language useful in writing programs for applications that involve manipulation of numerical data.
Form of Fortran programs

```
C PROGRAM FOR FINDING THE LARGEST VALUE
C
ATTAINED BY A SET OF NUMBERS
DIMENSION A(100)
READ (5,1) N,(A(I),I=1,N)
1 FORMAT (I5/(12F2.2))
BIG=A(1)
DO 20 I=2,N
2 IF (BIG-A(I)) 10,20,20
10 BIG=A(I)
20 CONTINUE
WRITE (6,2),BIG
2 FORMAT (2X,'THE LARGEST OF THESE, 10, 114 NUMBERS IS ',F7.3)
STOP
END
```
Ill-conditioned problems

\[ \cos(x) = b \]