CISC
COMPLEX INSTRUCTION SET COMPUTER

PERFORMANCE?
DESIGN TIME?
DESIGN ERRORS?
SINGLE CHIP?
PROGRAM SIZE?

RISC
REDUCED INSTRUCTION SET COMPUTER
1970's Design Principles

(1) Semiconductor Memory Growth + Microprogramming
   ⇒ "COSTS LITTLE FOR RICHER INSTRUCTION SETS"
(2) "Move software to "firmware" (microcode)"
   ⇒ FASTER & MORE RELIABLE SYSTEMS
(3) "Smaller programs are faster programs"
   ⇒ REDUCE CODE SIZE
(4) "Registers are old fashioned" (hard for compilers)
   ⇒ MEMORY-TO-MEMORY, STACKS

"One's eyebrows should rise whenever
a future architecture is developed
with a register oriented instruction set."

Glenford J. Myers
1978
RISC Design Principles

(1) **Keep functions simple** unless you have a very good reason not to.

10% increase in cycle time

$\Rightarrow$ > 10% fewer cycles?

(2) **Microinstructions are same speed as simple instructions.**

(3) **Microcode is not magic.**

(4) **Simple decoding and pipelined execution >> program size**

(5) **Use compiler technology to simplify instrs.**
292R
15 GRADS EXPLORE SIMPLE ARCH.
5 GRADS FORM RISC I

248
22 GRADS MEAD CONWAY DESIGN COMPONENTS

292X
11 GRADS REDESIGN COMPONENTS

292Y
9 GRADS INTEGRATE PARTS

June 22
"TAPEOUT"

July 20
"FIRST SILICON"

GOOD SILICON?

Spring Summer Fall Winter Spring Summer
1980 1981
<table>
<thead>
<tr>
<th>Tool</th>
<th>Institution</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCCS</td>
<td>Bell/UCB</td>
<td>Allman</td>
</tr>
<tr>
<td>Mail</td>
<td>UCB</td>
<td>Shoens</td>
</tr>
<tr>
<td>Msgs</td>
<td>UCB</td>
<td>Joy</td>
</tr>
<tr>
<td>Caesar</td>
<td>UCB</td>
<td>Ousterhout</td>
</tr>
<tr>
<td>Cifplot</td>
<td>UCB</td>
<td>Fitzpatrick</td>
</tr>
<tr>
<td>Mosext</td>
<td>UCB</td>
<td>Fitzpatrick</td>
</tr>
<tr>
<td>Moserc</td>
<td>Stanford</td>
<td>Baskett</td>
</tr>
<tr>
<td>Powest</td>
<td>UCB</td>
<td>Cmelik</td>
</tr>
<tr>
<td>Moslrc</td>
<td>MIT</td>
<td>Baker</td>
</tr>
<tr>
<td>Mossimi</td>
<td>MIT</td>
<td>Terman</td>
</tr>
<tr>
<td>Spice</td>
<td>UCB</td>
<td>Pederson</td>
</tr>
</tbody>
</table>

**RISC**
INSTRUCTION SET RATIONALE
FORMATS

GOAL WAS KEEPING SAME SIZE

<table>
<thead>
<tr>
<th>OP</th>
<th>DEST</th>
<th>SOURCE1</th>
<th>SOURCE2</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>5</td>
<td>5</td>
<td>14</td>
</tr>
</tbody>
</table>

32

\[ R_0 \leftarrow R_{s1} \quad \text{OP} \{ R_{s2} / \text{CONSTANT}\} \]

ADD ADDC SHLA SHL AND XOR

SUB SUBC SHRA SHR OR

LOAD (BYTE, WORD, LONG) (SIGNED/ UNSIGNED)

STORE (BYTE, WORD, LONG)

\[ R_0 \leftarrow M \cdot C R_{s1} + \{ R_{s2} / \text{CONSTANT}\} \]

CALL RETURN BRANCH

<table>
<thead>
<tr>
<th>OP</th>
<th>DEST</th>
<th>ADDRESS/CONSTANT</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>5</td>
<td>19</td>
</tr>
</tbody>
</table>

CALL, RETURN, LOAD IMMEDIATE, BRANCH

PC \leftarrow PC \pm \text{ADDRESS}

NOTE: NO ATTEMPT AT REDUCING CODE SIZE BUT COULD BE DONE MAKING FETCH MORE COMPLEX
INSTRUCTION FETCH

NORMAL BRANCH

100  LOAD X,A
101  ADD I,A
102  JUMP 105

NON-OPT. DELAYED BRANCH

100  LOAD X,A
101  ADD I,A
102  JUMP 105
103  ADD B,A
104  SUB C,A
105  STORE A,Z
106  STORE A,Z

OPTIMIZED DELAYED BRANCH

100  LOAD X,A
101  ADD I,A
102  JUMP 105
103  ADD B,A
104  ADD B,A
105  SUB C,A
106  STORE A,Z
106  STORE A,Z
IMPLEMENTATION

3-phase/cycle RISC:

CPU organization

GOLD

Manolis H.G. Katevenis

3 SEP 80

BUSES 32-BITS
(Actually mixed with logic)

Instruction decoder (g_3)

DATA IN

CONTROL SIGNALS

REG. FILE

SH

DATASH

ALU

INC

PC

NXPC

LAST PC

IMM OFFS

33 pins
Time for Berkeley to build microcomputer

Industry

4-7 years

$30,000,000.00

100 man years

Experience

U.C.B.

Berkeley Computer Aided Design

Reduced Instruction Set

2 Berkeley Student Years

Beginner's Luck
<table>
<thead>
<tr>
<th>NAME NO.</th>
<th>SPONSOR</th>
<th>CREW</th>
<th>ENGINE</th>
<th>VAX POWER</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAX 11 780</td>
<td>DEC</td>
<td>Pro:~30x3yr</td>
<td>'78 Bipolar</td>
<td>1</td>
</tr>
<tr>
<td>MC 68000</td>
<td>Motorola</td>
<td>Pro:~20x3yr</td>
<td>'79 NMOS</td>
<td>3</td>
</tr>
<tr>
<td>Z 8000</td>
<td>Zilog</td>
<td>Pro:~15x3yr</td>
<td>'77 NMOS</td>
<td>5</td>
</tr>
<tr>
<td>i 8086</td>
<td>Intel</td>
<td>Pro:~20x2yr</td>
<td>'78 NMOS</td>
<td>6</td>
</tr>
<tr>
<td>RISC I</td>
<td>U.C. Berkeley</td>
<td>Am:~10x1yr</td>
<td>'76 NMOS</td>
<td>?</td>
</tr>
</tbody>
</table>

If bet, one month salary March '80, what fraction of a VAX would be the goal?
Ave Speed for 11 C programs

m

i

n

i

n

i

m

i

n

i

r

c

i

o

VAX-11/780
PDP-11/70
BBN C/70
68000
Z8002
RISC I
WHY?

- COMPILER OPTIMIZATION?
  1.2x

- SIMPLE INSTRUCTION SET?
  3x to 5x

- REGISTER WINDOW?
  3x to 5x

- INHERENT SUPERIORITY OF BERKELEY?