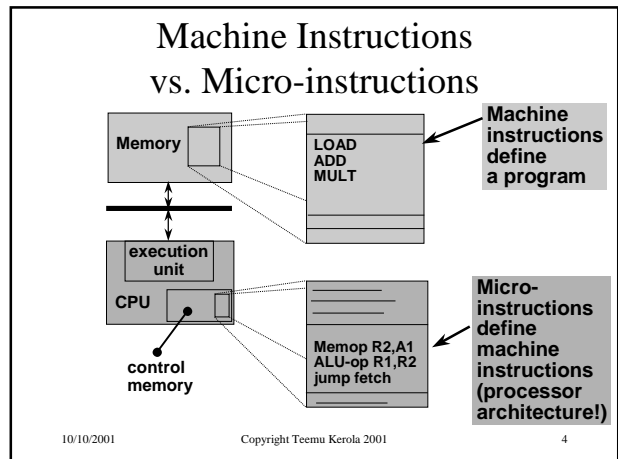


Micro-programmed Control Ch 15

Micro-instructions
Micro-programmed Control Unit
Sequencing
Execution
Characteristics

10/10/2001 Copyright Teemu Kerola 2001 1



Hardwired Control ⁽⁴⁾

- Complex
- Fast
- Difficult to design
- Difficult to modify
 - Lots of optimization done at implementation phase

10/10/2001 Copyright Teemu Kerola 2001 2

Machine Instructions vs. Micro-instructions ⁽²⁾

- Machine instruction fetch-execute cycle produces machine instructions to be executed at CPU
- Micro-instruction fetch-execute cycle produces control signals for data path

10/10/2001 Copyright Teemu Kerola 2001 5

Micro-programmed Control ⁽³⁾

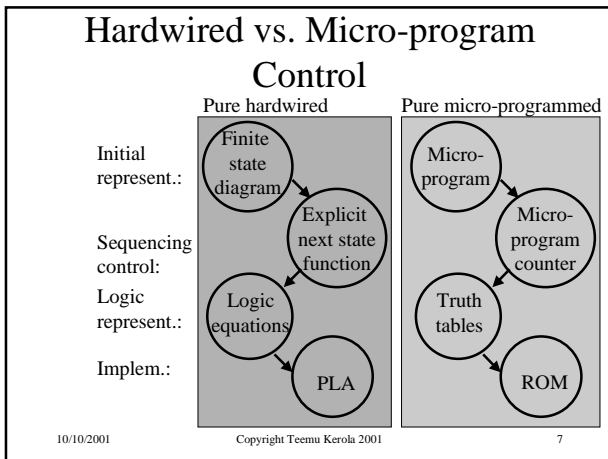
- Implement “execution engine” inside CPU
 - execute one micro-instruction at a time
- What to do now?
 - micro-instruction
 - control signals
 - stored in micro-instruction control memory
 - micro-program, firmware
- What to do next?
 - micro-instruction program counter
 - default (?): next micro-instruction
 - jumps or branches?

10/10/2001 Copyright Teemu Kerola 2001 3

Micro-program ⁽⁴⁾

- Stored in control memory Fig. 15.2
- ROM, PROM, EPROM
- One “subroutine” for each machine instruction
 - one or more micro-instructions
- Defines architecture
 - change instruction set?
 - ⇒ reload control memory

10/10/2001 Copyright Teemu Kerola 2001 6



- ### Micro-programming ⁽³⁾
- Simple design
 - Flexible
 - adapt to changes in organization, timing, technology
 - make changes late in design cycle, or even in the field
 - Very powerful instruction sets
 - use bigger control memory if needed
 - easy to have complex instruction sets
 - is this good?
- 10/10/2001 Copyright Teemu Kerola 2001 10

- ### Microcode ⁽³⁾
- Horizontal micro-code
 - control signals directly in micro-code Fig. 15.1 (a)
 - all control signals always there
 - lots of signals ⇒ many bits in micro-instruction
 - Vertical micro-code
 - each action encoded densely Fig. 15.1 (b)
 - actions need to be decoded to signals at execution time
 - takes less space but may be slower
 - Each micro-instruction is also a conditional branch?
- 10/10/2001 Copyright Teemu Kerola 2001 8

- ### Micro-programming ⁽²⁾
- Generality
 - multiple instruction sets on same machine
 - tailor instruction set to application?
 - Compatibility
 - easy to be backward compatible in one family
 - many organizations, same instruction set
- 10/10/2001 Copyright Teemu Kerola 2001 11

- ### Micro-programmed Control Unit ⁽⁴⁾
- Control Address Register
 - “micro-program PC” Fig. 15.4
 - Control Memory
 - Control Buffer Register
 - current micro-instruction
 - control signals
 - next address control
 - Sequencing logic
 - select next value for Control Address Reg
- 10/10/2001 Copyright Teemu Kerola 2001 9

- ### Micro-programming ⁽²⁾
- Costly to implement
 - need tools:
 - micro-program development environment
 - micro-program compiler
 - Slow
 - micro-instruction interpreted at execution time
 - interpretation is internal to CPU
 - interpret one instruction at a time
- 10/10/2001 Copyright Teemu Kerola 2001 12

RISC vs. Micro-programming ⁽⁸⁾

- Simple instructions can execute at very high clock rate
- Compilers can produce micro-instructions
 - machine dependent optimization
- Use only simple instructions and addressing mode
- Keep “micro-code” in RAM instead of ROM
- no micro-instruction interpretation logic needed
- Fast access to “micro-code” in RAM via caching
- Skip instruction interpretation of a micro-program and simply compile directly into lowest language of machine?
- ⇒ Compile to “micro-code” and use hardwired control for RISC

10/10/2001

Copyright Teemu Kerola 2001

13

Micro-instruction Implicit Address Generation

- Addresses not explicitly present
 - Mapping
 - map opcode in machine instruction into micro-instruction address
 - Addition Fig. 15.9
 - higher order bits directly from opcode
 - lower order bits based on current status and tag bits, or fields in current microinstruction
 - Residual Control
 - return from micro-program subroutine

10/10/2001

Copyright Teemu Kerola 2001

16

Micro-program Sequencing ⁽³⁾

- Two address format Fig. 15.6
 - default next micro-instruction address
 - waste of space most of the time?
 - conditional branch address
- One address format Fig. 15.7
 - (Conditional) branch address
- Variable format
 - only branch micro-instructions have addresses
 - waste of time many times?

10/10/2001

Copyright Teemu Kerola 2001

14

Micro-instruction Encoding

- Usually a compromise between pure horizontal and vertical formats
 - optimize on space with encoding multiple signals into a set of fields Fig. 15.11
 - each field defines control signals for certain separate actions
 - mutually exclusive actions are encoded into the same field
 - make design simpler by not using maximum encoding

10/10/2001

Copyright Teemu Kerola 2001

17

Micro-instruction Explicit Address Generation

- Addresses explicitly present
 - Two-field
 - select one of them
 - Unconditional branch
 - jump to this one
 - Conditional branch
 - select this one or default

10/10/2001

Copyright Teemu Kerola 2001

15

Micro-instruction Encoding ⁽²⁾

- Functional encoding
 - each field controls some function
 - load accumulator
 - load ALU operands
 - compute next PC
- Resource encoding
 - each field controls some resource
 - ALU
 - memory

10/10/2001

Copyright Teemu Kerola 2001

18

Example Micro-instruction Sets for a Simple Machine ⁽³⁾

- Micro-instruction types Fig. 15.12
 - 3 register transfers, 2 mem ops, 5 ALU ops, 3 seq. ops
- Vertical format

type	operation	reg
------	-----------	-----

 - 3 bits for type, 3 bits for operation Fig. 15.12 (a)
 - 2 bits for reg select (max 4 regs)
- Horizontal format

--	--	--	--	--	--	--	--	--	--

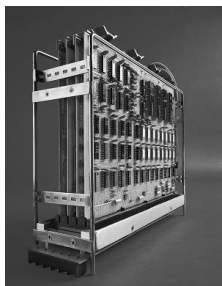
 - 2 bits for reg transfers (3 ops + “none”)
 - 2 bits for mem ops (2 ops + “none”)
 - 2 bits for seq. ops (3 ops + “none”) Fig. 15.12 (b)
 - 3 bits for ALU ops (5 ops + “none”)
 - 2 bits for reg select + 8 bits for constant

LSI-11 Micro-instruction Set ⁽²⁾

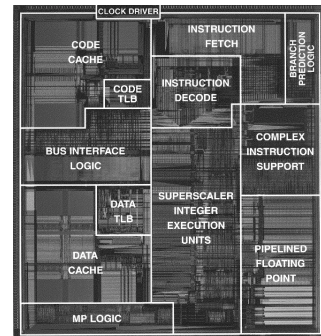
- Implements PDP-11 instruction set architecture for LSI-11 hardware
 - e.g., PDP-11 16 bit ALU vs. LSI-11 8-bit ALU
- 22 bit wide, extremely vertical set Fig. 15.15
 - 4 bits for special functions
 - 1 bit for testing interrupts
 - 1 bit for “micro-subroutine return”
 - 16 bits for variable format micro-ops
 - jump, cond. branch, literal ops, reg ops
 - ALU, logical, general, I/O ops

Table 15.5

LSI-11 Single Board Processor



-- End of Chapter 15 -- -- Micro-programmed Control --



16.10
http://infopad.EECS.Berkeley.EDU/CIC/die_photos/pentium.gif
Copyright Teemu Kerola 2001 23

LSI-11 (PDP-11) ⁽⁵⁾

- Three-chip single board processor
 - data chip
 - 26 8-bit regs
 - 8 16-bit general purpose regs,
 - PWS, MAR, MBR, ...
 - 8-bit ALU
 - (at least) 2 passes needed for 16-bit reg ops
 - control chip Fig. 15.14
 - control store chip
 - 22 bit wide control mem for micro-instructions
 - connected by micro-instruction bus Fig. 15.13

Summary ⁽¹⁰⁾

- How clock signals cause instruction executions?
- Low level stuff
 - gates, basic circuits, registers, memory
- Cache
- Virtual memory & TLB
- ALU, int & FP arithmetic's
- Instruction sets
- CPU structure & pipelining
- Branch prediction, limitations, hazards, issue
- RISC & superscalar processor
- Hardwired & micro-controlled control

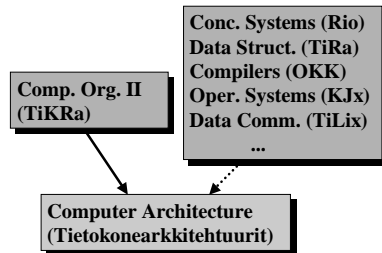
10/10/2001

Copyright Teemu Kerola 2001

25

Want to Know More?

- Read the text book completely
- 58070-8 Computer Architecture (4 cr)

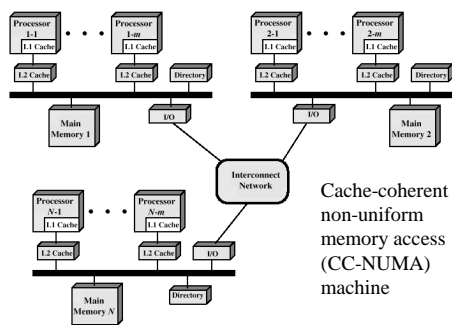


10/10/2001

Copyright Teemu Kerola 2001

26

-- The End --



Cache-coherent
non-uniform
memory access
(CC-NUMA)
machine

(Fig. 16.10)

10/10/2001

Copyright Teemu Kerola 2001

27