

Micro-programmed Control Ch 15

Micro-instructions
Micro-programmed Control Unit
Sequencing
Execution
Characteristics

26.11.1999 Copyright Teemu Kerola 1999 1

Hardwired Control ⁽⁴⁾

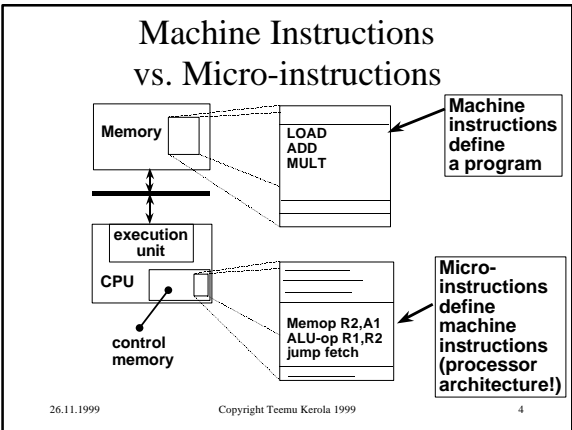
- Complex
- Fast
- Difficult to design
- Difficult to modify
 - Lots of optimization done at implementation phase

26.11.1999 Copyright Teemu Kerola 1999 2

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?

26.11.1999 Copyright Teemu Kerola 1999 3



Machine Instructions vs. Micro-instructions ⁽²⁾

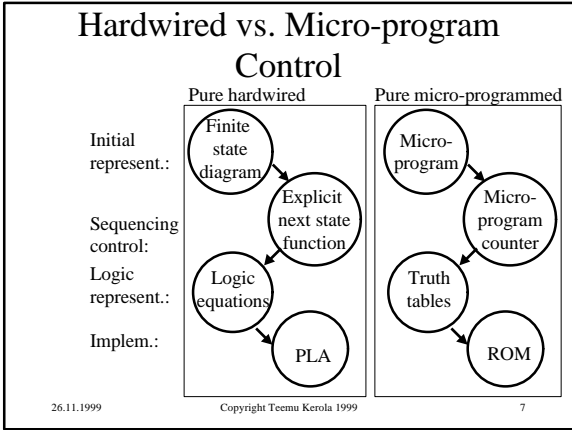
- Machine instruction fetch-execute cycle produces machine instructions to be executed at CPU
- Micro-instruction fetch-execute cycle produces control signals for data path

26.11.1999 Copyright Teemu Kerola 1999 5

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

26.11.1999 Copyright Teemu Kerola 1999 6



Microcode ⁽³⁾

Fig. 15.1

- Horizontal micro-code
 - control signals directly in micro-code
 - all control signals always there
 - lots of signals ⇒ many bits in micro-instruction
- Vertical micro-code
 - each action encoded densely
 - 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?

26.11.1999 Copyright Teemu Kerola 1999 8

Micro-programmed Control Unit ⁽⁴⁾

Fig. 15.4

- Control Address Register
 - “micro-program PC”
- Control Memory
- Control Buffer Register
 - current micro-instruction
 - control signals
 - next address control
- Sequencing logic
 - select next value for Control Address Reg

26.11.1999 Copyright Teemu Kerola 1999 9

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

26.11.1999 Copyright Teemu Kerola 1999 10

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

26.11.1999 Copyright Teemu Kerola 1999 11

Micro-programming ⁽²⁾

- Costly to implement
 - need tools:
 - micro-program development environment
 - micro-program compiler
- Slow
 - micro-instruction interpreted at execution time

26.11.1999 Copyright Teemu Kerola 1999 12

RISC vs. Micro-programming ⁽⁷⁾

- 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
- 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

26.11.1999 Copyright Teemu Kerola 1999 13

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?

26.11.1999 Copyright Teemu Kerola 1999 14

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

26.11.1999 Copyright Teemu Kerola 1999 15

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 machine opcode
 - lower order bits based on current status and tag bits
 - Residual Control
 - return from micro-program subroutine

26.11.1999 Copyright Teemu Kerola 1999 16

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

26.11.1999 Copyright Teemu Kerola 1999 17

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

26.11.1999 Copyright Teemu Kerola 1999 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
------	-----------	-----

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

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

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

26.11.1999 Copyright Teemu Kerola 1999 19

SLI-11 Single Board Processor



26.11.1999 Copyright Teemu Kerola 1999 20

LSI-11 (PDP-11) ⁽⁴⁾

- Three-chip single board processor
 - data
 - 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 Fig. 15.14
 - control store
 - 22 bit wide control mem for micro-instructions
 - connected by micro-instruction bus Fig. 15.13

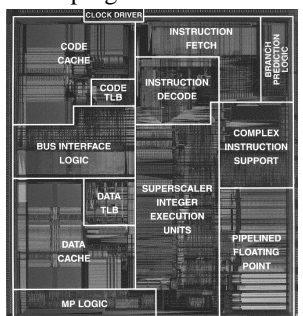
26.11.1999 Copyright Teemu Kerola 1999 21

LSI-11 Micro-instruction Set ⁽²⁾

- Implements PDP-11 instruction set architecture for LSI-11 hardware
- 22 bit wide, extremely vertical set
 - 4 bits for special functions
 - 1 bit for testing interrupts
 - 1 bit for "micro-subroutine return"
 - 16 bits for variable format micro-ops Fig. 15.15
 - jump, cond. branch, literal ops, reg ops
 - ALU, logical, general, I/O ops Table 15.5

26.11.1999 Copyright Teemu Kerola 1999 22

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



http://infopad.EECS.Berkeley.EDU/CIC/die_photos/pentium.gif 16.10)

26.11.1999 Copyright Teemu Kerola 1999 23

26.11.1999 Copyright Teemu Kerola 1999 24

Summary

- How does clock signal execute instructions?
- 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

26.11.1999 Copyright Teemu Kerola 1999 25

Want to Know More?

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

Comp. Org. II (TiKRra)

Computer Architecture

Conc. Systems (Rio)
Data Struct. (TiRa)
Compilers (OKK)
Oper. Systems (KJP)
Data Comm. (TiLi)
...

26.11.1999 Copyright Teemu Kerola 1999 26

