

Multi-cycle Processor*

Lecturer: Dr. Hui Annie Guo
huig@cse.unsw.edu.au
K17-501F (ext. 57136)

Overview

- **Microprogramming**

COMP3211/9211

2011S1 wk4_1 P2

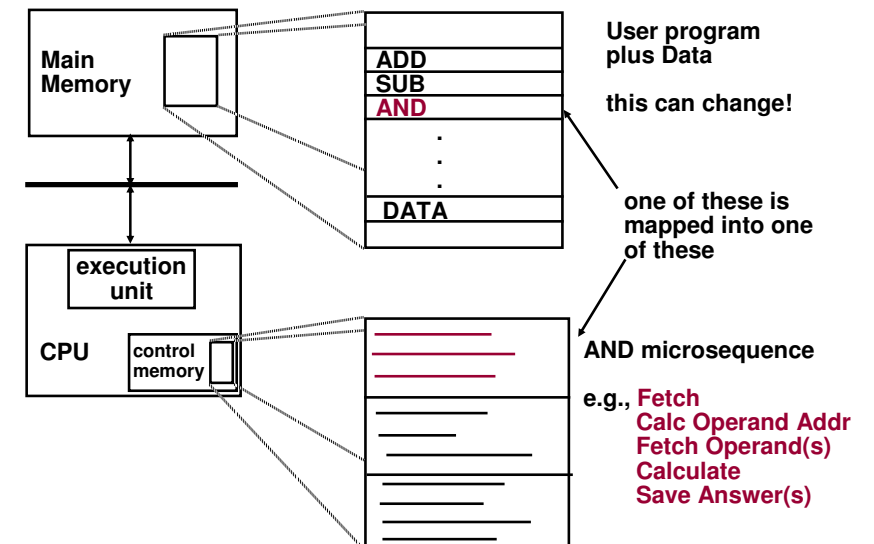
Microprogramming

- Each machine instruction is implemented by a series of microinstructions, called microcode
- The microgram for each machine instruction is written by the CPU designer
 - carefully designed and optimized for fastest possible execution
- Microinstruction implementation is easy and simple
- Execution of an instruction is to execute a sequence of microinstructions which is controlled by the sequencer.

COMP3211/9211

2011S1 wk4_1 P3

“Macroinstruction” Interpretation



COMP3211/9211

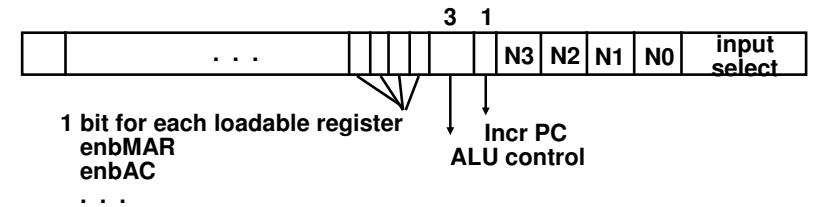
2011S1 wk4_1 P4

Variations on Microprogramming

- **“Horizontal” Microcode**
 - control field for each control point in the machine
- **“Vertical” Microcode**
 - compact microinstruction format for each class of micro-operations
 - local decode to generate all control points

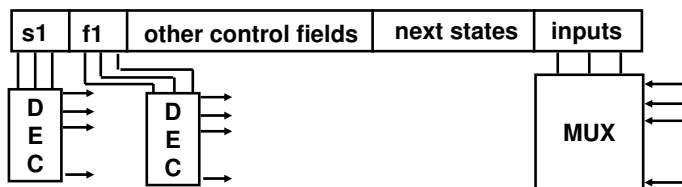
Extreme Horizontal

- **Allow all possible combinations of control signals and hence all possible combination of operations**



Vertical Format

- **Encoding some control signals with a set of bits**
- **Control signals are obtained by decoding**



Horizontal vs. Vertical Microprogramming

- **Most microprogramming-based controllers vary between**
 - horizontal organization (1 control bit per control point)
 - vertical organization (fields encoded in the control memory and must be decoded to control something)

Horizontal

- + more control over the potential parallelism of operations in the datapath
- Consume large storage space

Vertical

- + easier to program, not very different from programming a RISC machine in assembly language
- extra level of decoding may slow the machine down

Let's design a microprogram for our MIPS subset -- (R-type, SW, LW, BEQ, Jump)

- **Defining a microinstruction format**
 - Start with a list of control signals
 - Grouping signals together that make sense (vs. random): called “fields”
 - Place fields in some logical order
 - e.g., ALU operation & ALU operands first and microinstruction sequencing last
 - Determine symbolic value for each field
 - each of this value has corresponding control signals
- **Creating microprogram**
 - For a specified instruction set
- **Implementing the microprogram**

COMP3211/9211

2011S1 wk4_1 P9

Define a Microinstruction Format

- **Two design goals:**
 - **Readability**
 - E.g., one field for one functional block
 - **Compatibility**
 - Each field responsible for specifying a non-overlapping set of control signals
- **Our Microinstruction format**

Label	ALU control	SRC1	SRC2	Register ctrl	Memory Ctl	PCWrite Ctl	Sequencing
-------	-------------	------	------	---------------	------------	-------------	------------

COMP3211/9211

2011S1 wk4_1 P10

Microinstruction Format Field Values

Field name	Values
Label	Any string
ALU control	Add, Sub, Func code
SRC1	PC, A
SRC2	4, B, Extshft, Extend
Register control	Read, Write ALU, Write M
Memory	Read PC, Read ALU, Write ALU
PCWrite control	ALUOut-cond, Jump address
sequencing	Seq, Fetch, Dispatch i

COMP3211/9211

2011S1 wk4_1 P11

Sequencing

- **Determining what instruction should be executed next**
- **Unlike in normal program, the next instruction should be handled explicitly**
- **Three values in sequencing**
 - **Seq:** choose the next microinstruction sequentially
 - The one physically next to the current microinstruction in the control memory
 - **Fetch:** go to the microinstruction for a new instruction
 - The microinstruction is labeled as “Fetch”
 - **Dispatch:** jump to some microinstruction, according to the type of the instruction
 - Can be identified by the opcode field in the instruction

COMP3211/9211

2011S1 wk4_1 P12

Dispatch

- Implemented by two dispatch tables in our design

Opcode field	Opcode name	Value
000000	R-format	Rformat1
000010	jmp	JUMP1
000100	beq	BEQ1
100011	lw	Mem1
101011	sw	Mem1

Opcode field	Opcode name	Value
100011	lw	LW2
101011	sw	SW2

COMP3211/9211

2011S1 wk4_1 P13

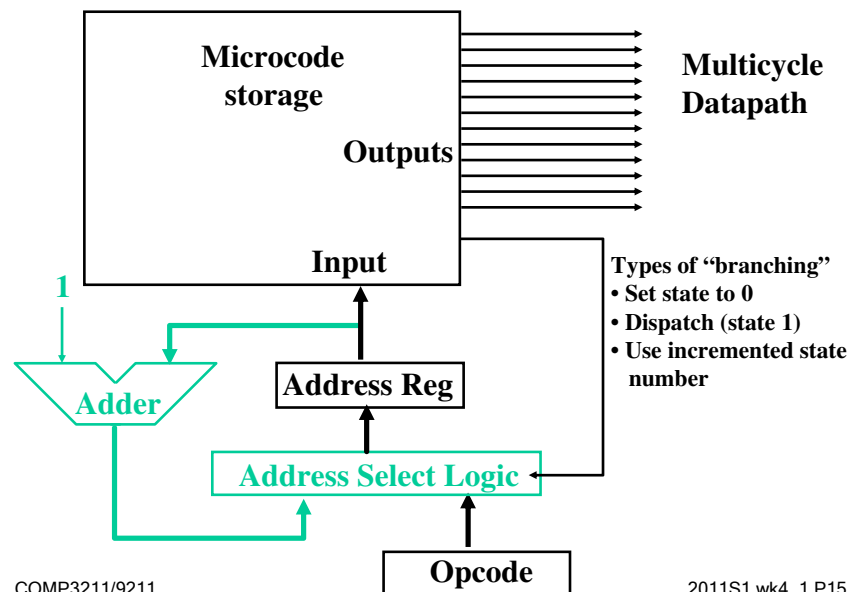
Creating Microprogram

Label	ALU control	SRC1	SRC2	Register control	Memory	PCWrite control	Sequencing
Fetch	Add	PC	4		Read PC	ALU	Seq
	Add	PC	Extshft	Read			Dispatch 1
Mem1	Add	A	Extend				Dispatch 2
LW2					Read ALU		Seq
				Write M			Fetch
SW2					Write ALU		Fetch
Rformat1	Func code	A	B				Seq
				Write ALU			Fetch
BEQ1	Sub	A	B			ALUOut-cond	Fetch
JUMP1						Jump address	Fetch

COMP3211/9211

2011S1 wk4_1 P14

Implementing Microprogram



COMP3211/9211

2011S1 wk4_1 P15

Exercise

For the given microprogram, list the addresses of the microinstructions that are read in order to execute the following program segment:

```

SW    $0, 10($3)
LW    $1, 10($3)
BEQ   $0, $1, L1
SUB   $5, $6, $7
L1:   SW    $5, 20($3)
    
```

COMP3211/9211

2011S1 wk4_1 P16

Addr	Label	ALU control	SRC1	SRC2	Register control	Memory	PCWrite control	Sequencing
0001	Fetch	Add	PC	4		Read PC	ALU	Seq
0010		Add	PC	Extshft	Read			Dispatch 1
0011	Mem1	Add	A	Extend				Dispatch 2
0100	LW2					Read ALU		Seq
0101					Write M			Fetch
0110	SW2					Write ALU		Fetch
0111	Rformat1	Func code	A	B				Seq
1000					Write ALU			Fetch
1001	BEQ1	Sub	A	B			ALUOut-cond	Fetch
1010	JUMP1						Jump address	Fetch