

CENG3420

## Lab 3-1: LC-3b Datapath

**Bei Yu**

Department of Computer Science and Engineering  
The Chinese University of Hong Kong

[byu@cse.cuhk.edu.hk](mailto:byu@cse.cuhk.edu.hk)

Spring 2018



香港中文大學

The Chinese University of Hong Kong

# Overview

Introduction

Lab3-1 Assignment

Golden Results



# Overview

Introduction

Lab3-1 Assignment

Golden Results



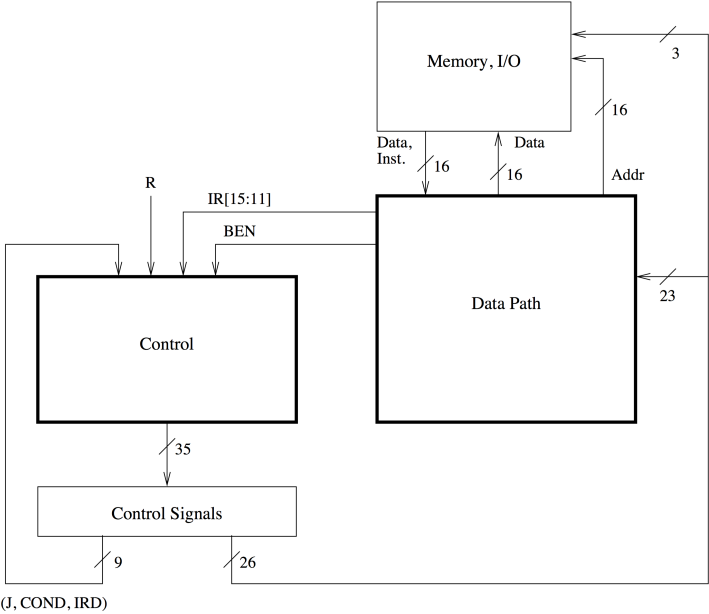
# The Slides are self-contained? NO!

Do please refer to following document:

- ▶ [LC-3b-datapath.pdf](#)
- ▶ [LC-3b-ISA.pdf](#)



# LC-3b Microarchitecture



## Input of Control Structure (7 bits)

- ▶ **R**: indicate whether memory data is ready (`System_Latches::READY`)
- ▶ **BEN**: indicate whether BR been taken (`System_Latches::BEN`)
- ▶ **IR[15:11]**: current instruction (`System_Latches::IR`)



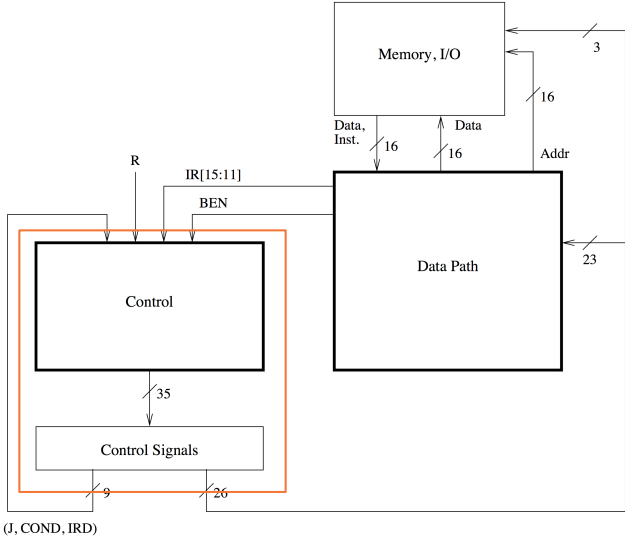
# Output of Control Structure: (35 bits)

```
45 /******  
46 /* Definition of bit order in control store word. */  
47 /******  
48 enum CS_BITS {  
49     IRD,  
50     COND1, COND0,  
51     J5, J4, J3, J2, J1, J0,  
52     LD_MAR,  
53     LD_MDR,  
54     LD_IR,  
55     LD_BEN,  
56     LD_REG,  
57     LD_CC,  
58     LD_PC,  
59     GATE_PC,  
60     GATE_MDR,  
61     GATE_ALU,  
62     GATE_MARMUX,  
63     GATE_SHF,  
64     PCMUX1, PCMUX0,  
65     DRMUX,  
66     SR1MUX,  
67     ADDR1MUX,  
68     ADDR2MUX1, ADDR2MUX0,  
69     MARMUX,  
70     ALUK1, ALUK0,  
71     MIO_EN,  
72     R_W,  
73     DATA_SIZE,  
74     LSHF1,  
75     CONTROL_STORE_BITS  
76 } CS_BITS;
```

- ▶ 26 bits to control data path
- ▶ J[5:0], COND[1:0],IRD: generate address of control structure for next clock cycle

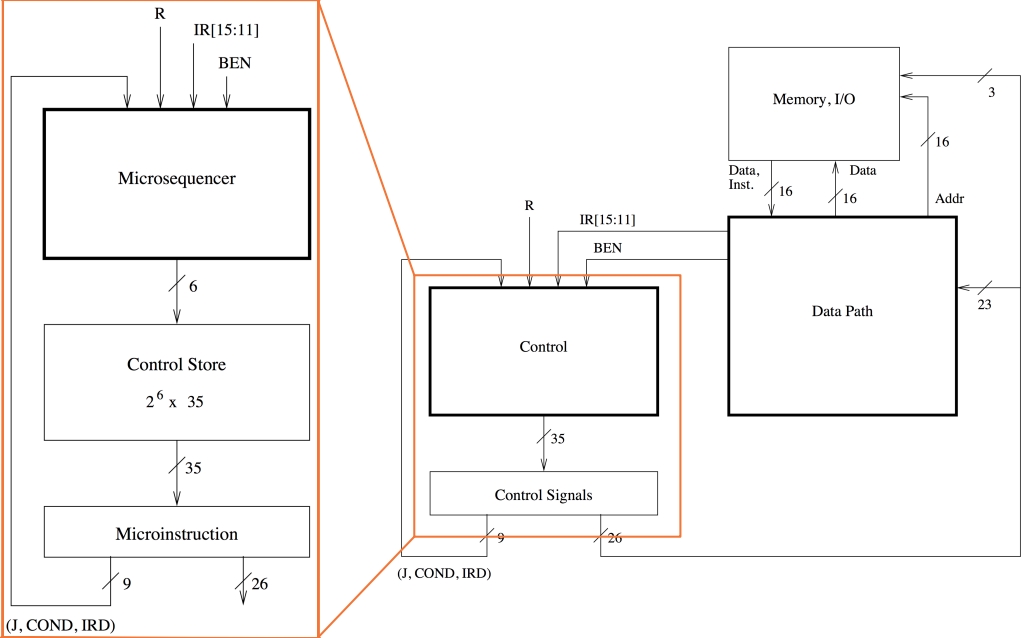


# LC-3b Control Structure

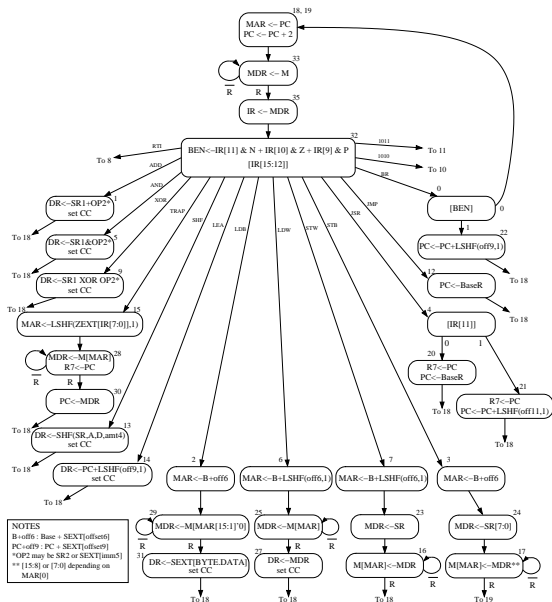




# LC-3b Control Structure



# How's Microsequencer Actually Working?





# How's Control Store Actually Implemented?

Row	Col	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
000000	(State 0)																																																															
000001	(State 1)																																																															
000010	(State 2)																																																															
000011	(State 3)																																																															
000100	(State 4)																																																															
000101	(State 5)																																																															
000110	(State 6)																																																															
000111	(State 7)																																																															
001000	(State 8)																																																															
001001	(State 9)																																																															
001010	(State 10)																																																															
001011	(State 11)																																																															
001100	(State 12)																																																															
001101	(State 13)																																																															
001110	(State 14)																																																															
001111	(State 15)																																																															
010000	(State 16)																																																															
010001	(State 17)																																																															
010010	(State 18)																																																															
010011	(State 19)																																																															
010100	(State 20)																																																															
010101	(State 21)																																																															
010110	(State 22)																																																															
010111	(State 23)																																																															
011000	(State 24)																																																															
011001	(State 25)																																																															
011010	(State 26)																																																															
011011	(State 27)																																																															
011100	(State 28)																																																															
011101	(State 29)																																																															
011110	(State 30)																																																															
011111	(State 31)																																																															
100000	(State 32)																																																															
100001	(State 33)																																																															
100010	(State 34)																																																															
100011	(State 35)																																																															
100100	(State 36)																																																															
100101	(State 37)																																																															
100110	(State 38)																																																															
100111	(State 39)																																																															
101000	(State 40)																																																															
101001	(State 41)																																																															
101010	(State 42)																																																															
101011	(State 43)																																																															
101100	(State 44)																																																															
101101	(State 45)																																																															
101110	(State 46)																																																															
101111	(State 47)																																																															
110000	(State 48)																																																															
110001	(State 49)																																																															
110010	(State 50)																																																															
110011	(State 51)																																																															
110100	(State 52)																																																															
110101	(State 53)																																																															
110110	(State 54)																																																															
110111	(State 55)																																																															
111000	(State 56)																																																															
111001	(State 57)																																																															
111010	(State 58)																																																															
111011	(State 59)																																																															
111100	(State 60)																																																															
111101	(State 61)																																																															
111110	(State 62)																																																															
111111	(State 63)																																																															

## Finite-State-Machine (FSM)

- ▶ States 10, 11 are empty
- ▶ 6 bits input enough
- ▶ Per state, output 35 bits

Hard to implement



# Good News!

- ▶ Part of FSM has been provided
- ▶ See file “*ucode3*”

```
107
108 /*****
109 /* The control store rom.          */
110 /*****
111 int CONTROL_STORE[CONTROL_STORE_ROWS][CONTROL_STORE_BITS];
112
```



# Overview

Introduction

Lab3-1 Assignment

Golden Results



# Operations in One Clock Cycle

In "lc3bsim3-1.c":

```
void cycle ()
{
    eval_micro_sequencer();
    cycle_memory();
    eval_bus_drivers();
    drive_bus();
    latch_datapath_values();

    CURRENT_LATCHES = NEXT_LATCHES;

    CYCLE_COUNT++;
}
```



# Lab3-1 Assignment

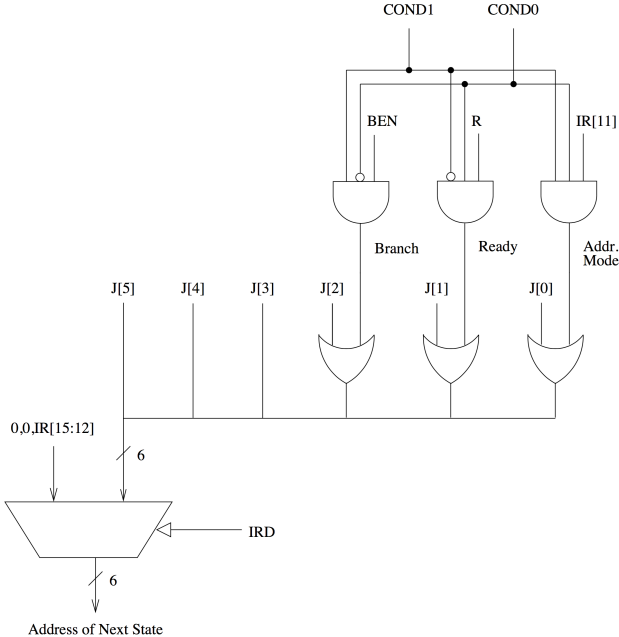
```
442
443 /*
444  * Evaluate the address of the next state according to the
445  * micro sequencer logic. Latch the next microinstruction.
446  */
447 void eval_micro_sequencer()
448 {
449     /*
450     * Lab3-1 assignment
451     *
452     * (the following lines are functionally wrong)
453     */
454
455     NEXT_LATCHES.STATE_NUMBER = 0;
456     memcpy(NEXT_LATCHES.MICROINSTRUCTION, CONTROL_STORE[NEXT_LATCHES.STATE_NUMBER], sizeof(int)*CONTROL_STORE_BITS);
457 }
458
```

- ▶ **Input:** CURRENT\_LATCHES
- ▶ **Output:** NEXT\_LATCHES.MICROINSTRUCTION





# Lab3-1 Assignment Tips



## Lab3-1 Assignment Tips (cont.)

Some functions may help:

- ▶ `GetCOND()`
- ▶ `GetIRD()`
- ▶ `GetJ()`
- ▶ `partVal()`



# Overview

Introduction

Lab3-1 Assignment

Golden Results



# Assignment Package

- ▶ `lc3bsim3-1.c`, `lc3bsim3-1.h`: codes to work on
- ▶ `libems3-1.a`: library
- ▶ `ucode3`: FSM
- ▶ Makefile
- ▶ `bench`: folder with benchmarks

## Run the simulator:

1. `make`, then binary “`lc3bsim3-1`” is generated
2. `./lc3bsim3-1 ucode3 bench/toupper.cod`



# Golden Results – case `toupper.cod`

## 1. run 6

```
Simulating for 6 cycles...
```

```
MemCycleCnt = 0  
MEM_EN = 0, R_W = 0, WE0 = 0, WE1 = 0  
MemCycleCnt = 0  
MEM_EN = 1, R_W = 0, WE0 = 0, WE1 = 0  
MemCycleCnt = 1  
MEM_EN = 1, R_W = 0, WE0 = 0, WE1 = 0  
MemCycleCnt = 2  
MEM_EN = 1, R_W = 0, WE0 = 0, WE1 = 0  
MemCycleCnt = 3  
MEM_EN = 1, R_W = 0, WE0 = 0, WE1 = 0  
MemCycleCnt = 4  
MEM_EN = 1, R_W = 0, WE0 = 0, WE1 = 0
```



# Golden Results – case toupper.cod (cont.)

## 2. rdump

Current register/bus values :

```
-----  
Cycle Count   : 6  
PC            : 0x3002  
IR            : 0x0000  
STATE_NUMBER  : 0x0023  
  
BUS           : 0x0000  
MDR           : 0xe00f  
MAR           : 0x3000  
CCs: N = 0   Z = 1   P = 0  
Registers:  
0: 0x0000  
1: 0x0000  
2: 0x0000  
3: 0x0000  
4: 0x0000  
5: 0x0000  
6: 0x0000  
7: 0x0000
```



# Golden Results – case `toupper.cod` (cont.)

## 3. Go on **run 1**

```
Simulating for 1 cycles...
```

```
MemCycleCnt = 1  
MEM_EN = 0, R_W = 0, WE0 = 0, WE1 = 0
```



# Golden Results – case toupper.cod (cont.)

## 4. rdump

Current register/bus values :

```
-----  
Cycle Count   : 7  
PC             : 0x3002  
IR             : 0xe00f  
STATE_NUMBER  : 0x0020  
  
BUS           : 0xe00f  
MDR           : 0xe00f  
MAR           : 0x3000  
CCs: N = 0   Z = 1   P = 0  
Registers:  
0: 0x0000  
1: 0x0000  
2: 0x0000  
3: 0x0000  
4: 0x0000  
5: 0x0000  
6: 0x0000  
7: 0x0000
```





# Golden Results – case toupper.cod (cont.)

## 5. Go on run 5

```
Simulating for 5 cycles...
```

```
MemCycleCnt = 0  
MEM_EN = 0, R_W = 0, WE0 = 0, WE1 = 0  
MemCycleCnt = 0  
MEM_EN = 0, R_W = 0, WE0 = 0, WE1 = 0  
MemCycleCnt = 0  
MEM_EN = 0, R_W = 0, WE0 = 0, WE1 = 0  
MemCycleCnt = 0  
MEM_EN = 1, R_W = 0, WE0 = 0, WE1 = 0  
MemCycleCnt = 1  
MEM_EN = 1, R_W = 0, WE0 = 0, WE1 = 0
```



# Golden Results – case toupper.cod (cont.)

## 6. rdump

Current register/bus values :

```
-----  
Cycle Count   : 12  
PC            : 0x3004  
IR           : 0xe00f  
STATE_NUMBER  : 0x0021  
  
BUS          : 0x0000  
MDR         : 0x0000  
MAR         : 0x3002  
CCs: N = 0  Z = 1  P = 0  
Registers:  
0: 0x3020  
1: 0x0000  
2: 0x0000  
3: 0x0000  
4: 0x0000  
5: 0x0000  
6: 0x0000  
7: 0x0000
```



Thanks. For any question: [byu@cse.cuhk.edu.hk](mailto:byu@cse.cuhk.edu.hk)

