18-447

Computer Architecture

Lecture 7: Microprogrammed Microarchitectures

Prof. Onur Mutlu
Carnegie Mellon University
Spring 2013, 1/30/2013

Homework 2

- Homework 2 out
 - Due February 11
 - LC-3b microcode
 - □ ISA concepts, ISA vs. microarchitecture, microcoded machines

Reminder: Lab Assignment 1

- Due this Friday (Feb 1), at the end of Friday lab
- A functional C-level simulator for a subset of the MIPS ISA
- Study the MIPS ISA Tutorial
- Go to the Lab Sessions, especially if you need help

Lookahead: Lab Assignment 2

- Lab Assignment 1.5
 - Verilog practice
 - Not to be turned in
- Lab Assignment 2
 - Due Feb 15
 - Single-cycle MIPS implementation in Verilog
 - All labs are individual assignments
 - No collaboration; please respect the honor code

Lookahead: Extra Credit for Lab Assignment 2

- Complete your normal (single-cycle) implementation first, and get it checked off in lab.
- Then, implement the MIPS core using a microcoded approach similar to what we will discuss in class.
- We are not specifying any particular details of the microcode format or the microarchitecture; you can be creative.
- For the extra credit, the microcoded implementation should execute the same programs that your ordinary implementation does, and you should demo it by the normal lab deadline.
- You will get partial credit for the extra credit
- Document what you have done and demonstrate well

Readings for Next Lecture

- Pipelining
 - P&H Chapter 4.5-4.8
- Pipelined LC-3b Microarchitecture
 - http://www.ece.cmu.edu/~ece447/s13/lib/exe/fetch.php? media=18447-lc3b-pipelining.pdf

Today's Agenda

- Finish the Microprogrammed LC-3b Design
- Do some microprogramming
- Start pipelining

Review: Last Lecture

- Finished single-cycle microarchitectures
- Microarchitecture design principles
- Basic performance evaluation (execution time equation)
- What does it mean to design for the common case (or bread and butter design)?
 - □ If memory takes 90% of execution time ...
- How does the single cycle microarchitecture make "critical path design" difficult?
- Remember the performance equation that consists of three components... How can you improve each component in a multi-cycle microarchitecture?

Review: Microarchitecture Design Principles

Critical path design

Find the maximum combinational logic delay and decrease it

Bread and butter (common case) design

- Spend time and resources on where it matters
 - i.e., improve what the machine is really designed to do
- Common case vs. uncommon case

Balanced design

- Balance instruction/data flow through hardware components
- Balance the hardware needed to accomplish the work

Review: Multi-Cycle Microarchitectures

 Goal: Let each instruction take (close to) only as much time it really needs

Idea

- Determine clock cycle time independently of instruction processing time
- Each instruction takes as many clock cycles as it needs to take
 - Multiple state transitions per instruction
 - The states followed by each instruction is different

Quick Review: A Microprogrammed Multi-Cycle Microarchitecture

Review: The Instruction Processing Cycle



Review: A Basic Multi-Cycle Microarchitecture

- Instruction processing cycle divided into "states"
 - A stage in the instruction processing cycle can take multiple states
- A multi-cycle microarchitecture sequences from state to state to process an instruction
 - The behavior of the machine in a state is completely determined by control signals in that state
- The behavior of the entire processor is specified fully by a finite state machine
- In a state (clock cycle), control signals control
 - How the datapath should process the data
 - How to generate the control signals for the next clock cycle

Review: Microprogrammed Control Terminology

- Control signals associated with the current state
 - Microinstruction
- Act of transitioning from one state to another
 - Determining the next state and the microinstruction for the next state
 - Microsequencing
- Control store stores control signals for every possible state
 - Store for microinstructions for the entire FSM
- Microsequencer determines which set of control signals will be used in the next clock cycle (i.e. next state)

Review: A Simple LC-3b Control and Datapath

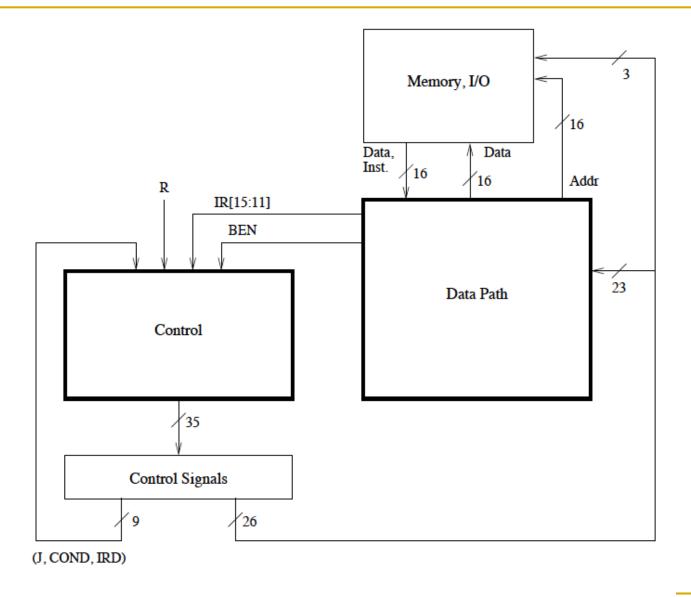


Figure C.1: Microarchitecture of the LC-3b, major components

Review: An LC-3b State Machine

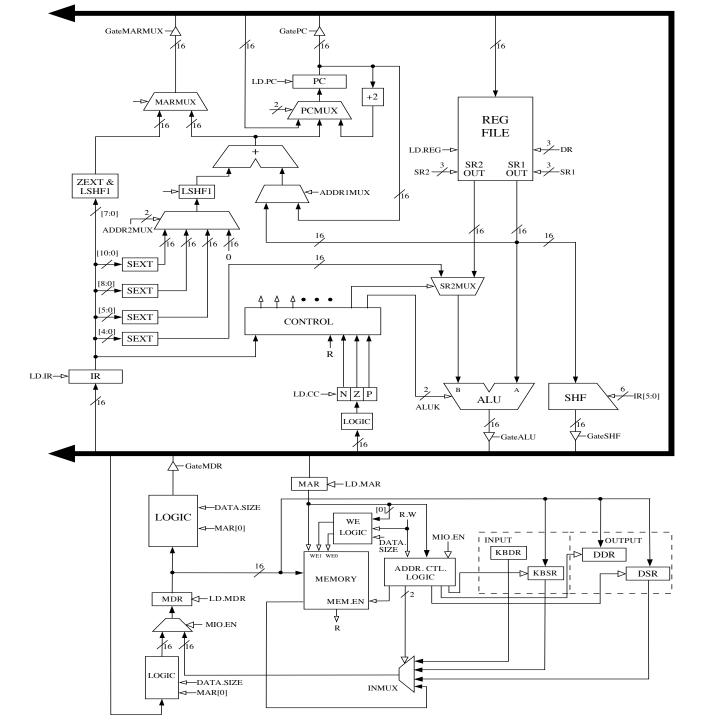
- Patt and Patel, App C, Figure C.2
- Each state must be uniquely specified
 - Done by means of state variables
- 31 distinct states in this LC-3b state machine
 - Encoded with 6 state variables
- Examples
 - State 18,19 correspond to the beginning of the instruction processing cycle
 - \square Fetch phase: state 18, 19 \rightarrow state 33 \rightarrow state 35
 - Decode phase: state 32

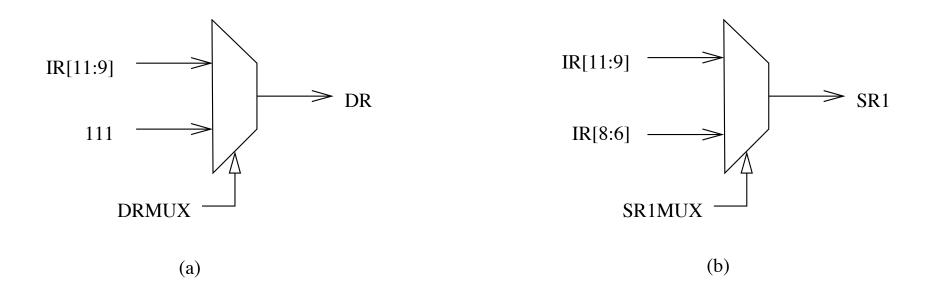
Review: LC-3b State Machine: Some Questions

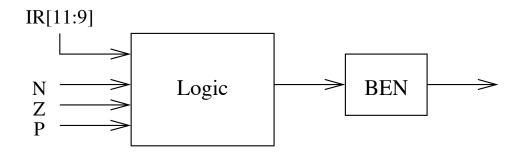
- How many cycles does the fastest instruction take?
- How many cycles does the slowest instruction take?
- Why does the BR take as long as it takes in the FSM?
- What determines the clock cycle?
- Is this a Mealy machine or a Moore machine?

LC-3b Datapath

- Patt and Patel, App C, Figure C.3
- Single-bus datapath design
 - At any point only one value can be "gated" on the bus (i.e., can be driving the bus)
 - Advantage: Low hardware cost: one bus
 - Disadvantage: Reduced concurrency if instruction needs the bus twice for two different things, these need to happen in different states
- Control signals (26 of them) determine what happens in the datapath in one clock cycle
 - Patt and Patel, App C, Table C.1







Signal Name	Signal Values	
LD.MAR/1: LD.MDR/1: LD.IR/1: LD.BEN/1:	NO, LOAD NO, LOAD NO, LOAD NO, LOAD	
LD.REG/1: LD.CC/1: LD.PC/1:	NO, LOAD NO, LOAD NO, LOAD	
GatePC/1: GateMDR/1: GateALU/1: GateMARMUX/1: GateSHF/1:	NO, YES NO, YES NO, YES NO, YES NO, YES	
PCMUX/2:	PC+2 BUS ADDER	;select pc+2 ;select value from bus ;select output of address adder
DRMUX/1:	11.9 R7	;destination IR[11:9] ;destination R7
SR1MUX/1:	11.9 8.6	;source IR[11:9] ;source IR[8:6]
ADDR1MUX/1:	PC, BaseR	
ADDR2MUX/2:	ZERO offset6 PCoffset9 PCoffset11	;select the value zero ;select SEXT[IR[5:0]] ;select SEXT[IR[8:0]] ;select SEXT[IR[10:0]]
MARMUX/1:	7.0 ADDER	;select LSHF(ZEXT[IR[7:0]],1) ;select output of address adder
ALUK/2:	ADD, AND, X	OR, PASSA
MIO.EN/1: R.W/1: DATA.SIZE/1: LSHF1/1:	NO, YES RD, WR BYTE, WORL NO, YES)

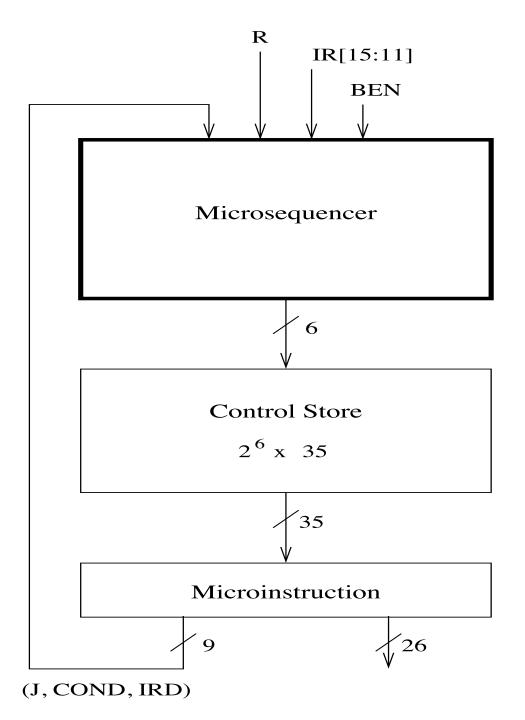
Table C.1: Data path control signals

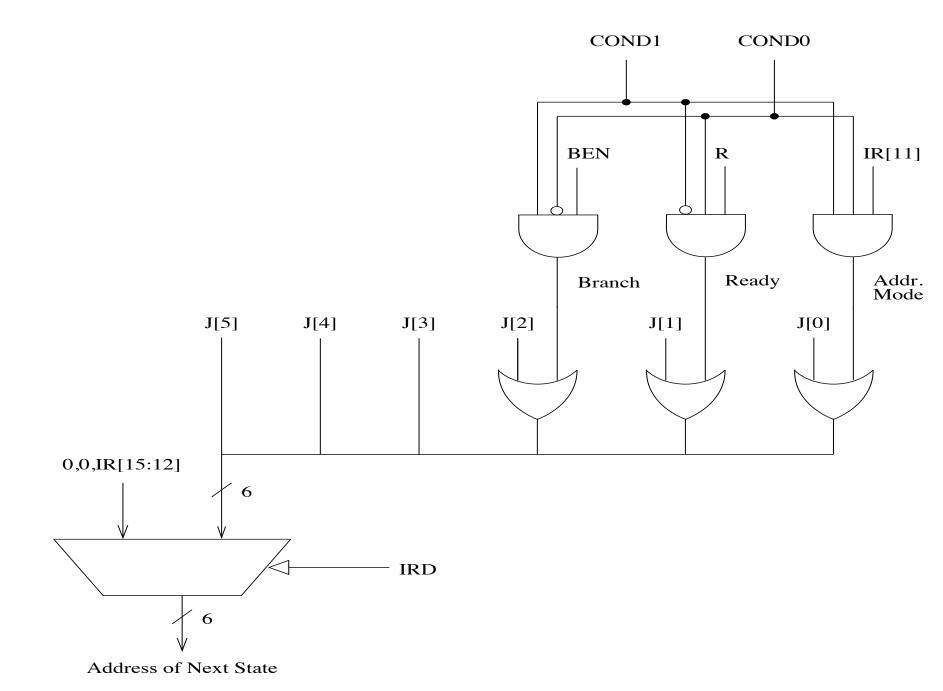
LC-3b Datapath: Some Questions

- How does instruction fetch happen in this datapath according to the state machine?
- What is the difference between gating and loading?
- Is this the smallest hardware you can design?

LC-3b Microprogrammed Control Structure

- Patt and Patel, App C, Figure C.4
- Three components:
 - Microinstruction, control store, microsequencer
- Microinstruction: control signals that control the datapath (26 of them) and determine the next state (9 of them)
- Each microinstruction is stored in a unique location in the control store (a special memory structure)
- Unique location: address of the state corresponding to the microinstruction
 - Remember each state corresponds to one microinstruction
- Microsequencer determines the address of the next microinstruction (i.e., next state)





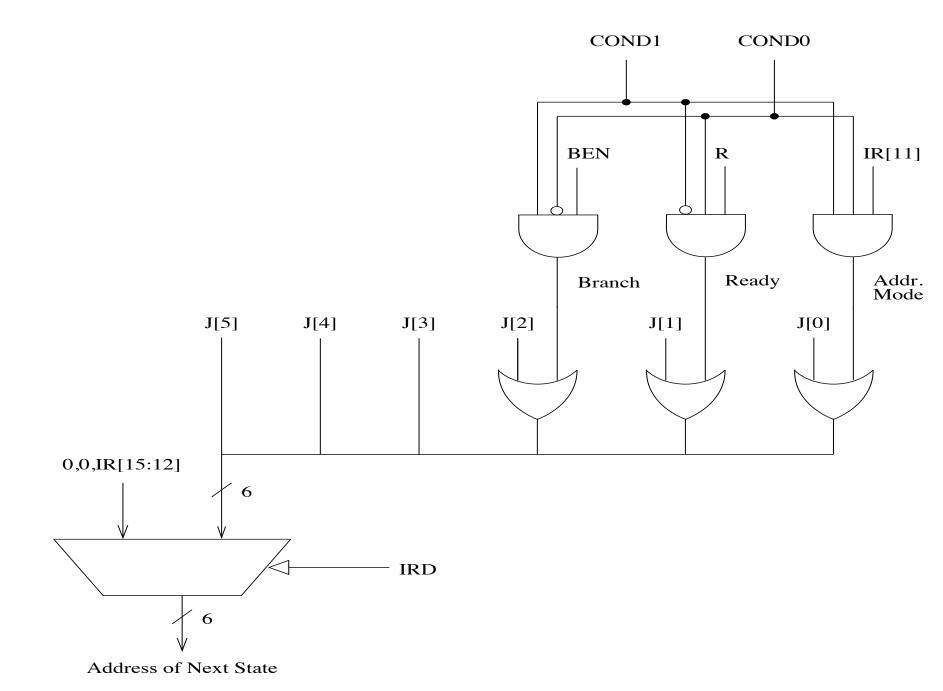
																			£	5 ^V					Д		4							
_		6						-	2	Q Q	÷ &	i d	ی دی	ي ج	ئو ن	پ ن	De S	÷		, \$	St.	Ś	\$	ST CE	A STANDARD STANDARD	A S	چ	A TOP	4	5	P. W. O.	Á		, <u>.</u>
\$	څ	3		~	,			9	9	\$	3	S.	3	3	ويقي	المحقق	ر نقعی	ر روس	على الم	, &	,	\$	ويجي	Ş	4	₹,	Źż	4	γ,	\$0	4.	4	\$,
	-	П		1	1	т -	1	П													Т	-1			П	T		Т	Т				Г	000000 (State 0)
	-	T		1	1	1	1														1	T				T		-	寸					000001 (State 1)
	-	T		1	1	1	1														T							-	寸					000010 (State 2)
	- '			1	1	'	1													_								-						000011 (State 3)
	_ '		_			<u>ا</u>	_																											000100 (State 4)
_		_	_	<u>'</u>			'														_	_			L'	_			4					000101 (State 5)
_		_				<u>. </u>															_	_			Ľ				_					000110 (State 6)
_		4		· -	· -																_	_	_		L.	_			4					000111 (State 7)
\dashv	-	-	_	_	-	-	1	H				_		-	_			-		-	\dashv	-	\dashv		-	-	-	_	\dashv	\dashv	-		-	001000 (State 8) 001001 (State 9)
\dashv		\dashv	_	1	1	_	1	H				_		-		_					\dashv	\dashv	\dashv		—	\dashv			\dashv	\dashv	\dashv	_		001001 (State 9)
\dashv	- 1	_		1	1	1	т	H												-	\dashv	\dashv	\dashv		 	T			\dashv				H	001011 (State 11)
	-			ı	1	1	1														_					T			寸					001100 (State 12)
	- 1			ı	ı	1	1													- 1								-	一					001101 (State 13)
				'	1	'	'																											001110 (State 14)
_	<u>'</u>		_	! !	! !	<u> </u>		Ш													_	_	_		Ľ	_			4					001111 (State 15)
\dashv	<u> </u>	4		<u>. </u>	·	.	-	┡	_			_		_	_	_		-			\dashv	-	\dashv		<u> </u>	-	_		\dashv	_	_		-	010000 (State 16)
\dashv		-	_																		\dashv	-	-		-	-	_		+	\dashv				010001 (State 17) 010010 (State 18)
\dashv	-	-	_	1			Т													-	\dashv	-	\dashv		 			-	\dashv	\dashv				010010 (State 18) 010011 (State 19)
\dashv	-	\dashv	_	1	1		т	H			H										\dashv	\dashv	\dashv		-	\dashv	-	-	\dashv	\dashv				01011 (State 19)
\dashv	- 1	+		1	1	1	ı													-	_	_			H	_		-	\dashv	_				010100 (State 20)
\neg			_	1	т —	т —	Т	H													寸	\neg	\neg			T			\dashv					010110 (State 22)
	-	T		1	1	1	1													Т	1				Т			-	寸					010111 (State 23)
				_	_	_																												011000 (State 24)
_			_	<u>'</u>		<u>'</u>	<u>'</u>														_				<u>'</u>				_					011001 (State 25)
		4																			_	_			Ľ	_			_					011010 (State 26)
_					· -			L												÷	_	_	_		Ľ	_			\dashv					011011 (State 27)
\dashv		4	_	1	1			L												-	_	\dashv	_			_			\dashv	-				011100 (State 28)
\dashv	_	\dashv						H						-						-	\dashv	\dashv	\dashv		—	\dashv	-	_	+	\dashv	-			011101 (State 29)
\dashv	-	-		1	ı		ı													-	-		_		 	-		-	\dashv	\dashv				011110 (State 30) 011111 (State 31)
\dashv	-	_	_	1	1	1	Т	H												-	\dashv	\dashv	\dashv		Η-	_		-	\dashv				\vdash	100000 (State 32)
	- 1			1	1	1	1													-	T				_	T		-	T					100001 (State 33)
				'	'		'																					'						100010 (State 34)
			_	_		<u>' </u>	_																											100011 (State 35)
_		_						Ш													_	_			Ľ	_			\dashv					100100 (State 36)
_		4		· -	· -									_						÷	_	_	_		L.	_	_		_	_	_			100101 (State 37)
-		-					1													_	-		-		—			-	\dashv	_				100110 (State 38)
\dashv	-	\dashv	_	-	-	_	1							_						-	\dashv	-	\dashv		<u> </u>	-	_	-	\dashv	\dashv	_		_	100111 (State 39) 101000 (State 40)
	-	\dashv	_	1	1	1	1													-	\dashv	\dashv				7		-	\dashv					101000 (State 40)
T	- 1			1	1		т —													- 1	7	T				T			\dashv					101010 (State 42)
				_	_	_	1													_ '	╛				Ľ									101011 (State 43)
	- 1			1	1	1	1																		'			- '						101100 (State 44)
				_	_	_															\Box				╚	耳			\Box					101101 (State 45)
_]		_[_					L		Ш	Ш		Ш			Ш	Ш				_[[_]		<u> </u>	[_[_]				101110 (State 46)
_	÷	4				.		⊢	_	Н	Н		\vdash	_	\vdash		\vdash	Щ		<u> </u>	\dashv	_	\dashv		 '	\dashv	_		\dashv	4	_		<u> </u>	101111 (State 47)
\dashv	÷	-	_		1	1	1	\vdash	-	Н	\vdash			-				Н	\vdash	Ť	\dashv	-	-		H	\dashv	\dashv		+	\dashv	-		\vdash	110000 (State 48) 110001 (State 49)
\dashv	-	\dashv	_	1	1		1	H			H							\vdash		-	\dashv	\dashv	\dashv		 	\dashv	-	-	\dashv	\dashv				110001 (State 49) 110010 (State 50)
\dashv	-	-	_	1	1	-	1	Н		Н	Н		Н	\exists	Н			H	Н	-	\dashv	\dashv	\dashv			\dashv	_	_	\dashv	\dashv	\exists			110010 (State 50)
\dashv	- 1	\dashv	_	1	1	1	ı								П			П	Н	-	\dashv					7	7	-	\dashv	\dashv				110100 (State 52)
\dashv		\exists	_	1	1		_	Г			П										\dashv	\neg	\neg			寸	\exists		\dashv	\exists				110101 (State 53)
			_	_	1	_	_														Ţ				Ľ	丁			丁	╗				110110 (State 54)
	'		_	_	_	_	_													'								'						110111 (State 55)
		_[_	_										\Box						7	4	\Box	\Box		Γ.		\Box		4	_]	\Box			111000 (State 56)
_	- 1	4									Ш									- ;	\dashv	_			L.	_			_	_			_	111001 (State 57)
\dashv	<u> </u>	4	 _				1	H	-		Н		Н							·	\dashv	\dashv	-		H	\dashv	4		+	\dashv	-		_	111010 (State 58)
\dashv		\dashv	_				1	\vdash			Н	H	Н			H	H	\vdash		-	\dashv	-	-		H	\dashv		_	+	\dashv	_		-	111011 (State 59) 111100 (State 60)
\dashv		-	_					Н			H	Н	Н	\dashv	\vdash	Н		H	\vdash	-	\dashv	\dashv	-	\vdash	H	\dashv	-	_	+	\dashv	\dashv		\vdash	111100 (State 60) 111101 (State 61)
\dashv	-	\dashv	_	1	1		1	Н			Н							\vdash	\vdash	Т	\dashv	\dashv	\dashv		-	\dashv		_	\dashv	\dashv				111101 (State 61) 111110 (State 62)
\exists	- 1	7	_	1		1	1	t		П	П		П				П	П		-	\dashv	7				T		-	\dashv					111111 (State 63)

LC-3b Microsequencer

- Patt and Patel, App C, Figure C.5
- The purpose of the microsequencer is to determine the address of the next microinstruction (i.e., next state)
- Next address depends on 9 control signals

Signal Name	Signal Val	ues
J/6: COND/2:	COND ₀ COND ₁ COND ₂ COND ₃	;Unconditional ;Memory Ready ;Branch ;Addressing Mode
IRD/1:	NO, YES	

Table C.2: Microsequencer control signals



The Microsequencer: Some Questions

- When is the IRD signal asserted?
- What happens if an illegal instruction is decoded?
- What are condition (COND) bits for?
- How is variable latency memory handled?
- How do you do the state encoding?
 - Minimize number of state variables
 - Start with the 16-way branch
 - Then determine constraint tables and states dependent on COND

An Exercise in Microprogramming

Handouts

- 7 pages of Microprogrammed LC-3b design
- http://www.ece.cmu.edu/~ece447/s13/doku.php? id=manuals
- http://www.ece.cmu.edu/~ece447/s13/lib/exe/fetch.php? media=lc3b-figures.pdf

A Simple LC-3b Control and Datapath

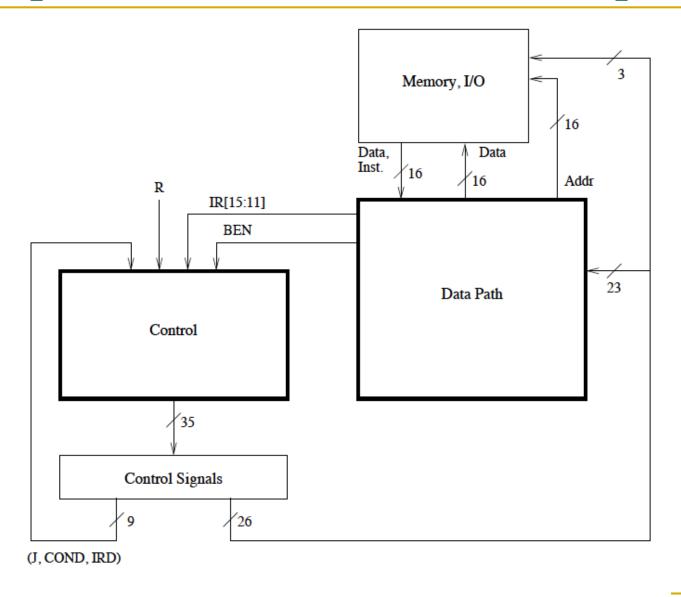
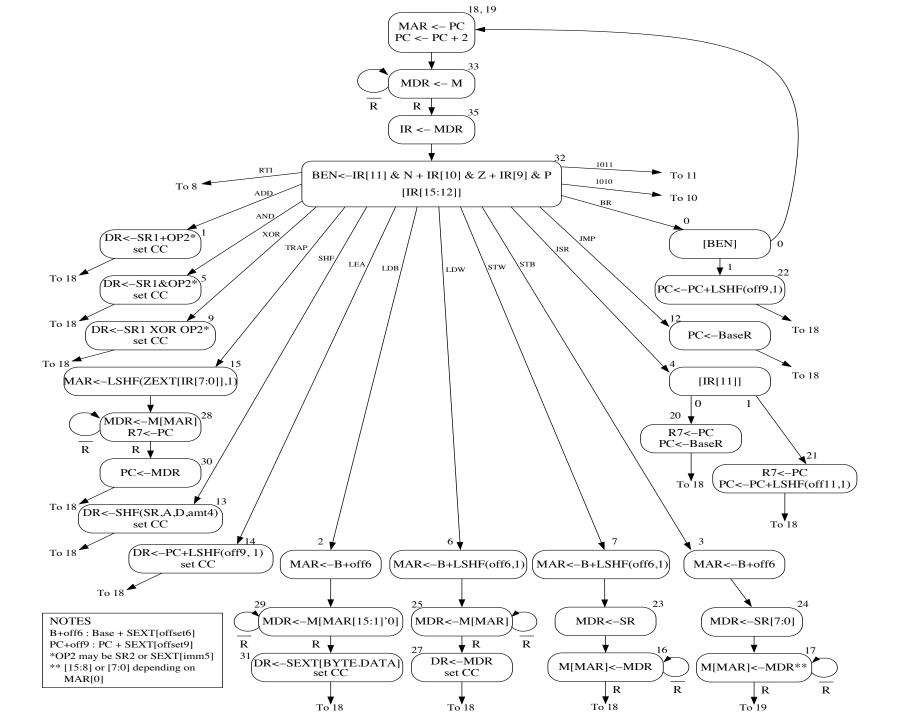
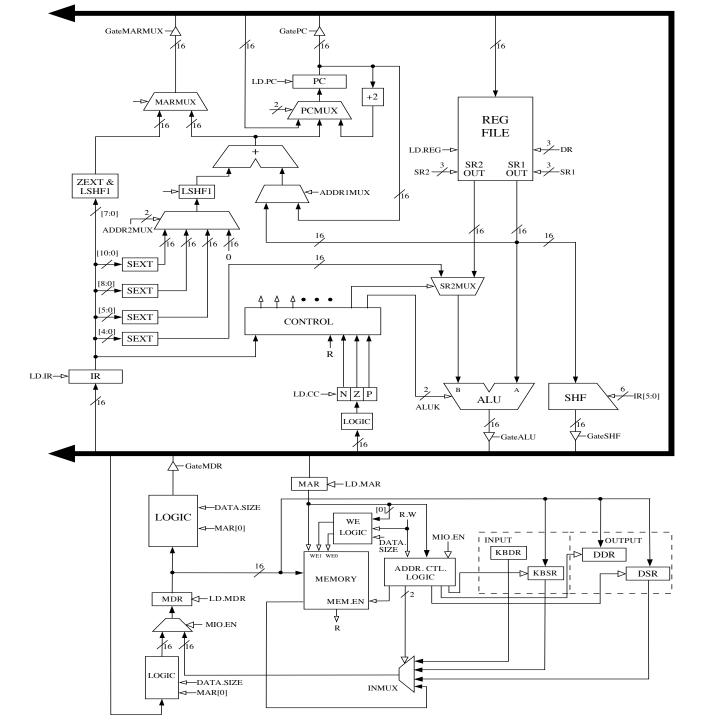
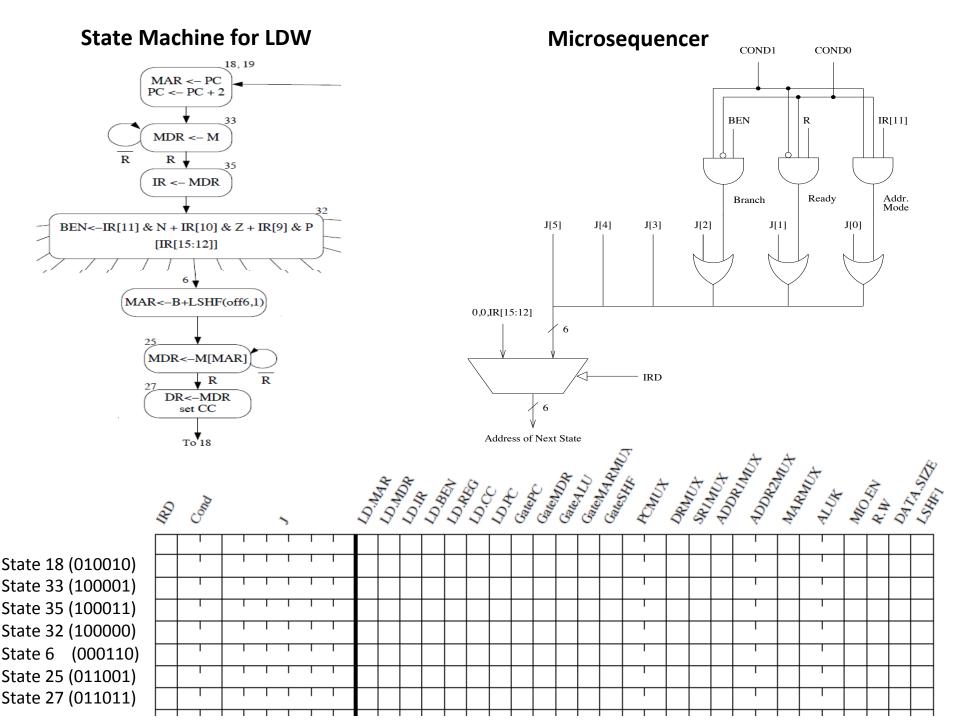
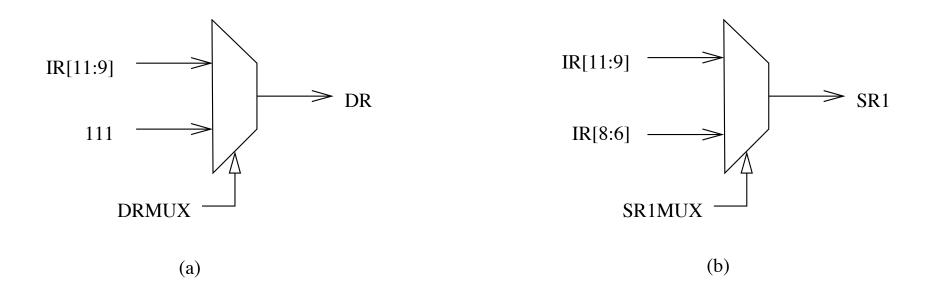


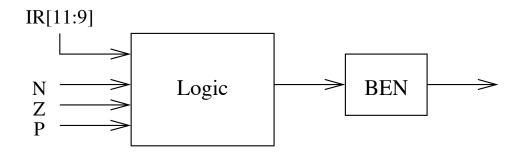
Figure C.1: Microarchitecture of the LC-3b, major components





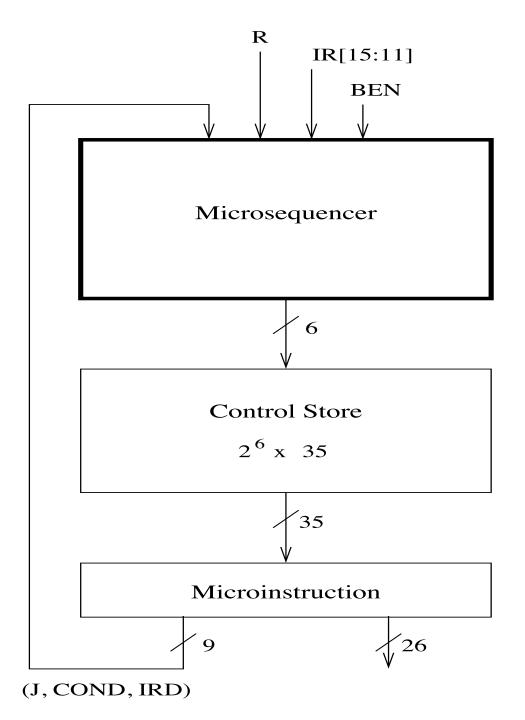


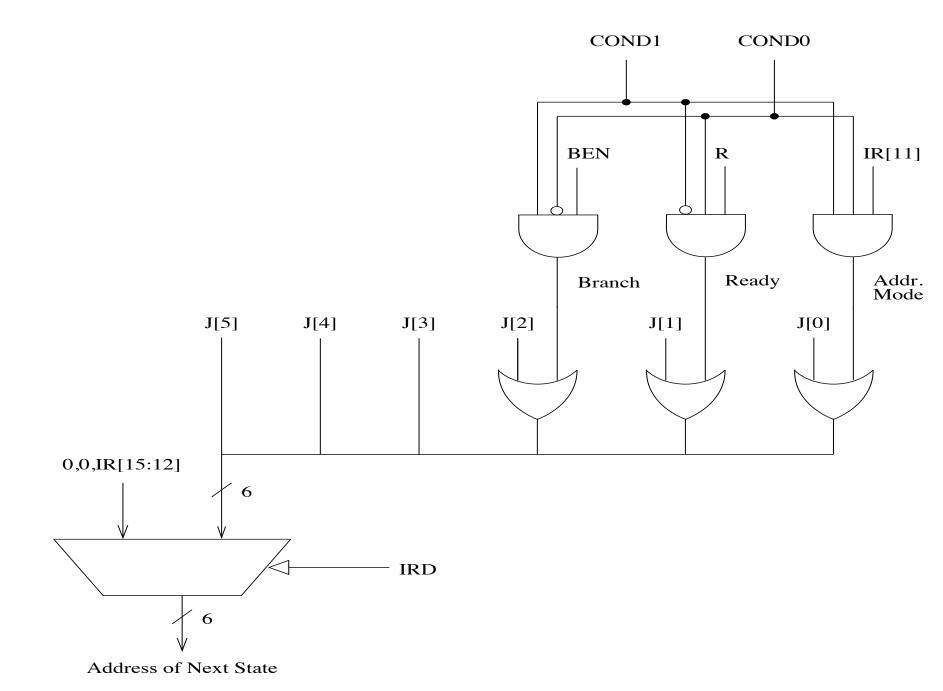




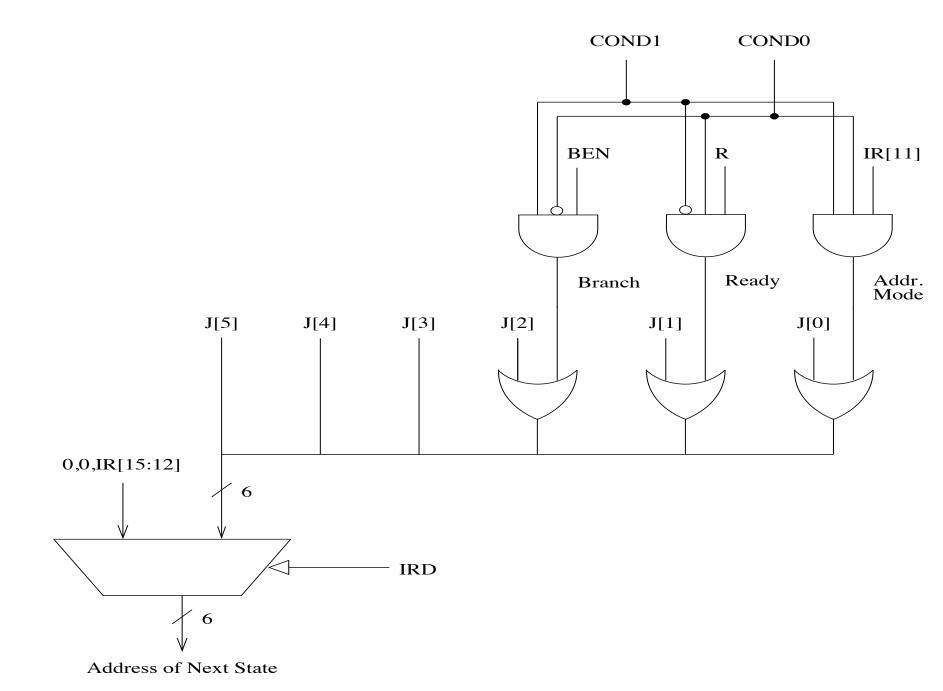
Signal Name	Signal Values	
LD.MAR/1: LD.MDR/1: LD.IR/1: LD.BEN/1:	NO, LOAD NO, LOAD NO, LOAD NO, LOAD	
LD.REG/1: LD.CC/1: LD.PC/1:	NO, LOAD NO, LOAD NO, LOAD	
GatePC/1: GateMDR/1: GateALU/1: GateMARMUX/1: GateSHF/1:	NO, YES NO, YES NO, YES NO, YES NO, YES	
PCMUX/2:	PC+2 BUS ADDER	;select pc+2 ;select value from bus ;select output of address adder
DRMUX/1:	11.9 R7	;destination IR[11:9] ;destination R7
SR1MUX/1:	11.9 8.6	;source IR[11:9] ;source IR[8:6]
ADDR1MUX/1:	PC, BaseR	
ADDR2MUX/2:	ZERO offset6 PCoffset9 PCoffset11	;select the value zero ;select SEXT[IR[5:0]] ;select SEXT[IR[8:0]] ;select SEXT[IR[10:0]]
MARMUX/1:	7.0 ADDER	;select LSHF(ZEXT[IR[7:0]],1) ;select output of address adder
ALUK/2:	ADD, AND, X	OR, PASSA
MIO.EN/1: R.W/1: DATA.SIZE/1: LSHF1/1:	NO, YES RD, WR BYTE, WORL NO, YES)

Table C.1: Data path control signals





																			£	5 ^V					Д		4							
_		6						-	2	Q Q	÷ &	i d	ی دی	ي ج	ئو ن	پ ن	De S	÷		, \$	St.	Ś	\$	ST CE	A STANDARD STANDARD	A S	چ	A TOP	4	5	P. W. O.	Á		, <u>.</u>
\$	څ	3		~	,			9	9	\$	\$	S.	3	3	ويجي	المحقق	ر نقعی	ر روس	على الم	, &	,	\$	وجي	Ş	4	₹,	Źż	4	γ,	\$0	4.	4	\$,
	-	П		1	1	т -	1	П													Т	-1			П	T		Т	Т				Г	000000 (State 0)
	-	T	_	1	1	1	1														1	T				T		-	寸					000001 (State 1)
	-	T		1	1	1	1														T							-	寸					000010 (State 2)
	- '			1	1	'	1													_								-						000011 (State 3)
	_ '		_			<u>ا</u>	_																											000100 (State 4)
_		_	_	<u>'</u>			'														_	_			L'	_			4					000101 (State 5)
_		_				<u>. </u>															_	_			Ľ				_					000110 (State 6)
_		4		· -	· -																_	_	_		L.	_			4					000111 (State 7)
\dashv	-	-	_	_	-	-	1	H						-	_			_		-	\dashv	-	\dashv		-	-	-	_	\dashv	\dashv	-		-	001000 (State 8) 001001 (State 9)
\dashv		\dashv	_	1	1	_	1	H				_		-		_					\dashv	\dashv	\dashv		—	\dashv			\dashv	\dashv	\dashv	_		001001 (State 9)
\dashv	- 1	_		1	1	1	т	H												-	\dashv	\dashv	\dashv		 	T			\dashv				H	001011 (State 11)
	-			ı	1	1	1														_					T			寸					001100 (State 12)
	- 1			ı	ı	1	1													- 1								-	一					001101 (State 13)
				'	1	'	'																											001110 (State 14)
_	<u>'</u>		_	! !	! !			Ш													_	_	_		Ľ	_			4					001111 (State 15)
\dashv	<u> </u>	4		<u>. </u>	·	.	-	┡	_			_		_	_	_		-			\dashv	-	\dashv		<u> </u>	-	_		\dashv	4	_		-	010000 (State 16)
\dashv		-	_								\vdash										\dashv	-	-		-	-	_		+	\dashv				010001 (State 17) 010010 (State 18)
\dashv	-	-	_	1			Т													-	\dashv	-	\dashv		 			-	\dashv	\dashv				010010 (State 18) 010011 (State 19)
\dashv	-	\dashv	_	1	1		т —	H			H										\dashv	\dashv	\dashv		-	\dashv	-	-	\dashv	\dashv				01011 (State 19)
\dashv	- 1	+		1	1	1	ı													-	_	_			H	_		-	\dashv	_				010100 (State 20)
\neg			_	1	т —	т —	Т	H													寸	\neg	\neg			T			\dashv					010110 (State 22)
	-	T		1	1	1	1													Т	T				Т			- 1	寸					010111 (State 23)
				_	_	_																												011000 (State 24)
_			_	<u>'</u>		<u>'</u>	<u>'</u>																		<u>'</u>				_					011001 (State 25)
		4																			_	_			Ľ	_			_					011010 (State 26)
_								L												÷	_	_	_		Ľ	_			\dashv					011011 (State 27)
\dashv		4	_	1	1			L												-	_	\dashv	_			_			\dashv	-				011100 (State 28)
\dashv	_	\dashv						H						-						-	\dashv	\dashv	\dashv		—	\dashv	-	_	+	\dashv	-			011101 (State 29)
\dashv	-	-		1	ı		ı													-	-		_		 	-		-	\dashv	\dashv				011110 (State 30) 011111 (State 31)
\dashv	-	_	_	1	1	1	Т	H												-	\dashv	\dashv	\dashv		Η-	_		-	\dashv				\vdash	100000 (State 32)
	- 1			1	1	1	1													-	T				_	T		-	T					100001 (State 33)
				'	'		'																					'						100010 (State 34)
			_	_		<u>' </u>	_																											100011 (State 35)
_		_						Ш													_	_			Ľ	_			\dashv					100100 (State 36)
_		4		· -	· -									_						÷	-	_	_		L.	_	_		_	_	_			100101 (State 37)
-		-					1													_	-		-		—			-	-	-				100110 (State 38)
\dashv	-	\dashv	_	-	-	_	1							_						-	\dashv	-	\dashv		<u> </u>	-	_	-	\dashv	\dashv	_		_	100111 (State 39) 101000 (State 40)
	-	\dashv	_	1	1	1	1													-	\dashv	\dashv				7		-	\dashv					101000 (State 40)
T	- 1			1	1		т —													- 1	7	T	_			T			\dashv					101010 (State 42)
				_	_	_	1													_ '	╛				Ľ									101011 (State 43)
	- 1			1	1	1	1																		'			-						101100 (State 44)
				_	_	_															\Box				╚	耳			\Box					101101 (State 45)
_]		_[_					L		Ш	Ш		Ш			Ш	Ш				_[[_]		<u> </u>	[_[_]				101110 (State 46)
_	÷	4				.		⊢	_	Н	Н		\vdash	_	\vdash		\vdash	Щ		<u> </u>	\dashv	_	\dashv		 '	\dashv	_	÷	\dashv	4	_		<u> </u>	101111 (State 47)
\dashv	÷	-	_		1	1	1	\vdash	-	Н	\vdash			-				Н	\vdash	Ť	\dashv	-	-		H	\dashv	\dashv		+	\dashv	-		\vdash	110000 (State 48) 110001 (State 49)
\dashv	-	\dashv	_	1	1		1	H			H							\vdash		-	\dashv	\dashv	\dashv		 	\dashv	-	-	\dashv	\dashv				110001 (State 49) 110010 (State 50)
\dashv	-	-	_	1	1	-	1	Н		Н	Н		Н	\exists	Н			H	Н	-	\dashv	\dashv	\dashv			\dashv	_	_	\dashv	\dashv	\exists			110010 (State 50)
\dashv	- 1	\dashv	_	1	1	1	ı								П			П	Н	-	\dashv					7	7	-	\dashv	\dashv				110100 (State 52)
\dashv		\exists	_	1	1		_	Г			П										\dashv	\neg	\neg			寸	\exists		\dashv	\exists				110101 (State 53)
			_	_	1	_	_													_ '	Ţ				Ľ	丁			丁	╗				110110 (State 54)
	'		_	_	_	_	_													'								'						110111 (State 55)
		_[_	_										\Box						7	4	\Box	\Box		Γ.		\Box		4	_]	\Box			111000 (State 56)
_	- 1	4									Ш									- ;	\dashv	_			L.	_			_	_			_	111001 (State 57)
\dashv	<u> </u>	4	 _				1	H	-		Н		Н							·	\dashv	\dashv	-		H	\dashv	4		+	\dashv	-		_	111010 (State 58)
\dashv		\dashv	_				1	\vdash			Н	H	Н			H	H	\vdash		-	\dashv	-	-		H	\dashv		_	+	\dashv	_		-	111011 (State 59) 111100 (State 60)
\dashv		-	_					Н			H	Н	Н	\dashv	\vdash	Н		H	\vdash	-	\dashv	\dashv	-	\vdash	H	\dashv	-	_	+	\dashv	\dashv		\vdash	111100 (State 60) 111101 (State 61)
\dashv	-	\dashv	_	1	1		1	Н			Н							\vdash	\vdash	Т	\dashv	\dashv	\dashv		-	\dashv		_	\dashv	\dashv				111101 (State 61) 111110 (State 62)
\exists	- 1	7	_	1		1	1	t		П	П		П				П	П		-	\dashv	7				T		-	\dashv					111111 (State 63)



End of the Exercise in Microprogramming

Homework 2

 You will write the microcode for the entire LC-3b as specified in Appendix C