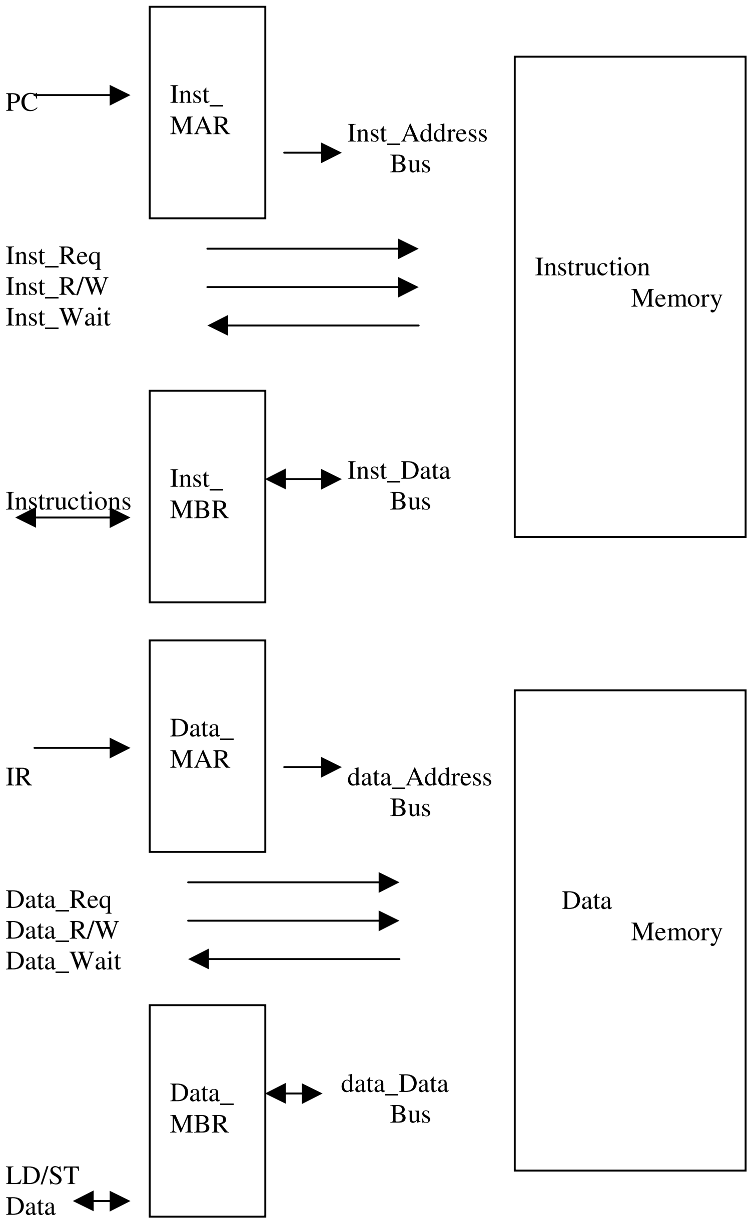
CS150 Fall 1997
HW #9 Solutions

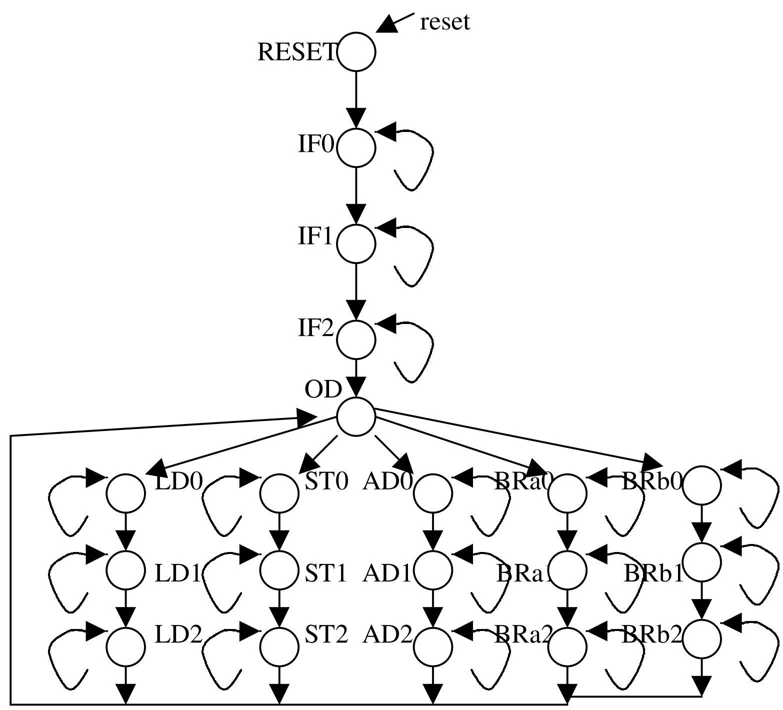
HW#9 Solutions

1) 11.4: Instruction pre-fetch

Harvard Archetecture:



State Diagram:



Register Transfer Operations: Mealy					
STATE:	I/O				NS
RESET:	Reset/0->PC				RESET
	Reset*/PC->inst_MAR,PC+1->PC				IF0
IF0:	Inst_Wait*				IF0
	Inst_Wait/Inst_MAR->Memory,				IF1
		1->Inst_R/W,			
		1->Inst_Req			
IF1:	Inst_Wait/Inst_MAR->Memory,				IF1
		1->Inst_R/W,			
		1->Inst_Req			
	Inst_Wait*/Mem->Inst_MBR				IF2
IF2:	Inst_Wait*				IF2
	Inst_Wait/Inst_MBR->IR				OD
OD:	IR<15:14>=00/IR<13:0>->data_MAR,				LD0
		PC->inst-MAR,			
		PC+1->PC			
	IR<15:14>=01/IR<13:0>->data_MAR,				ST0
		AC->data_MBR,			
		PC->inst_MAR,			
		PC+1->PC			
	IR<15:14>=10/IR,13:0>->data_MAR				AD0
		PC->inst-MAR,			
		PC+1->PC			
	IR<15:14>=11 AC<15>=1/				BRa0
		IR<13:0>->inst_MAR			

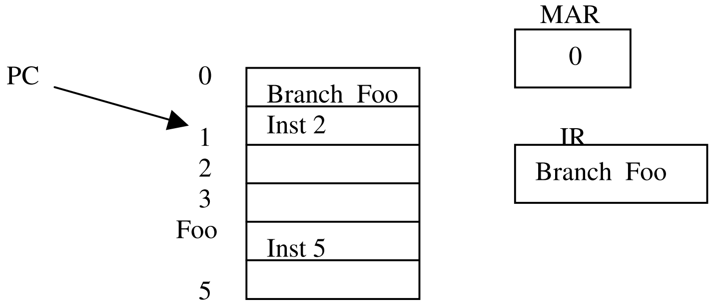
	IR<15:14>=11 AC<15>=0/		BRb0
		PC->inst_MAR,	
		PC+1->PC	
LD0:	inst_Wait* data_Wait*/		LD0
	inst_Wait data_Wait/		LD1
		Data_MAR->data_memory,	
		1->data_R/W,	
		1->data_Req	
		Inst_MAR->inst_memory,	
		1->inst_R/W	
		1->inst_Req	
LD1:	inst_Wait data_Wait/		LD1
	inst_Wait* data_Wait*/		LD2
		Data_memory->data_MBR	
		Inst_memory->inst_MBR	
LD2:	inst_Wait* data_Wait*/		LD2
	inst_Wait data_Wait/		OD
		Data_MBR->AC	
		Inst_MBR->IR	
ST0:	inst_Wait* data_Wait*/		ST0
	inst_Wait data_Wait/		ST1
		Data_MAR->data_memory	
		Data_MBR->data_memory	
		0->data_R/W	
		1->data_Req	
		Inst_MAR->inst-memory	
		1->inst_R/W	
		1->inst_Req	
ST1:	inst_Wait data_Wait/		ST1
		Data_MAR->data_memory	
		Data_MBR->data_memory	
		0->data_R/W	
		1->data_Req	
		inst_MAR->inst_memory	
		1->inst_R/W	
		1->inst_Req	
	inst_Wait* data_Wait*/		ST2
		0->data_Req	
		inst_memory->inst_MBR	
ST2:	inst_Wait* data_Wait*/		ST2
	inst_Wait data_Wait/		OD
		inst_MBR->IR	
AD0:	inst_Wait* data_Wait*/		AD0
	inst_Wait data_Wait/		AD1
		data_MAR->data_memory	

		1->data_R/W		
		1->data_Req		
		inst_MAR->inst-memory		
		1->inst_R/W		
		1->inst_Req		
AD1:	inst_Wait data_Wait/			AD1
		data_MAR->data_memory		
		1->data_R/W		
		1->data_Req		
		inst_MAR->inst-memory		
		1->inst_R/W		
		1->inst_Req		
	inst_Wait* data_Wait*/			AD2
		data_memory->data_MBR		
		inst_memory->inst_MBR		
AD2:	inst_Wait* data_Wait*/			AD2
	inst_Wait data_Wait/			OD
		data_MBR+AC->AC		
		inst_MBR->IR		
BRb0	Inst_Wait*			BRb0
	Inst_Wait/			BRb1
		Inst_MAR->inst-memory		
		1->inst_R/W		
		1->inst_Req		
BRb1	Inst_Wait			BRb1
		Inst_MAR->inst-memory		
		1->inst_R/W		
		1->inst_Req		
	Inst_Wait*			BRb2
		Inst-memory->inst_MBR		
BRb2	Inst_Wait*			BRb2
	Inst_Wait			OD
		Inst_MBR->IR		
BRa0:	inst_Wait*			BRa0
	inst_Wait/			BRa1
		inst_MAR->inst-memory		
		1->inst_R/W		
		1->inst_Req		
		IR<13:0>->PC		
		PC+1->PC		
BRa1:	inst_Wait/			BRa1
		inst_MAR->inst-memory		
		1->inst_R/W		
		1->inst_Req		
	inst_Wait*/			BRa2
		inst_memory->inst_MBR		

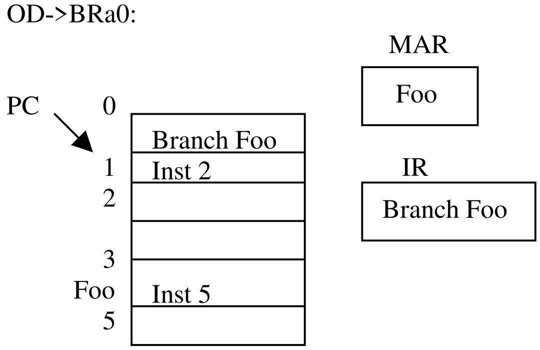
BRa2:	inst_Wait*/				BRa2
	inst_Wait/				OD
		inst_MBR->IR			

Diagrams for Branch:

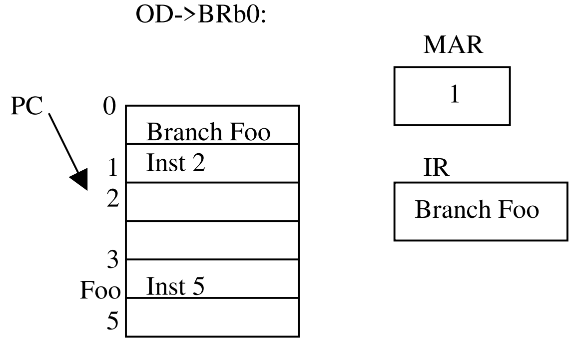
Stack after IF2:



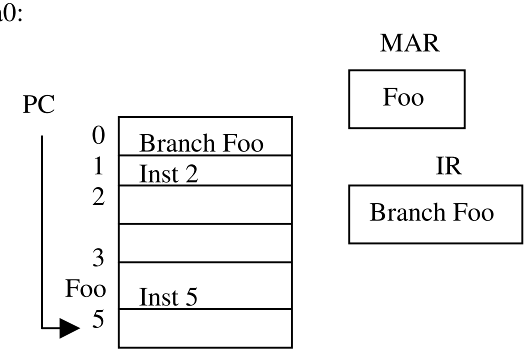
OD->BRa0:



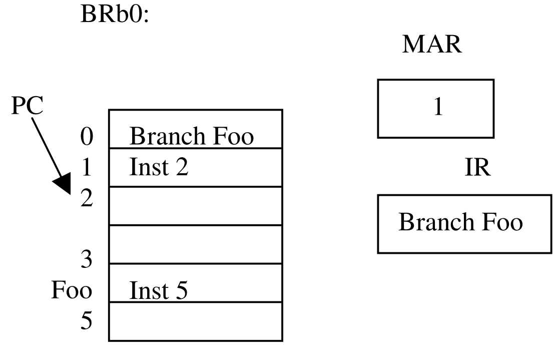
OD->BRb0:



BRa0:



BRb0:



2a) 11.13: Processor Datapath and Control

Microoperations:		grouped by sources
Data Bus->MBR (delayed)		
MBR->Data Bus (immediate)		
MBR->CBus (immediate)		
MBR->Bbus (immediate)		
MAR->Address Bus (immediate)		

PC->Bbus (immediate)
IR->Bbus (immediate)
SHFT->Cbus (immediate)
SHFT SHIFT (immediate)
SHFT PASS SHIFT (immediate)
ALU RES->Cbus (immediate)
ALU ADD, SUB, etc. (immediate)
ALU PASS A (immediate)
ALU PASS B (immediate)
Ra->Bbus (immediate)
Ra->Abus (immediate)
Rb->Bbus (immediate)
Rb->Abus (immediate)
Abus->ALU A (immediate)
Bbus->ALU B (immediate)
Bbus->SHFT (immediate)
Bbus->PC (delayed)
Bbus->MAR (delayed)
Cbus->Ra (delayed)
Cbus->Rb (delayed)
Cbus->IR (delayed)
Cbus->PC (delayed)
Cbus->MAR (delayed)
Cbus->MBR (delayed)

b) ADD Ra,(Rb)offset

Rb+offset->MAR	IR<offset>->Bbus (immediate)
	Rb->Abus (immediate)
	Abus->ALU A
	Bbus->ALU B
	ALU ADD
	ALU RES->Cbus (immediate)
	Cbus->MAR (delayed)
RAM[MAR]->MBR	MAR->address bus (immediate)
	data bus->MBR (delayed)
Ra+MBR->Ra	Ra->Abus (immediate)
	MBS->Bbus (immediate)
	Abus->ALU A (immediate)
	Bbus->ALU B (immediate)
	ALU ADD
	ALU RES->Cbus (immediate)
	Cbus->Ra

c) BRN Ra, (Rb)offset : If Ra < 0 then PC:=Rb + Offset

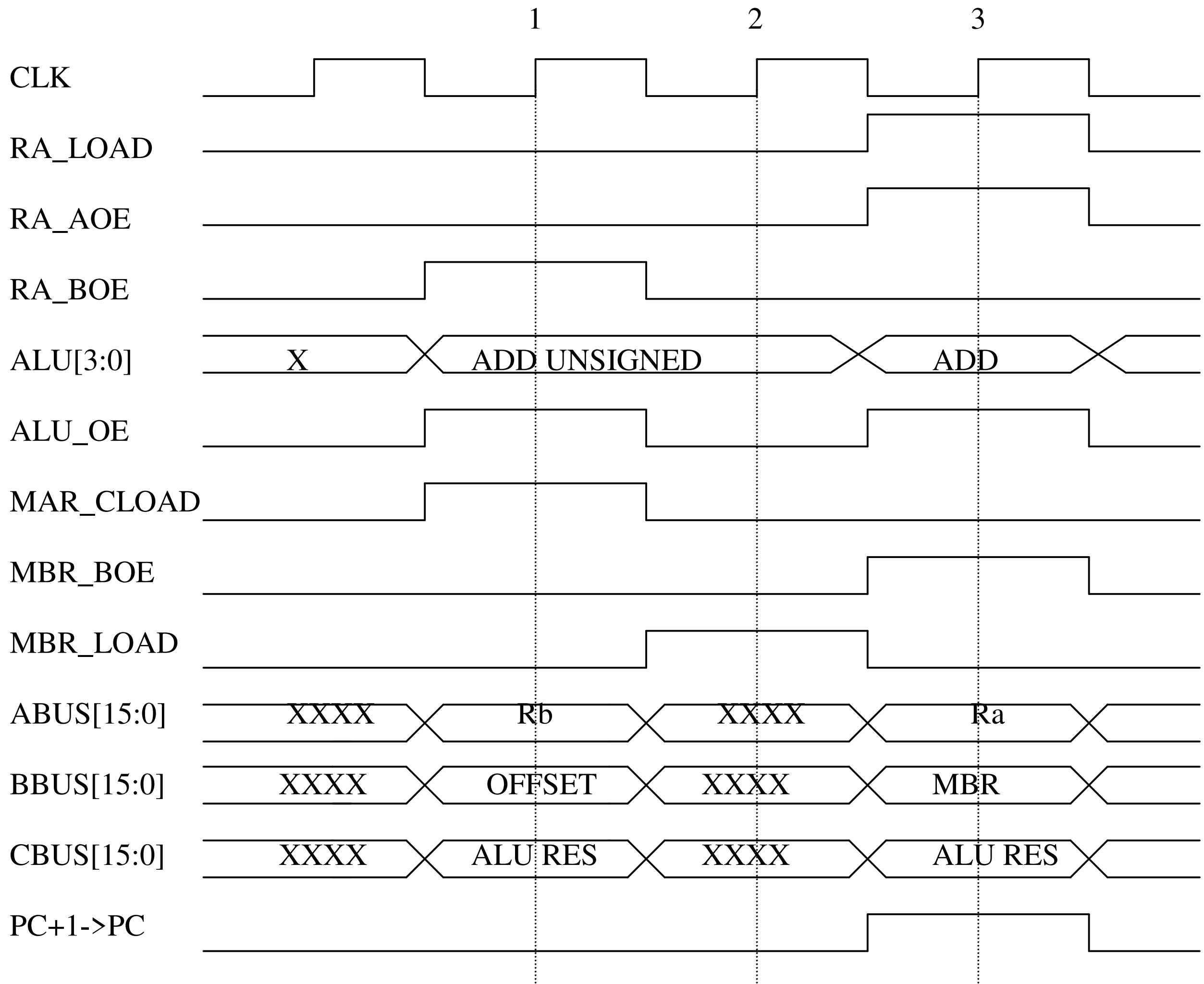
(if Ra[15]=1)	{	
	Rb+offset->PC	IR<offset>->Bbus(immediate)
		Rb->Abus (immediate)
		Abus->ALU A
		Bbus->ALU B
		ALU ADD SIGN_EXTEND
		ALU RES->Cbus (immediate)
		Cbus->MAR (delayed)
	}	

Assumptions:

Cbus->PC & MBR->Bbus microoperations were added.
For twos complement offset, ALU ADD is 16-bit sign-extended
Memory is always in read mode

3)

Control Signals:				
Ra_LOAD	Ra_AOE	Ra_BOE		
Rb_LOAD	Rb_AOE	Rb_BOE		
ALU[3:0]	ALU_OE			
IR_LOAD	IR_OE			
PC_BLOAD	PC_CLOAD	PC_OE	PC+1->PC	
MAR_BLOAD	MAR_CLOAD			
MBR_BOE	MBR_COE	MBR_W/R	MBR_DATA_OE	MBR_LOAD
SHFT_AMT[3:0]	SHFT_OE	SHFT_L/R		



1. Rb+offset->MAR
2. RAM[MAR]->MBR
3. MBR+Ra->Ra