Mohd Abdulla
932-104-847
ECE 375

H.W#4

Solutions:

Question 1)

a) The microoperation for the CPI Rd,K instruction.

EX: Rd - K

b) The microoperations and the RAL output.

Control Signals	IF	CPI
		EX
MJ	00	xx
MK	0	x
ML	0	x
IR_en	1	x
PC_en	1	0
PCh_en	0	0
PCl_en	0	0
NPC_en	1	x

SP_en	0	0
DEMUX	x	x
MA	x	0
MB	x	x
ALU_f	xxxx	0010
MC	xx	xx
RF_wA	0	0
RF_wB	0	0
MD	x	x
ME	x	x
DM_r	x	x
DM_w	0	0
MF	x	x
MG	x	x
Adder_f	xx	xx
Inc_Dec	x	x

MH	x	x
MI	x	x

RAL Output	CPI
	EX
wA	x
wB	x
rA	Rd
rB	x

Explanation: firs of all, the content of Rd is read from the Register File by providing the register identifier Rd to rA. At the same time, the constant K from the instruction (is routed through MUXA by setting it to 0. The ALU then performs a subtract operation (ALU_f = 0010), which then sets the appropriate condition flags (e.g., C, Z, N, and S flags in SREG). All other control signals can be don't cares except DM_w, RF_wA, and RF_wB, which need to be set to 0's so that the memory and the register file are not updated, and PC_en (as well as PCh_en and PCl_en) and SP_en are set to 0's to prevent PC and SP from being overwritten.

Question 2)

a) 1) DMAR ← Yh:YL, Yh:Yl ← Yh:Yl + 1

2) Rd ← M[DMAR]

b)

Control Signals	IF	LD Rd, Y+	
		EX1	EX2
MJ	0	xx	xx
MK	0	x	x
ML	0	x	x
IR_en	1	0	x
PC_en	1	0	0
PCh_en	0	0	0
PCl_en	0	0	0
NPC_en	1	x	x
SP_en	0	0	0
DEMUX	x	x	x
MA	x	x	x
MB	x	x	1
ALU_f	xxxx	xxxx	xxxx
MC	xx	01	01
RF_wA	0	1	0
RF_wB	0	1	1
MD	x	x	x
ME	x	x	1
DM_r	x	x	1
DM_w	0	0	0
MF	x	x	x
MG	x	1	x
Adder_f	xx	01	xx
Inc_Dec	x	x	x
MH	x	0	x
MI	x	x	x

RAL Output	LD Rd, Y+	
	EX1	EX2
wA	Yh	x
wB	Yl	Rd
rA	Yh	x

Explanation:

1) first of all ,the contents of Yh and Yl are read from the Register File by providing Yh and Yl to rA and rB, respectively. Yh:Yl or Y is routed to DMAR by setting MH to 0. Y is incremented by one by the Address Adder by setting Adder_f to 01, and then latched onto YH and YL (via MUXC) by setting both RF_wA and RF_wB to 1s and providing Yh and Yl to wA and wB, respectively. All other control signals can be don't cares except DM_w, which needs to be set to 0 so that the memory is overwritten, and IR_en, PC_en, and SP_en, which are all set to 0's to prevent IR, PC, and SP, respectively, from being overwritten. The RAL output for rA and rB are set to Yh and Yl, respectively, so that the upper and lower bytes of Y can be read from the register file.

2) The content of DMAR is routed through MUXE and used to fetch the operand from Data Memory. The fetched operand is routed through MUXB and MUXC to the inB of the register file and written by setting RF_wB to 1. PC_en and SP_en, need to be set to 0 to prevent the PC register and the SP register, respectively, from being overwritten. The RAL output for wA has to be set to Rd because the loaded value from memory has to be written to the destination register.

Question 3)

a) 1) M[SP]←RARl,SP←SP-1

2) M[SP]←RARh,SP←SP-1,PC←NPC+sek

b)

Control Signals	IF	RCALL	
		EX1	EX2
MJ	0	x	1
MK	0	x	x
ML	0	x	x
IR_en	1	0	x
PC_en	1	x	1
PCh_en	0	0	0
PCl_en	0	0	0
NPC_en	1	0	x
SP_en	0	1	1
DEMUX	x	x	x
MA	x	x	x
MB	x	x	x
ALU_f	xxxx	xxxx	xxxx
MC	xx	01	01

RF_wA	0	0	0
RF_wB	0	0	0
MD	x	0	0
ME	x	0	0
DM_r	x	0	0
DM_w	0	1	1
MF	x	x	0
MG	x	x	0
Adder_f	xx	xx	00
Inc_Dec	x	1	1
MH	x	x	x
MI	x	0	1

RAL Output	RCALL	
	EX1	EX2
wA	x	x
wB	x	x
rA	x	x
rB	x	x

Question 4)

a) 1) SP ←SP +1

2) PCh←M[SP],SP←SP+1

3) PCl ← M[SP]

b)

Control Signals	IF	RET		
		EX1	EX2	EX3
MJ	0	x	x	x
MK	0	x	x	x
ML	0	x	x	x

IR_en	1	0	0	x
PC_en	1	0	0	0
PCh_en	0	0	1	0
PCl_en	0	0	0	1
NPC_en	1	x	x	x
SP_en	0	1	1	0
DEMUX	x	x	1	0
MA	x	x	x	x
MB	x	x	x	x
ALU_f	xxxx	xxxx	xxxx	xxxx
MC	xx	xx	xx	xx
RF_wA	0	0	0	0
RF_wB	0	0	0	0
MD	x	x	x	x
ME	x	x	0	0
DM_r	x	x	1	1
DM_w		0		0

	0		0	
MF	x	x	x	x
MG	x	x	x	x
Adder_f	xx	xx	xx	xx
Inc_Dec	x	0	0	x
MH	x	x	x	x
MI	x	x	x	x

Output EX1 EX2 EX3

wA	x	x	x
wB	x	x	x
rA	x	x	x
rB	x	x	x

Explanation :

1) first of all, the content of SP is routed to the increment and decrement unit, and incremented by setting Inc_Dec to zero. The incremented SP is then relatched onto SP by setting SP_en to one. For the other control signals can be “don’t cares” except RF_wA/RF_wB, DM_w, IR_en, and PC_en , also PCh_en and PCl_en, which all need to be set to 0’s to prevent the register file, Data Memory, IR, and PC being overwritten with unwanted values

2) The content of SP is routed to the increment and decrement unit, which means it incremented by setting Inc_Dec to zero. The incremented SP is then relatched onto SP by setting SP_en to one. At the same time, the Data Memory location pointed to by SP, which it has the higher byte of the return address, is read by providing SP as an address to the Data Memory by setting ME to zero and DM_r to one . For the read value, it will routed to DEMUX to the upper byte of PC, PCh by setting DEMUX to one and PCh_en to one. And all other control signals can be don’t cares except DM_w, IR_en, and PC_en, which all need to be set to 0’s to prevent the Data Memory, IR, and PC being overwritten with unwanted values

3) The Data Memory location pointed to by SP, M(SP), which is the lower byte of the return address, is read by providing SP as an address to the Data Memory by setting ME to 0 and DM_r to 1. The read value is then routed to DEMUX to the lower byte of PC, PCl, by setting DEMUX to 0 and PCl_en to 1. DM_w and PC_en, need to be set to 0’s to prevent the Data Memory and PC being overwritten with unwanted values. Note that IR_en can be “don’t care” since this is the last execute cycle and IR register will be overwritten in the Fetch cycle.