ISTANBUL TECHNICAL UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

BLG 222E
COMPUTER ORGANIZATION PROJECT REPORT

PROJECT NO	:	2
PROJECT DATE	:	28.05.2022 - 01.06.2022
LAB SESSION	:	FRIDAY - 10.30
GROUP NO	:	G56

GROUP MEMBERS:

150180089	:	Harun ÇİFCİ
150190724	:	Umut TÖLEK
150200705	:	Helin Aslı AKSOY

SPRING 2022

Contents

FRONT COVER

CONTENTS

1 INTRODUCTION [10 points] 1

2 MATERIALS AND METHODS [40 points] 1

2.1 Decode Modules . . . 1
2.1.1 Instruction with address reference . . . 1
2.1.2 Instruction without address reference . . . 1
2.2 BRA . . . 1
2.3 LD . . . 1
2.3.1 Immediate . . . 1
2.3.2 Direct . . . 2
2.4 ST . . . 2
2.5 MOV . . . 2
2.5.1 If source is RF . . . 2
2.5.2 If source is ARF . . . 3
2.5.3 If destination is RF . . . 3
2.5.4 If destination is ARF . . . 3
2.6 AND . . . 3
2.6.1 If source 1 is RF . . . 3
2.6.2 If source 1 is ARF . . . 3
2.6.3 Since we know Source2 is RF . . . 4
2.6.4 If destination is RF . . . 4
2.6.5 If destination is ARF . . . 4
2.7 OR . . . 4
2.7.1 If source 1 is RF . . . 4
2.7.2 If source 1 is ARF . . . 4
2.7.3 Since we know Source2 is RF . . . 5
2.7.4 If destination is RF . . . 5
2.7.5 If destination is ARF . . . 5
2.8 NOT . . . 5
2.8.1 If source is RF . . . 5
2.8.2 If source is ARF . . . 6
2.8.3 If destination is RF . . . 6

2.8.4 If destination is ARF 6
2.9 ADD 6
2.9.1 If source 1 is RF 6
2.9.2 If source 1 is ARF 6
2.9.3 Since we know Source2 is RF 7
2.9.4 If destination is RF 7
2.9.5 If destination is ARF 7
2.10 SUB 7
2.10.1 If source 1 is RF 7
2.10.2 If source 1 is ARF 7
2.10.3 Since we know Source2 is RF 8
2.10.4 If destination is RF 8
2.10.5 If destination is ARF 8
2.11 LSR 8
2.11.1 If source is RF 8
2.11.2 If source is ARF 9
2.11.3 If destination is RF 9
2.11.4 If destination is ARF 9
2.12 LSL 9
2.12.1 If source is RF 9
2.12.2 If source is ARF 10
2.12.3 If destination is RF 10
2.12.4 If destination is ARF 10
2.13 PUL 10
2.14 PSH 10
2.15 INC 11
2.15.1 If source is RF 11
2.15.2 If source is ARF 11
2.15.3 If destination is RF 11
2.15.4 If destination is ARF 11
2.16 DEC 12
2.16.1 If source is RF 12
2.16.2 If source is ARF 12
2.16.3 If destination is RF 12
2.16.4 If destination is ARF 13
2.17 BNE 13

3 RESULTS [15 points] 14

4 DISCUSSION [25 points] 16

5 CONCLUSION [10 points] 20

1 INTRODUCTION [10 points]

2 MATERIALS AND METHODS [40 points]

2.1 Decode Modules

With the help of these modules, we are decoding the Instruction set in two different way by looking if it is with or without address reference.

2.1.1 Instruction with address reference

With using this module we have decoded IR to 4 parts. We have taken first 8 bits as address part and assigned it. Then we have taken following 2 bits as register selector bits and assigned them. After that following one bit has assigned as address mode. After not caring following bit, we have taken last 4 bits as opcode and assigned them.

2.1.2 Instruction without address reference

With the help of that module we have decoded IR to 4 parts. We have taken first 4 bits as Source Register 2 and assigned it. Then we have taken following 4 bits as Source Register 1 and assigned them also. After that next 4 bits are assigned as Destination Register. And same with above part, last 4 bits are assigned as opcode.

2.2 BRA

We have implemented a branch operation according to this opcode(0x00) which loads Address value to PC ”Immediately”. In order to achieve this, we have taken first 8 bits of IR and selected IR output(0-7) at Multiplexer B with 01 value of selector. Then we have activated PC in ARF and then we have selected load function(10). In conclusion we have loaded Address bits of IR to the PC.

2.3 LD

We have implemented load operation for Direct and Immediate mode. To determine if our operation works as direct or immediate mode we have used addressing mode bit(IR[10]).

2.3.1 Immediate

We have taken first 8 bits of IR and selected IR output(0-7) at Multiplexer A with 00 value of selector. Then we have activated related Register in RF with Regsel. And then we have selected load function(10). In conclusion we have loaded Address bits of IR to the Rx.

2.3.2 Direct

Firstly, we have selected AR register as output D of ARF which is the input address of memory. We have enabled read mode of memory and activated memory by giving chip select bit 0. To load related registers we have selected memory output in Multiplexer A by giving 01 to selector of MUXA. Then we have activated related Register in RF with Regsel. And then we have selected load function(10). In conclusion we have loaded M[AR] to the Rx.

2.4 ST

We have selected related register by regsel bits of IR[9:8]. According to selected register we have selected Output B of RF. By using output channel B we have sent the value to ALU and take it as output without changing it. Then we sent output of alu to the memory. We have selected AR register as output D of ARF which is the input address of memory. Then we have activated write mode and chip select 0.

2.5 MOV

Firstly, we have selected our source register by decoding IR's SRCREG1 part(IR[4:7]) bits. Then we have determined whether it is RF or ARF by, checking IR[6]. If it equals 1 Register file is selected else Address register file selected. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]). Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation. Then we have loaded selected register. In order to determine if our destination is RF or ARF we have checked IR[10]. If it is 1 Register file is selected, else Address register file is selected. Then we have checked last 2 bits of DESTREG part of IR(IR[9:8]).

2.5.1 If source is RF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our MOV operation. Then we selected related register as output B of RF. Then we sent it directly to the ALU and take it as Output of ALU without changing it. After that to send it to RF and ARF we have selected ALU output in Multiplexer A and Multiplexer B by giving selector bits as 11.

2.5.2 If source is ARF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our MOV operation. Then we selected related register as output C of ARF. Then we sent it directly to the ALU by giving ARF out to Multiplexer C and selecting 0. And then we took it as Output of ALU without changing it. After that to send it back to ARF we have selected ALU output in Multiplexer B by giving selector bits as 11. In order to send it to RF we have used Multiplexer A. We have selected ARF out in MuxA(10).

2.5.3 If destination is RF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation and we have activated related register by using RFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.5.4 If destination is ARF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation and we have activated related register by using ARFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.6 AND

Firstly, we have selected our source register by decoding IR's SRCREG1 part(IR[4:7]) bits. Then we have determined whether it is RF or ARF by, checking IR[6].If it equals 1 Register file is selected else Address register file selected and MUXC has a selector according to this bit of IR[6].

2.6.1 If source 1 is RF

Multiplexer C has a selector 1 which gives the OutputA of RF. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]) in order to determine which register is our SRCREG1.

2.6.2 If source 1 is ARF

Multiplexer C has a selector 0 which gives the OutputC of ARF. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]) in order to determine which register is our SRCREG1.

2.6.3 Since we know Source2 is RF

For SRCREG2 we check first IR[2] in order to make sure it is a register in RF. After that we check IR[5:4] which register is SRCREG2 and we change RFOutBSel according to SRCREG2. After selecting inputs of ALU (A and B) we took it as Output of ALU by making "A AND B" operation in ALU with AluFunsel 0111.

2.6.4 If destination is RF

Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our AND operation. We set MUXA to OutAlu(11) which has the value (SRCREG1 AND SRCREG2) and we set RFFunsel to 10 which loads that value to DESTREG.

2.6.5 If destination is ARF

Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our AND operation. We set MUXB to OutAlu(11) which has the value (SRCREG1 AND SRCREG2) and we set ARFFunsel to 10 which loads that value to DESTREG.

2.7 OR

Firstly, we have selected our source register by decoding IR's SRCREG1 part(IR[4:7]) bits. Then we have determined whether it is RF or ARF by, checking IR[6].If it equals 1 Register file is selected else Address register file selected and MUXC has a selector according to this bit of IR[6].

2.7.1 If source 1 is RF

Multiplexer C has a selector 1 which gives the OutputA of RF. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]) in order to determine which register is our SRCREG1.

2.7.2 If source 1 is ARF

Multiplexer C has a selector 0 which gives the OutputC of ARF. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]) in order to determine which register is our SRCREG1.

2.7.3 Since we know Source2 is RF

For SRCREG2 we check first IR[2] in order to make sure it is a register in RF. After that we check IR[5:4] which register is SRCREG2 and we change RFOutBSel according to SRCREG2. After selecting inputs of ALU (A and B) we took it as Output of ALU by making "A OR B" operation in ALU with AluFunsel 1000.

2.7.4 If destination is RF

Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our AND operation. We set MUXA to OutAlu(11) which has the value (SRCREG1 OR SRCREG2) and we set RFFunsel to 10 which loads that value to DESTREG.

2.7.5 If destination is ARF

Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our AND operation. We set MUXB to OutAlu(11) which has the value (SRCREG1 OR SRCREG2) and we set ARFFunsel to 10 which loads that value to DESTREG.

2.8 NOT

Firstly, we have selected our source register by decoding IR's SRCREG1 part(IR[4:7]) bits. Then we have determined whether it is RF or ARF by, checking IR[6]. If it equals 1 Register file is selected else Address register file selected. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]). Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation. Then we have loaded selected register. In order to determine if our destination is RF or ARF we have checked IR[10]. If it is 1 Register file is selected, else Address register file is selected. Then we have checked last 2 bits of DESTREG part of IR(IR[9:8]).

2.8.1 If source is RF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our MOV operation. Then we selected related register as output B of RF. Then we sent it directly to the ALU and take it as Output of ALU by making "NOT B" operation in ALU with AluFunsel 0011. After that to send it to RF and ARF we have selected ALU output in Multiplexer A and Multiplexer B by giving selector bits as 11.

2.8.2 If source is ARF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our MOV operation. Then we selected related register as output C of ARF. Then we sent it directly to the ALU by giving ARF out to Multiplexer C and selecting 0. And then we took it as Output of ALU by making "NOT A" operation in ALU with AluFunsel 0010. After that to send it to RF and ARF we have selected ALU output in Multiplexer A and Multiplexer B by giving selector bits as 11. In order to send it to RF we have used Multiplexer A. We have selected ARF out in MuxA(10).

2.8.3 If destination is RF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation and we have activated related register by using RFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.8.4 If destination is ARF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation and we have activated related register by using ARFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.9 ADD

Firstly, we have selected our source register by decoding IR's SRCREG1 part(IR[4:7]) bits. Then we have determined whether it is RF or ARF by, checking IR[6].If it equals 1 Register file is selected else Address register file selected and MUXC has a selector according to this bit of IR[6].

2.9.1 If source 1 is RF

Multiplexer C has a selector 1 which gives the OutputA of RF. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]) in order to determine which register is our SRCREG1.

2.9.2 If source 1 is ARF

Multiplexer C has a selector 0 which gives the OutputC of ARF. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]) in order to determine which register is our SRCREG1.

2.9.3 Since we know Source2 is RF

For SRCREG2 we check first IR[2] in order to make sure it is a register in RF. After that we check IR[5:4] which register is SRCREG2 and we change RFOutBSel according to SRCREG2. After selecting inputs of ALU (A and B) we took it as Output of ALU by making "A + B" operation in ALU with AluFunsel 0100.

2.9.4 If destination is RF

Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our AND operation. We set MUXA to OutAlu(11) which has the value (SRCREG1 OR SRCREG2) and we set RFFunsel to 10 which loads that value to DESTREG.

2.9.5 If destination is ARF

Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our AND operation. We set MUXB to OutAlu(11) which has the value (SRCREG1 OR SRCREG2) and we set ARFFunsel to 10 which loads that value to DESTREG.

2.10 SUB

Firstly, we have selected our source register by decoding IR's SRCREG1 part(IR[4:7]) bits. Then we have determined whether it is RF or ARF by, checking IR[6].If it equals 1 Register file is selected else Address register file selected and MUXC has a selector according to this bit of IR[6].

2.10.1 If source 1 is RF

Multiplexer C has a selector 1 which gives the OutputA of RF. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]) in order to determine which register is our SRCREG1.

2.10.2 If source 1 is ARF

Multiplexer C has a selector 0 which gives the OutputC of ARF. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]) in order to determine which register is our SRCREG1.

2.10.3 Since we know Source2 is RF

For SRCREG2 we check first IR[2] in order to make sure it is a register in RF. After that we check IR[5:4] which register is SRCREG2 and we change RFOutBSel according to SRCREG2. After selecting inputs of ALU (A and B) we took it as Output of ALU by making "A - B" operation in ALU with AluFunsel 0110.

2.10.4 If destination is RF

Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our AND operation. We set MUXA to OutAlu(11) which has the value (SRCREG1 OR SRCREG2) and we set RFFunsel to 10 which loads that value to DESTREG.

2.10.5 If destination is ARF

Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our AND operation. We set MUXB to OutAlu(11) which has the value (SRCREG1 OR SRCREG2) and we set ARFFunsel to 10 which loads that value to DESTREG.

2.11 LSR

Firstly, we have selected our source register by decoding IR's SRCREG1 part(IR[4:7]) bits. Then we have determined whether it is RF or ARF by, checking IR[6]. If it equals 1 Register file is selected else Address register file selected. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]). Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation. Then we have loaded selected register. In order to determine if our destination is RF or ARF we have checked IR[10]. If it is 1 Register file is selected, else Address register file is selected. Then we have checked last 2 bits of DESTREG part of IR(IR[9:8]).

2.11.1 If source is RF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our MOV operation. Then we selected related register as output A of RF. Then we sent it directly to the ALU by giving RF out to Multiplexer C and selecting 1. And then we took it as Output of ALU by making "LSR A" operation in ALU with AluFunsel 1011. After that to send it to RF and ARF we have selected ALU output in Multiplexer A and Multiplexer B by giving selector bits as 11.

2.11.2 If source is ARF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our MOV operation. Then we selected related register as output C of ARF. Then we sent it directly to the ALU by giving ARF out to Multiplexer C and selecting 0. And then we took it as Output of ALU by making ”LSR A” operation in ALU with AluFunsel 1011. After that to send it to RF and ARF we have selected ALU output in Multiplexer A and Multiplexer B by giving selector bits as 11.

2.11.3 If destination is RF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation and we have activated related register by using RFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.11.4 If destination is ARF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation and we have activated related register by using ARFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.12 LSL

Firstly, we have selected our source register by decoding IR's SRCREG1 part(IR[4:7]) bits. Then we have determined whether it is RF or ARF by, checking IR[6]. If it equals 1 Register file is selected else Address register file selected. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]). Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation. Then we have loaded selected register. In order to determine if our destination is RF or ARF we have checked IR[10]. If it is 1 Register file is selected, else Address register file is selected. Then we have checked last 2 bits of DESTREG part of IR(IR[9:8]).

2.12.1 If source is RF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our MOV operation. Then we selected related register as output A of RF. Then we sent it directly to the ALU by giving RF out to Multiplexer C and selecting 1. And then we took it as Output of ALU by making ”LSL A” operation in ALU with AluFunsel 1010. After that to send it to RF and ARF we have selected ALU output in Multiplexer A and Multiplexer B by giving selector bits as 11.

2.12.2 If source is ARF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our MOV operation. Then we selected related register as output C of ARF. Then we sent it directly to the ALU by giving ARF out to Multiplexer C and selecting 0. And then we took it as Output of ALU by making ”LSL A” operation in ALU with AluFunsel 1010. After that to send it to RF and ARF we have selected ALU output in Multiplexer A and Multiplexer B by giving selector bits as 11.

2.12.3 If destination is RF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation and we have activated related register by using RFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.12.4 If destination is ARF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation and we have activated related register by using ARFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.13 PUL

Firstly, we have selected SP register as output D of ARF which is the input address of memory. We have enabled read mode of memory and activated memory by giving chip select bit 0. To load related registers we have selected memory output in Multiplexer A by giving 01 to selector of MUXA. Then we have activated related Register in RF with Regsel. And then we have selected load function(10). We have already selected SP register, so we have incremented it. In conclusion we have loaded M[SP] to the Rx and then incremented SP by 1.

2.14 PSH

We have selected related register by regsel bits of IR[9:8]. According to selected register we have selected Output B of RF. By using output channel B we have sent the value to ALU and take it as output without changing it. Then we sent output of alu to the memory. We have selected SP register as output D of ARF which is the input address of memory. Then we have activated write mode and chip select 0. Then we have decremented already selected SP register. In conclusion we have loaded Rx to M[SP] and then decremented SP by 1.

2.15 INC

Firstly, we have selected our source register by decoding IR's SRCREG1 part(IR[4:7]) bits. Then we have determined whether it is RF or ARF by, checking IR[6]. If it equals 1 Register file is selected else Address register file selected. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]). Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation. Then we have loaded selected register. In order to determine if our destination is RF or ARF we have checked IR[10]. If it is 1 Register file is selected, else Address register file is selected. Then we have checked last 2 bits of DESTREG part of IR(IR[9:8]).

2.15.1 If source is RF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our INC operation. Firstly, we have selected related register and incremented it with RFFunsel 01. Then we have sent it as output B of RF. After that we sent it directly to the ALU and take it as Output of ALU without changing it. After that to send it to RF and ARF we have selected ALU output in Multiplexer A and Multiplexer B by giving selector bits as 11.

2.15.2 If source is ARF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our INC operation.Firstly, we have selected related register and incremented it with ARFFunsel 01. Then we have sent it as output C of ARF. After that we sent it directly to the ALU by giving ARF out to Multiplexer C and selecting 0. And then we took it as Output of ALU without changing it. After that to send it back to ARF we have selected ALU output in Multiplexer B by giving selector bits as 11. In order to send it to RF we have used Multiplexer A. We have selected ARF out in MuxA(10).

2.15.3 If destination is RF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our INC operation and we have activated related register by using RFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.15.4 If destination is ARF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our INC operation and we have activated related register by using ARFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.16 DEC

Firstly, we have selected our source register by decoding IR's SRCREG1 part(IR[4:7]) bits. Then we have determined whether it is RF or ARF by, checking IR[6]. If it equals 1 Register file is selected else Address register file selected. Then we have checked last 2 bits of SRCREG1 part of IR(IR[5:4]). Then in order to select destination register, we have checked first two bits of DESTREG part of IR to determine which register will be the destination of our MOV operation. Then we have loaded selected register. In order to determine if our destination is RF or ARF we have checked IR[10]. If it is 1 Register file is selected, else Address register file is selected. Then we have checked last 2 bits of DESTREG part of IR(IR[9:8]).

2.16.1 If source is RF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our DEC operation. Firstly, we have selected related register and incremented it with RFFunsel 00. Then we have sent it as output B of RF. After that we sent it directly to the ALU and take it as Output of ALU without changing it. After that to send it to RF and ARF we have selected ALU output in Multiplexer A and Multiplexer B by giving selector bits as 11.

2.16.2 If source is ARF

We have checked first two bits of SRCREG1 part of IR to determine which register will be the source of our DEC operation.Firstly, we have selected related register and incremented it with ARFFunsel 00. Then we have sent it as output C of ARF. After that we sent it directly to the ALU by giving ARF out to Multiplexer C and selecting 0. And then we took it as Output of ALU without changing it. After that to send it back to ARF we have selected ALU output in Multiplexer B by giving selector bits as 11. In order to send it to RF we have used Multiplexer A. We have selected ARF out in MuxA(10).

2.16.3 If destination is RF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our DEC operation and we have activated related register by using RFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.16.4 If destination is ARF

We have checked first two bits of DESTREG part of IR to determine which register will be the destination of our DEC operation and we have activated related register by using ARFRegsel. Then we have loaded related register by giving RFFunsel 10.

2.17 BNE

We have implemented a conditional branch operation according to this opcode(0x0F) which loads Address value to PC "Immediately" if the zeroflag is equal 0. In order to achieve this, we have taken first 8 bits of IR and selected IR output(0-7) at Multiplexer B with 01 value of selector. Then we have activated PC in ARF and then we have selected load function(10). In conclusion we have loaded Address bits of IR to the PC.

3 RESULTS [15 points]

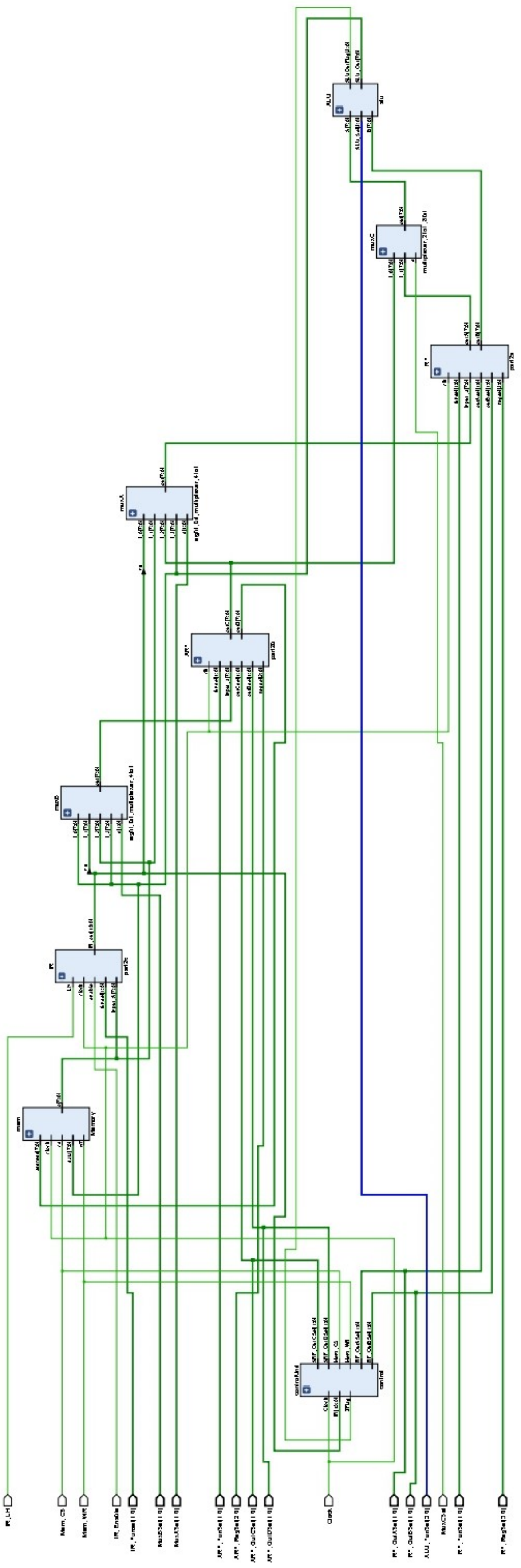

Figure 1: Project 2 Elaborated Design

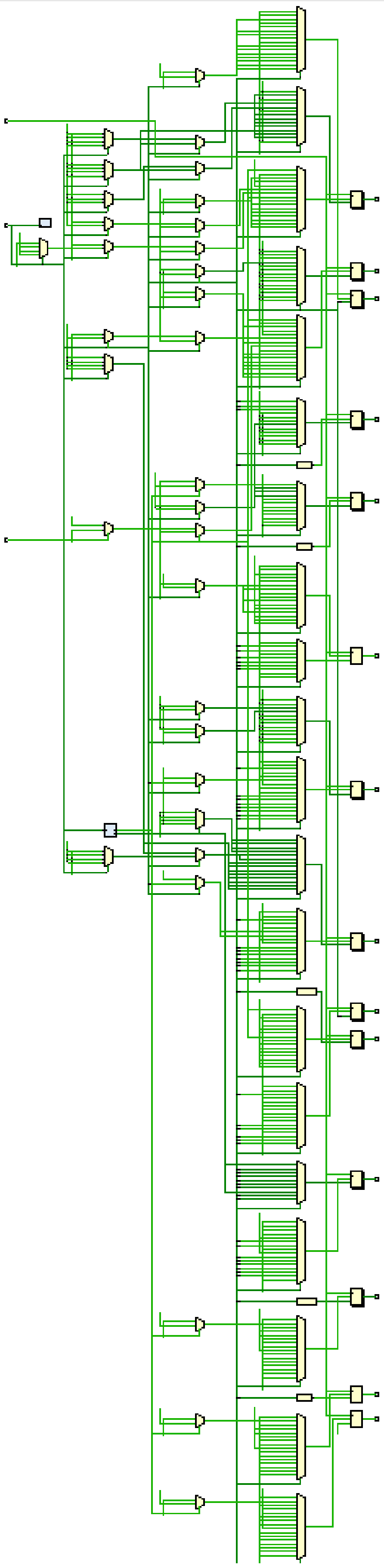

Figure 2: Control Unit Elaborated Design

4 DISCUSSION [25 points]

Mainly during the project we have designed a hardwired control unit which prepares the system for set of Instructions.We have used behavioral approach while designing the system. Firstly we have written decode modules for 2 different Instruction mode. Then according to decoded Instructions we have implemented operations in control unit module case by case. Our cases are:

BRA: In branch operation we have written M[AR] into PC register.

LD: In load operation we have selected the register to be loaded with regsel cases. IF it is direct, we have written M[AR] to selected register. Else we have loaded address part of IR[7:0] to selected register.

ST: In store operation we have selected the register to be loaded with regsel cases. Then we have stored the selected register into M[AR].

MOV: In move operation firstly we have selected the source register 1 by decoding IR's source register 1 part(IR[7:4]). Then we have selected destination register according to IR's destination register bits (IR[11:8]). Then we have loaded destination register with source register.

AND: In AND operation firstly we have selected the source register 1 by decoding IR's source register 1 part(IR[7:4]). If SRCREG1 was one of the RF registers we have selected RFOutputA according to that and send to MUXC and selected MUXC RFOutputA by giving MUXC selector 1. If SRCREG1 was one of the ARF registers we have selected ARFOutputC according to that and send to MUXC and selected MUXC ARFOutput by giving MUXC selector 0. Since SRCREG2 can only be one of the RF registers we have selected RFOutputB according to that and sent it to the ALU. In ALU we have selected AND operation with setting ALUFunsel 0111. After that we have selected destination regsiter by decoding IR's DSTREG part (IR[11:8]). If it is RF we have selected MUXA as ALUOut by giving MUXA selector 11. If it was ARF we have selected MUXB as ALUOut by giving MUXB selector 11. After doing that if DSTREG was RF we set RFFunsel 10 to load the wanted register, else we set ARFFunsel 10 to load the wanted register.

OR: In OR operation firstly we have selected the source register 1 by decoding IR's source register 1 part(IR[7:4]). If SRCREG1 was one of the RF registers we have selected

RFOutputA according to that and send to MUXC and selected MUXC RFOutputA by giving MUXC selector 1. If SRCREG1 was one of the ARF registers we have selected ARFOutputC according to that and send to MUXC and selected MUXC ARFOutput by giving MUXC selector 0. Since SRCREG2 can only be one of the RF registers we have selected RFOutputB according to that and sent it to the ALU. In ALU we have selected OR operation with setting ALUFunsel 1000. After that we have selected destination regsiter by decoding IR's DSTREG part (IR[11:8]). If it is RF we have selected MUXA as ALUOut by giving MUXA selector 11. If it was ARF we have selected MUXB as ALUOut by giving MUXB selector 11. After doing that if DSTREG was RF we set RFFunsel 10 to load the wanted register, else we set ARFFunsel 10 to load the wanted register.

NOT: In NOT operation firstly we have selected the source register 1 by decoding IR's source register 1 part(IR[7:4]). If SRCREG1 was one of the RF registers we have selected RFOutputB according to that and send to ALU directly. If SRCREG1 was one of the ARF registers we have selected ARFOutputC according to that and send to MUXC and selected MUXC ARFOutput by giving MUXC selector 0. After sending to ALU, In ALU we have selected NOT operation with setting ALUFunsel 0010. After that we have selected destination regsiter by decoding IR's DSTREG part (IR[11:8]). If it is RF we have selected MUXA as ALUOut by giving MUXA selector 11. If it was ARF we have selected MUXB as ALUOut by giving MUXB selector 11. After doing that if DSTREG was RF we set RFFunsel 10 to load the wanted register, else we set ARFFunsel 10 to load the wanted register.

ADD: In ADD operation firstly we have selected the source register 1 by decoding IR's source register 1 part(IR[7:4]). If SRCREG1 was one of the RF registers we have selected RFOutputA according to that and send to MUXC and selected MUXC RFOutputA by giving MUXC selector 1. If SRCREG1 was one of the ARF registers we have selected ARFOutputC according to that and send to MUXC and selected MUXC ARFOutput by giving MUXC selector 0. Since SRCREG2 can only be one of the RF registers we have selected RFOutputB according to that and sent it to the ALU. In ALU we have selected ADD operation with setting ALUFunsel 0100. After that we have selected destination regsiter by decoding IR's DSTREG part (IR[11:8]). If it is RF we have selected MUXA as ALUOut by giving MUXA selector 11. If it was ARF we have selected MUXB as ALUOut by giving MUXB selector 11. After doing that if DSTREG was RF we set RFFunsel 10 to load the wanted register, else we set ARFFunsel 10 to load the wanted register.

SUB: In SUB operation firstly we have selected the source register 1 by decoding IR's source register 1 part(IR[7:4]). If SRCREG1 was one of the RF registers we have selected RFOutputA according to that and send to MUXC and selected MUXC RFOutputA by giving MUXC selector 1. If SRCREG1 was one of the ARF registers we have selected ARFOutputC according to that and send to MUXC and selected MUXC ARFOutput by giving MUXC selector 0. Since SRCREG2 can only be one of the RF registers we have selected RFOutputB according to that and sent it to the ALU. In ALU we have selected SUB operation with setting ALUFunsel 0110. After that we have selected destination regsiter by decoding IR's DSTREG part (IR[11:8]). If it is RF we have selected MUXA as ALUOut by giving MUXA selector 11. If it was ARF we have selected MUXB as ALUOut by giving MUXB selector 11. After doing that if DSTREG was RF we set RFFunsel 10 to load the wanted register, else we set ARFFunsel 10 to load the wanted register.

LSR: In LSR operation firstly we have selected the source register 1 by decoding IR's source register 1 part(IR[7:4]). If SRCREG1 was one of the RF registers we have selected RFOutputB according to that and send to ALU directly. If SRCREG1 was one of the ARF registers we have selected ARFOutputC according to that and send to MUXC and selected MUXC ARFOutput by giving MUXC selector 0. After sending to ALU, In ALU we have selected LSR operation with setting ALUFunsel 1011. After that we have selected destination regsiter by decoding IR's DSTREG part (IR[11:8]). If it is RF we have selected MUXA as ALUOut by giving MUXA selector 11. If it was ARF we have selected MUXB as ALUOut by giving MUXB selector 11. After doing that if DSTREG was RF we set RFFunsel 10 to load the wanted register, else we set ARFFunsel 10 to load the wanted register.

LSL: In LSL operation firstly we have selected the source register 1 by decoding IR's source register 1 part(IR[7:4]). If SRCREG1 was one of the RF registers we have selected RFOutputB according to that and send to ALU directly. If SRCREG1 was one of the ARF registers we have selected ARFOutputC according to that and send to MUXC and selected MUXC ARFOutput by giving MUXC selector 0. After sending to ALU, In ALU we have selected LSL operation with setting ALUFunsel 1010. After that we have selected destination regsiter by decoding IR's DSTREG part (IR[11:8]). If it is RF we have selected MUXA as ALUOut by giving MUXA selector 11. If it was ARF we have selected MUXB as ALUOut by giving MUXB selector 11. After doing that if DSTREG was RF we set RFFunsel 10 to load the wanted register, else we set ARFFunsel 10 to load the wanted register.

PUL: In PUL operation firstly, we have selected SP register as output D of ARF which is the input address of memory. Then we have incremented SP in ARF by giving ARFFunsel 01 and ARFRegsel 110. We set the Memory read mode and activated it by chip selecting 0. After that we set MUXA as MemoryOutput since we want to load the Rx in RF. Then we have selected wanted register and RFFunsel as 10 to load the wanted register.

PSH: In PSH operation firstly, we have selected Rx decoding the IR's Regsel part (IR[9:8]). After that we activated Rx and selecting the RFOutBSel according to that and then we sent it to ALU. In ALU we have set ALUFunsel as 0001 because we wanted B without changing it. After that we have selected SP register as output D of ARF which is the input address of memory. Then we have decremented SP in ARF by giving ARFFunsel 00 and ARFRegsel 110. We set the Memory write mode and activated it by chip selecting 0. After that we set MUXA as MemoryOutput since we want to load the Rx in RF. Then we have selected wanted register and RFFunsel as 10 to load the wanted register.

INC: In INC operation firstly we have selected the source register 1 by decoding IR's source register 1 part(IR[7:4]). If SRCREG1 was one of the RF registers we have selected RFRegsel according to that and RFFunsel as 01 because we increment it in the register itself. Then we set RFOutputB according to that and send to ALU directly. If SRCREG1 was one of the ARF registers we have selected ARFRegsel according to that and ARFFunsel as 01 because we increment it in the register itself. Then we set ARFOutputC according to that and send to MUXC and selected MUXC ARFOutput by giving MUXC selector 0. After sending to ALU, if SRCREG1 was RF we have selected 0001 without changing the RFOutputB value, if SRCREG1 was ARF then we have selected ALUFunsel 0000 without changing the A input of ALU. After that we have selected destination regsiter by decoding IR's DSTREG part (IR[11:8]). If it is RF we have selected MUXA as ALUOut by giving MUXA selector 11. If it was ARF we have selected MUXB as ALUOut by giving MUXB selector 11. After doing that if DSTREG was RF we set RFFunsel 10 to load the wanted register, else we set ARFFunsel 10 to load the wanted register.

DEC: In DEC operation firstly we have selected the source register 1 by decoding IR's source register 1 part(IR[7:4]). If SRCREG1 was one of the RF registers we have selected RFRegsel according to that and RFFunsel as 00 because we decrement it in the register itself. Then we set RFOutputB according to that and send to ALU directly. If SRCREG1 was one of the ARF registers we have selected ARFRegsel according to that and ARFFunsel as 00 because we decrement it in the register itself. Then we set ARFOutputC according to that and send to MUXC and selected MUXC ARFOutput by giving MUXC selector 0. After sending to ALU, if SRCREG1 was RF we have selected 0001 without changing the RFOutputB value, if SRCREG1 was ARF then we have selected ALUFunsel 0000 without changing the A input of ALU. After that we have selected destination regsiter by decoding IR's DSTREG part (IR[11:8]). If it is RF we have selected MUXA as ALUOut by giving MUXA selector 11. If it was ARF we have selected MUXB as ALUOut by giving MUXB selector 11. After doing that if DSTREG was RF we set RFFunsel 10 to load the wanted register, else we set ARFFunsel 10 to load the wanted register.

BNE: In BNE operation firstly we check the Zero flag of ALU is 0 or not. If it is 0 then we have written address bits of IR into PC register.

5 CONCLUSION [10 points]

Since we have only designed Hardwired Control Unit, we did not implement the fetch decode phase. We couldn't complete that stage due to time constraints but we have learned how to do it and how it works. By doing this project we have learned how basic hardwired control unit works and gets implemented.