ISTANBUL TECHNICAL UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

BLG 222E

COMPUTER ORGANIZATION
PROJECT REPORT

Project : 3

DATE : 15.05.2019

GROUP NO : 34

GROUP MEMBERS:

150170005 : SONER OZTURK

150170062 : MEHMET FATIH YILDIRIM

150170098 : SAHIN AKKAYA

150170715 : MUSTAFA TEMURTAS

SPRING 2019

Contents

1 INTRODUCTION

In this project we design a hardware control unit which operates operations in given Figure1. We decided on which operations to do according to the 5 bit OPCODE in the instructions. In order to appropriate implementation Project2 has modified(which can be seen in Figure3) and Control Unit which control the inputs of components of the Basic Computer in Figure5 has implemented.

OPCODE (HEX)	SYMB	ADDRESSING MODE	DESCRIPTION
0x00	ADD	N/A	DESTREG ← SRCREG1 + SRCREG2
0x01	SUB	N/A	DESTREG ← SRCREG2 - SRCREG1
0x02	DEC	N/A	DESTREG ← SRCREG1 - 1
0x03	INC	N/A	DESTREG ← SRCREG1 + 1
0x04	BRA	IM	PC ← Value
0x05	BEQ	IM	IF Z=1 THEN PC ← Value
0x06	BNE	IM	IF Z=0 THEN PC ← Value
0x07	LSL	N/A	DESTREG ← LSL SRCREG1
0x08	LSR	N/A	DESTREG ← LSR SRCREG1
0x09	LD	IM, D	Rx ← Value (Value is described in Table 3)
0x0A	ST	D	Value ← Rx
0x0B	MOV	N/A	DESTREG ← SRCREG1
0x0C	NOT	N/A	DESTREG ← NOT SRCREG1
0xOD	OR	N/A	DESTREG ← SRCREG1 OR SRCREG2
0x0E	AND	N/A	DESTREG ← SRCREG1 AND SRCREG2

Figure 1: OPCODE field and symbols for operations and their descriptions

ADDRESSING MODE	MODE	SYMB	Value
0	Direct	D	M[AR]
1	Immediate	IM	ADDRESS Field

Figure 2: Addressing modes

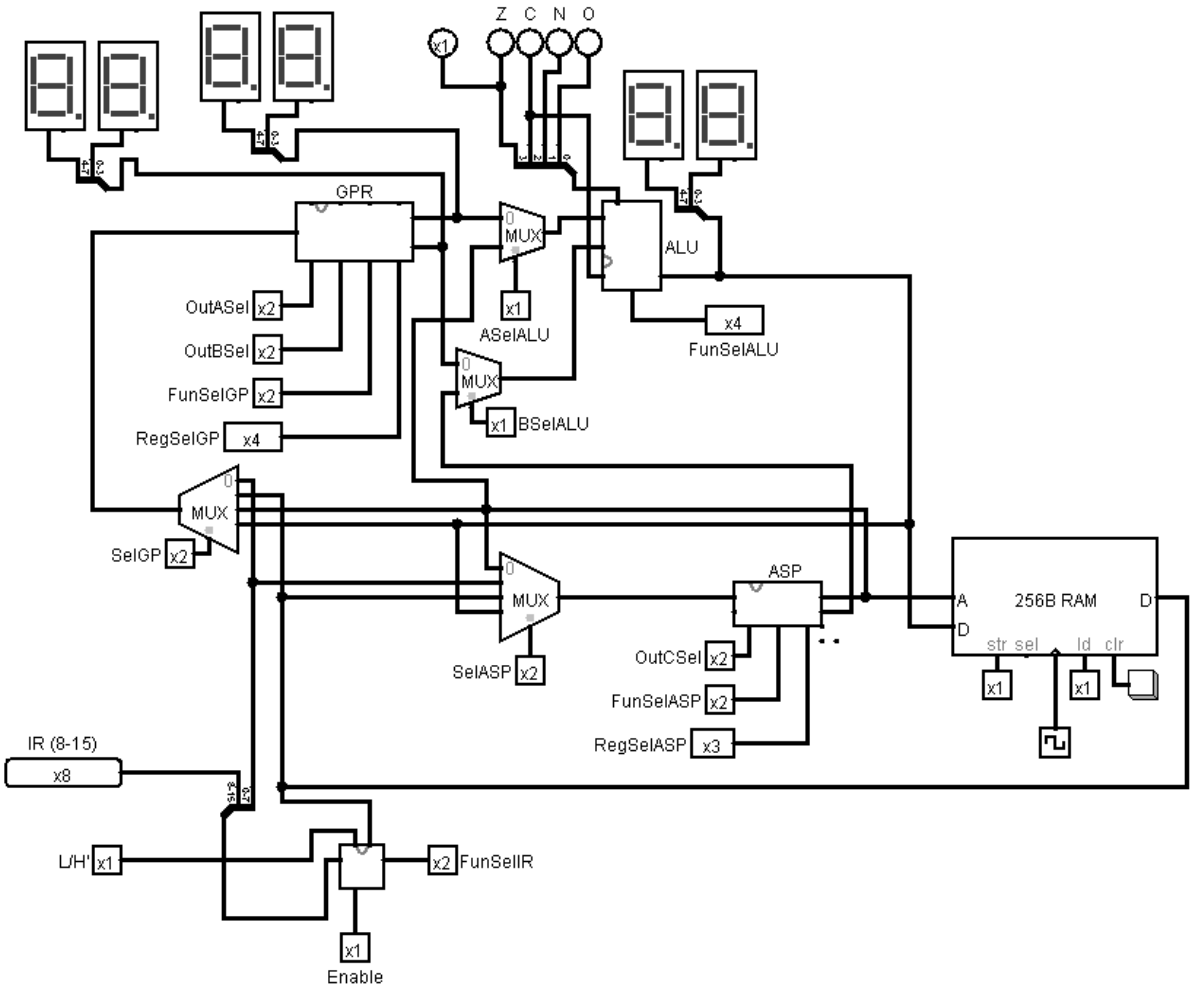

Figure 3: ALU System

2 Project

We design and implement a Control unit which can be seen in Figure 4 and Figure5. Control unit provides necessary inputs during operations according to specific timing signal in that operation. In order to understand outputs of the Control Unit, we are going to analyze each operation below.

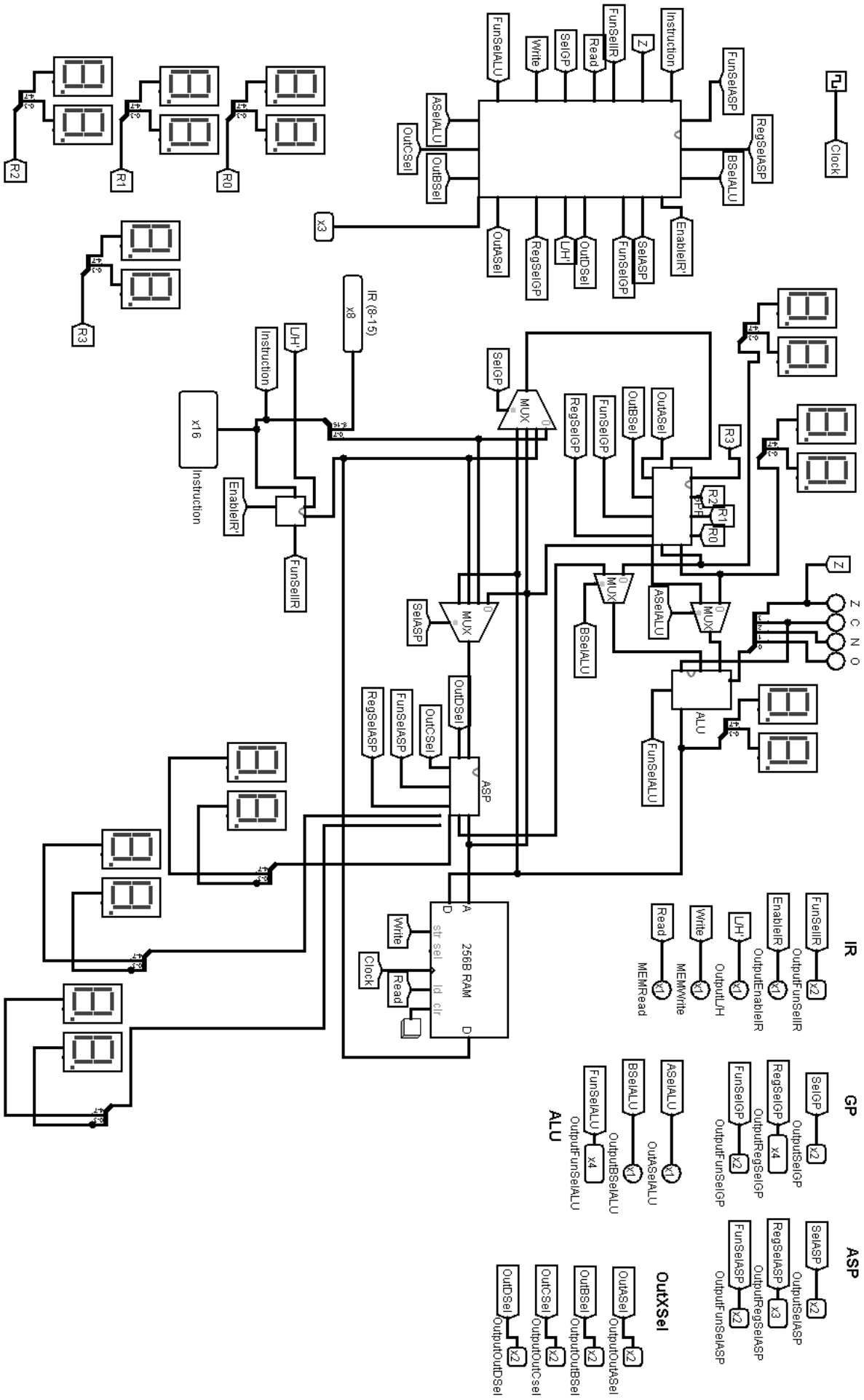

Figure 4: Control Unit and ALU System

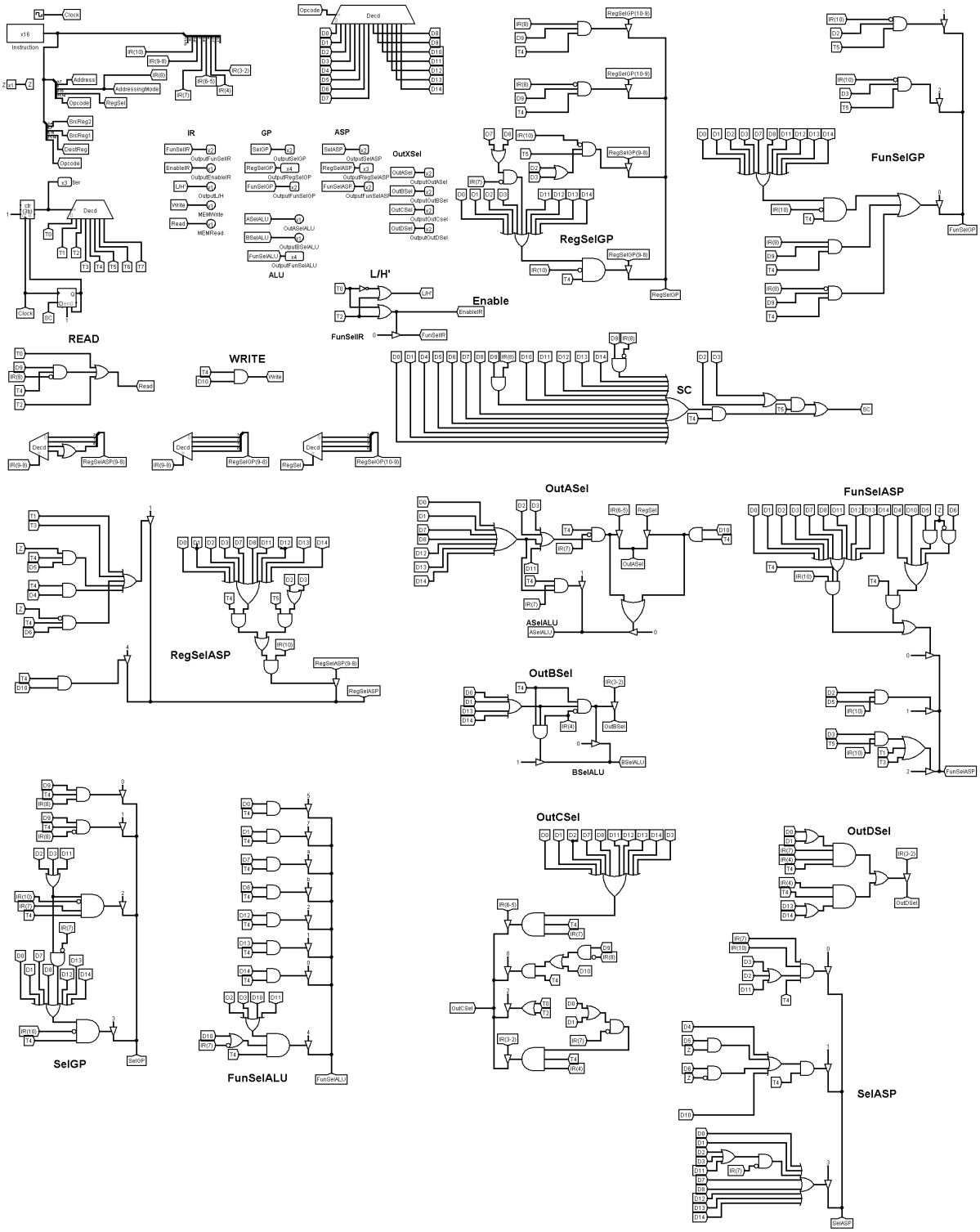

Figure 5: Control Unit

Fetch and Decode: this part is common in each of the operations from T0 to T3.

IR(8-15) = M[PC]

T0:

OutCSel: 10;

Read: 1;

L/H': 0;

Enable: 1;

FunSelIR: 00;

PC = PC + 1

T1:

RegSelASP: 001;

FunSelASP: 10;

IR(0-7) = M[PC]

T2:

OutCSel: 10;

Read: 1;

L/H': 1;

Enable: 1;

FunSelIR: 00;

PC = PC + 1

T3:

RegSelASP: 001;

FunSelASP: 10;

After Fetch and Decode is done, each operation continues according to decoded Opcode.

```
D0: DestReg = SrcReg1 + SrcReg2
IR10'IR7'IR4' (GPR = GPR + GPR)
D0T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutBSel: IR(3,2); // Select SrcReg2
BSelALU: 0; // Send B output of GPR to ALU as B
FunSelALU: 0101; // A + B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10'IR7'IR4 (GPR = GPR + ASP)
D0T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutCSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send C output of ASP to ALU as B
FunSelALU: 0101; // A + B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10'IR7IR4' (GPR = ASP + GPR)
D0T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutBSel: IR(3,2); // Select SrcReg2
BSelALU: 0; // Send A output of GPR to ALU as B
FunSelALU: 0101; // A + B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
```

```
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10'IR7IR4 (GPR = ASP + ASP)
D0T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0101; // A + B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10IR7'IR4' (ASP = GPR + GPR)
D0T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutBSel: IR(3,2); // Select SrcReg2
BSelALU: 0; // Send B output of GPR to ALU as B
FunSelALU: 0101; // A + B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10IR7'IR4 (ASP = GPR + ASP)
D0T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutCSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send C output of ASP to ALU as B
FunSelALU: 0101; // A + B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
```

```
SC = 0; // Done, Reset the counter

IR10IR7IR4' (ASP = ASP + GPR)
D0T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutBSel: IR(3,2); // Select SrcReg2
BSelALU: 0; // Send A output of GPR to ALU as B
FunSelALU: 0101; // A + B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10IR7IR4 (ASP = ASP + ASP)
D0T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0101; // A + B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter


D1: DestReg = SrcReg1 - SrcReg2
IR10'IR7'IR4' (GPR = GPR - GPR)
D1T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutBSel: IR(3,2); // Select SrcReg2
BSelALU: 0; // Send B output of GPR to ALU as B
FunSelALU: 0111; // A - B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
```

```
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10'IR7'IR4 (GPR = GPR - ASP)
D1T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutCSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send C output of ASP to ALU as B
FunSelALU: 0111; // A - B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10'IR7IR4' (GPR = ASP - GPR)
D1T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutBSel: IR(3,2); // Select SrcReg2
BSelALU: 0; // Send A output of GPR to ALU as B
FunSelALU: 0111; // A - B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10'IR7IR4 (GPR = ASP - ASP)
D1T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0111; // A - B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
```

```
SC = 0; // Done, Reset the counter

IR10IR7'IR4' (ASP = GPR - GPR)
D1T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutBSel: IR(3,2); // Select SrcReg2
BSelALU: 0; // Send B output of GPR to ALU as B
FunSelALU: 0111; // A - B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10IR7'IR4 (ASP = GPR - ASP)
D1T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutCSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send C output of ASP to ALU as B
FunSelALU: 0111; // A - B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10IR7IR4' (ASP = ASP - GPR)
D1T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutBSel: IR(3,2); // Select SrcReg2
BSelALU: 0; // Send A output of GPR to ALU as B
FunSelALU: 0111; // A - B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter
```

```
IR10IR7IR4 (ASP = ASP - ASP)
D1T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0111; // A - B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter


D2: DestReg = SrcReg1  1
IR10'IR7' (GPR = GPR - 1)
D2T4:
OutASel: IR(6,5); //Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 0100; // A
SelGP: 11; //Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load SrcReg1
D2T5:
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 01; // Decrement DestReg
SC = 0; // Done, Reset the counter

IR10'IR7 (GPR = ASP - 1)
D2T4:
OutCSel: IR(6,5); // Select SrcReg1
SelGP: 10; // Send output of ASP to GPR
RegSelGP: IR(9,8) // D- Select DestReg
FunSelGP: 00; // Load SrcReg1
D2T5:
RegSelGP: IR(9,8) // D- Select DestReg
FunSelGP: 01; // Decrement DestReg
```

```
SC = 0; // Done, Reset the counter

IR10IR7' (ASP = GPR - 1)
D2T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 0100; // A
SelASP: 11; // Send output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load SrcReg1
D2T5:
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 01; // Decrement SrcReg1
SC = 0; // Done, Rest the counter

IR10IR7  (ASP = ASP - 1)
D2T4:
OutCSel: IR(6,5); // Select SrcReg1
SelASP: 01; // Send SrcReg1 to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load SrcReg1
D2T5:
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 01; // Decrement SrcReg1
SC = 0; // Done, Rest the counter


D3: DestReg = SrcReg1 + 1
IR10'IR7' (GPR = GPR + 1)
D3T4:
OutASel: IR(6,5); //Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 0100; // A
SelGP: 11; //Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load DestReg
D3T5:
```

```
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 10; // Increment DestReg
SC = 0; // Done, Reset the counter

IR10'IR7 (GPR = ASP + 1)
D3T4:
OutCSel: IR(6,5); // Select SrcReg1
SelGP: 10; // Send output of ASP to GPR
RegSelGP: IR(9,8) // D- Select DestReg
FunSelGP: 00; // Load SrcReg1
D3T5:
RegSelGP: IR(9,8) // D- Select DestReg
FunSelGP: 10; // Increment DestReg
SC = 0; // Done, Reset the counter

IR10IR7' (ASP = GPR + 1)
D3T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 0100; // A
SelASP: 11; // Send output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load SrcReg1
D3T5:
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 10; // Increment SrcReg1
SC = 0; // Done, Rest the counter

IR10IR7  (ASP = ASP + 1)
D3T4:
OutCSel: IR(6,5); // Select SrcReg1
SelASP: 00; // Send SrcReg1 to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load SrcReg1
D3T5:
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 10; // Increment SrcReg1
```

```
SC = 0; // Done, Rest the counter


D4: PC = Value
D4T4:
SelASP: 01; // Send IR(0,7) to ASP
RegSelASP: 001; // Enable PC
FunSelASP: 00; // Load PC
SC = 0; // Done, Reset the counter


D5: If Z=1 then PC = Value
Z
D5T4:
SelASP: 01; // Send IR(0,7) to ASP
RegSelASP: 001; // Enable PC
FunSelASP: 00; // Load PC
SC = 0; // Done, Reset the counter
Z'
D5T4:
SC = 0; // Done, Reset the counter


D6: If Z=0 then PC = Value
Z'
D6T4:
SelASP: 01; // Send IR(0,7) to ASP
RegSelASP: 001; // Enable PC
FunSelASP: 00; // Load PC
SC = 0; // Done, Reset the counter
Z
D6T4:
SC = 0; // Done, Reset the counter


D7: DestReg = LSL SrcReg1
IR10'IR7' (GPR = LSL GPR)
```

```
D7T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 1010; // LSL A
SelGP: 11; //Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10'IR7 (GPR = LSL ASP)
D7T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
FunSelALU: 1010; // LSL A
SelGP: 11; // Send output of ALU to GP
RegSelGP: IR(9,8); //D- Select Destreg
FunSelGP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10IR7' (ASP = LSL GPR)
D7T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 1010; // LSL A
SelASP: 11; // Send output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10IR7 (ASP = LSL ASP)
D7T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
FunSelALU: 1010; // LSL A
SelASP: 11; // Send output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load DestReg
```

```
SC = 0; // Done, Reset the counter


D8: DestReg = LSR SrcReg1
IR10'IR7' (GPR = LSR GPR)
D8T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 1011; // LSR A
SelGP: 11; //Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10'IR7 (GPR = LSR ASP)
D8T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
FunSelALU: 1011; // LSR A
SelGP: 11; // Send output of ALU to GP
RegSelGP: IR(9,8); //D- Select Destreg
FunSelGP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10IR7' (ASP = LSR GPR)
D8T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 1011; // LSR A
SelASP: 11; // Send output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10IR7 (ASP = LSR ASP)
D8T4:
OutCSel: IR(6,5); // Select SrcReg1
```

```
ASelALU: 1; // Send C output of ASP to ALU as A
FunSelALU: 1011; // LSR A
SelASP: 11; // Send output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load DestReg
SC = 0; // Done, Reset the counter


D9: Rx = Value
IR8 (Immediate, value is equal to ADDRESS field)
D9T4:
SelGP: 00; // Send IR(0,7) to GPR
RegSelGP: IR(10,9) //D- Select Rx
FunSelGP: 00; // Load Rx
SC = 0; // Done, Reset the counter

IR8' (Direct, value is equal to M[AR])
D9T4:
SelASP: 01; // Send IR(0,7) to ASP
RegSelASP: 100; // Enable AR
FunSelASP: 00; // Load AR
OutCSel: 00; // Select AR
D9T5:
Read: 1;
SelGP: 01; // Send M[AR] to GPR
RegSelGP: IR(10,9) //D- Enable Rx
FunSelGP: 00; // Load Rx
SC = 0; // Done, Reset the counter


D10: Value = Rx
D10T4:
OutASel: IR(10,9); // Select Rx
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 0100; // A
SelASP: 01; // Send IR(0,7) to ASP
RegSelASP: 100; // Enable AR
```

```
FunSelASP: 00; // Load AR
OutCSel: 00; // Select AR
Write: 1;
SC = 0; // Done, Reset the counter


D11: DestReg = SrcReg1
IR10'IR7' (GPR = GPR)
D11T4:
OutASel: IR(6,5); //Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 0100; // A
SelGP: 11; //Send Output of ALU to GPR
RegSelGP: IR(9,8); //Select DestReg
FunSelGP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10'IR7 (GPR = ASP)
D11T4:
OutCSel: IR(6,5); // Select SrcReg1
SelGP: 10; // Send C output of ASP to GPR
RegSelGP: IR(9,8); // D- Select DestReg
FunSelGP: 00; // Load SrcReg1
SC = 0; // Done, Reset the counter

IR10IR7' (ASP = GPR)
D11T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 0100; // A
SelASP: 11; // Send output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load SrcReg1
SC = 0; // Done, Reset the counter

IR10IR7 (ASP = ASP)
D11T4:
```

```
OutCSel: IR(6,5); // Select SrcReg1
SelASP: 00; // Send C output of ASP to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load SrcReg1
SC = 0; // Done, Reset the counter


D12: DestReg = NOT SrcReg1

IR10'IR7' (GPR = NOT GPR)
D12T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 0010; // NOT A
SelGP: 11; // Send output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10'IR7 (GPR = NOT ASP)
D12T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
FunSelALU: 0010; // NOT A
SelGP: 11; // Send output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10IR7' (ASP = NOT GPR)
D12T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
FunSelALU: 0010; // NOT A
SelASP: 11; // Send output of ALU to GPR
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load DestReg
```

```
SC = 0; // Done, Reset the counter

IR10IR7 (ASP = NOT ASP)
D12T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
FunSelALU: 0010; // NOT A
SelASP: 11; // Send output of ALU to GPR
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load DestReg
SC = 0; // Done, Reset the counter


D13: DestReg = SrcReg1 OR SrcReg2

IR10'IR7'IR4' (GPR = GPR OR GPR)
D13T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutBSel: IR(3,2); //Select SrcReg2
BSelALU: 0; // Send B output of GPR to ALU as B
FunSelALU: 0001; // A OR B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10'IR7'IR4 (GPR = GPR OR ASP)
D13T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0001; // A OR B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
```

```
SC = 0; // Done, Reset the counter

IR10'IR7IR4' (GPR = ASP OR GPR)
D13T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutBSel: IR(3,2); //Select SrcReg2
BSelALU: 0; // Send B output of GPR to ALU as B
FunSelALU: 0001; // A OR B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10'IR7IR4 (GPR = ASP OR ASP)
D13T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0001; // A OR B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10IR7'IR4' (ASP = GPR OR GPR)
D13T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutBSel: IR(3,2); //Select SrcReg2
BSelALU: 0; // Send B output of GPR to ALU as B
FunSelALU: 0001; // A OR B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load DestReg
SC = 0; // Done, Reset the counter
```

```
IR10IR7'IR4 (ASP = GPR OR ASP)
D13T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0001; // A OR B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10IR7IR4' (ASP = ASP OR GPR)
D13T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutBSel: IR(3,2); // Select SrcReg2
BSelALU: 0; // Send A output of GPR to ALU as B
FunSelALU: 0001; // A OR B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10IR7IR4 (ASP = ASP OR ASP)
D13T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0001; // A OR B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter
```

```
D14: DestReg = SrcReg1 AND SrcReg2

IR10'IR7'IR4' (GPR = GPR AND GPR)
D14T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutBSel: IR(3,2); //Select SrcReg2
BSelALU: 0; // Send B output of GPR to ALU as B
FunSelALU: 0000; // A AND B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10'IR7'IR4 (GPR = GPR AND ASP)
D14T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0000; // A AND B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10'IR7IR4' (GPR = ASP AND GPR)
D14T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutBSel: IR(3,2); //Select SrcReg2
BSelALU: 0; // Send B output of GPR to ALU as B
FunSelALU: 0000; // A AND B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter
```

```
IR10'IR7IR4 (GPR = ASP AND ASP)
D14T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0000; // A AND B
SelGP: 11; // Send Output of ALU to GPR
RegSelGP: IR(9,8); //D- Select DestReg
FunSelGP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10IR7'IR4' (ASP = GPR AND GPR)
D14T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutBSel: IR(3,2); //Select SrcReg2
BSelALU: 0; // Send B output of GPR to ALU as B
FunSelALU: 0000; // A AND B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load DestReg
SC = 0; // Done, Reset the counter

IR10IR7'IR4 (ASP = GPR AND ASP)
D14T4:
OutASel: IR(6,5); // Select SrcReg1
ASelALU: 0; // Send A output of GPR to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0000; // A AND B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter
```

```
IR10IR7IR4' (ASP = ASP AND GPR)
D14T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutBSel: IR(3,2); // Select SrcReg2
BSelALU: 0; // Send A output of GPR to ALU as B
FunSelALU: 0000; // A AND B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter

IR10IR7IR4 (ASP = ASP AND ASP)
D14T4:
OutCSel: IR(6,5); // Select SrcReg1
ASelALU: 1; // Send C output of ASP to ALU as A
OutDSel: IR(3,2); // Select SrcReg2
BSelALU: 1; // Send D output of ASP to ALU as B
FunSelALU: 0000; // A AND B
SelASP: 11; // Send Output of ALU to ASP
RegSelASP: IR(9,8); //D- Select DestReg
FunSelASP: 00; // Load enabled registers
SC = 0; // Done, Reset the counter
```

Simplified version of logical expressions of the control unit components;

```
OutASel:
IR(10-9): D10T4
IR(6-5): T4IR7'(D0 + D1 + D2 + D3 + D7 + D8 + D11 + D12 + D13 + D14)

OutBSel:
IR(3-2): T4IR4'(D0 + D1 + D13 + D14)

OutCSel:
00: T4(D10 + D9IR8)
10:T0 + T2
IR(6-5): T4IR7(D0 + D1 + D2 + D3 + D7 + D8 + D11 + D12 + D13 + D14)
IR(3-2):T4IR4(IR7'(D0 + D1))

OutDSel:
IR(3-2): T4IR4IR7(D0 + D1) + T4IR4(D13 + D14)


FunSelASP:
00: T4IR10(D0+D1+D2+D3+D11+D12+D13+D14+D7+D8) + T4(D4 + D5Z + D6Z' + D9IR8' + D10)
01: D2IR10T5
10: T1 + T3 + (D3IR10T5)


FunSelGP:
00: T4IR10'(D0+D1+D2+D3+D7+D8+D11+D12+D13+D14) + D9T4IR8 + D9T5IR8'
01: D2T5IR10
10: D3T5IR10'

RegSelASP:
IR(9,8): T4IR10(D0+D1+D2+D3+D7+D8+D11+D12+D13+D14) + T5IR10(D2+D3)
001: T1 + T3 + T4D4 + ZT4D5 + ZT4D6
100: T4D10

RegSelGP:
```

IR(10-9): T4D9(IR8 + IR8')

IR(9-8): T5IR10'(D2 + D3) + T4IR10'(D0+D1+D2+D3+D11+D12+D13+D14+IR7'(D7+D8))

ASelAlu:

0: T4IR7'(D0+D1+D2+D3+D7+D8+D11+D12+D13+D14) + D10T4

1: T4IR7(D0+D1+D7+D8+D12+D13+D14)

BSelAlu:

0: T4IR4'(D0 + D1 + D13 + D14)

1: T4IR4(D0 + D1 + D13 + D14)

FunSelAlu:

0000: D14T4

0001: D13T4

0010: D12T4

0100: T4(IR7'(D2 + D3 + D11) + D10)

0111: D1T4

0101: D0T4

1010: D7T4

1011: D8T4

SelGP:

00: D9IR8T4

01: D9T5IR8'

10: IR7T4IR10'(D11 + D2 + D3)

11: T4IR10'(D0 + D1 + IR7'(D2 + D3 + D11) + D7 + D8 + D12 + D13 + D14)

SelASP:

00: T4IR10IR7(D2+D3+D11)

01: T4(D4 + D10 + D5Z D6Z')

11: D0 + D1 + D7 + D8 + D12 + D13 + D14 + IR7'(D2 + D3 + D11)

SC:

T4(D0 + D1 + D4 + D5 + D6 + D7 + D8 + D9 + D10 + D11 + D12 + D13 + D14) +

T5(D2 + D3 + D9IR8')

```
Read:
1: T0 + T2 + T4D9IR8'

Write:
1: T4D10
```

3 CONCLUSION

In this project, we designed a hardware control unit that controls the basic computer we made in the second project. We also made some changes on the basic computer to do the project. While we were designing this project, we gained experience on hard-wired implementing of basic computer and processing operations.