Computer Organization Project 2

Group Number: 18
Mehmed Esad AKÇAM: 150190725
Talha Sezer ÇAKIR: 150180027
Mehmet Yiğit ATEŞ: 150180049

Introduction

In this project a basic computer’s which was designed in the first project hardwired control unit is implemented. While designing the control unit, operations in the project description are considered. Control operations consist of three stages: Fetch, Decode and Execute. Decoded operations are shown with *D* letter. For timing purposes *T* letter is used.

FETCH

IR(0-7) ← M[PC]

T0:

OutDSel: 00; Select PC

Load: 1;

L/H’:1; Select the low part of IR

Enable of IR: 1;

IR FunSel:10 Load to Selected part of the IR

PC ← PC + 1

T1:

RegselARF: 011; only PC enabled

FunselARF: 01; increase enabled registers

IR(8-15) ← M[PC]

T2:

OutDSel: 00; Select PC

Load: 1;

L/H’:0; Select the high part of IR

Enable of IR: 1;

FunselIR:10 Load to Selected part of the IR

PC ← PC + 1

T3: RegselARF: 011; only PC enabled

FunselARF: 01; increase enabled registers

OPERATIONS

D0: PC<- VALUE

D0T4:

MuxASel: 11 Select IR(0,7)

FunSelRf: 10 Load

RegSelRF 1111 Disable all register

TempSel 0111 Select temporary register

D0T5

Registerfile OutAsel: 100 Select Temp Reg. 1

FunselAlu: 0000 Transfer via ALU

MuxBSel: 01 Select OutALU

FunSel ARF: 10 Load

RegSel: 011 Select PC

SC<-0 Clear the SC

D1: Rx<-Value

D1T4IR(10): Rx <- IR(0-7) 00010100

RegSelRF: Decode IR(8,9) reverse it and Xor with 1

FunSelRF: 10 Load

MuxASel: 11 Select IR(0,7)

SC<-0 Clear SC

D1T4IR(10)’ Rx <- M(AR)

Memory Load:1

MuxASel: 10 Select M[AR]

OutDSel: 10 Select AR

FunSel of Registerfile: 10 Load

RegSel of Registerfile: Decode IR(8,9) reverse it and Xor with 1

Sc<-0 Clear SC

D2: Value <-Rx

D2T4

Memory store = 1

OutDSel of ARF: 10 AR Selected

OutASel of Registerfile: {0,IR(8,9)} register selected

FunSel of ALU: 0000 (ALU output is RX) Rx is sended to M[AR]

SC<-0

D3: Destreg <- SRCREG1

D3T4IR(6)IR(10) (Reg<-reg)

OutASel of Registerfile: {0,IR(5,4)} transfer R as A

FunSel ALU: 0000 OutAlu becomes R

MuxASel: 00 Select OutAlu

FunSel of Registerfile: 10 Load

RegSel of Registerfile: Decode IR(9,8) reverse it and Xor with 1

SC<-0 Clear SC

D3T4IR(6)’IR(10) (Reg<-Add. reg)

OutCSel of ARF: IR(5,4) Select register

MuxASel: 01 OutC is selected

RegSel of Registerfile: Decode IR(9,8) reverse it and Xor with 1

FunSel of Registerfile: 10 Load

SC<-0 Clear SC

D3T4IR(6)IR(10)’(Add .Reg <- reg)

OutASel of Registerfile: {0,IR(5,4)}

FunSelALU: 0000 transfer R

MuxBSel: 01 OutAlu selected

FunSel of ARF: 10 Load

RegSel of ARF: Decode IR(9,8)

SC<-0 Clear SC

D3T4IR(6)’IR(10)’(Add .Reg <- Add. reg)

OutCSel of ARF: IR(5,4) Select register

MuxASel: 01 OutC Selected

FunSel of Registerfile: 10 Load

RegSel of Registerfile: 1111 Disable all registers

TempSel of Registerfile: 0111 Enable temp reg. 1

D3T5IR(6)’IR(10)’

OutASel of Registerfile: 100 Select temp reg. 1

FunSel of Alu: 0000 Transfer R

MuxBSel: 01 Selecet OutAlu

FunSel of ARF: 10 Load

RegSel of ARF: decode IR(9,8)

SC<-0 Clear

D4: DestReg <-Src1 and Src2

D4T4IR(10) IR(6) IR(2)(Reg <- Reg and Reg)

OutASel of Registerfile: {0,IR(1,0)} Src1

OutBSel of Registerfile:{0,IR(5,4)} Src2

FunSel of ALU: 0111 Src1 and Src2

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, Sor with 1

SC<-0 Clear

D4T4IR(10) IR(6) IR(2)'(Reg <- Reg and Add. Reg)

OutCSel of ARF: IR(1,0) Select register

MuxASel:01 OutC

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile:0111 Enable temp reg. 1

D4T5IR(10) IR(6) IR(2)'

OutBSel of Registerfile:{0,IR(5,4)} Src1

OutASel Of RegisterFile: 100 Src2

FunSel of ALU: 0111 (Src1 and Src2)

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, Xor with 1

SC<-0 Clear SC

D4T4IR(10) IR(6)’ IR(2)(Reg <- Add.Reg and Reg)

OutCSel of ARF: IR(5,4) Select Register

MuxASel:01 Select OutC

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile: 0111 Enable temp reg. 1

D4T5IR(10) IR(6)’ IR(2)

OutBSel of Registerfile:{0,IR(1,0)} Src2

OutASel Of RegisterFile:100 Src1

FunSel of ALU: 0111 (Src1 and Src2)

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, Xor with 1

SC<-0 Clear SC

D4T4IR(10) IR(6)’ IR(2)’(Reg <- Add.Reg and Add. Reg)

OutCSel of ARF: IR(1,0) Select Register

MuxASel: 01 Select OutC

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile:0111 Enable temp reg. 1

D4T5IR(10) IR(6)’ IR(2)’

OutCSel of ARF: IR(5,4) Select Register

MuxASel: 01 Select OutC

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile:1011 Enable temp reg. 2

D4T6IR(10) IR(6)’ IR(2)’

OutBSel of Registerfile:100 Select temp reg. 1

OutASel Of RegisterFile:101 Select temp reg. 2

FunSel of ALU: 0111 Src1 and Src2

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, Xor with 1

SC<-0 Clear SC

D4T4IR(10)’ IR(6) IR(2)(Add. Reg <- Reg and Reg)

OutASel of Registerfile: {0,IR(1,0)} Src1

OutBSel of Registerfile:{0,IR(5,4)} Src2

FunSel of ALU: 0111 Src1 and Src2

MuxBSel = 01 OutAlu selected

FunSel of ARF = 10 Load

RegSel of ARF: decode IR(9,8)

SC<-0 Clear SC

D4T4IR(10)’ IR(6) IR(2)’(Add. Reg <- Reg and Add. Reg)

OutCSel of ARF: IR(1,0) Select Register

MuxASel:01 OutC selected

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile:0111 Enable temp reg. 1

D4T5IR(10)’ IR(6) IR(2)’

OutBSel of Registerfile:{0,IR(5,4)} Src2

OutASel Of RegisterFile:100 Src1

FunSel of ALU: 0111 Src1 and Src2

MuxBSel: 01 OutAlu selected

FunSel of ARF: 10 Load

RegSel of ARF: Decode IR(9,8)

SC<-0 Clear SC

D4T4IR(10)’ IR(6)’ IR(2) (Add. Reg <-Add. Reg and Reg)

OutCSel of ARF: IR(5,4) Select register

MuxASel: 01 OutC selected

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile: 0111 Enable temp reg. 1

D4T5IR(10)’ IR(6)’ IR(2)

OutBSel of Registerfile:{0,IR(1,0)} Src2

OutASel Of RegisterFile: 100 Select temp reg.1

FunSel of ALU: 0111 Src1 and Src2

MuxBSel = 01 OutAlu selected

FunSel of ARF = 10 Load

RegSel of ARF: decode IR(9,8)

SC<-0 Clear SC

D4T4IR(10)’ IR(6)’ IR(2)’(Add. Reg <-Add. Reg and Add. Reg)

OutCSel of ARF: IR(1,0) Select Register

MuxASel: 01 OutC Selected

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile:0111 Enable Temp reg. 1

D4T5IR(10)’ IR(6)’ IR(2)’

OutCSel of ARF: IR(5,4) Select Register

MuxASel: 01 OutC Selected

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile: 1011 Enable temp reg. 2

D4T6IR(10)’ IR(6)’ IR(2)’

OutBSel of Registerfile: 100 Temp reg. 1

OutASel Of RegisterFile: 101 Temp reg. 2

FunSel of ALU: 0111 Src1 and Src2

MuxBSel = 01 OutAlu selected

FunSel of ARF = 10 Load

RegSel of ARF: decode IR(9,8)

SC<-0 Clear SC

D5: DestReg <-Src1 or Src2

D5T4IR(10) IR(6) IR(2)(Reg <- Reg or Reg)

OutASel of Registerfile: {0,IR(1,0)} Src1

OutBSel of Registerfile:{0,IR(5,4)} Src2

FunSel of ALU: 1000 Src1 or Src2

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, Sor with 1

SC<-0 Clear

D5T4IR(10) IR(6) IR(2)’(Reg <- Reg or Add. Reg)

OutCSel of ARF: IR(1,0) Select register

MuxASel:01 OutC

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile:1000 Enable temp reg. 1

D5T5IR(10) IR(6) IR(2)’

OutBSel of Registerfile:{0,IR(5,4)} Src1

OutASel Of RegisterFile: 100 Src2

FunSel of ALU: 1000 Src1 or Src2

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, Xor with 1

SC<-0 Clear SC

D5T4IR(10) IR(6)’ IR(2)(Reg <- Add.Reg or Reg)

OutCSel of ARF: IR(5,4) Select Register

MuxASel:01 Select OutC

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile: 0111 Enable temp reg. 1

D5T5IR(10) IR(6)’ IR(2)

OutBSel of Registerfile:{0,IR(1,0)} Src2

OutASel Of RegisterFile:100 Src1

FunSel of ALU: 1000 Src1 or Src2

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, Xor with 1

SC<-0 Clear SC

D5T4IR(10) IR(6)’ IR(2)’(Reg <- Add. Reg or Add. Reg)

OutCSel of ARF: IR(1,0) Select Register

MuxASel: 01 Select OutC

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile:0111 Enable temp reg. 1

D5T5IR(10) IR(6)’ IR(2)’

OutCSel of ARF: IR(5,4) Select Register

MuxASel: 01 Select OutC

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile:1011 Enable temp reg. 2

D5T6IR(10) IR(6)’ IR(2)’

OutBSel of Registerfile:100 Select temp reg. 1

OutASel Of RegisterFile:101 Select temp reg. 2

FunSel of ALU: 1000 Src1 or Src2

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, Xor with 1

SC<-0 Clear SC

D5T4IR(10)’ IR(6) IR(2)(Add. Reg <- Reg or Reg)

OutASel of Registerfile: {0,IR(1,0)} Src1

OutBSel of Registerfile:{0,IR(5,4)} Src2

FunSel of ALU: 1000 Src1 or Src2

MuxBSel = 01 OutAlu selected

FunSel of ARF = 10 Load

RegSel of ARF: decode IR(9,8)

SC<-0 Clear SC

D5T4IR(10)’ IR(6) IR(2)’(Add. Reg <- Reg or Add. Reg)

OutCSel of ARF: IR(1,0) Select Register

MuxASel:01 OutC selected

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile:0111 Enable temp reg. 1

D5T5IR(10)’ IR(6) IR(2)’

OutBSel of Registerfile:{0,IR(5,4)} Src2

OutASel Of RegisterFile:100 Src1

FunSel of ALU: 1000 Src1 or Src2

MuxBSel: 01 OutAlu selected

FunSel of ARF: 10 Load

RegSel of ARF: Decode IR(9,8)

SC<-0 Clear SC

D5T4IR(10)’ IR(6)’ IR(2) (Add. Reg <-Add. Reg or Reg)

OutCSel of ARF: IR(5,4) Select register

MuxASel: 01 OutC selected

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile: 0111 Enable temp reg. 1

D5T5IR(10)’ IR(6)’ IR(2)

OutBSel of Registerfile:{0,IR(1,0)} Src2

OutASel Of RegisterFile: 100 Select temp reg.1

FunSel of ALU: 1000 Src1 or Src2

MuxBSel = 01 OutAlu selected

FunSel of ARF = 10 Load

RegSel of ARF: decode IR(9,8)

SC<-0 Clear SC

D5T4IR(10)’ IR(6)’ IR(2)’(Add. Reg <-Add. Reg or Add. Reg)

OutCSel of ARF: IR(1,0) Select Register

MuxASel: 01 OutC Selected

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile:0111 Enable Temp reg. 1

D5T5IR(10)’ IR(6)’ IR(2)’

OutCSel of ARF: IR(5,4) Select Register

MuxASel: 01 OutC Selected

FunSel of Registerfile: 10 Load

Regsel of Registerfile: 1111 Disable all registers

TempSel of Registerfile: 1011 Enable temp reg. 2

D5T6IR(10)’ IR(6)’ IR(2)’

OutBSel of Registerfile: 100 Temp reg. 1

OutASel Of RegisterFile: 101 Temp reg. 2

FunSel of ALU: 1000 Src1 or Src2

MuxBSel = 01 OutAlu selected

FunSel of ARF = 10 Load

RegSel of ARF: decode IR(9,8)

SC<-0 Clear SC

D6: DESTREG <—NOT SRCREG1

D6T4IR(6)IR(10) (Reg<-reg)

OutASel of Registerfile: {0,IR(5,4)} transfer R as A

FunSel ALU: 0010 OutAlu becomes R

MuxASel:00

FunSel of Registerfile: 10 (load)

RegSel of registerfile: decode IR(9,8) reverse it and xor with 1

SC<-0

D6T4IR(6)’IR(10) (Reg<-Add. reg)

OutCSel of ARF(5,4) (select register)

MuxASel: 01 OutC is selected

RegSel of registerfile: decode IR(9,8) reverse it and xor with 1

FunSel of registerfile: 10 (load)

SC<-0

D6T4IR(6)IR(10)’(Add .Reg <- reg)

OutASel of Registerfile: {0,IR(5,4)}

FunSelALU: 0010 transfer R

MuxBSel: 01 OutAlu selected

FunSel of ARF: 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D6T4IR(6)’IR(10)’(Add .Reg <- Add. reg)

OutCSel of ARF: IR(5,4) (select register)

MuxASel: 01

FunSel of Registerfile: 10

RegSel of Registerfile: 1111

TempSel of Registerfile: 0111

D6T5IR(6)’IR(10)’

OutASel of registerfile: 100

FunSel of Alu: 0010

MuxBSel: 01

FunSel of ARF: 10

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D7: DestReg <— Src1 + Src2

D7T4IR(10) IR(6) IR(2)(Reg <- Reg + Reg)

OutASel of Registerfile: {0,IR(1,0)} (src1)

OutBSel of Registerfile:{0,IR(5,4)} (src2)

FunSel of ALU: 0100 (src1 + src2)

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, xor with 1

00, 0001, 1000, 0111

SC<-0

D7T4IR(10) IR(6) IR(2)’(Reg <- Reg + Add. Reg)

OutCSel of ARF: IR(1,0)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D7T5IR(10) IR(6) IR(2)’

OutBSel of Registerfile:{0,IR(5,4)} (src2)

OutASel Of RegisterFile:100

FunSel of ALU: 0100 (src1 + src2)

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, xor with 1

SC<-0

D7T4IR(10) IR(6)’ IR(2)(Reg <- Add.Reg + Reg)

OutCSel of ARF: IR(5,4)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D7T5IR(10) IR(6)’ IR(2)

OutBSel of Registerfile:{0,IR(1,0)} (src2)

OutASel Of RegisterFile:100

FunSel of ALU: 0100 (src1 + src2)

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, xor with 1

SC<-0

D7T4IR(10) IR(6)’ IR(2)’(Reg <- Add.Reg + Add. Reg)

OutCSel of ARF: IR(1,0)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D7T5IR(10) IR(6)’ IR(2)’

OutCSel of ARF: IR(5,4)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:1011

D7T6IR(10) IR(6)’ IR(2)’

OutBSel of Registerfile:100

OutASel Of RegisterFile:101

FunSel of ALU: 0100 (src1 + src2)

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, xor with 1

SC<-0

D7T4IR(10)’ IR(6) IR(2)(Add. Reg <- Reg + Reg)

OutASel of Registerfile: {0,IR(1,0)} (src1)

OutBSel of Registerfile:{0,IR(5,4)} (src2)

FunSel of ALU: 0100 (src1 + src2)

MuxBSel = 01 outAlus selected

FunSel of ARF = 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D7T4IR(10)’ IR(6) IR(2)’(Add. Reg <- Reg + Add. Reg)

OutCSel of ARF: IR(1,0)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D7T5IR(10)’ IR(6) IR(2)’

OutBSel of Registerfile:{0,IR(5,4)} (src2)

OutASel Of RegisterFile:100

FunSel of ALU: 0100 (src1 + src2)

MuxBSel = 01 outAlu selected

FunSel of ARF = 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D7T4IR(10)’ IR(6)’ IR(2) (Add. Reg <-Add. Reg + Reg)

OutCSel of ARF: IR(5,4)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D7T5IR(10)’ IR(6)’ IR(2)

OutBSel of Registerfile:{0,IR(1,0)} (src2)

OutASel Of RegisterFile:100

FunSel of ALU: 0100 (src1 + src2)

MuxBSel = 01 outAlu selected

FunSel of ARF = 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D7T4IR(10)’ IR(6)’ IR(2)’(Add. Reg <-Add. Reg + Add. Reg)

OutCSel of ARF: IR(1,0)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D7T5IR(10)’ IR(6)’ IR(2)’

OutCSel of ARF: IR(5,4)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:1011

D7T6IR(10)’ IR(6)’ IR(2)’

OutBSel of Registerfile:100

OutASel Of RegisterFile:101

FunSel of ALU: 0100 (src1 + src2)

MuxBSel = 01 outAlus selected

FunSel of ARF = 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D8 DESTREG <— SRCREG2 - SRCREG1

D8T4IR(10) IR(6) IR(2)(Reg <- Reg - Reg)

OutASel of Registerfile: {0,IR(1,0)} --

OutBSel of Registerfile: {0,IR(5,4)} --

FunSel of ALU: 0110 (src1 - src2)

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, xor with 1

SC<-0

D8T4IR(10) IR(6) IR(2)’(Reg <- Add. Reg - Reg)

OutCSel of ARF: IR(1,0)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D8T5IR(10) IR(6) IR(2)’

OutBSel of Registerfile: {0,IR(5,4)} --

OutASel Of RegisterFile: 100 --

FunSel of ALU: 0110

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, xor with 1

SC<-0

D8T4IR(10) IR(6)’ IR(2)(Reg <- Reg – Add. Reg)

OutCSel of ARF: IR(5,4)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D8T5IR(10) IR(6)' IR(2)

OutBSel of Registerfile: 100 --

OutASel Of Registerfile: {0,IR(1,0)} --

FunSel of ALU: 0110 (src1 - src2)

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, xor with 1

SC<-0

D8T4IR(10) IR(6)' IR(2)'(Reg <- Add.Reg - Add. Reg)

OutCSel of ARF: IR(1,0)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D8T5IR(10) IR(6)' IR(2)'

OutCSel of ARF: IR(5,4)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:1011

D8T6IR(10) IR(6)' IR(2)'

OutBSel of Registerfile:101--

OutASel Of RegisterFile:100--

FunSel of ALU: 0110 (src1 - src2)

MuxASel: 00 OutAlu selected

FunSel of Registerfile: 10

RegSel of Registerfile: Decode IR(9,8), reverse it, xor with 1

SC<-0

D8T4IR(10)’ IR(6) IR(2)(Add. Reg <- Reg - Reg)

OutASel of Registerfile: {0,IR(1,0)} --

OutBSel of Registerfile: {0,IR(5,4)} --

FunSel of ALU: 0110

MuxBSel = 01 outAlus selected

FunSel of ARF = 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D8T4IR(10)’ IR(6) IR(2)’(Add. Reg <- Add.Reg - Reg)

OutCSel of ARF: IR(1,0)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D8T5IR(10)’ IR(6) IR(2)’

OutBSel of Registerfile: {0,IR(5,4)}--

OutASel Of Registerfile: 100--

FunSel of ALU: 0110 (src1 - src2)

MuxBSel = 01 outAlu selected

FunSel of ARF = 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D8T4IR(10)’ IR(6)’ IR(2) (Add. Reg <-Reg – Add.Reg)

OutCSel of ARF: IR(5,4)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D8T5IR(10)’ IR(6)’ IR(2)

OutBSel of Registerfile: 100 --

OutASel Of RegisterFile: {0,IR(1,0)} --

FunSel of ALU: 0110 (src1 - src2)

MuxBSel = 01 outAlu selected

FunSel of ARF = 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D8T4IR(10)’ IR(6)’ IR(2)’(Add. Reg <-Add. Reg - Add. Reg)

OutCSel of ARF: IR(1,0)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:0111

D8T5IR(10)’ IR(6)’ IR(2)’

OutCSel of ARF: IR(5,4)

MuxASel:01

FunSel of Registerfile: 10

Regsel of Registerfile: 1111

TempSel of Registerfile:1011

D8T6IR(10)’ IR(6)’ IR(2)’

OutBSel of Registerfile:101 --

OutASel Of RegisterFile:100 --

FunSel of ALU: 0110

MuxBSel = 01 outAlus selected

FunSel of ARF = 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D9 DESTREG <— LSR SRCREG1

D9T4IR(6)IR(10) (Reg<-reg)

OutASel of Registerfile: {0,IR(5,4)} transfer R as A

FunSel ALU: 1011 OutAlu becomes R

MuxASel:00

FunSel of Registerfile: 10 (load)

RegSel of registerfile: decode IR(9,8) reverse it and xor with 1

SC<-0

D9T4IR(6)'IR(10) (Reg<-Add. reg)

OutCSel of ARF(5,4) (select register)

MuxASel: 01 OutC is selected

RegSel of registerfile: decode IR(9,8) reverse it and xor with 1

FunSel of registerfile: 10 (load)

SC<-0

D9T4IR(6)IR(10)'(Add .Reg <- reg)

OutASel of Registerfile: {0,IR(5,4)}

FunSelALU: 1011 transfer R

MuxBSel: 01 OutAlu selected

FunSel of ARF: 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D9T4IR(6)’IR(10)’(Add .Reg <- Add. reg)

OutCSel of ARF: IR(5,4) (select register)

MuxASel: 01

FunSel of Registerfile: 10

RegSel of Registerfile: 1111

TempSel of Registerfile: 0111

D9T5IR(6)’IR(10)’

OutASel of registerfile: 100

FunSel of Alu: 1011

MuxBSel: 01

FunSel of ARF: 10

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D10 DESTREG <— LSL SRCREG1

D10T4IR(6)IR(10) (Reg<-reg)

OutASel of Registerfile: {0,IR(5,4)} transfer R as A

FunSel ALU: 1010 OutAlu becomes R

MuxASel:00

FunSel of Registerfile: 10 (load)

RegSel of registerfile: decode IR(9,8) reverse it and xor with 1

SC<-0

D10T4IR(6)’IR(10) (Reg<-Add. reg)

OutCSel of ARF(5,4) (select register)

MuxASel: 01 OutC is selected

RegSel of registerfile: decode IR(9,8) reverse it and xor with 1

FunSel of registerfile: 10 (load)

SC<-0

D10T4IR(6)IR(10)’(Add .Reg <- reg)

OutASel of Registerfile: {0,IR(5,4)}

FunSelALU: 1010 transfer R

MuxBSel: 01 OutAlu selected

FunSel of ARF: 10 (load)

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D10T4IR(6)’IR(10)’(Add .Reg <- Add. reg)

OutCSel of ARF: IR(5,4) (select register)

MuxASel: 01

FunSel of Registerfile: 10

RegSel of Registerfile: 1111

TempSel of Registerfile: 0111

D10T5IR(6)’IR(10)’

OutASel of registerfile: 100

FunSel of Alu: 1010

MuxBSel: 01

FunSel of ARF: 10

RegSel of ARF: decode IR(9,8), reverse and xor with 1 (least 3 bits)

SC<-0

D11: M[SP] <- Rx, SP <- SP -1

IR6 (Address is on Register File):

D11T4IR6:

OutASel: IR(4,5)	Select SRCREG1
FunSelALU: 0000	Transfer A input of ALU
OutDSel: 11	Get value of SP
STR: 1	Write to Memory

IR6’ (Address is on ARF):

D11T4IR6’:

OutCSel: IR(4,5)	Select SRCREG1
MuxASel: 01	Select Output of ARF
FunSelRF: 10	Load to RF
RegSel: 1111	Disable general Registers
TmpSel: 0111	Load to T1

D11T5IR6’:

OutASel: 100	Output T1
FunSelALU: 0000	Transfer A
OutDSel: 11	Get value of SP
STR: 1	Write to Memory

(T6 common for both)

D11T6IR6’, D11T6IR6:

FunSelARF: 00	decrement selected Register
RegSelARF: 110	SP is selected

SC <- 0

D12: SP <- SP + 1, Rx <- M[SP]

(T4 is common for both)

D12T4IR10, D12T4IR10’:

FunSelARF: 01	increment selected Register
RegSelARF: 110	SP is selected

IR10 (Address is on Register File):

D12T5IR10:

Load: 1	Read From RAM
OutDSel: 11	Get value of SP
MuxASel: 10	Select output of RAM
FunSelRF: 10	Load to RF
RegSelRF:	Decode IR(9,8) to RF address

SC <- 0

IR10’ (Address is on ARF):

D12T5IR10‘:

Load: 1	Read From RAM
OutDSel: 11	Get value of SP
MuxBSel: 10	Select output of RAM

FunSelARF: 10	Load to ARF
RegSelARF:	Decode IR(9,8) to ARF address

SC <- 0

D13: DESTREG <- SRCREG1 + 1

IR10IR6 (reg <- reg + 1):

D13T4IR10IR6:	load from reg to reg
OutASel: IR(4,5)	Select SRCREG1
FunSelAlu: 0000	Transfer A
MuxASel: 00	Select output of ALU
FunSelRF: 10	Load to RF
RegSelRF:	Decode IR(8,9) to RF address
D13T5IR10IR6:	increment DESTREG
FunSelRF: 01	
RegSelRF:	Decode IR(8,9) to RF address

SC <- 0

IR10IR6’ (reg <- add. reg + 1):

D13T4 IR10IR6’:	load from add. reg to reg
OutCSel: IR(4,5)	Select SRCREG1
MuxASel: 01	Select output of ARF
FunSelRF: 10	Load to RF
RegSelRF:	Decode IR(8,9) to RF address

D13T5 IR10IR6':	increment DESTREG
FunSelRF: 01	
RegSelRF:	Decode IR(8,9) to RF address
SC <- 0	

IR10'IR6 (add. reg <- reg + 1):

D13T4 IR10'IR6:	load from reg to add. reg
OutASel: IR(4,5)	Select SRCREG1
FunSelAlu: 0000	Transfer A
MuxBSel: 01	Select output of RF
FunSelARF: 10	Load to ARF
RegSelARF:	Decode IR(8,9) to ARF address
D13T5 IR10'IR6:	increment DESTREG
FunSelARF: 01	
RegSelARF:	Decode IR(8,9) to ARF address
SC <- 0	

IR10'IR6' (add. reg <- add. reg + 1):

D13T4 IR10'IR6':	load from add. reg to add. reg
OutCSel: IR(4,5)	Select SRCREG1
MuxBSel: 00	Select output of ARF
FunSelARF: 10	Load to ARF
RegSelARF:	Decode IR(8,9) to ARF address

D13T5 IR10’IR6’:	increment DESTREG
FunSelARF: 01	
RegSelARF:	Decode IR(8,9) to ARF address
SC <- 0	

D14: DESTREG <- SRCREG1 - 1

IR10IR6 (reg <- reg - 1):

D14T4 IR10IR6:

OutASel: IR(4,5)	Select SRCREG1
FunSelAlu: 0000	Transfer A
MuxASel: 00	Select output of ALU
FunSelRF: 10	Load to RF
RegSelRF:	Decode IR(8,9) to RF address
D14T5 IR10IR6:	decrement DESTREG
FunSelRF: 00	
RegSelRF:	Decode IR(8,9) to RF address
SC <- 0	

IR10IR6’ (reg <- add. reg - 1):

D14T4 IR10IR6’:

OutCSel: IR(4,5)	Select SRCREG1
MuxASel: 01	Select output of ARF
FunSelRF: 10	Load to RF
RegSelRF:	Decode IR(8,9) to RF address

D14T5 IR10IR6’:	decrement DESTREG
FunSelRF: 00	
RegSelRF:	Decode IR(8,9) to RF address
SC <- 0	

IR10’IR6 (add. reg <- reg - 1):

D14T4 IR10’IR6:

OutASel: IR(4,5)	Select SRCREG1
FunSelAlu: 0000	Transfer A
MuxBSel: 01	Select output of RF
FunSelARF: 10	Load to ARF
RegSelARF:	Decode IR(8,9) to ARF address
D14T5 IR10’IR6:	decrement DESTREG
FunSelARF: 00	
RegSelARF:	Decode IR(8,9) to ARF address
SC <- 0	

IR10’IR6’ (add. reg <- add. reg - 1):

D14T4 IR10’IR6’:

OutCSel: IR(4,5)	Select SRCREG1
MuxBSel: 00	Select output of ARF
FunSelARF: 10	Load to ARF
RegSelARF:	Decode IR(8,9) to ARF address

D14T5 IR10'IR6': decrement DESTREG

FunSelARF: 00

RegSelARF: Decode IR(8,9) to ARF address

SC <- 0

D15: IF Z=0 THEN PC <- Value

D15T4:

MuxBSel: 11 Select IR(0-7)

FunSelARF: 10 Load to ARF

RegSelARF: Z11

IF Z=0 THEN 011 ELSE 111 (if Z=0 select PC, else don't select)

SC <- 0

Decoding of RF addresses

----

00 -> 0111

01 -> 1011

10 -> 1101

11 -> 1110

Decode using decoder 00 -> 0001

Reverse 0001 -> 1000

NOT 1000 -> 0111

Decoding of ARF addresses

00, 01 -> 011

10 -> 101

11 -> 110

Address is 2 bit, Decoded Address is 3 bit.

Equations:

Decoded(0) = (address(0).address(1))'

Decoded(1) = (address(0)'.address(1))' = address(0) + address(1)’

Decoded(2) = address(1)

Conclusion

In this project, we have gained experience on designing hardwired control unit. Determining the control operations’ expressions was incredibly time consuming. To simplify implementation of circuits, Boolean Algebra Theorems and Axioms are used.