PROJECT IITB_RISC_PIPELINE

E KRITHEESH (200070018)

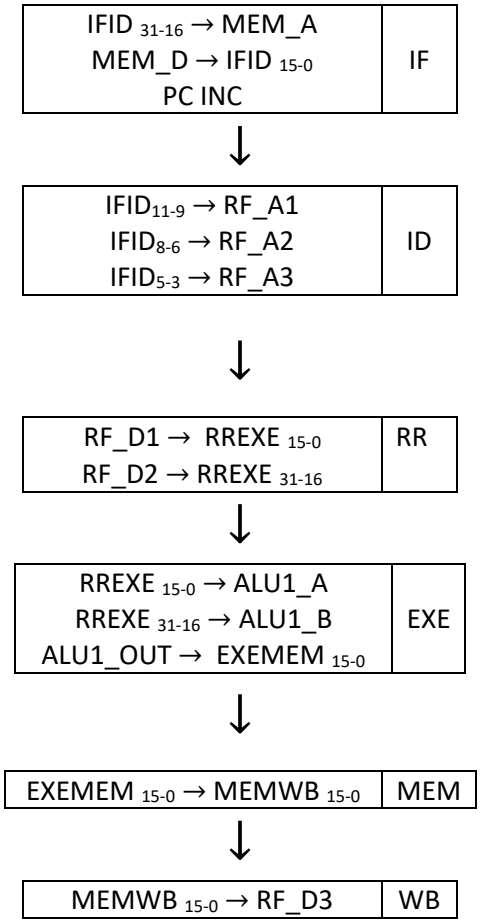
AKSHAY VERMA (200070005)

ANKITH R (200070006)

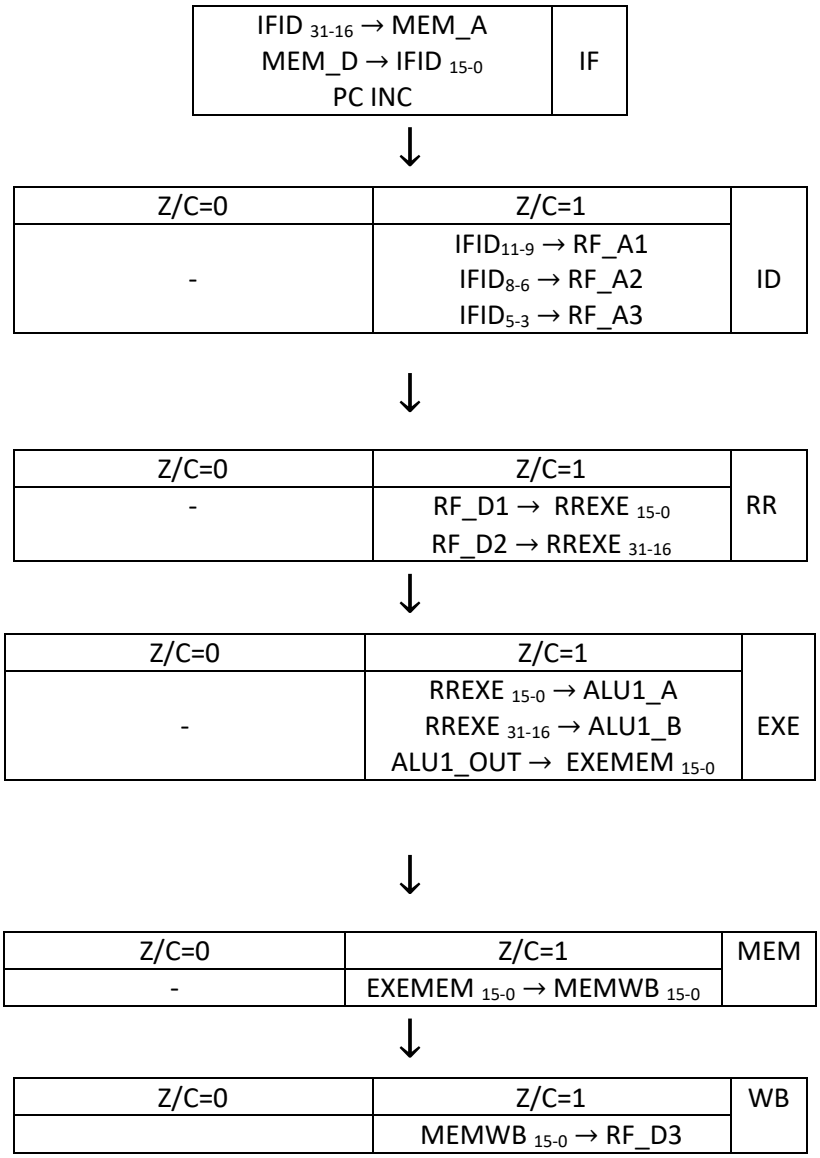
GOWTHAM S (20D070031)

Level 2 Flowcharts

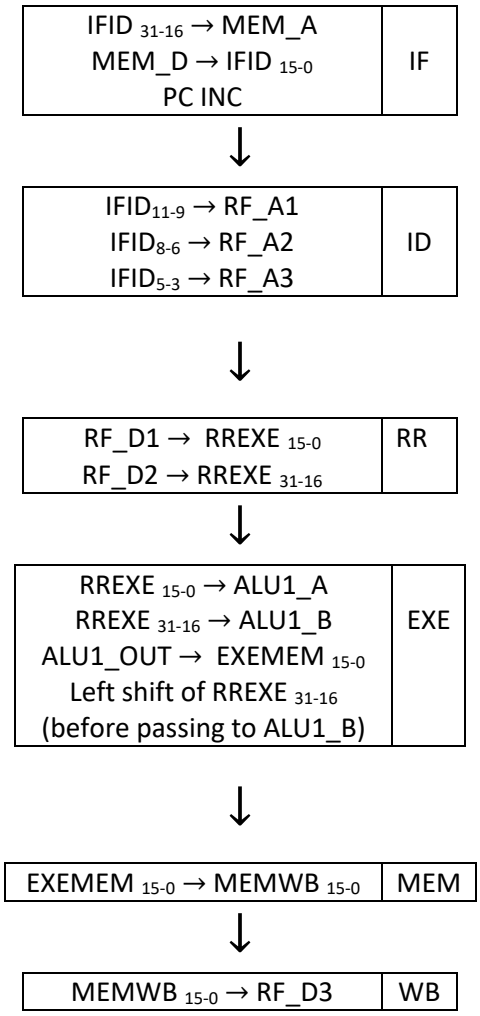
ADD & NDU:



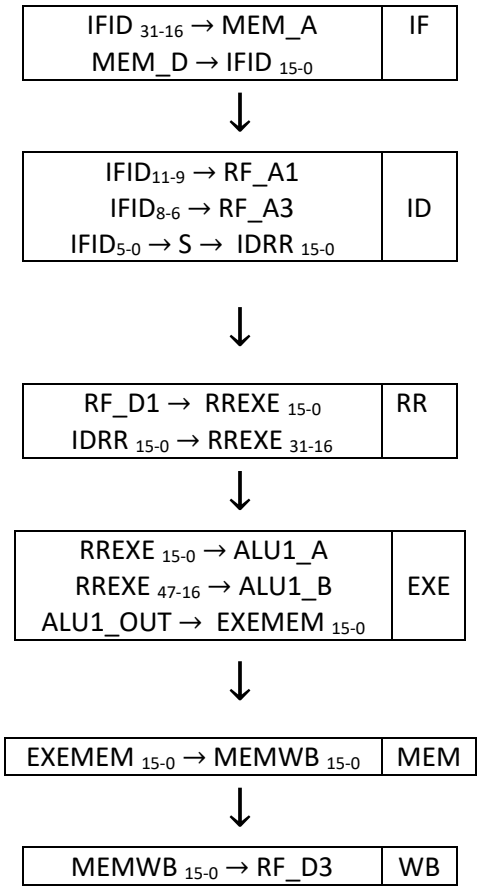
ADC/ADZ/NDC/NDZ:



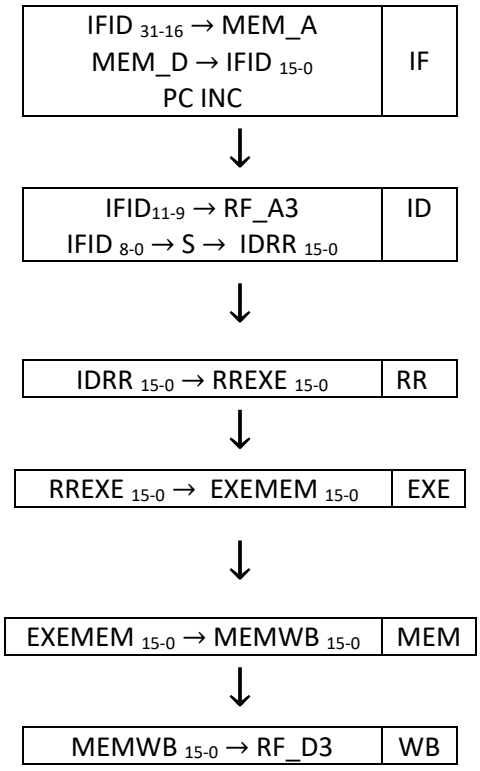
ADL:


IFID 31-16 → MEM_A
MEM_D → IFID 15-0
PC INC
IF
IFID11-9 → RF_A1
IFID8-6 → RF_A2
IFID5-3 → RF_A3
ID
RF_D1 → RREXE 15-0
RF_D2 → RREXE 31-16
RR
RREXE 15-0 → ALU1_A
RREXE 31-16 → ALU1_B
ALU1_OUT → EXEMEM 15-0
Left shift of RREXE 31-16
(before passing to ALU1_B)
EXE
EXEMEM 15-0 → MEMWB 15-0
MEM
MEMWB 15-0 → RF_D3
WB

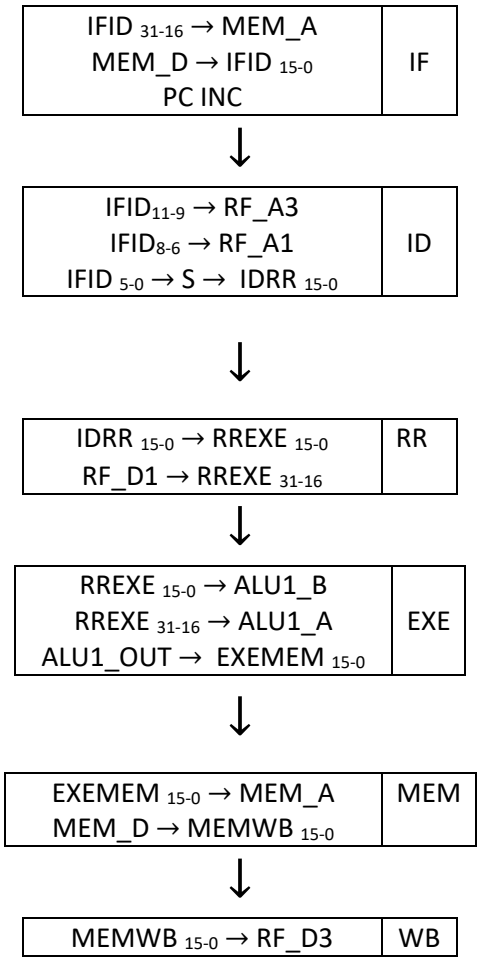
ADI:


IFID 31-16 → MEM_A
MEM_D → IFID 15-0
IF
IFID11-9 → RF_A1
IFID8-6 → RF_A3
IFID5-0 → S → IDRR 15-0
ID
RF_D1 → RREXE 15-0
IDRR 15-0 → RREXE 31-16
RR
RREXE 15-0 → ALU1_A
RREXE 47-16 → ALU1_B
ALU1_OUT → EXEMEM 15-0
EXE
EXEMEM 15-0 → MEMWB 15-0
MEM
MEMWB 15-0 → RF_D3
WB

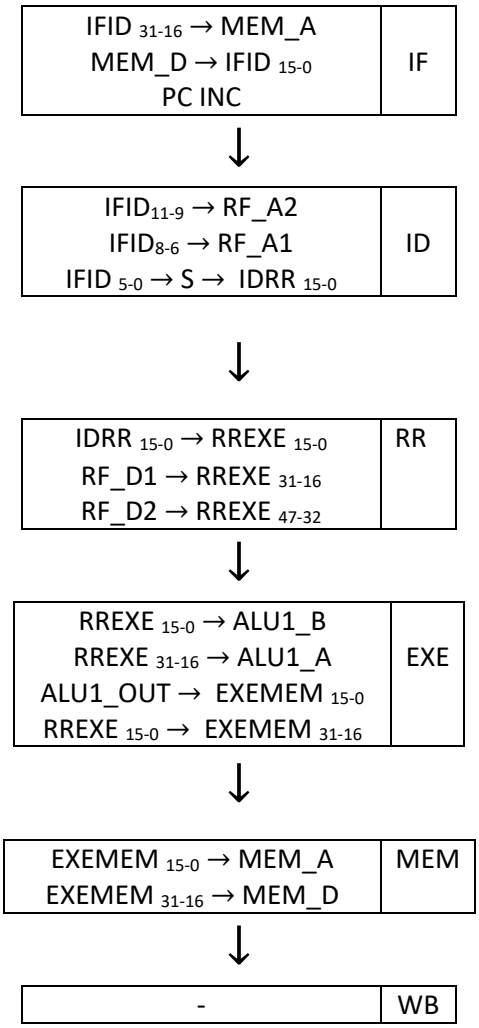
LHI:


IFID 31-16 → MEM_A
MEM_D → IFID 15-0
PC INC
IF
IFID11-9 → RF_A3
IFID 8-0 → S → IDRR 15-0
ID
IDRR 15-0 → RREXE 15-0
RR
RREXE 15-0 → EXEMEM 15-0
EXE
EXEMEM 15-0 → MEMWB 15-0
MEM
MEMWB 15-0 → RF_D3
WB

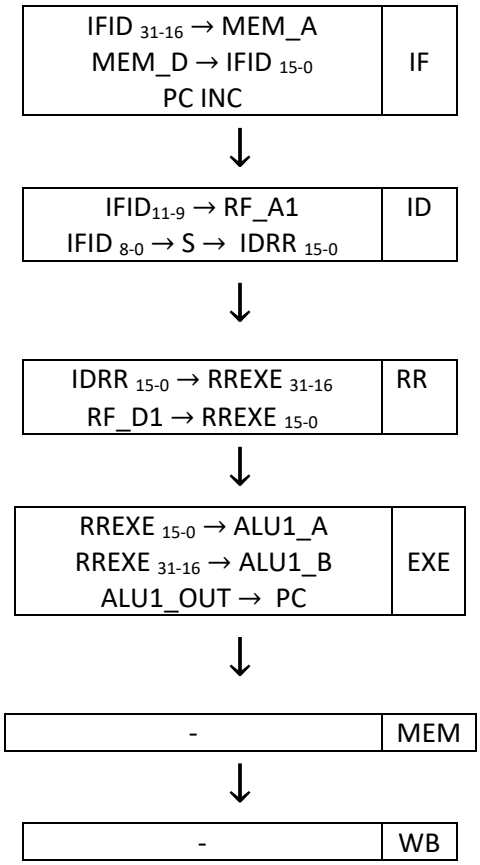
LW:


IFID 31-16 → MEM_A
MEM_D → IFID 15-0
PC INC
IF
IFID11-9 → RF_A3
IFID8-6 → RF_A1
IFID 5-0 → S → IDRR 15-0
ID
IDRR 15-0 → RREXE 15-0
RF_D1 → RREXE 31-16
RR
RREXE 15-0 → ALU1_B
RREXE 31-16 → ALU1_A
ALU1_OUT → EXEMEM 15-0
EXE
EXEMEM 15-0 → MEM_A
MEM_D → MEMWB 15-0
MEM
MEMWB 15-0 → RF_D3
WB

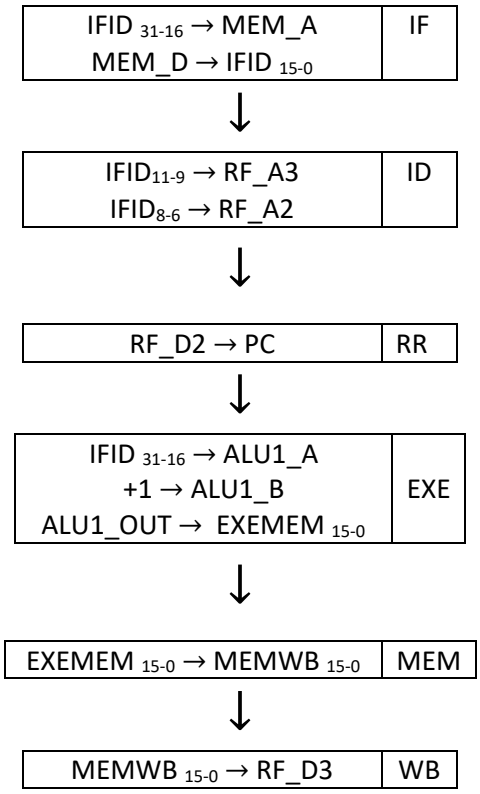
SW:


IFID 31-16 → MEM_A
MEM_D → IFID 15-0
PC INC
IF
IFID11-9 → RF_A2
IFID8-6 → RF_A1
IFID 5-0 → S → IDRR 15-0
ID
IDRR 15-0 → RREXE 15-0
RF_D1 → RREXE 31-16
RF_D2 → RREXE 47-32
RR
RREXE 15-0 → ALU1_B
RREXE 31-16 → ALU1_A
ALU1_OUT → EXEMEM 15-0
RREXE 15-0 → EXEMEM 31-16
EXE
EXEMEM 15-0 → MEM_A
EXEMEM 31-16 → MEM_D
MEM
-
WB

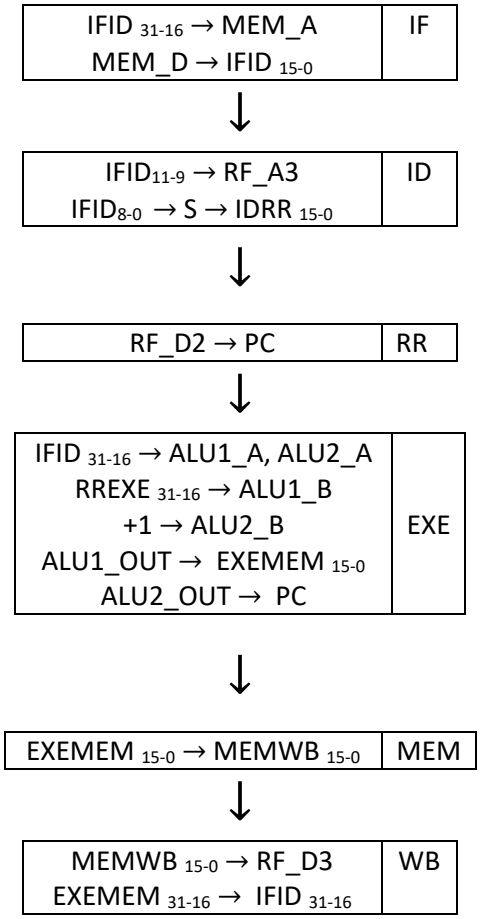
JRI:


IFID 31-16 → MEM_A
MEM_D → IFID 15-0
PC INC
IF
IFID11-9 → RF_A1
IFID 8-0 → S → IDRR 15-0
ID
IDRR 15-0 → RREXE 31-16
RF_D1 → RREXE 15-0
RR
RREXE 15-0 → ALU1_A
RREXE 31-16 → ALU1_B
ALU1_OUT → PC
EXE
-
MEM
-
WB

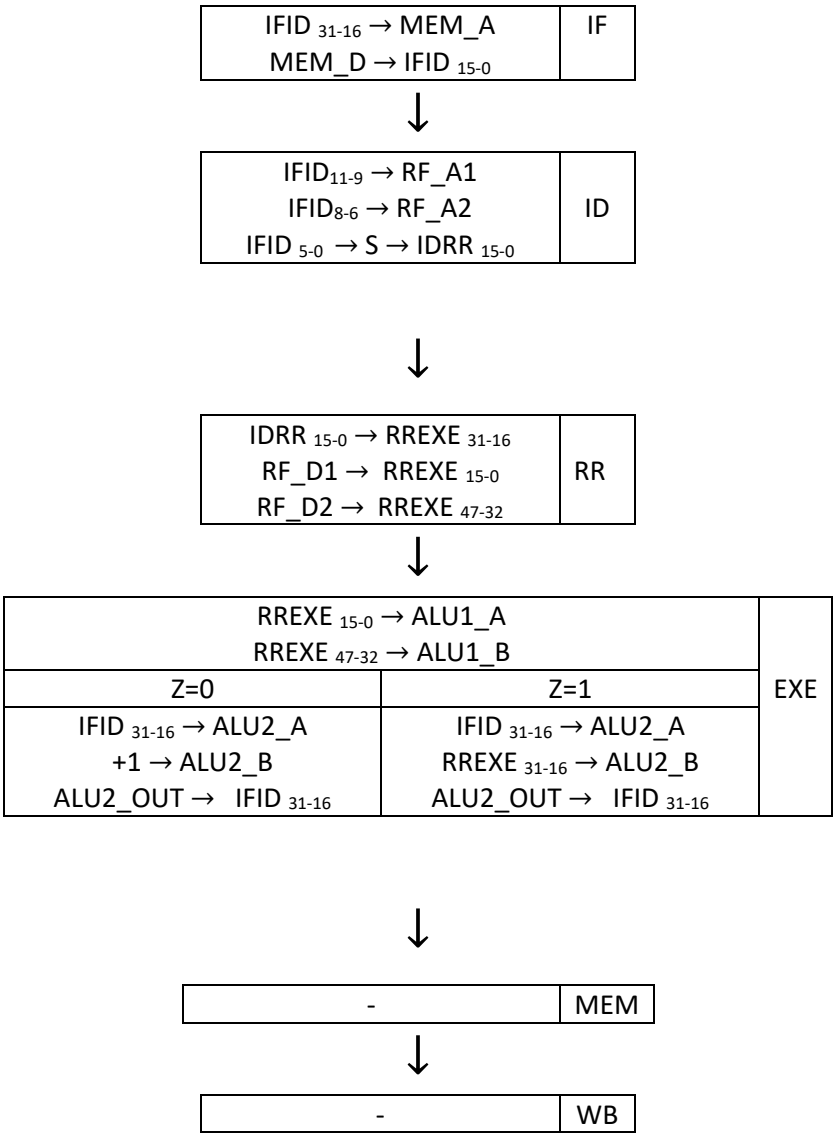
JLR:


IFID 31-16 → MEM_A
MEM_D → IFID 15-0
IF
IFID11-9 → RF_A3
IFID8-6 → RF_A2
ID
RF_D2 → PC
RR
IFID 31-16 → ALU1_A
+1 → ALU1_B
ALU1_OUT → EXEMEM 15-0
EXE
EXEMEM 15-0 → MEMWB 15-0
MEM
MEMWB 15-0 → RF_D3
WB

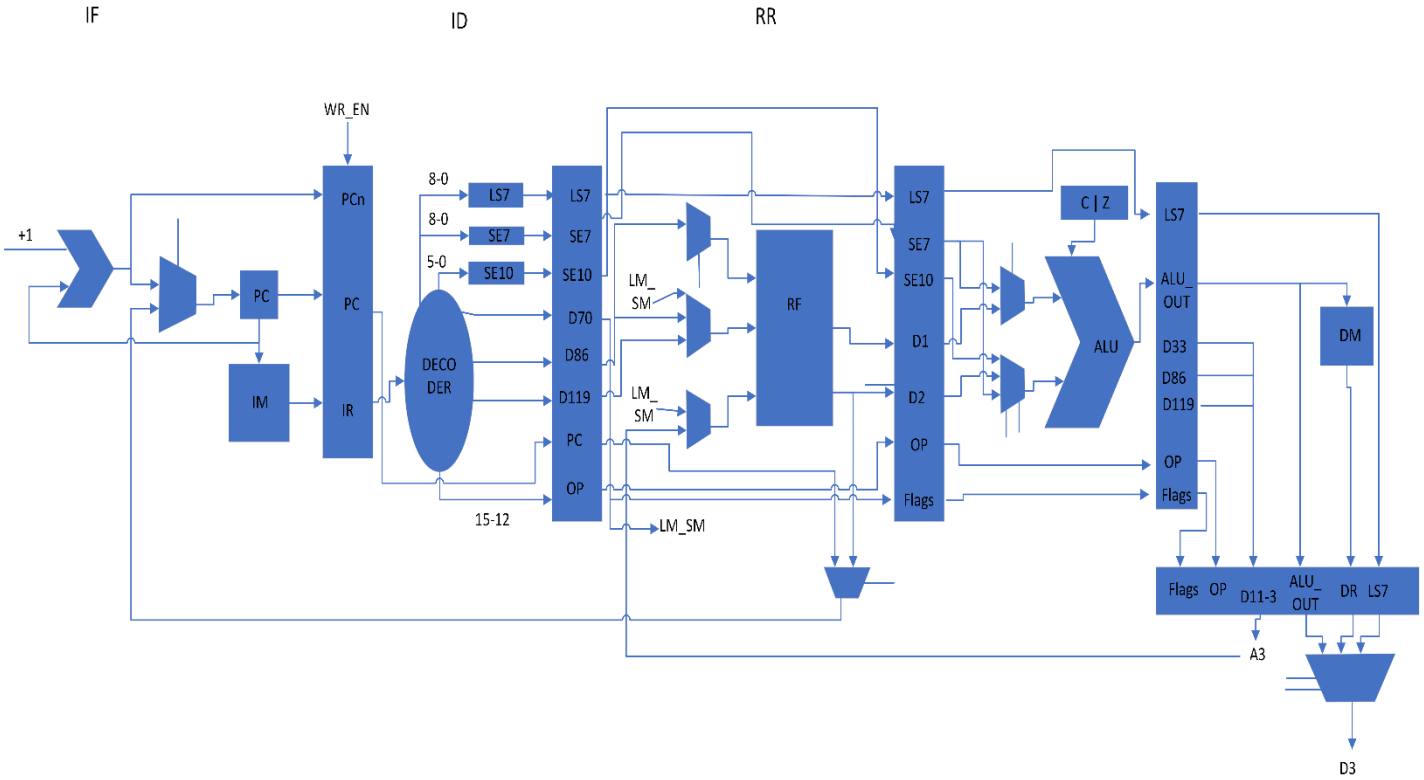
JAL:


IFID 31-16 → MEM_A
MEM_D → IFID 15-0
IF
IFID11-9 → RF_A3
IFID8-0 → S → IDRR 15-0
ID
RF_D2 → PC
RR
IFID 31-16 → ALU1_A, ALU2_A
RREXE 31-16 → ALU1_B
+1 → ALU2_B
ALU1_OUT → EXEMEM 15-0
ALU2_OUT → PC
EXE
EXEMEM 15-0 → MEMWB 15-0
MEM
MEMWB 15-0 → RF_D3
EXEMEM 31-16 → IFID 31-16
WB

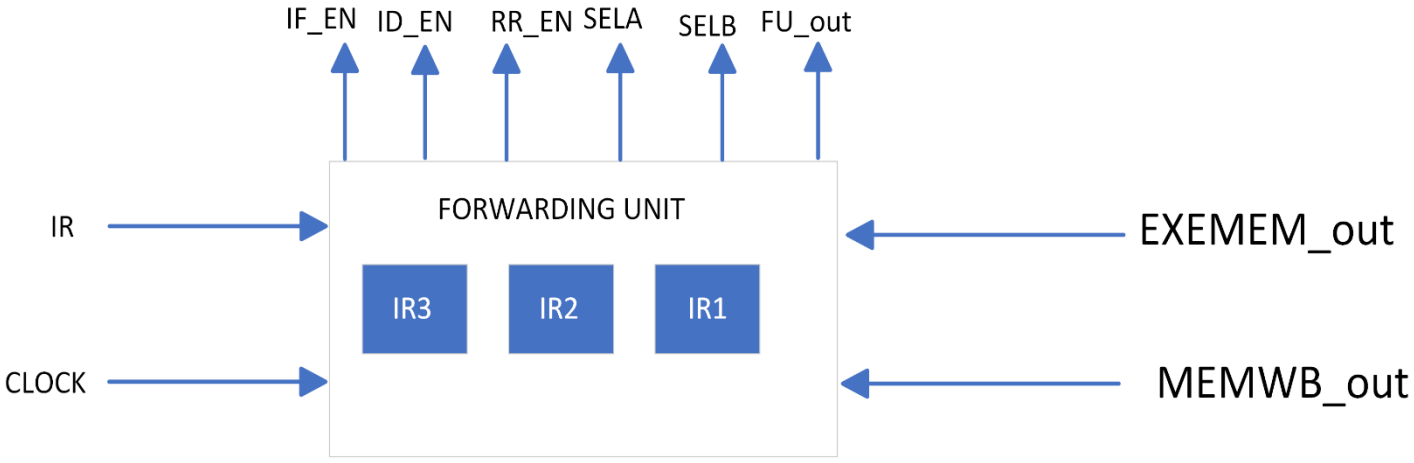
BEQ:


IFID 31-16 → MEM_A
MEM_D → IFID 15-0
IF
IFID11-9 → RF_A1
IFID8-6 → RF_A2
IFID 5-0 → S → IDRR 15-0
ID
IDRR 15-0 → RREXE 31-16
RF_D1 → RREXE 15-0
RF_D2 → RREXE 47-32
RR
RREXE 15-0 → ALU1_A
RREXE 47-32 → ALU1_B
Z=0
Z=1
IFID 31-16 → ALU2_A
+1 → ALU2_B
ALU2_OUT → IFID 31-16
IFID 31-16 → ALU2_A
RREXE 31-16 → ALU2_B
ALU2_OUT → IFID 31-16
EXE
-
MEM
-
WB

DATAPATH



FORWARDING UNIT



FORWARDING UNIT TABLE

Current Instruction(I1)	**Next or Next to Next Instruction(I2)**	**Forwarding**
ADD/ADC (with C=1)/ ADZ (with Z=1)/ NDU/NDC/ NDZ/ADL	ADD/ADC/ADZ/ADL/ ADI/NDU/NDC/NDZ/ BEQ/SW	if IR^1_{5-3} = = IR^2_{11-9} or IR^2_{8-6} then ALU_A(or ALU_B) = EXE_MEM_ALU_out if IR^1_{5-3} = = IR^3_{11-9} or IR^3_{8-6} then ALU_A(or ALU_B) = MEM_WB_ALU_out
ADD/ADC (with C=1)/ ADZ (with Z=1)/ NDU/NDC/ NDZ/ADL	JRI/SM/LM/ADI	if IR^1_{5-3} == IR^2_{11-9} then ALU_A = EXE_MEM_ALU_out if IR^1_{5-3} == IR^3_{11-9} then ALU_A = MEM_WB_ALU_out
ADD/ADC (with C=1)/ ADZ (with Z=1)/ NDU/NDC/ NDZ/ADL	LW/JLR	if IR^1_{5-3} == IR^2_{8-6} then ALU_B = EXE_MEM_ALU_out if IR^1_{5-3} == IR^3_{8-6} then ALU_B = MEM_WB_ALU_out
ADI	ADD/ADC/ADZ/ADL/ ADI/NDU/NDC/NDZ/ BEQ/SW	if IR^1_{8-6} = = IR^2_{11-9} or IR^2_{8-6} then ALU_A(or ALU_B) = EXE_MEM_ALU_out if IR^1_{8-6} = = IR^3_{11-9} or IR^3_{8-6} then ALU_A(or ALU_B) = MEM_WB_ALU_out
ADI	JRI/SM/LM/ADI	if IR^1_{8-6} == IR^2_{11-9} then ALU_A = EXE_MEM_ALU_out if IR^1_{8-6} == IR^3_{11-9} then ALU_A = MEM_WB_ALU_out
ADI	LW/JLR	if IR^1_{8-6} == IR^2_{8-6} then ALU_B = EXE_MEM_ALU_out if IR^1_{8-6} == IR^3_{8-6} then ALU_B = MEM_WB_ALU_out
JLR/LHI/JAL	ADD/ADC/ADZ/ADL/ ADI/NDU/NDC/NDZ/ BEQ/SW	if IR^1_{11-9} = = IR^2_{11-9} or IR^2_{8-6} then ALU_A(or ALU_B) = EXE_MEM_ALU_out if IR^1_{11-9} = = IR^3_{11-9} or IR^3_{8-6} then ALU_A(or ALU_B) = MEM_WB_ALU_out
JLR/LHI/JAL	JRI/SM/LM/ADI	if IR^1_{11-9} == IR^2_{11-9} then ALU_A = EXE_MEM_ALU_out if IR^1_{11-9} == IR^3_{11-9} then ALU_A = MEM_WB_ALU_out

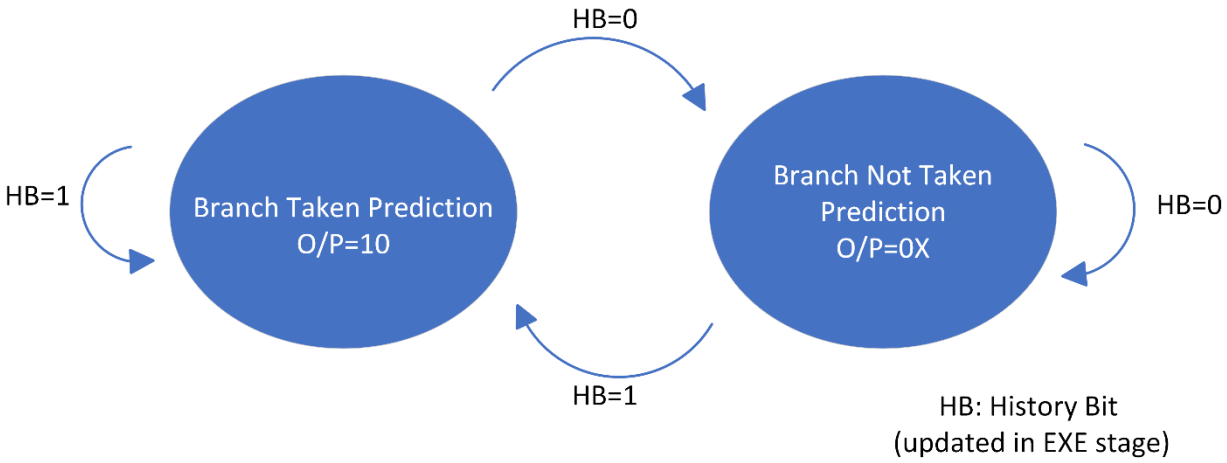
JLR/LHI/JAL	LW/JLR	if IR^1_{11-9} == IR^2_{8-6} then
		ALU_B = EXE_MEM_ALU_out
		
		if IR^1_{11-9} == IR^3_{8-6} then
		ALU_B = MEM_WB_ALU_out

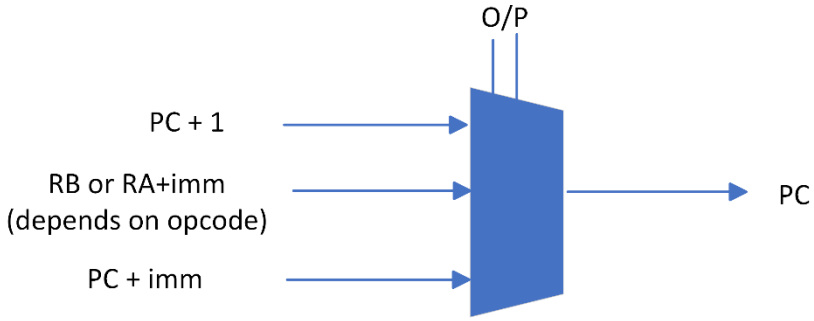
LW	ADD/ADC/ADZ/ADL/
	ADI/NDU/NDC/NDZ/
	BEQ/SW
		if IR^1_{11-9} == IR^2_{11-9} or IR^2_{8-6} then
		1) STOP FOR ONE CLOCK CYCLE
		2) ALU_A (or ALU_B) =
		MEM_WB_ALU_out
		
		if IR^1_{11-9} == IR^3_{11-9} or IR^3_{8-6} then
		ALU_A (or ALU_B) = MEM_WB_ALU_out

LW	JRI/SM/LM/ADI	if IR^1_{11-9} == IR^2_{11-9} or IR^2_{8-6} then
		1) STOP FOR ONE CLOCK CYCLE
		2) ALU_A = MEM_WB_ALU_out
		
		if IR^1_{11-9} == IR^3_{11-9} or IR^3_{8-6} then
		ALU_A = MEM_WB_ALU_out

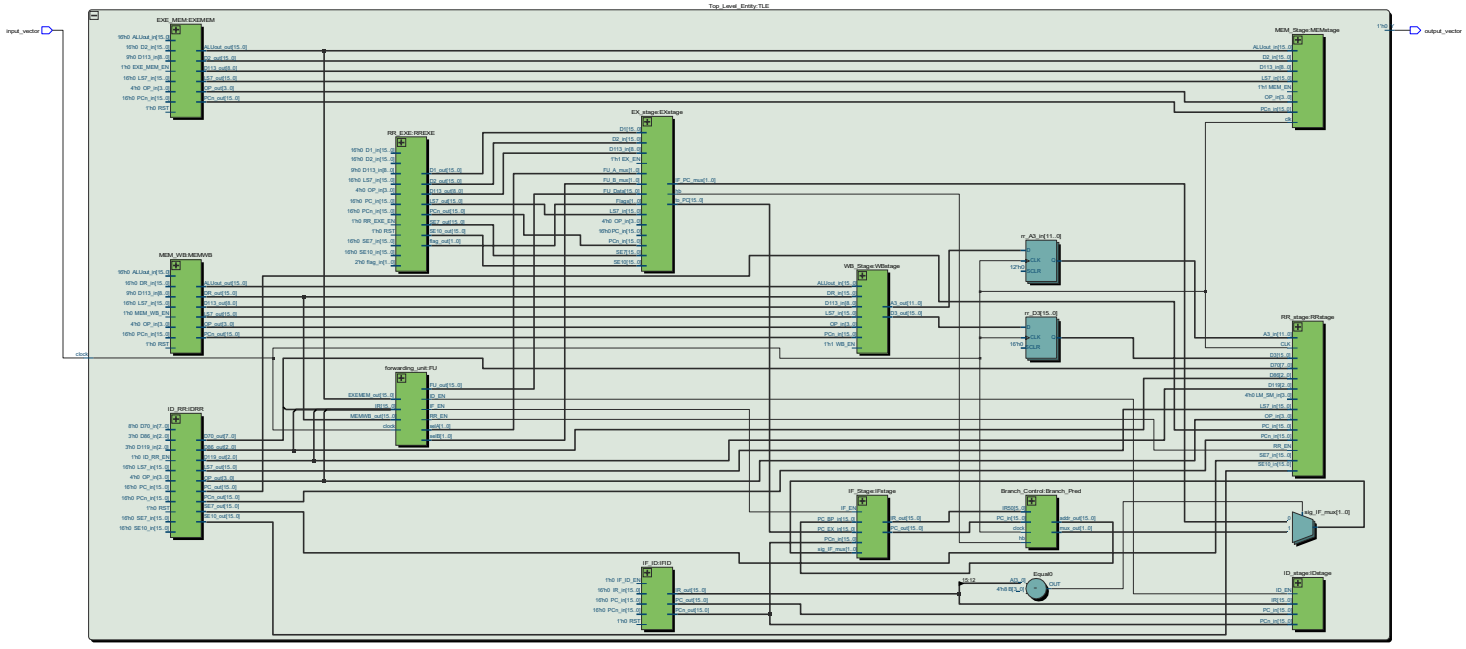
LW	LW/JLR	if IR^1_{11-9} == IR^2_{8-6} or IR^2_{8-6} then
		1) STOP FOR ONE CLOCK CYCLE
		2) ALU_B = MEM_WB_ALU_out
		
		if IR^1_{11-9} == IR^3_{8-6} or IR^3_{8-6} then
		ALU_B = MEM_WB_ALU_out

BRANCH PREDICTOR





RTL View


Top_Level_Entity:TLE
input_vector
output_vector
clock
EXE_MEM:EXEMEM
MEM_WB:MEMWB
ID_RR:IDRR
RR_EXE:RREXE
EX_stage:EXstage
MEM_Stage:MEMstage
WB_Stage:WBstage
RR_stage:RRstage
IF_Stage:IFstage
IF_ID:IFID
ID_stage:IDstage
forwarding_unit:FU
Branch_Control:Branch_Pred
Equal0
sig_IF_mux[1..0]