MINI PROJECT 6 – Instruction Decoder

EE 477

Yexuan Chen, Yuxiang Chen, Xinyi Chang

1. Introduction:

The purpose of this project is to design and synthesize the CPU core controller, Instruction Decoder(ID), in verilog using the SAPR flow. The Instruction Decoder takes in a 16-bit binary instruction in each cycle, decodes the instruction and outputs appropriate control signals and operands to the datapath for further processes. Also, in this project, we implemented and synthesized a Program Counter(PC) in conjunction with the Instruction Decoder module.

2. Analytical Approach

We designed our ID module based on the Figure 2.1, which shows all assembly instructions expected to support in the CPU design. In order to designed and implemented the CPU, we connected datapath, Data Memory, Instruction Memory module and all other verilog modules with our PC and ID module. We use the CPU to simulate and test the functionality of ID.

16-bit Instruction																Assembler	Operation	S updates	Action
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
0	0	1	0	0	Rd			imm8								MOVS Rd, #<imm>	Move	N Z	Rd := imm8, ZeroExt(<imm8>, 16)
0	1	0	0	0	1	1	0	0	Rm			Rd				MOV Rd, Rm			Rd = Rm
0	0	0	1	1	1	0	imm3		Rn			Rd				ADDS Rd, Rn, #<imm3>	Add	N Z C V	Rd := Rn + imm3
0	0	0	1	1	0	0	Rm		Rn			Rd				ADDS Rd, Rn, Rm		N Z C V	Rd := Rn + Rm
1	0	1	1	0	0	0	0	0	imm7							ADD SP, SP, #<imm7>			SP := SP + ZeroExt(<imm7>, 16)
0	0	0	1	1	0	1	Rm		Rn			Rd				SUBS Rd, Rn, Rm	Subtract	N Z C V	Rd := Rn + ~(Rm) + 1
0	0	0	1	1	1	1	imm3		Rn			Rd				SUBS Rd, Rn, #<imm3>		N Z C V	Rd := Rn + ~(imm3) + 1
1	0	1	1	0	0	0	0	1	imm7							SUB SP, SP, #<imm7>			Rd := SP + ~(ZeroExt(<imm7>, 16)) + 1
0	1	0	0	0	0	1	0	1	0	Rm		Rn				CMP Rn, Rm	Compare	N Z C V	Result := Rn + ~(Rm) + 1; Result is not used
0	1	0	0	0	0	0	0	0	0	Rm		Rdn				ANDS Rdn, Rm	Logical	N Z	Rd := Rn AND Rm
0	1	0	0	0	0	0	0	0	1	Rm		Rdn				EORS Rdn, Rm		N Z	Rd := Rd XOR Rm
0	1	0	0	0	0	1	1	0	0	Rm		Rdn				ORRS Rdn, Rm		N Z	Rd := Rd OR Rm
0	1	0	0	0	0	1	1	1	1	Rm		Rdn				MVNS Rdn, Rm		N Z	Rd := ~(Rm)
0	1	0	0	0	0	0	0	1	0	Rm		Rdn				LSLS Rd, Rd, Rm	Shift	N Z C	Rd := Rd << Rm
0	1	0	0	0	0	0	0	1	1	Rm		Rdn				LSRS Rd, Rd, Rm		N Z C	Rd := Rd >> Rm
0	1	0	0	0	0	0	1	0	0	Rm		Rdn				ASRS Rd, Rd, Rm		N Z C	Rd := Rd >>> Rm
0	1	0	0	0	0	0	1	1	1	Rm		Rdn				RORS Rd, Rd, Rm	Rotate		Right shift Rm amount of bits, the shifted out bits are inserted into the vacated bits on the left
0	1	1	0	0	imm5					Rn		Rd				STR Rd, [Rn, #<imm5>]	Store		Mem[Rn + ZeroExtend(imm5, 16), 2] = Rd
0	1	1	0	1	imm5					Rn		Rd				LDR Rd, [Rn, #<imm5>]	Load	N Z C	Rd = Mem[Rn + ZeroExtend(imm5, 16), 2]
1	1	0	1	cond				imm8								B<cc> <label>	Branch		If the condition is satisfied, PC = PC + SignExt(imm8, 16)
1	1	1	0	0	imm11											B <label>			PC = PC + SignExt(imm11, 16)
0	1	0	0	0	1	0	1	0	0	imm6						BL <label>			LR = PC+1; PC = PC + SignExt(imm6, 16)
0	1	0	0	0	1	1	1	0	Rm				0	0	0	BX Rm			PC = Rm
1	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	NOOP	No Operation		Execution stalls for one cycles

Figure 2.1 All Assembly Instruction Support in CPU

3. Behaviorial Verilog Design

We used the provided test decks(test0.bin) to test the functionality. Figure 3.1 shows the simulation waveform of our behaviour verilog design, and it is funcionally correct.

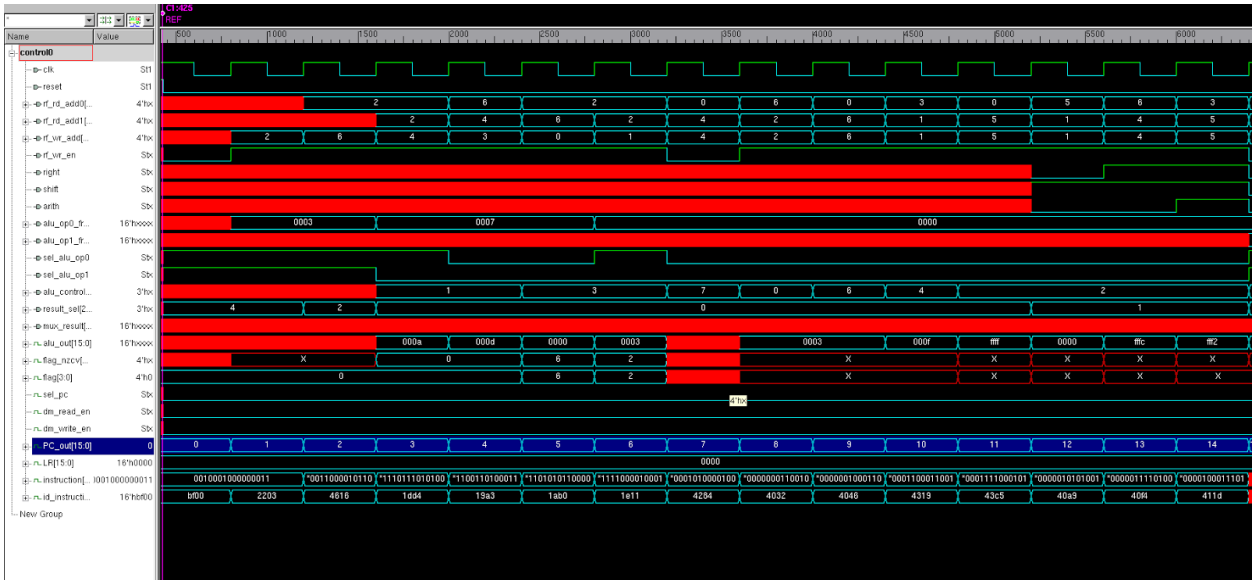

Figure 3.1 Simulation Waveform of Behaviorial Verilog Design

4. Synthesize Verilog Design

Figure 4.1 shows the synthesized design simulation.

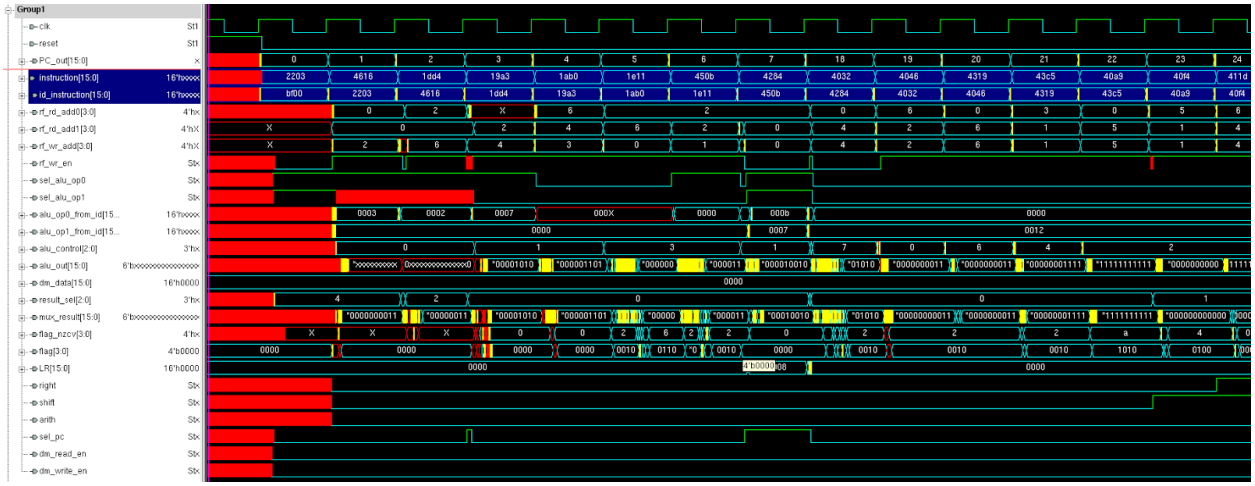

Figure 4.1 Simulation Waveform of Behaviorial Verilog Design

5. Post-layout Design

Figure 5.1 shows the auto-generated layout simulation result. The minimum clock cycle time is 8 ns.

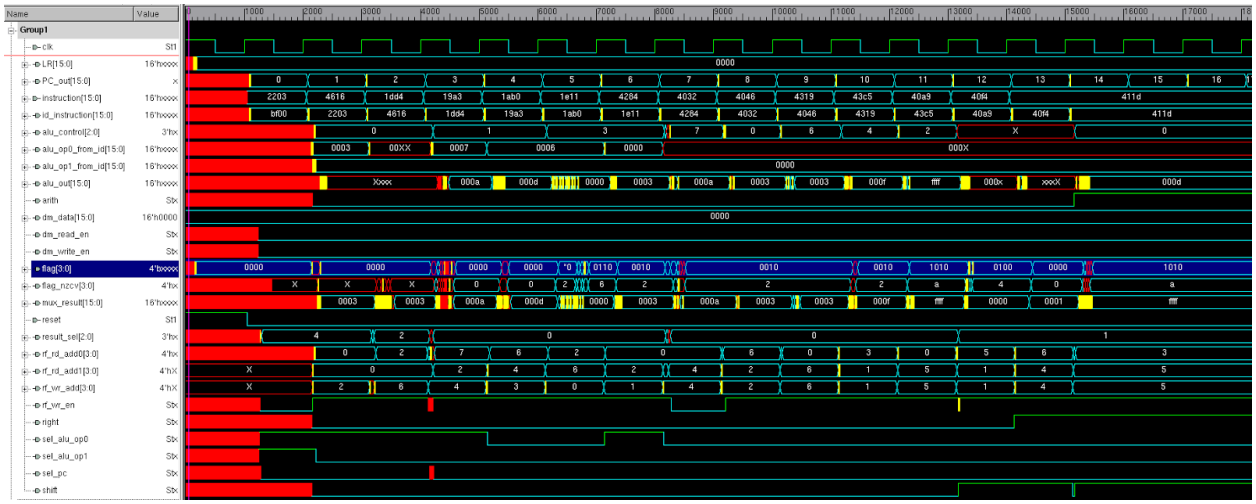

Figure 5.1 SAPR Simulation

Figure 5.2 shows the auto-generated layout of the **complete CPU.** And Figure 5.3 shows the auto-generated layout of the instruction decoder.

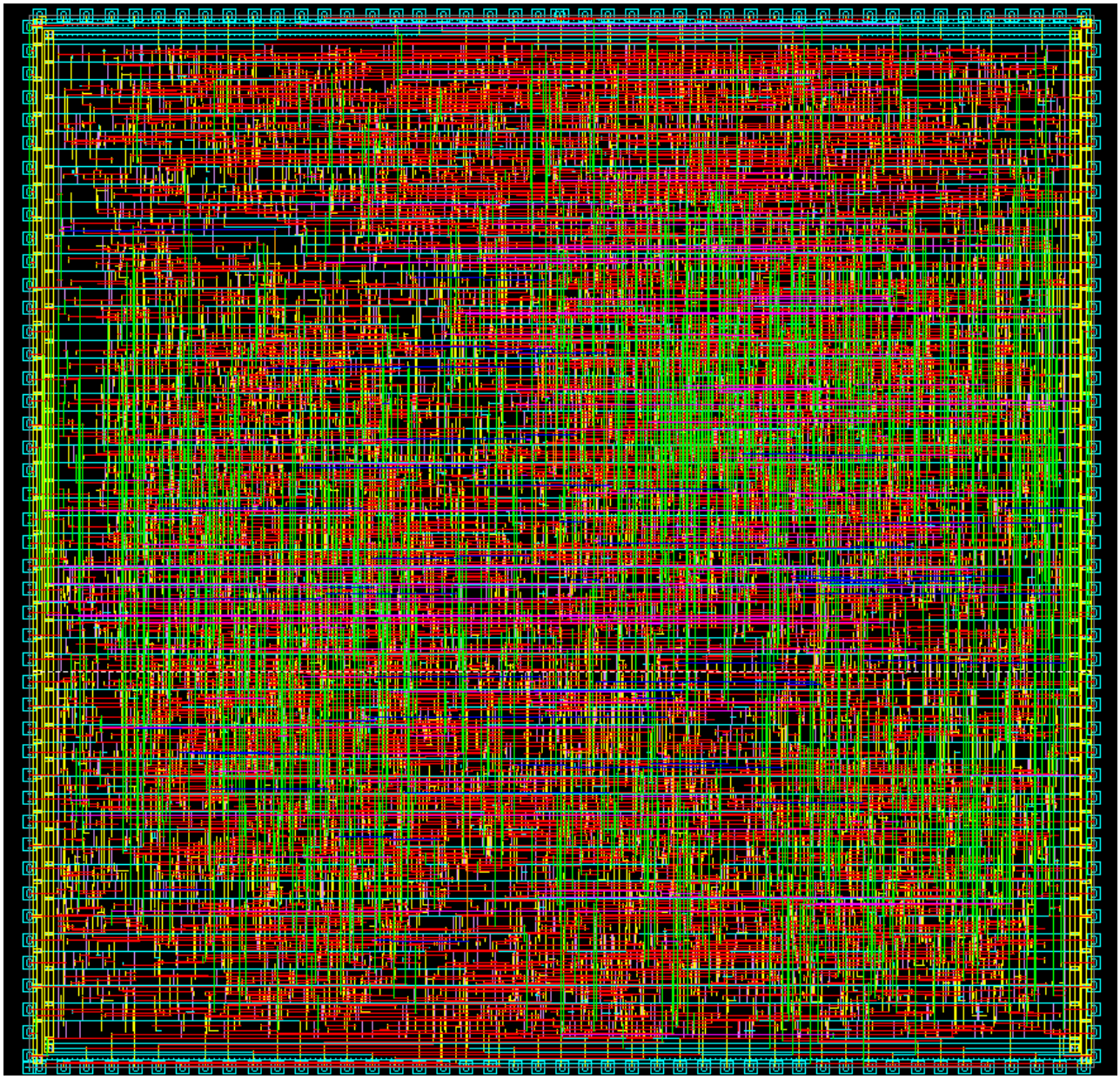

Figure 5.2 Auto-generated Layout of CPU

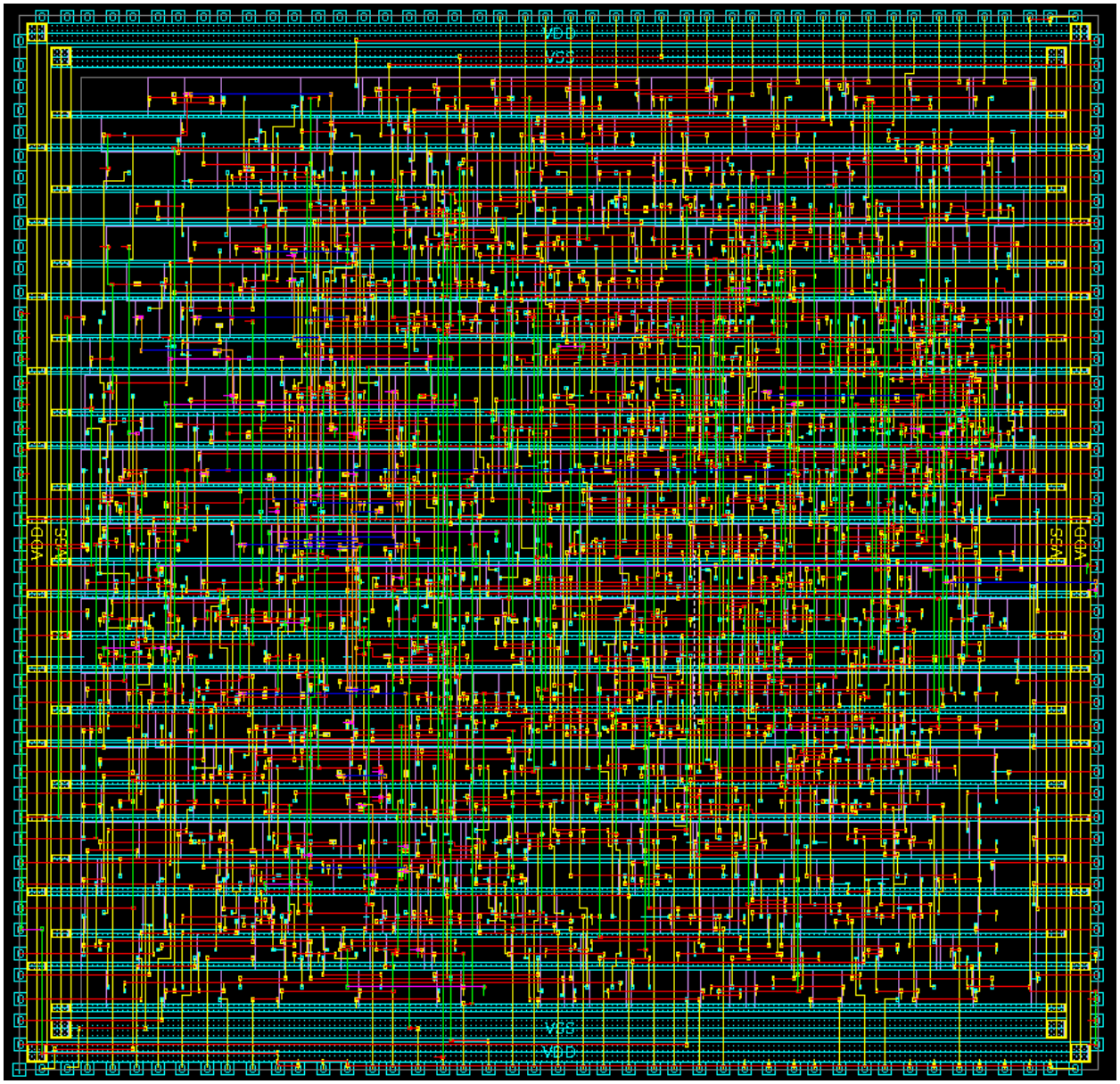

Figure 5.3 Auto-generated Layout of the ID

6. Measurements and Analysis

	Synthesized
Maximum Operation Frequency	1.44 ns
Slack	0.06 ns
Critical Path	from reset ->pc reg[0]

Table 6.1 Measurement based on gate-level synthesis report

```
Startpoint: reset (input port clocked by clk)
Endpoint: IR1/id_instruction_reg[12]
          (rising edge-triggered flip-flop clocked by clk)
Path Group: clk
Path Type: max

Point                                                    Incr       Path
-----------------------------------------------------------------------------
clock clk (rise edge)                                    0.00       0.00
clock network delay (ideal)                              0.00       0.00
input external delay                                     1.00       1.00 r
reset (in)                                               0.04       1.04 r
U335/Z (BUFFD2)                                          0.11 *     1.15 r
U355/Z (CKBD4)                                           0.09 *     1.24 r
U354/Z (CKBD4)                                           0.06 *     1.30 r
U415/Z (OR2XD1)                                          0.06 *     1.36 r
IR1/id_instruction_reg[12]/D (DFQD2)                     0.00 *     1.36 r
data arrival time                                                   1.36

clock clk (rise edge)                                    1.50       1.50
clock network delay (ideal)                              0.00       1.50
clock uncertainty                                       -0.05       1.45
IR1/id_instruction_reg[12]/CP (DFQD2)                    0.00       1.45 r
library setup time                                      -0.03       1.42
data required time                                                  1.42
-----------------------------------------------------------------------------
data required time                                                  1.42
data arrival time                                                  -1.36
-----------------------------------------------------------------------------
slack (MET)                                                         0.06
```

Figure 6.1 Synthesized Design Report

7. Directory

/home/projects/ee477/yuxiangc/cad6/result

Behavior Verilog Design:

netlist file : control.v

test file: tb.v

simulation file: vcdplus_behavior.vpd

Synthesize Verilog Design

netlist file : control_syn.v

test file: tb_syn.v

simulation file: control_syn.vcd

sdf file: control_syn.sdf
tcl file: control_syn.tcl

Post- layout Design
netlist file : control_apr.v
test file: tb_apr.v
simulation file: control_apr.vcd, control_apr.vcd.vpd
sdf file: control_apr.sdf
tcl file: netlist file : control_apr.tcl