Nhóm 1 - CE118.Q11.VMTN

THIẾT KẾ VI XỬ LÝ 16-BIT

theo kiến trúc tập lệnh đơn giản

GV hướng dẫn: Tạ Trí Đức

DANH SÁCH THÀNH VIÊN

24520344	Nguyễn Việt Dũng (Nhóm trưởng)
24520301	Tạ Vương Điền
24521177	Huỳnh Hữu Nguyện
24520839	Phan Công Đăng Khoa
24520816	Lê Đăng Khoa

Nhóm 1 - CE118.Q11.VMTN

-MỤC LỤC-

TẬP LỆNH

- **RRR**:

Ins	Opcode 15 13	Rs 12 10	Rt 9 7	Rd 6 4	Funct 3 0	Assembly-Code Format	Meaning
add	000	Rs	Rt	Rd	0000	add Rd, Rs, Rt	Reg[Rd] ← Reg[Rs] + Reg[Rt]
inc	000	Rs	000	Rd	0001	inc Rd, Rs	Reg[Rd] ← Reg[Rs] - Reg[Rt]
sub	000	Rs	Rt	Rd	0010	sub Rd, Rs, Rt	Reg[Rd] ← Reg[Rs] + 1
dec	000	Rs	000	Rd	0011	dec Rd, Rs	Reg[Rd] ← Reg[Rs] - 1
and	000	Rs	Rt	Rd	0100	and Rd, Rs, Rt	Reg[Rd] ← Reg[Rs] & Reg[Rt]
or	000	Rs	Rt	Rd	0101	or Rd, Rs, Rt	Reg[Rd] ← Reg[Rs] | Reg[Rt]
nand	000	Rs	Rt	Rd	0110	nand Rd, Rs, Rt	Reg[Rd] ← Reg[Rs] ~& Reg[Rt]
xor	000	Rs	Rt	Rd	0111	xor Rd, Rs, Rt	Reg[Rd] ← Reg[Rs] ^ Reg[Rt]
shfl	000	Rs	000	Rd	1000	shfl Rd, Rs	Reg[Rd] ← Reg[Rs] << 1
shfr	000	Rs	000	Rd	1001	shfr Rd, Rs	Reg[Rd] ← Reg[Rs] >> 1
sll	000	Rs	000	Rd	1010	shll Rd, Rs	Reg[Rd] ← Reg[Rs] << 2
srr	000	Rs	000	Rd	1011	shrr Rd, Rs	Reg[Rd] ← Reg[Rs] >> 2
sra	000	Rs	000	Rd	1100	shra Rd, Rs	Reg[Rd] ← Reg[Rs] >> 1 (giữ bit dấu)
mul	000	Rs	Rt	Rd	1101	mul Rd, Rs, Rt	Reg[Rd] ← Reg[Rs] × Reg[Rt]
slt	000	Rs	Rt	Rd	1110	slt Rd, Rs, Rt	Reg[Rd] ← (Reg[Rs] < Reg[Rt]) ? 1 : 0
sgt	000	Rs	Rt	Rd	1111	sgt Rd, Rs, Rt	Reg[Rd] ← (Reg[Rs] > Reg[Rt]) ? 1 : 0

- **RI:**

Ins	Opcode 15 13	Rs 12 10	Imm 9 0	Assembly-Code Format	Meaning
lw	100	Rs	Imm	lw Rs, Imm	Reg[Rs] ← Mem[Imm]
sw	101	Rs	Imm	sw Rs, Imm	Mem[Imm] ← Reg[Rs]
jnz	110	Rs	Imm	jnz Rs, Imm	PC ← (Reg[Rs] != 0) ? Imm : PC + 1
li	111	Rs	Imm	li Rs, Imm	Reg[Rs] ← Imm

- **Extend (Mã giả):**

Ins	Opcode 15 13	Rs 12 10	Rt 9 7	Rd 6 4	Funct 3 0	Assembly-Code Format	Meaning
not	000	Rs	Rs	Rd	1000	not Rd, Rs	Reg[Rd] ← Reg[Rs] ~& Reg[Rs]
mov	000	Rs	Rs	Rd	0110	mov Rd, Rs	Reg[Rd] ← Reg[Rs] & Reg[Rs]
nop	001	Rs	Rs	0000000		nop	Reg[Rs] ← Reg[Rs] + 0
jmp	011	Rs	Rs	Imm		jmp Imm	PC ← (Reg[Rs] == Reg[Rs]) ? Imm : PC + 1

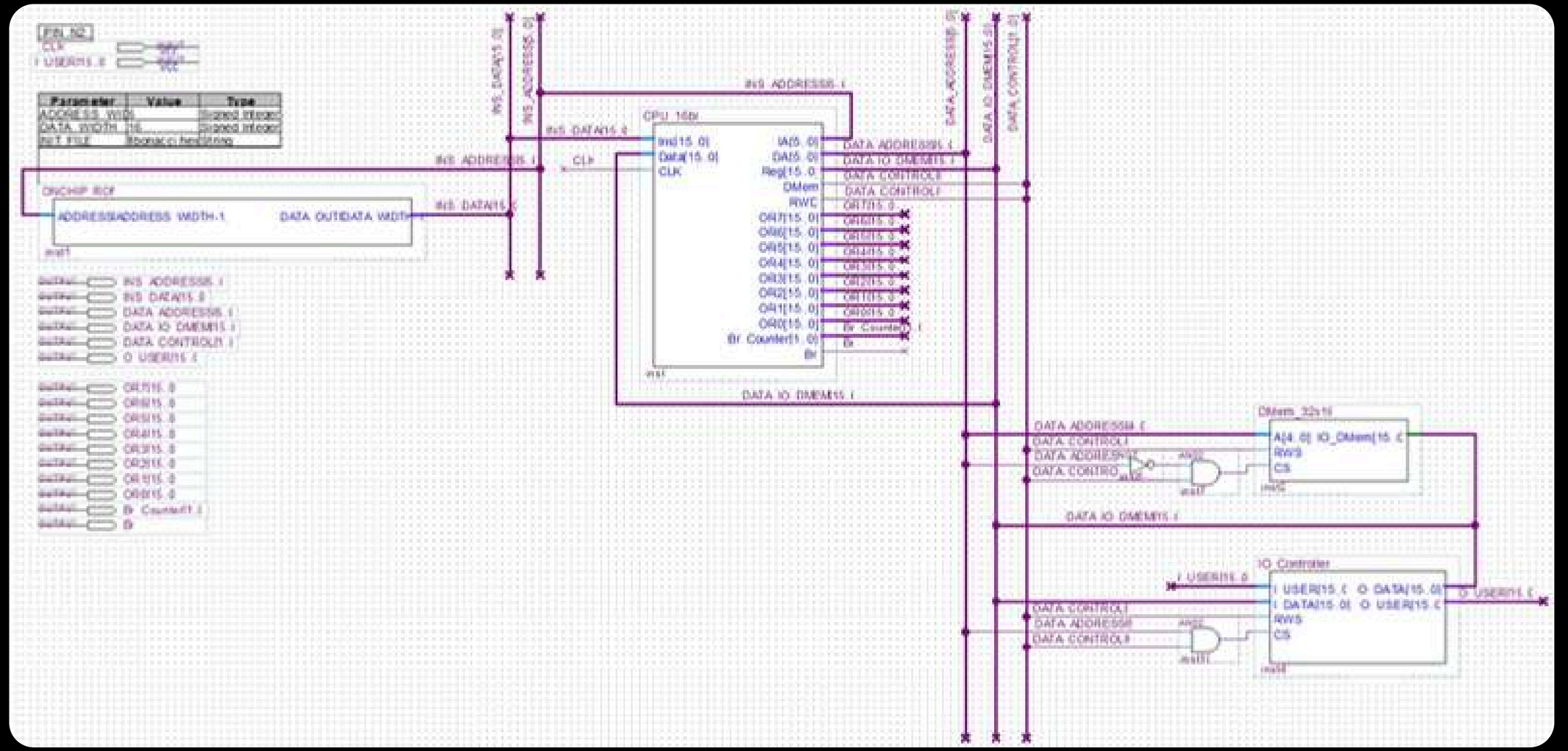
- **RRI:**

Ins	Opcode 15 13	Rs 12 10	Rt 9 7	Imm 6 0	Assembly-Code Format	Meaning
addi	001	Rs	Rt	Imm	addi Rs, Rt, Imm	Reg[Rt] ← Reg[Rs] + SignExtImm
subi	010	Rs	Rt	Imm	subi Rs, Rt, Imm	Reg[Rt] ← Reg[Rs] - SignExtImm
beq	011	Rs	Rt	Imm	beq Rs, Rt, Imm	PC ← (Reg[Rs] == Reg[Rt]) ? Imm : PC + 1

01

TỔNG QUAN THIẾT KẾ

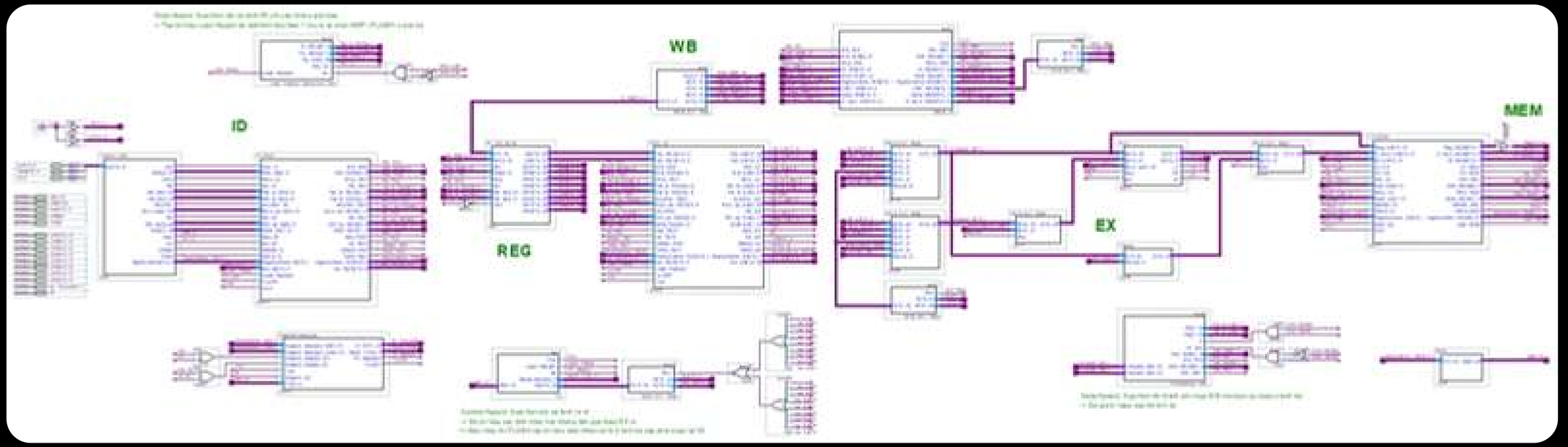
DATAPATH



01

TỔNG QUAN THIẾT KẾ

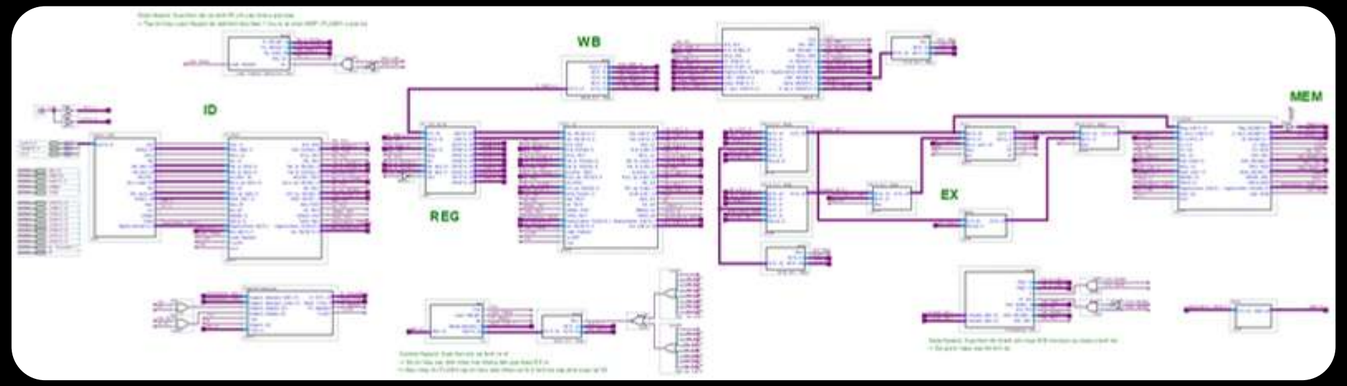
DATAPATH – CPU 16-BIT

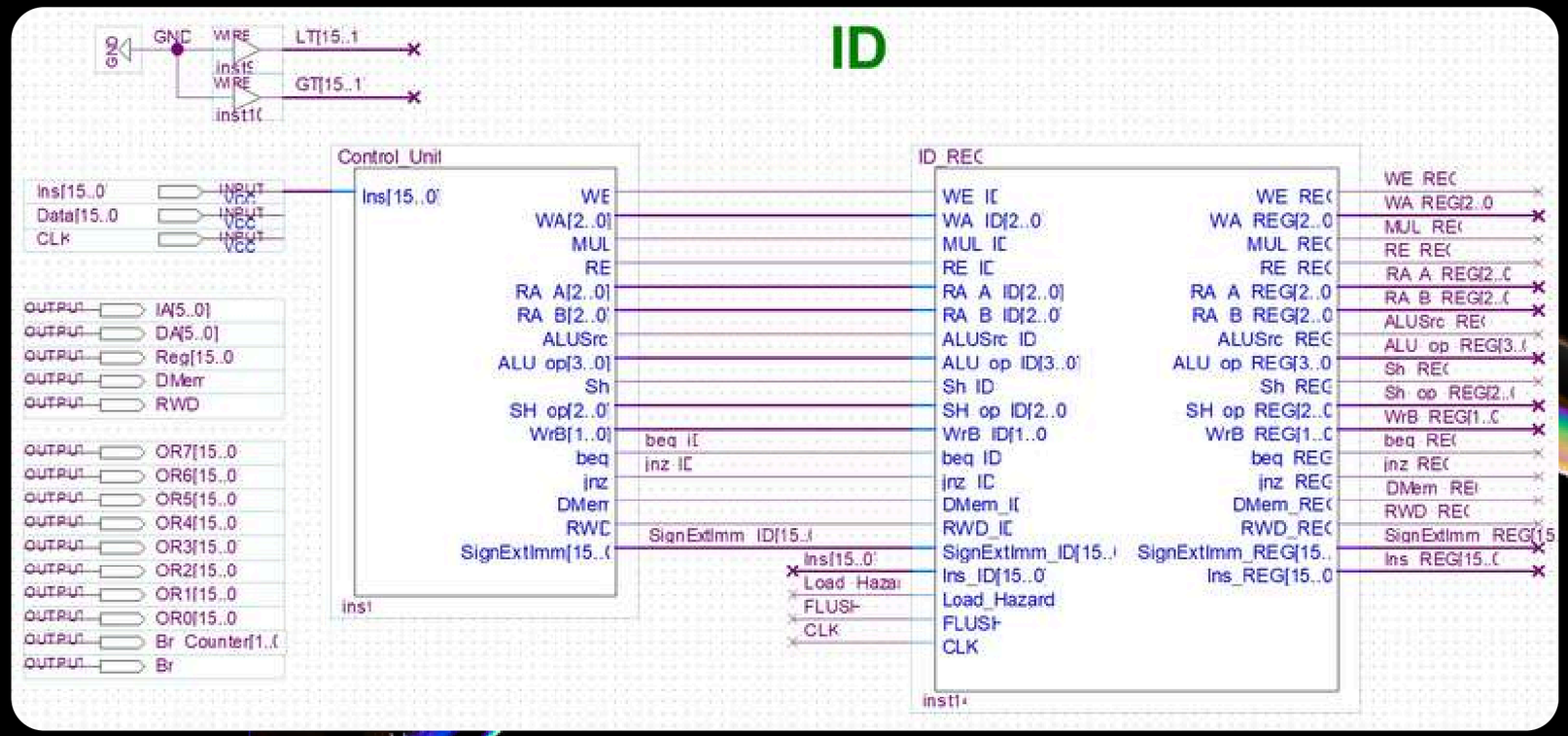


02

PIPELINE & MẠCH NHÂN

PIPELINE – IF & ID

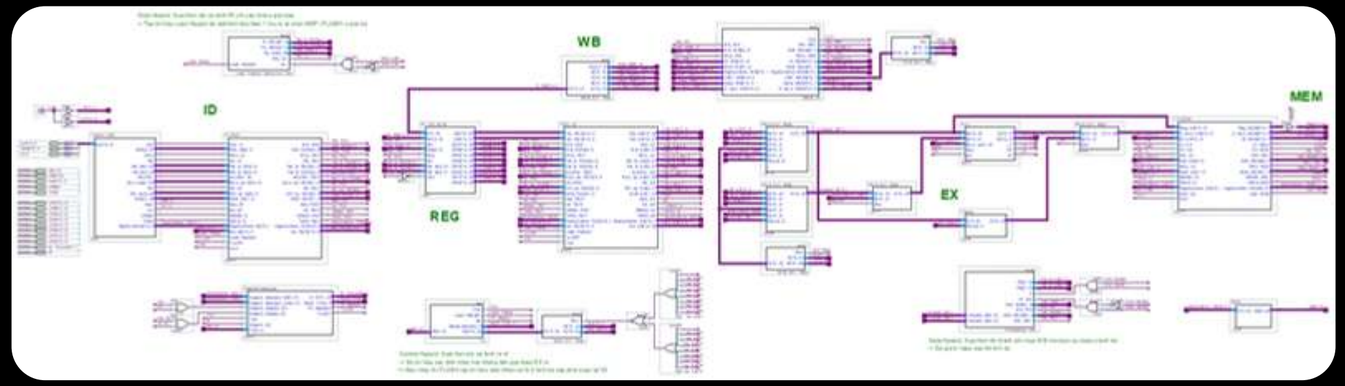


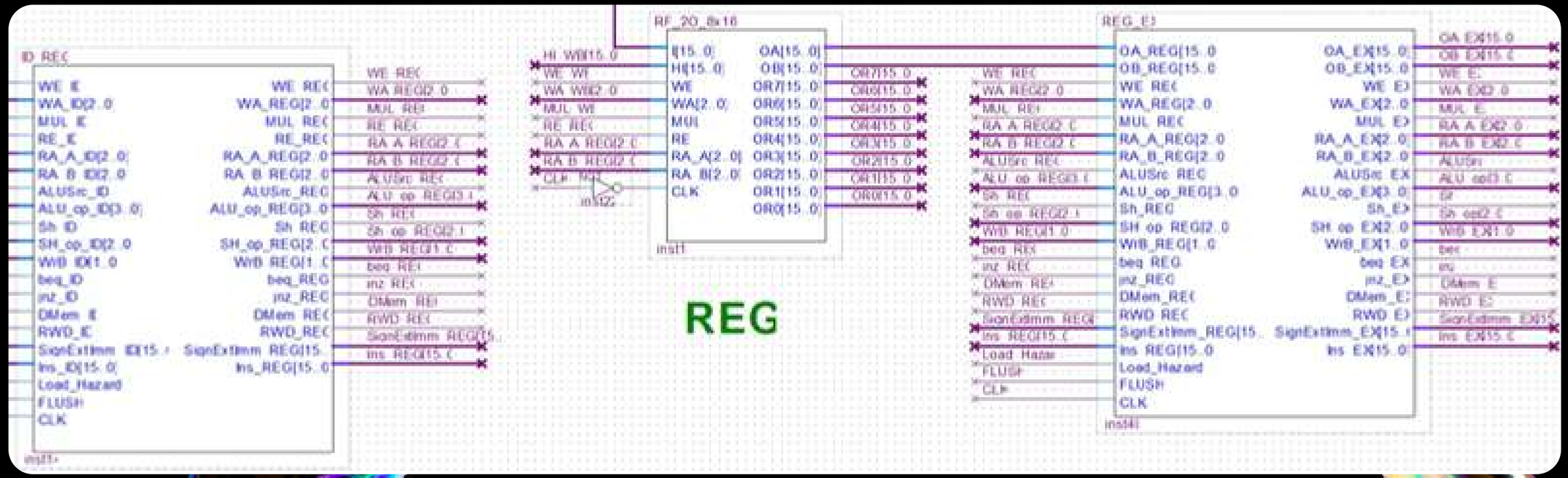


02

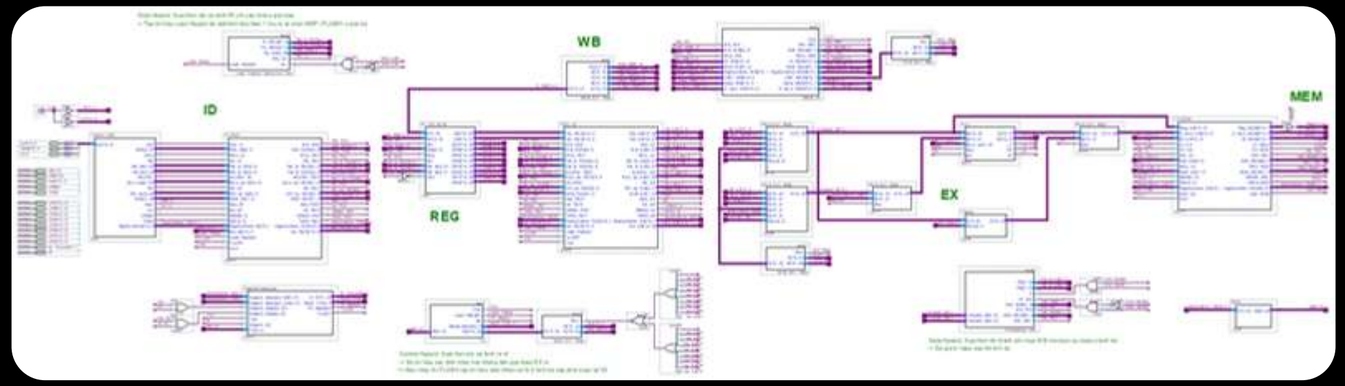
PIPELINE & MẠCH NHÂN

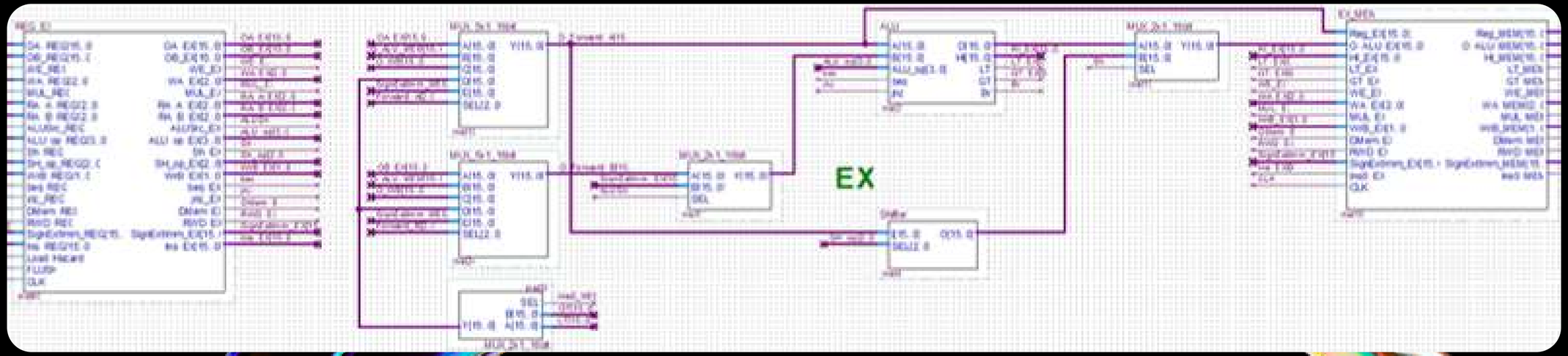
PIPELINE – REG



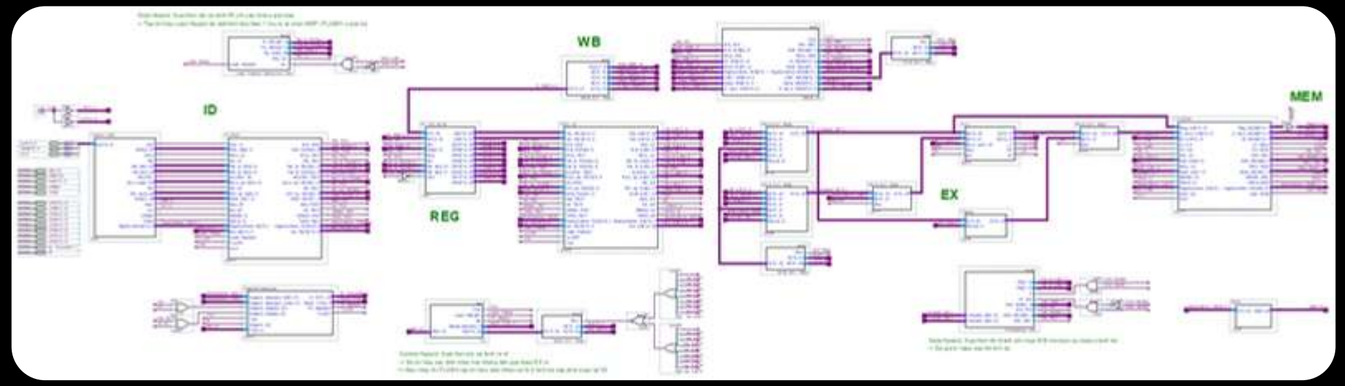


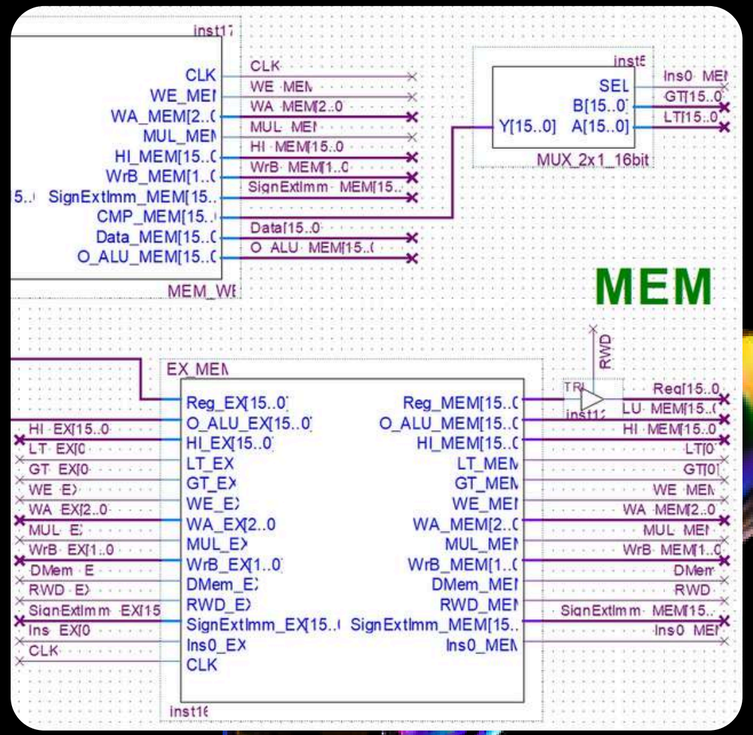
PIPELINE – EX

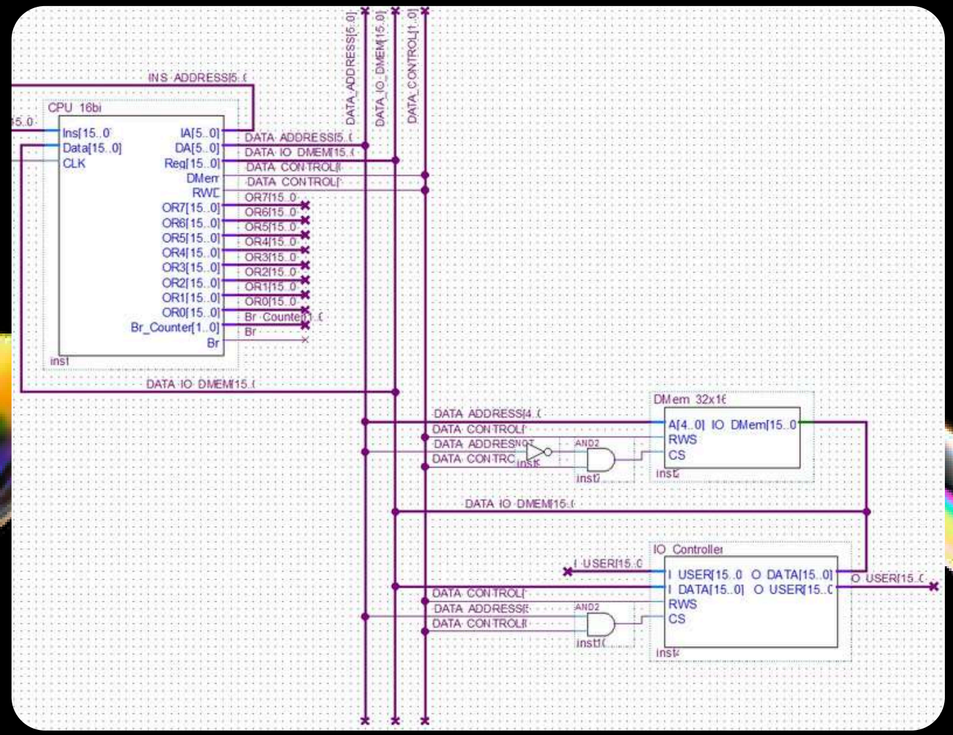




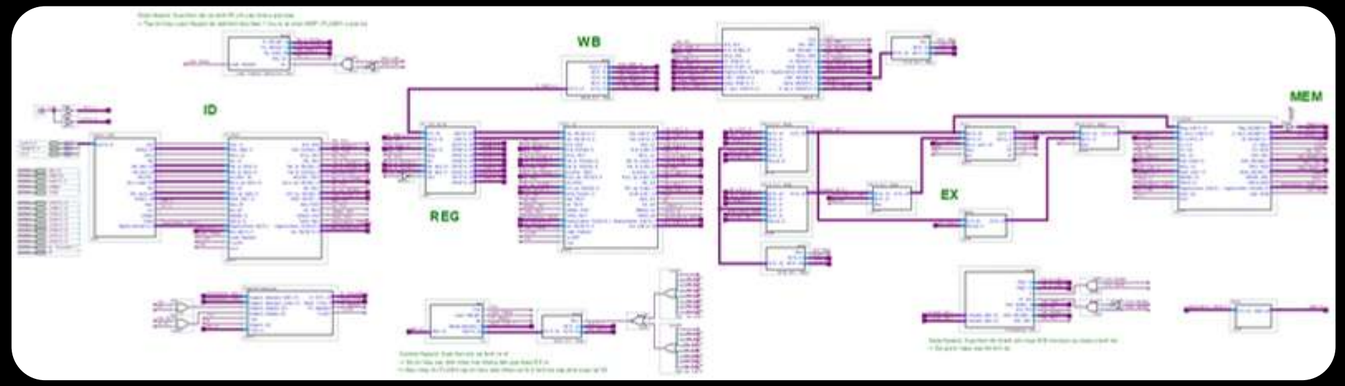
PIPELINE – MEM

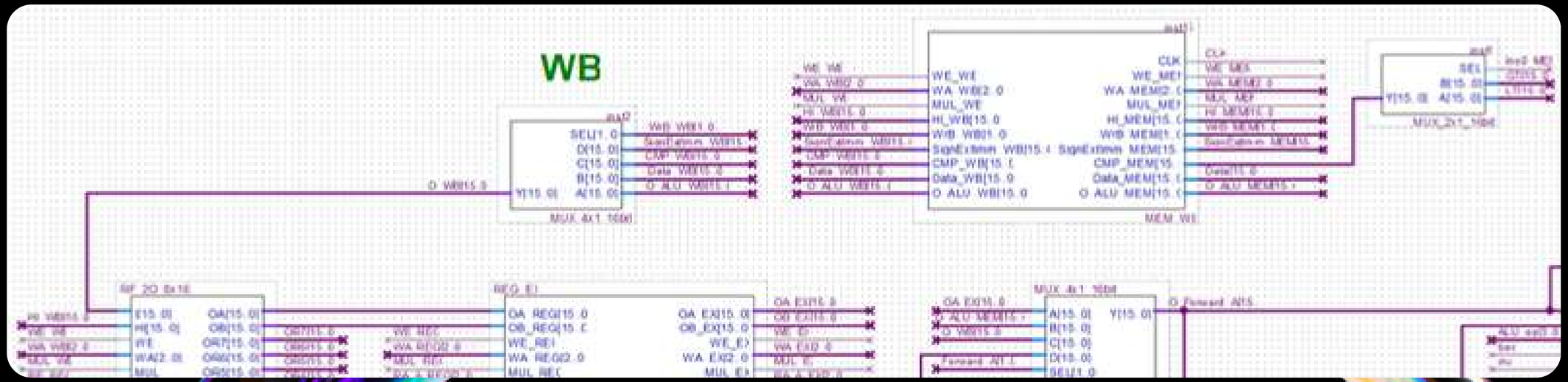




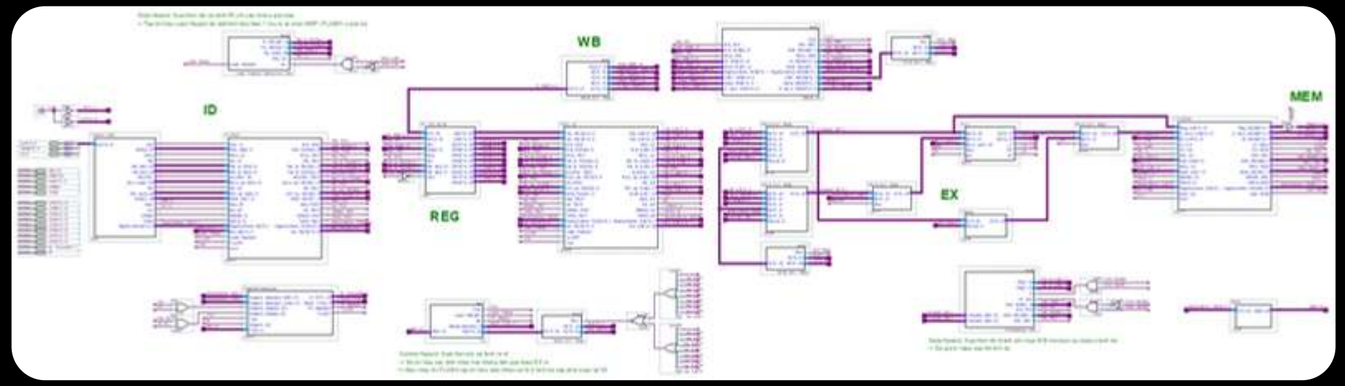


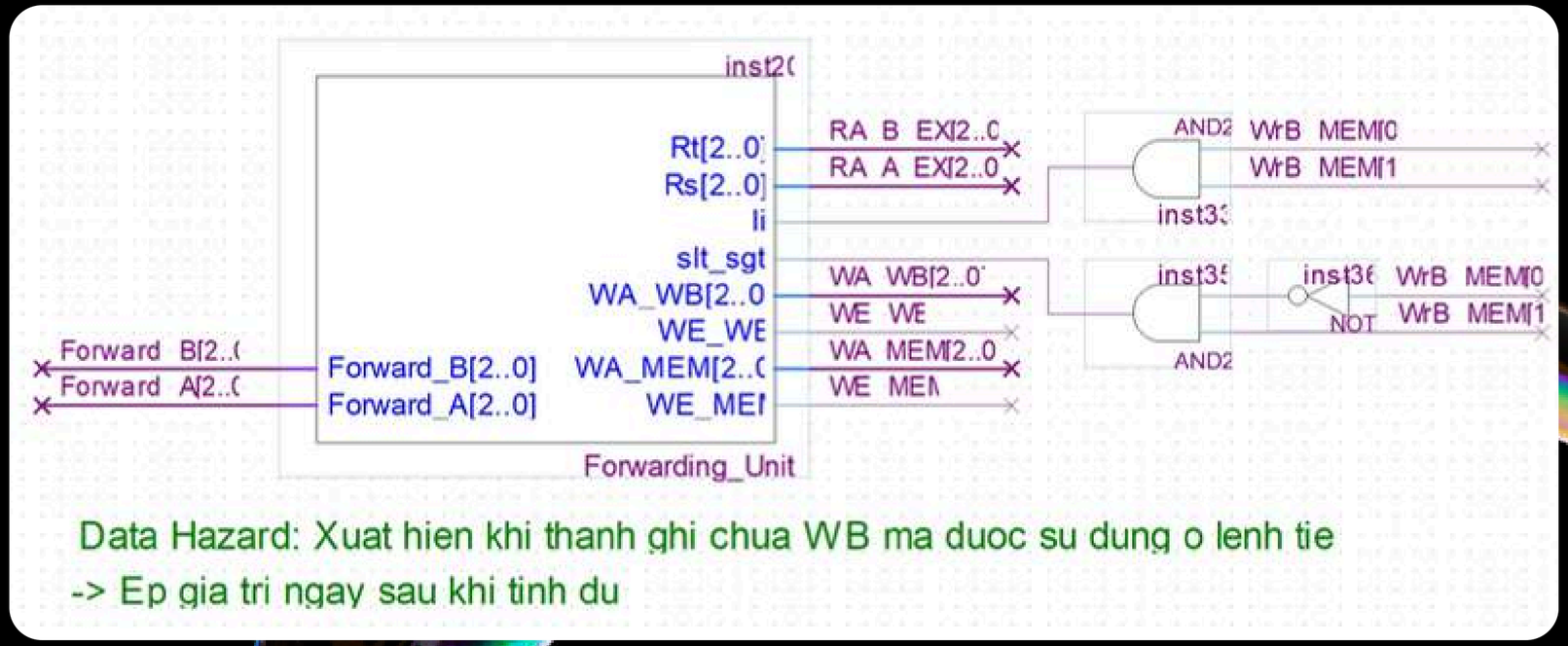
PIPELINE – WB


WB
ID
REG
EX
MEM


WB
MUX_4x1_16bit
MEM_WB
MUX_2x1_16bit
RF_2O_8x16
REG_E

PIPELINE – FORWARDING UNIT



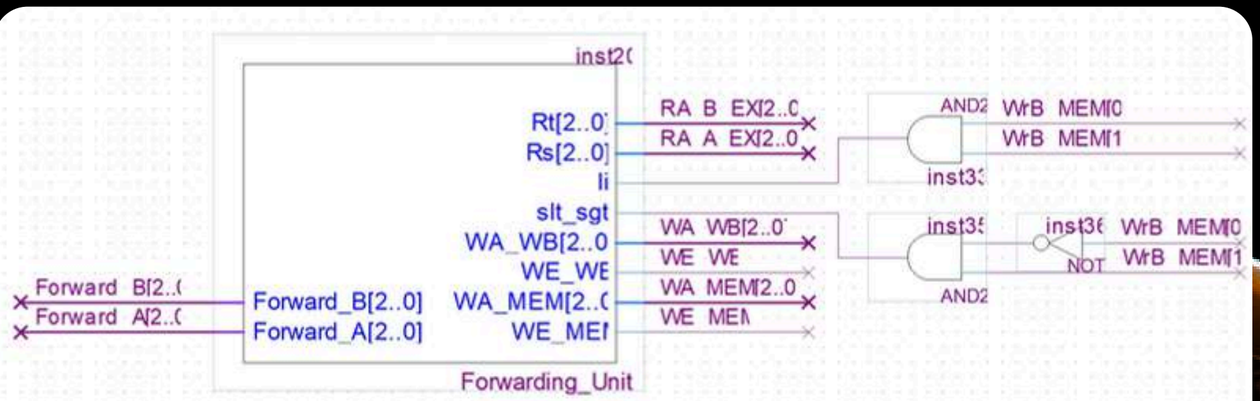


02

PIPELINE & MẠCH NHÂN

PIPELINE – FORWARDING UNIT

```
sub   $2,   $1, $3
and   $12,  $5, $2    # 1b
or    $13,  $2, $6    # 2a
add   $14,  $2, $2    # No hazard
sw    $15,  100($2)   # No hazard
```



Data Hazard: Xuat hien khi thanh ghi chua WB ma duoc su dung o lenh tie

-> Ep gia tri ngay sau khi tinh du

```
lw   $2, 20($1)
nop
or   $8, $2, $6
add  $9, $4, $2
slt  $1, $6, $7
```

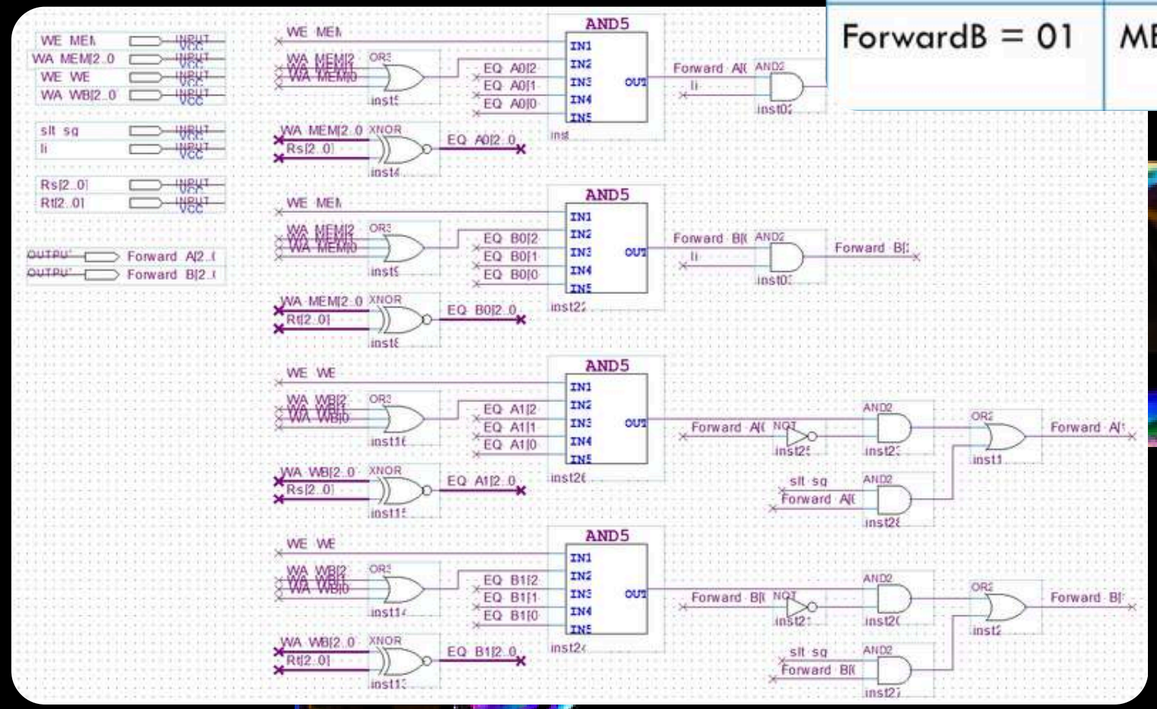
```
add $1, $1, $2
add $1, $1, $3
add $1, $1, $4
```

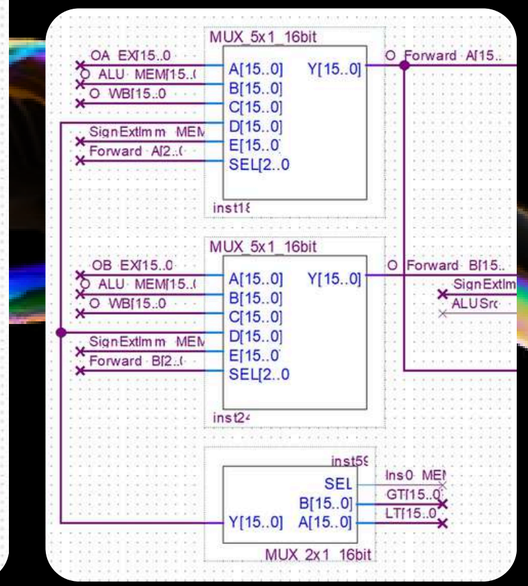
02

PIPELINE & MẠCH NHÂN

PIPELINE – FORWARDING UNIT

Mux Control	Source	Explanation
ForwardA = 00	ID/EX	First ALU input comes from register file.
ForwardA = 10	EX/MEM	First ALU input is forwarded from the prior ALU result.
ForwardA = 01	MEM/WB	First ALU input is forwarded from the data memory or a prior ALU result.
ForwardB = 00	ID/EX	Second ALU input comes from register file.
ForwardB = 10	EX/MEM	Second ALU input is forwarded from the prior ALU result.
ForwardB = 01	MEM/WB	Second ALU input is forwarded from the data memory or a prior ALU result.

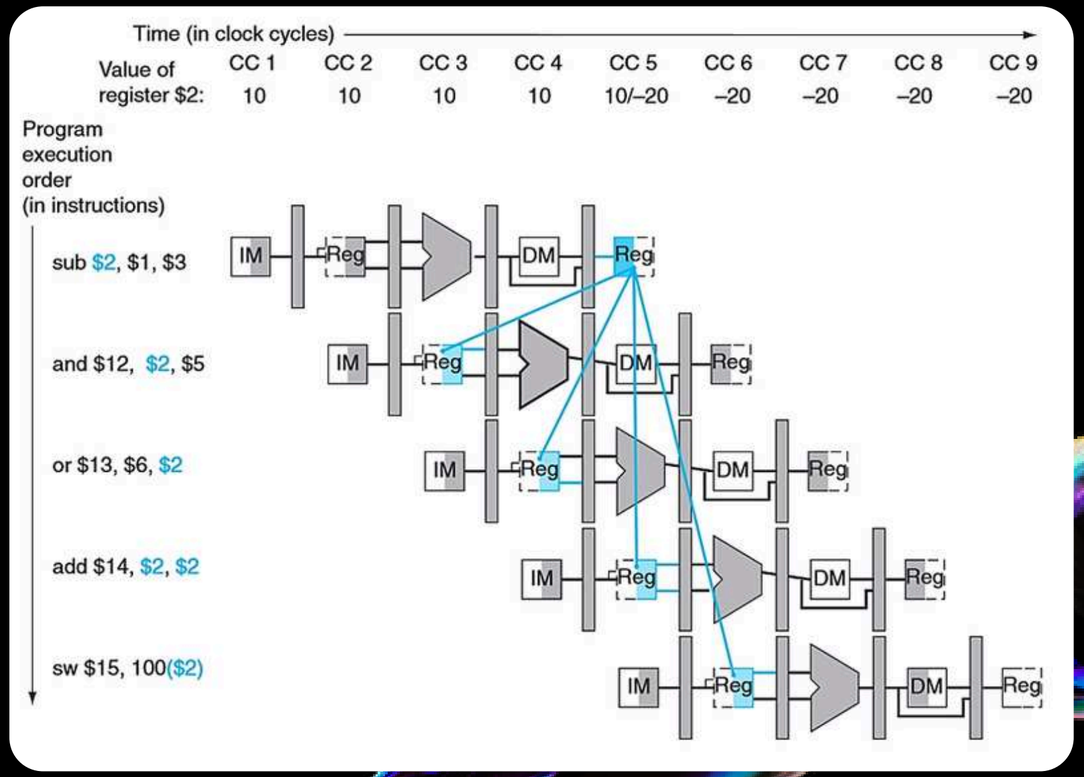


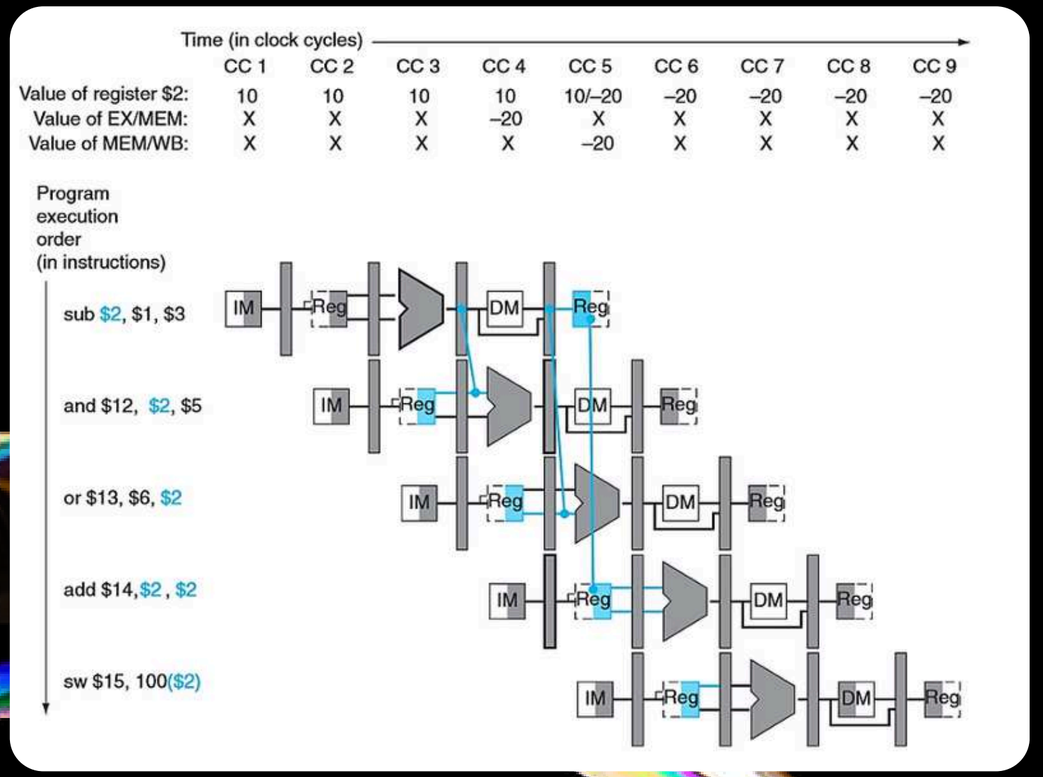


02

PIPELINE & MẠCH NHÂN

PIPELINE – FORWARDING UNIT

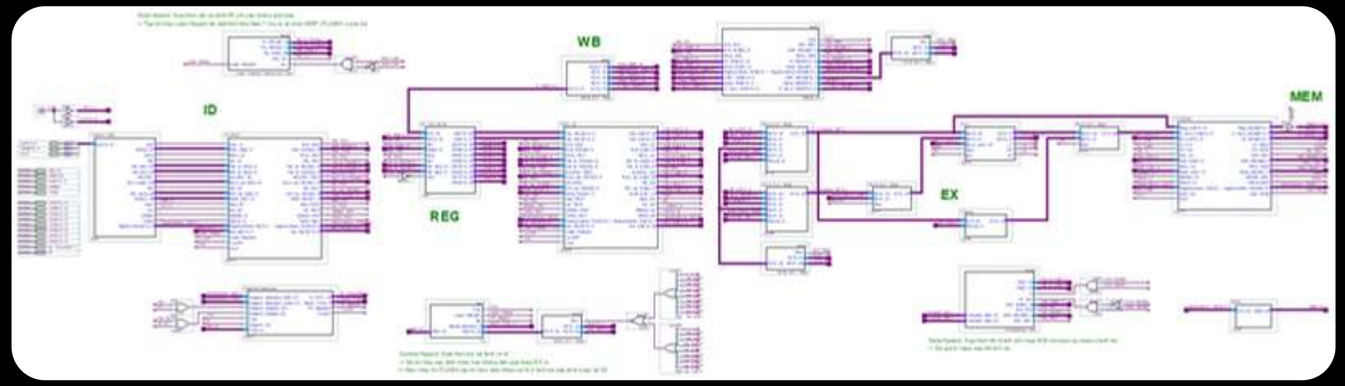




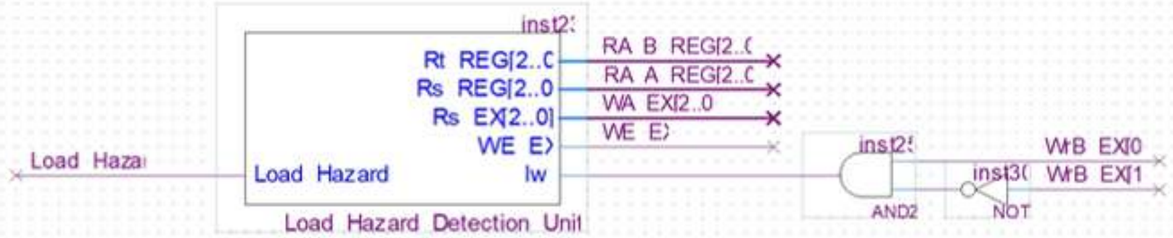
02

PIPELINE & MẠCH NHÂN

PIPELINE – HAZARD DETECTION UNIT



Data Hazard: Xuat hien doi voi lenh RI chi cap nhat o giai doa
-> Tao tin hieu Load Hazard de stall lenh tiep theo 1 chu ki va chen NOP (FLUSH) o giai do

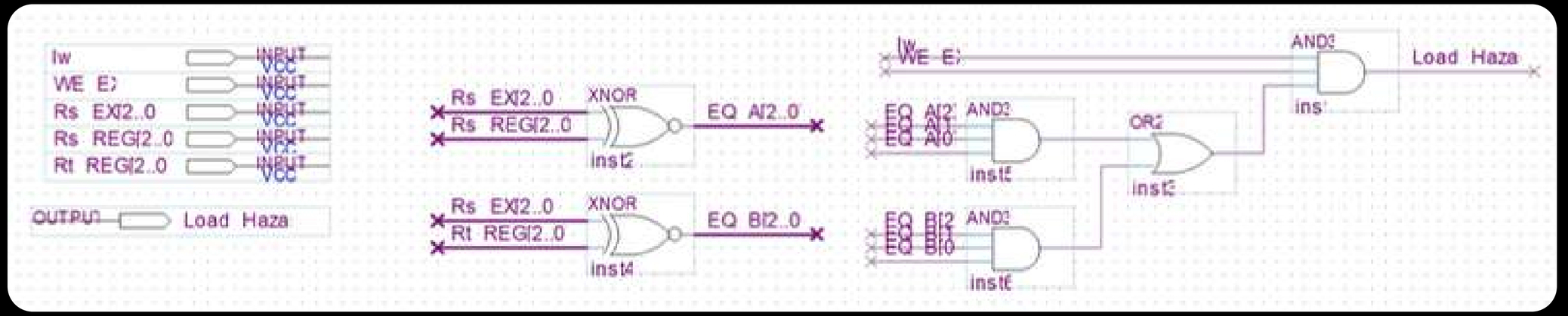


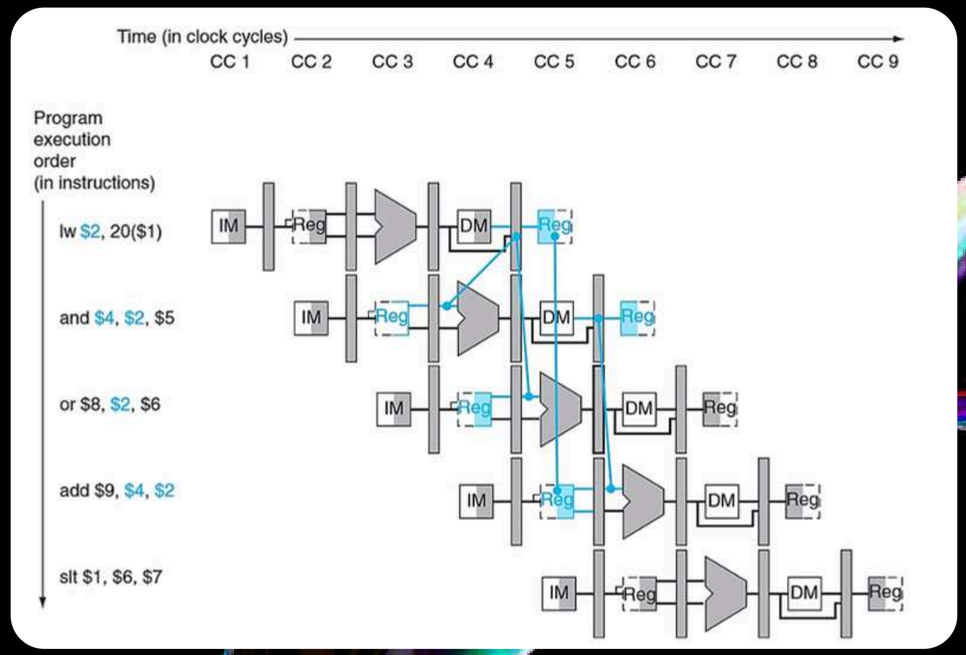

```
lw   $2, 20($1)
and  $4, $2, $5
or   $8, $2, $6
add  $9, $4, $2
slt  $1, $6, $7
```

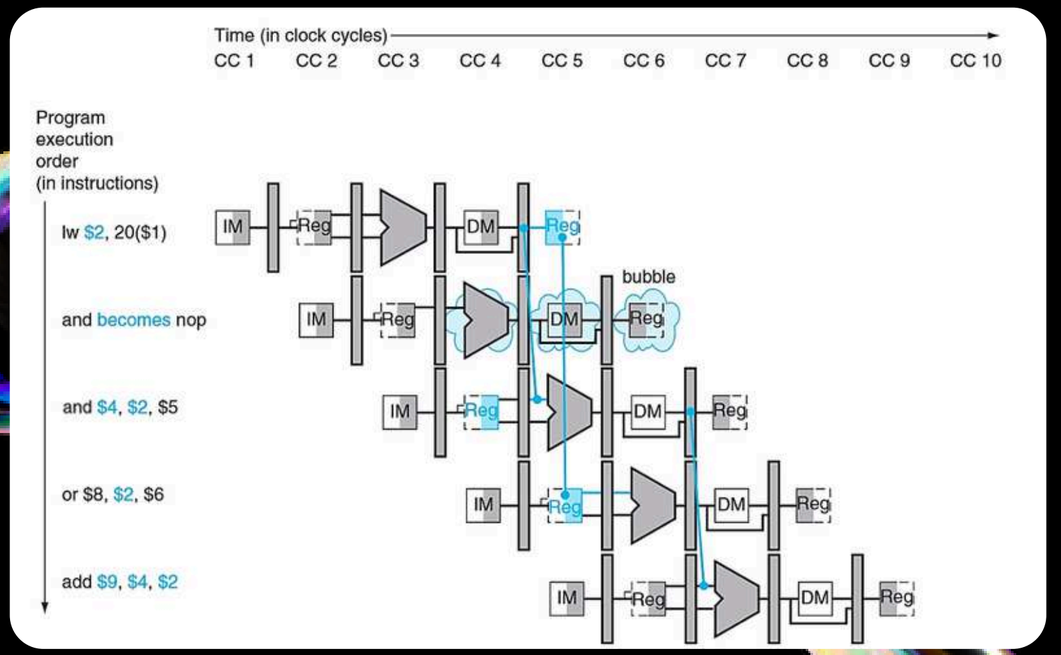
02

PIPELINE & MẠCH NHÂN

PIPELINE – HAZARD DETECTION UNIT



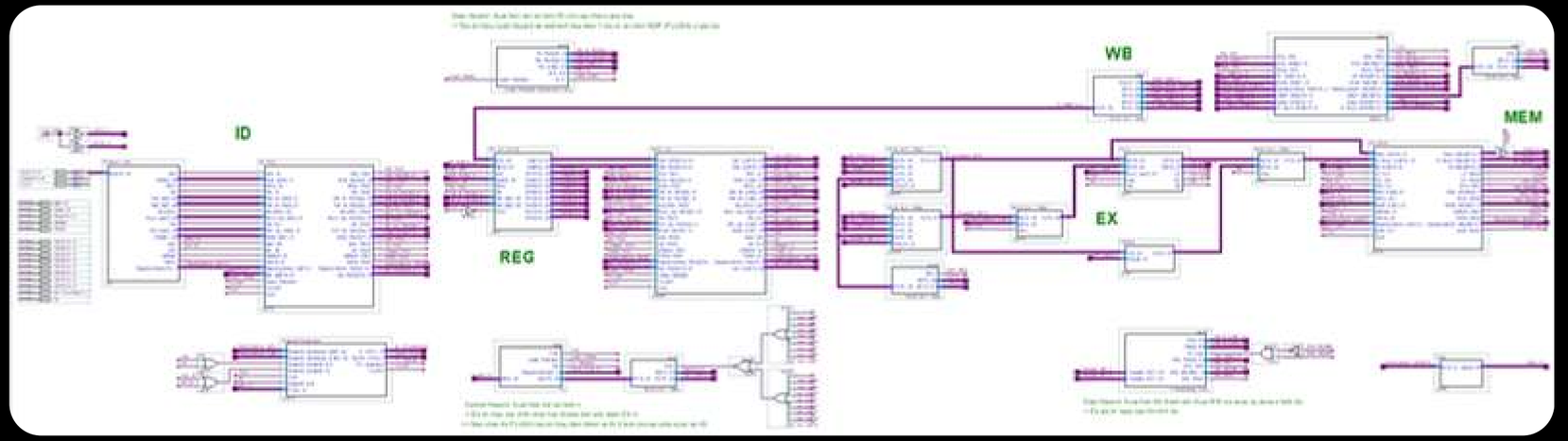


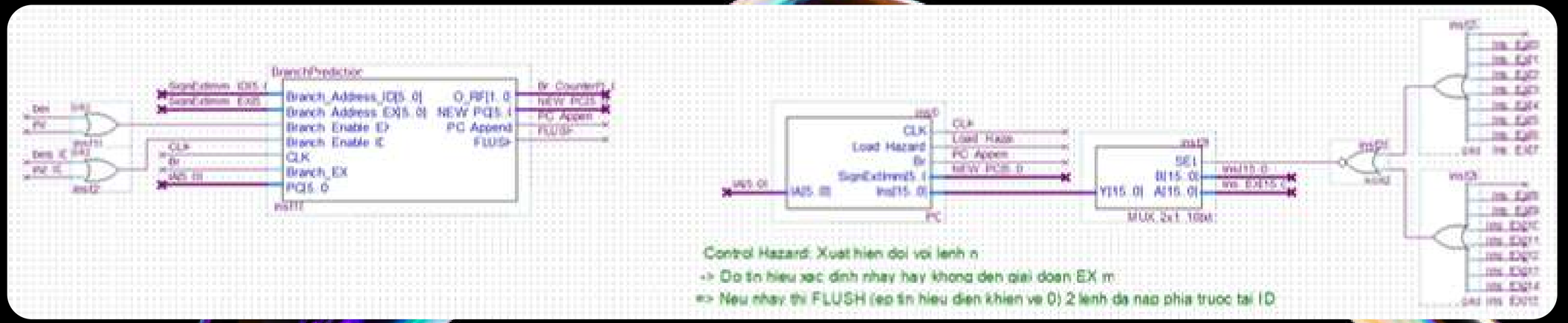


02

PIPELINE & MẠCH NHÂN

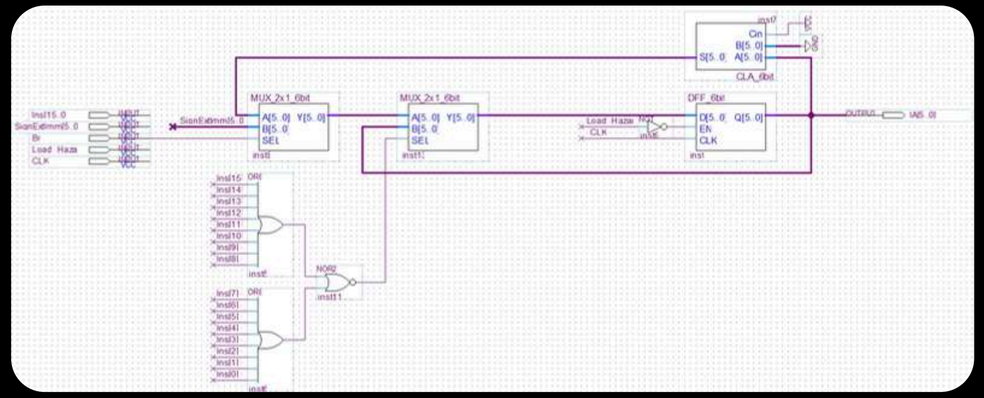
PIPELINE – PC + BRANCH PREDICTION

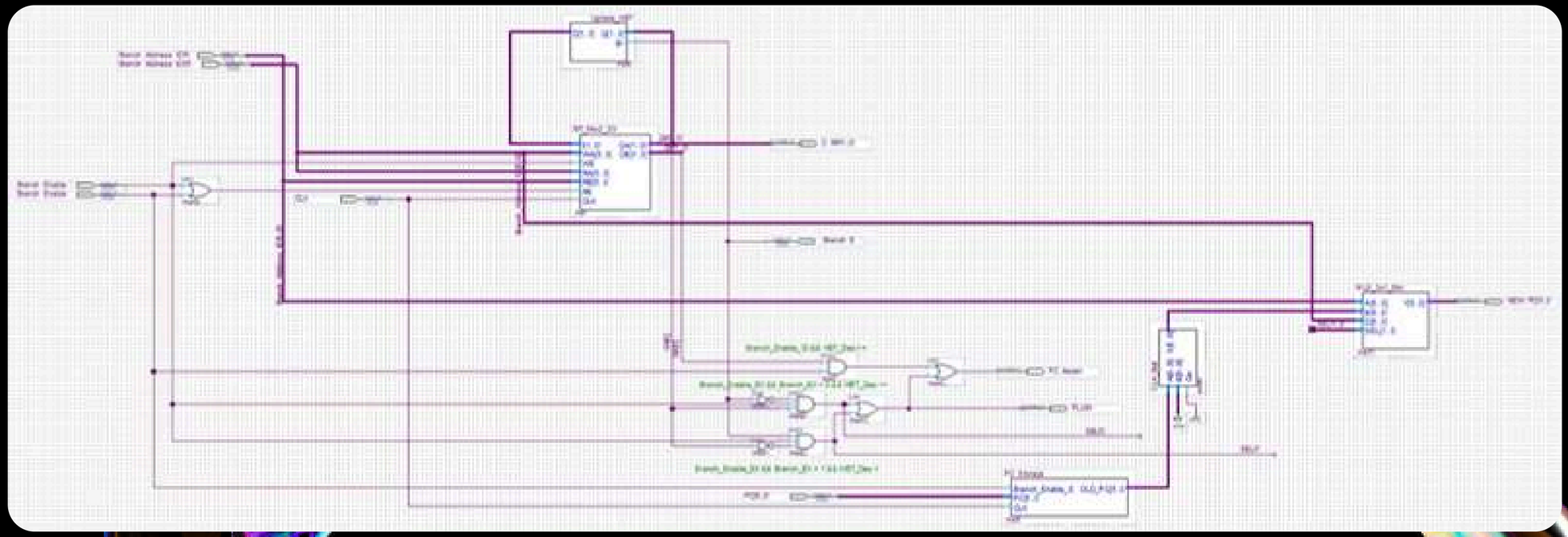




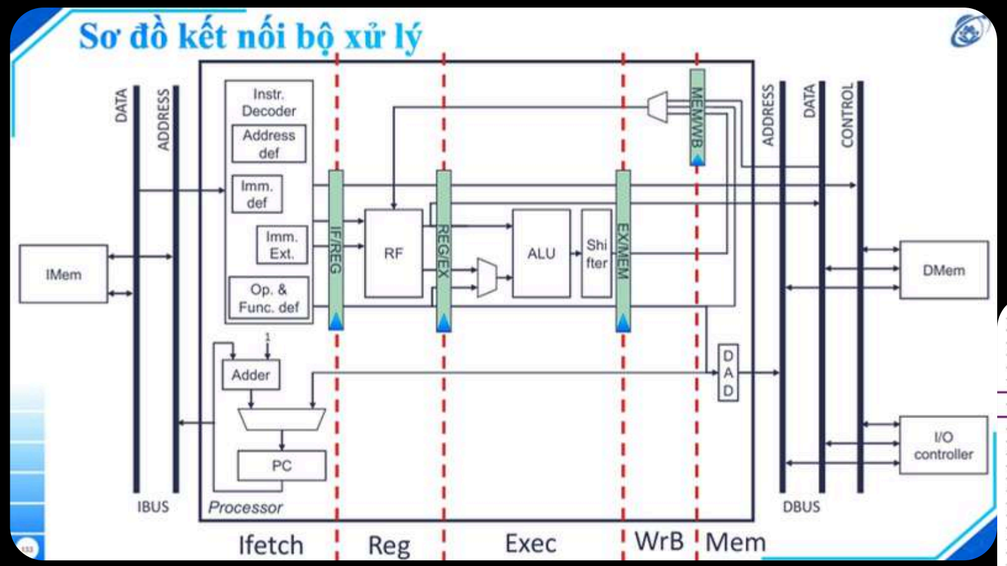
PIPELINE – PC + BRANCH PREDICTION

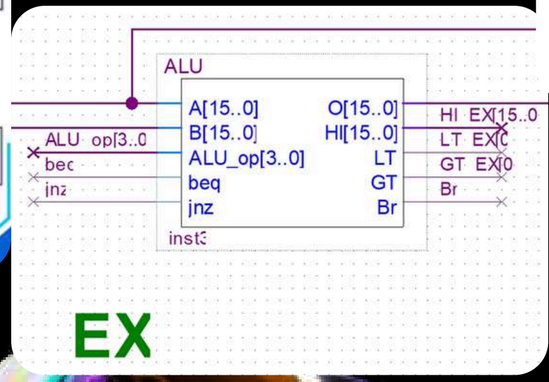
```
int i = 0;                          add    $t0, $0, $0
do{                           L1:   /* loop body*/
    /* loop body */                 addi   $t0, $t0, 1
    i = i + 1                       slti   $t1, $t0, 10
}while(i < 10);                     bne    $t1, $0, L1
```

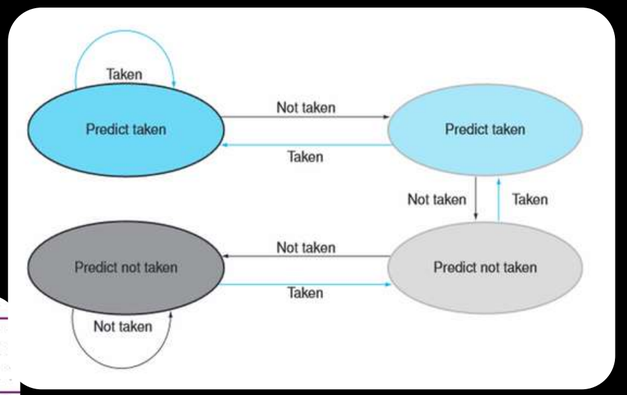


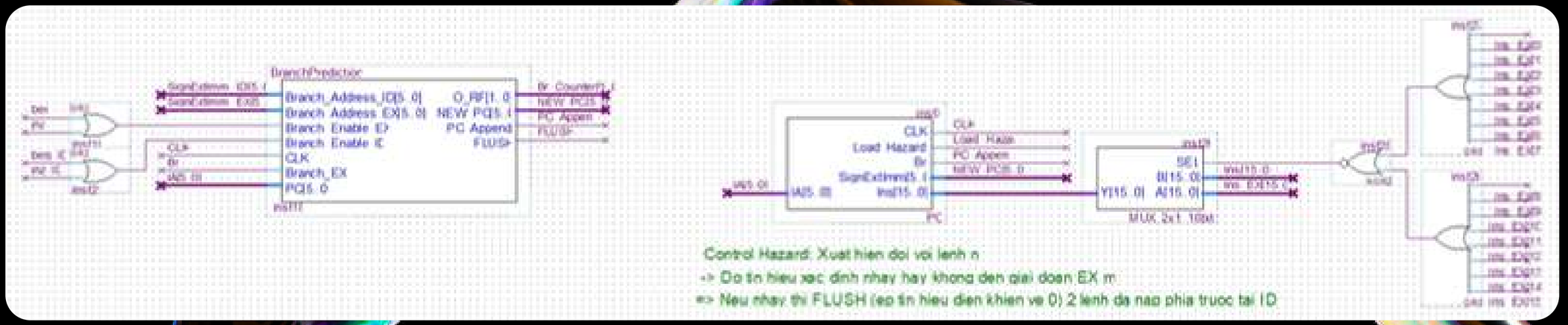


PIPELINE – PC + BRANCH PREDICTION





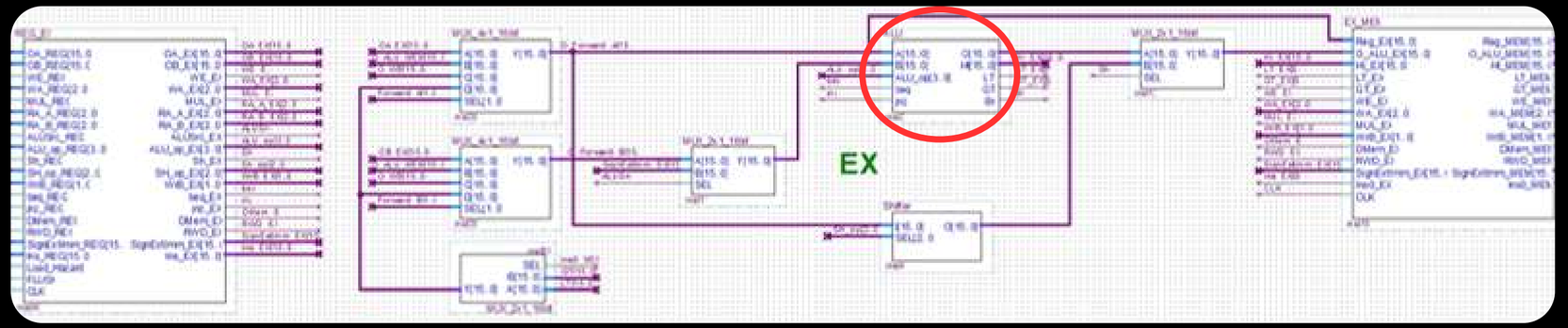


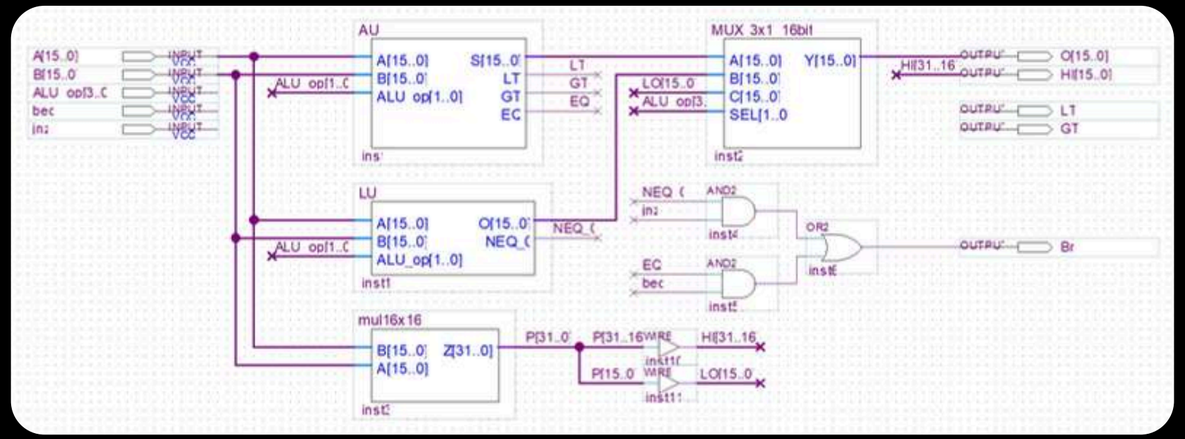


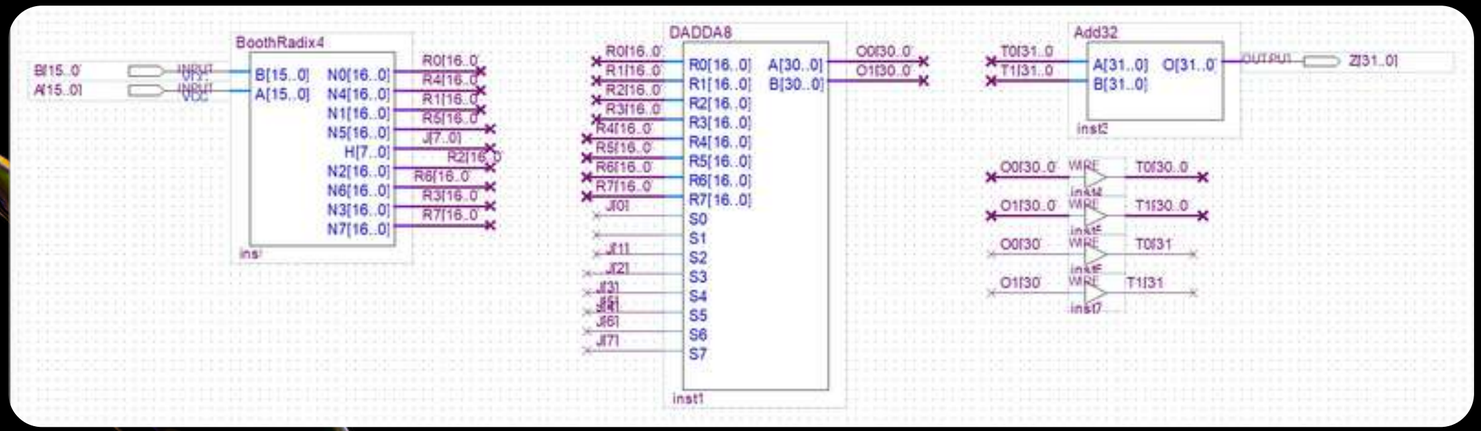
02

PIPELINE & MẠCH NHÂN

MẠCH NHÂN



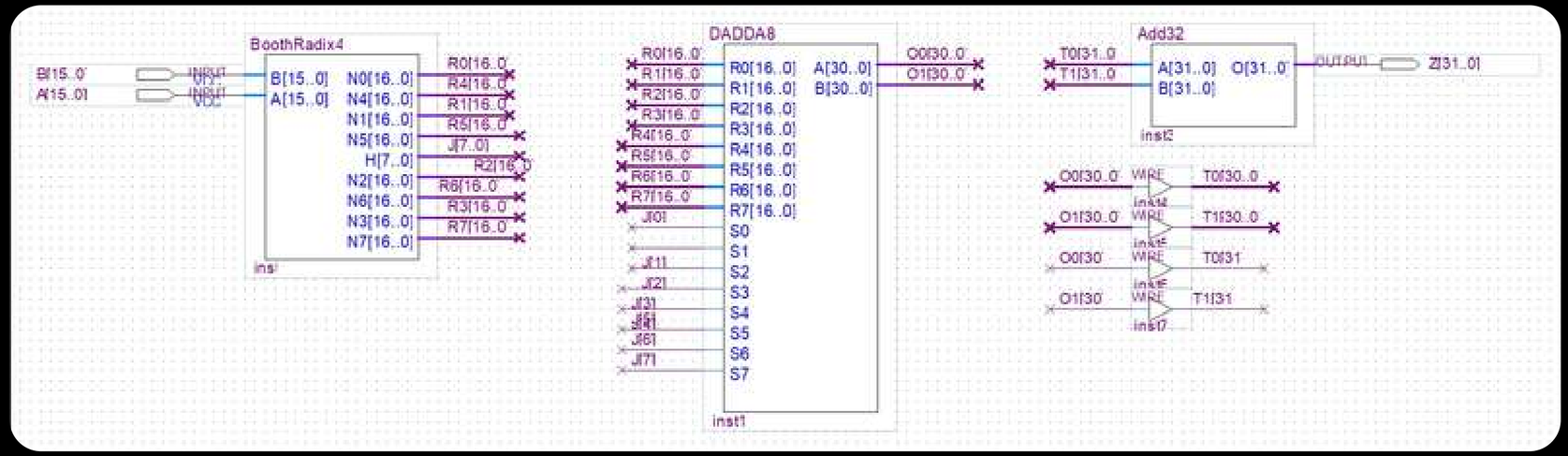




02

PIPELINE & MẠCH NHÂN

MẠCH NHÂN



→ Thuật toán Booth?

<u>VD:</u> 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth: 16 - 1 = 15 → 1 phép toán

???

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-2): 16 - 1 = 15 → 1 phép toán

???

0000 1111

i = 7 6 5 4 3 2 1 0

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-2): 16 - 1 = 15 → 1 phép toán

???

	0	0	0	0	1	1	1	1	0
i =	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*	*-1*

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-2): 16 - 1 = 15 → 1 phép toán

???

0000 1111 1 0

i = 7 6 5 4 3 2 1 0 -1

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-2): 16 - 1 = 15 → 1 phép toán

???

	0	0	0	0	1	1	1	1	0	
i =	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*		*-1*

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-2): 16 - 1 = 15 → 1 phép toán

???

	0	0	0	0	1	1	1	1	0	
i =	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*	*-1*	

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-2): 16 - 1 = 15 → 1 phép toán

???

	0	0	0	0	1	1	1	1	0	
i =	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*	*-1*	

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-2): 16 - 1 = 15 → 1 phép toán

???

0000 1111 0

i = 7 6 5 4 3 2 1 0 -1

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-2): 16 - 1 = 15 → 1 phép toán

???

0000 1111 0

i = 7 6 5 4 3 2 1 0 -1

→ Xét từ phải qua trái tại bit i:

+) 0 → 1: -2^i

+) 1 → 0: +2^i

⇒ Công thức Booth (đúng với mọi số có dấu):

$$\sum_{i=0}^{N-1}(y_{i-1}-y_i)2^i$$

MẠCH NHÂN

→ Cách nhân thông thường:

0000 0000 0000 1111 (15)
x
0111 1111 1111 1111 (32767)

0000 ... 0000 0000 0000 1111 (32 bits)

MẠCH NHÂN

→ Cách nhân thông thường:

0000 0000 0000 1111 (15)

x 0111 1111 1111 1111 (32767)

0000 ... 0000 0000 0000 1111

...

0000 ... 0000 0000 0000 0000

} x16

⇒ Phức tạp, không tối ưu.

MẠCH NHÂN

→ Áp dụng Booth (Booth Radix-2):

$$
\begin{array}{rl}
 & 0000\ 0000\ 0000\ 1111 \quad (15) \rightarrow M \\
\times & 0111\ 1111\ 1111\ 1110\ (32767) \\
\hline
 & -M << 0
\end{array}
$$

MẠCH NHÂN

→ Áp dụng Booth (Booth Radix-2):

0000 0000 0000 1111 (15) → M

x

0111 1111 1111 1111 0 (32767)

-M << 0

0000 ... 0000 0000 0000 0000

...

0000 ... 0000 0000 0000 0000

M << 15

} x16

⇒ Giảm được độ phức tạp khi cộng nhưng chưa hoàn toàn tối ưu.

Độ phức tạp không giảm nếu chuỗi bit thay đổi liên tục (TH xấu nhất).

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-4): 16 - 1 = 15 → 1 phép toán

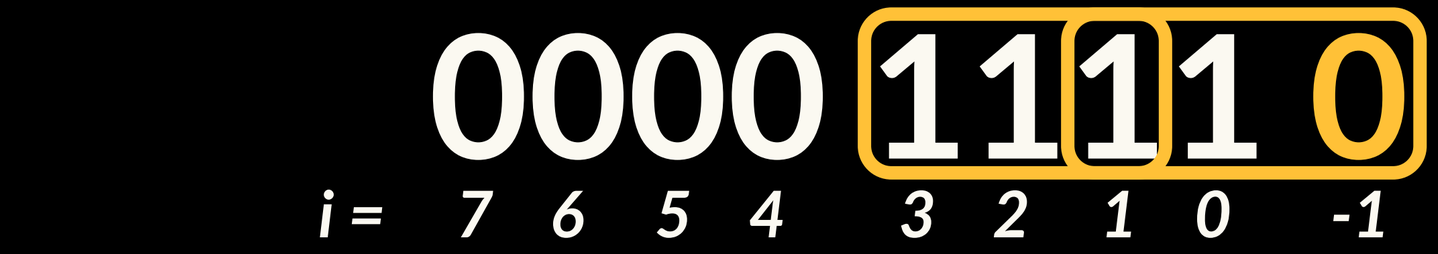
	0	0	0	0	1	1	1	1	0
i =	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*	*-1*

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-4): 16 - 1 = 15 → 1 phép toán



MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-4): 16 - 1 = 15 → 1 phép toán

MẠCH NHÂN

VD: 0000 1111 (15 hệ 10)

Thông thường: 8 + 4 + 2 + 1 = 15 → 3 phép toán

Booth (Radix-4): 16 - 1 = 15 → 1 phép toán

0 0 0 0 1 1 1 1 0

i = 7 6 5 4 3 2 1 0 -1

Booth (Radix-2):

$$\sum_{i=0}^{N-1}(y_{i-1}-y_i)2^i$$

⇒

Booth (Radix-4):

$$\sum_{i=0}^{N/2-1}(-2y_{2i+1}+y_{2i}+y_{2i-1})\cdot 4^i$$

MẠCH NHÂN

$$-2y_{2i+1} + y_{2i} + y_{2i-1}$$

→ Áp dụng Booth (Booth Radix-4):

0000 0000 0000 1111 (15) → M

x

0111 1111 1111 1111 0 (32767)

-M << 0

MẠCH NHÂN

$$-2y_{2i+1} + y_{2i} + y_{2i-1}$$

→ Áp dụng Booth (Booth Radix-4):

0000 0000 0000 1111 (15) → M

x

0111 1111 1111 1111 0 (32767)

-M << 0

0000 ... 0000 0000 0000 0000

...

0000 ... 0000 0000 0000 0000

+2M << 14

} x8 (Sử dụng Dadda để cộng đồng thời 8 hàng)

⇒ Mức độ tối ưu cao.

Thuật toán dễ dàng thiết kế.

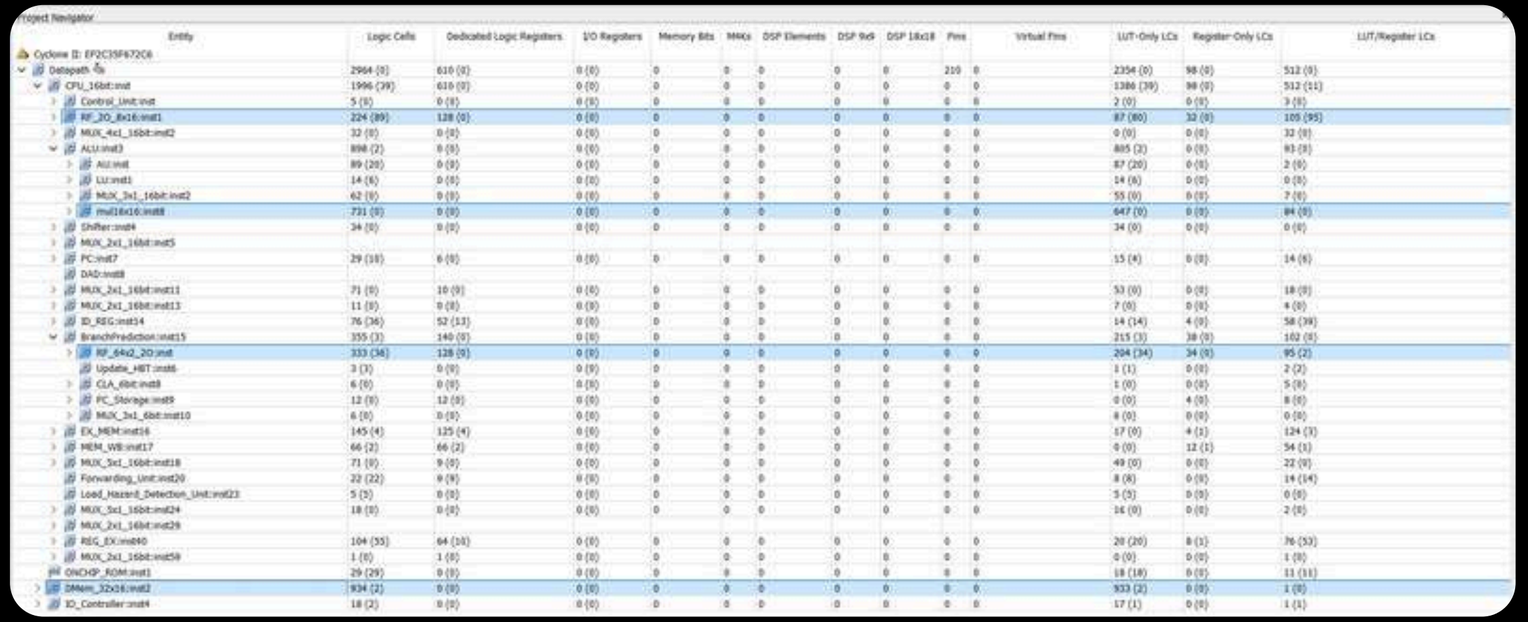
03

KIỂM ĐỊNH

KIỂM TRA TÀI NGUYÊN

Analysis & Synthesis Resource Usage Summary

	Resource	Usage
1	Estimated Total logic elements	3,109
2		
3	Total combinational functions	2774
4	Logic element usage by number of LUT inputs	
1	-- 4 input functions	1830
2	-- 3 input functions	846
3	-- <=2 input functions	98
5		
6	Logic elements by mode	
1	-- normal mode	2774
2	-- arithmetic mode	0
7		
8	Total registers	541
1	-- Dedicated logic registers	541
2	-- I/O registers	0
9		
10	I/O pins	210
11	Embedded Multiplier 9-bit elements	0
12	Maximum fan-out node	CLK
13	Maximum fan-out	541
14	Total fan-out	11707
15	Average fan-out	3.32



KIỂM TRA ĐỊNH THỜI

```
1  create_clock -name clk -period 14.7 [get_ports {CLK}]
```

Slow Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	76.95 MHz	76.95 MHz	clk	

Slow Model Setup Summary

	Clock	Slack	End Point TNS
1	clk	1.704	0.000

Slow Model Hold Summary

	Clock	Slack	End Point TNS
1	clk	0.391	0.000

KIỂM TRA ĐỊNH THỜI

Slow Model Setup Summary

	Clock	Slack	End Point TNS
1	clk	1.704	0.000

Slow Model Hold Summary

	Clock	Slack	End Point TNS
1	clk	0.391	0.000

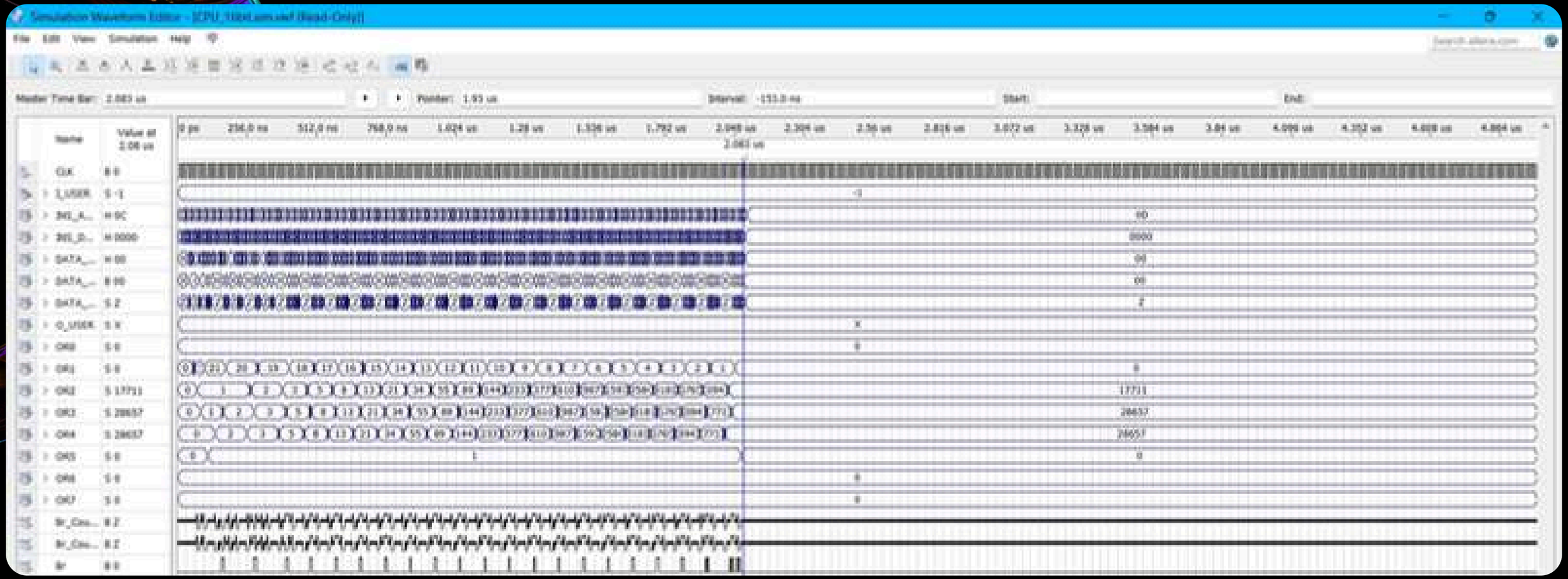
Slow Model Setup: 'clk'

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	1.704	CPU_16bit:inst|EX_MEM:inst16|DFF_2bit:inst11|inst	CPU_16bit:inst|Forwarding_Unit:inst20|inst2~3_OTERM206	clk	clk	14.700	0.003	13.035
2	1.719	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	CPU_16bit:inst|Forwarding_Unit:inst20|inst2~3_OTERM206	clk	clk	14.700	0.000	13.017
3	1.813	CPU_16bit:inst|EX_MEM:inst16|DFF_2bit:inst11|inst	CPU_16bit:inst|MUX_2x1_16bit:inst11|M...UX_2x1:inst1|inst~2_OTERM200_OTERM666	clk	clk	14.700	0.003	12.926
4	1.813	CPU_16bit:inst|MUX_2x1_16bit:inst11|M...UX_2x1:inst1|inst~2_OTERM200_OTERM666	CPU_16bit:inst|Forwarding_Unit:inst20|inst2~3_OTERM206	clk	clk	14.700	0.000	12.923
5	1.828	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	CPU_16bit:inst|MUX_2x1_16bit:inst11|M...UX_2x1:inst1|inst~2_OTERM200_OTERM666	clk	clk	14.700	0.000	12.908
6	1.909	CPU_16bit:inst|MEM_WB:inst17|DFF_2bit:inst11|inst	CPU_16bit:inst|Forwarding_Unit:inst20|inst2~3_OTERM206	clk	clk	14.700	0.003	12.830
7	1.917	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst1|inst15	CPU_16bit:inst|Forwarding_Unit:inst20|inst2~3_OTERM206	clk	clk	14.700	0.003	12.822
8	1.922	CPU_16bit:inst|MUX_2x1_16bit:inst11|M...UX_2x1:inst1|inst~2_OTERM200_OTERM666	CPU_16bit:inst|MUX_2x1_16bit:inst11|M...UX_2x1:inst1|inst~2_OTERM200_OTERM666	clk	clk	14.700	0.000	12.814
9	1.956	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst24|inst13	CPU_16bit:inst|Forwarding_Unit:inst20|inst2~3_OTERM206	clk	clk	14.700	0.003	12.783
10	1.984	CPU_16bit:inst|EX_MEM:inst16|DFF_2bit:inst11|inst	CPU_16bit:inst|Forwarding_Unit:inst20|inst~1_OTERM130	clk	clk	14.700	-0.001	12.751
11	1.992	CPU_16bit:inst|EX_MEM:inst16|DFF_2bit:inst11|inst	CPU_16bit:inst|Forwarding_Unit:inst20|inst~3_OTERM192	clk	clk	14.700	-0.001	12.743
12	1.999	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	CPU_16bit:inst|Forwarding_Unit:inst20|inst~1_OTERM130	clk	clk	14.700	-0.004	12.733
13	2.006	CPU_16bit:inst|EX_MEM:inst16|DFF_2bit:inst11|inst	CPU_16bit:inst|MUX_5x1_16bit:inst18|M...X_2x1:inst2|inst3~3_OTERM524_OTERM528	clk	clk	14.700	-0.001	12.729
14	2.007	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	CPU_16bit:inst|Forwarding_Unit:inst20|inst~3_OTERM192	clk	clk	14.700	-0.004	12.725
15	2.016	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst1|inst12	CPU_16bit:inst|Forwarding_Unit:inst20|inst2~3_OTERM206	clk	clk	14.700	0.003	12.723
16	2.018	CPU_16bit:inst|MEM_WB:inst17|DFF_2bit:inst11|inst	CPU_16bit:inst|MUX_2x1_16bit:inst11|M...UX_2x1:inst1|inst~2_OTERM200_OTERM666	clk	clk	14.700	0.003	12.721
17	2.021	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	CPU_16bit:inst|MUX_5x1_16bit:inst18|M...X_2x1:inst2|inst3~3_OTERM524_OTERM528	clk	clk	14.700	-0.004	12.711
18	2.026	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst1|inst15	CPU_16bit:inst|MUX_2x1_16bit:inst11|M...UX_2x1:inst1|inst~2_OTERM200_OTERM666	clk	clk	14.700	0.003	12.713
19	2.034	CPU_16bit:inst|EX_MEM:inst16|DFF_2bit:inst11|inst	CPU_16bit:inst|MUX_5x1_16bit:inst18|MUX...bit:inst3|MUX_2x1:inst|inst3~1_OTERM134	clk	clk	14.700	-0.001	12.701
20	2.049	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	CPU_16bit:inst|MUX_5x1_16bit:inst18|MUX...bit:inst3|MUX_2x1:inst|inst3~1_OTERM134	clk	clk	14.700	-0.004	12.683
21	2.053	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst3|inst13	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst25|inst14_OTERM306	clk	clk	14.700	-0.162	12.521
22	2.054	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst3|inst13	CPU_16bit:inst|EX_MEM:inst16|DFF_16b...25|inst12_OTERM278_OTERM496_OTERM552	clk	clk	14.700	-0.162	12.520
23	2.065	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst24|inst13	CPU_16bit:inst|MUX_2x1_16bit:inst11|M...UX_2x1:inst1|inst~2_OTERM200_OTERM666	clk	clk	14.700	0.003	12.674
24	2.071	CPU_16bit:inst|MEM_WB:inst17|DFF_2bit:inst11|inst1	CPU_16bit:inst|Forwarding_Unit:inst20|inst2~3_OTERM206	clk	clk	14.700	0.003	12.668
25	2.090	CPU_16bit:inst|EX_MEM:inst16|DFF_2bit:inst11|inst	CPU_16bit:inst|MUX_2x1_16bit:inst11|MUX...bit:inst2|MUX_2x1:inst1|inst~0_OTERM708	clk	clk	14.700	0.003	12.649
26	2.091	CPU_16bit:inst|EX_MEM:inst16|DFF_2bit:inst11|inst	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	clk	clk	14.700	0.003	12.648
27	2.092	CPU_16bit:inst|EX_MEM:inst16|DFF_2bit:inst11|inst	CPU_16bit:inst|MUX_2x1_16bit:inst11|MUX...49bit:inst3|MUX_2x1:inst|inst~0_OTERM696	clk	clk	14.700	0.003	12.647
28	2.093	CPU_16bit:inst|EX_MEM:inst16|DFF_2bit:inst11|inst	CPU_16bit:inst|MUX_2x1_16bit:inst11|MUX...bit:inst3|MUX_2x1:inst2|inst~1_OTERM694	clk	clk	14.700	0.003	12.646
29	2.093	CPU_16bit:inst|MUX_2x1_16bit:inst11|M...UX_2x1:inst1|inst~2_OTERM200_OTERM666	CPU_16bit:inst|Forwarding_Unit:inst20|inst~1_OTERM130	clk	clk	14.700	-0.004	12.639
30	2.101	CPU_16bit:inst|MUX_2x1_16bit:inst11|M...UX_2x1:inst1|inst~2_OTERM200_OTERM666	CPU_16bit:inst|Forwarding_Unit:inst20|inst~3_OTERM192	clk	clk	14.700	-0.004	12.631
31	2.105	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	CPU_16bit:inst|MUX_2x1_16bit:inst11|MUX...bit:inst2|MUX_2x1:inst1|inst~0_OTERM708	clk	clk	14.700	0.000	12.631
32	2.106	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	clk	clk	14.700	0.000	12.630
33	2.107	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	CPU_16bit:inst|MUX_2x1_16bit:inst11|MUX...49bit:inst3|MUX_2x1:inst|inst~0_OTERM696	clk	clk	14.700	0.000	12.629
34	2.108	CPU_16bit:inst|Forwarding_Unit:inst20|inst22~2_OTERM132	CPU_16bit:inst|MUX_2x1_16bit:inst11|MUX...bit:inst3|MUX_2x1:inst2|inst~1_OTERM694	clk	clk	14.700	0.000	12.628

Slow Model Hold: 'clk'

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship	Clock Skew	Data Delay
1	0.391	CPU_16bit:inst|REG_EX:inst40|DFF_3bit:inst4|inst_OTERM616	CPU_16bit:inst|REG_EX:inst40|DFF_3bit:inst4|inst_OTERM616	clk	clk	0.000	0.000	0.657
2	0.391	CPU_16bit:inst|REG_EX:inst40|DFF_3bit:inst4|inst1_OTERM614	CPU_16bit:inst|REG_EX:inst40|DFF_3bit:inst4|inst1_OTERM614	clk	clk	0.000	0.000	0.657
3	0.511	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst3|inst13	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst3|inst13	clk	clk	0.000	0.163	0.940
4	0.519	CPU_16bit:inst|ID_REG:inst14|DFF_16bit:inst|inst15	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst3|inst15	clk	clk	0.000	0.000	0.785
5	0.520	CPU_16bit:inst|ID_REG:inst14|DFF_3bit:inst2|inst1_OTERM690	CPU_16bit:inst|REG_EX:inst40|DFF_3bit:inst4|inst1_OTERM614	clk	clk	0.000	0.000	0.786
6	0.522	CPU_16bit:inst|EX_MEM:inst16|inst10	CPU_16bit:inst|MEM_WB:inst17|inst6	clk	clk	0.000	0.000	0.788
7	0.522	CPU_16bit:inst|ID_REG:inst14|inst17_OTERM584	CPU_16bit:inst|REG_EX:inst40|inst14_OTERM596	clk	clk	0.000	0.000	0.788
8	0.523	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst25|inst13_OTERM292	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst25|inst13	clk	clk	0.000	0.000	0.789
9	0.525	CPU_16bit:inst|ID_REG:inst14|DFF_16bit:inst19|inst1	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst18|inst1	clk	clk	0.000	0.000	0.791
10	0.527	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst25|inst11_OTERM266	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst25|inst11	clk	clk	0.000	0.000	0.793
11	0.527	CPU_16bit:inst|ID_REG:inst14|DFF_16bit:inst19|inst14	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst18|inst14	clk	clk	0.000	0.000	0.793
12	0.530	CPU_16bit:inst|ID_REG:inst14|DFF_16bit:inst19|inst13	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst18|inst13	clk	clk	0.000	0.000	0.796
13	0.533	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst25|inst9_OTERM228	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst25|inst9	clk	clk	0.000	0.000	0.799
14	0.535	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst25|inst10_OTERM240	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst25|inst10	clk	clk	0.000	0.000	0.801
15	0.538	CPU_16bit:inst|ID_REG:inst14|DFF_16bit:inst19|inst12	CPU_16bit:inst|REG_EX:inst40|inst7_OTERM572	clk	clk	0.000	0.000	0.804
16	0.544	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst3|inst8	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst|inst8	clk	clk	0.000	0.000	0.810
17	0.556	CPU_16bit:inst|ID_REG:inst14|DFF_4bit:inst8|inst_OTERM586	CPU_16bit:inst|REG_EX:inst40|inst9_OTERM568	clk	clk	0.000	0.000	0.822
18	0.650	CPU_16bit:inst|ID_REG:inst14|inst5_OTERM578	CPU_16bit:inst|REG_EX:inst40|inst7_OTERM572	clk	clk	0.000	0.000	0.916
19	0.657	CPU_16bit:inst|ID_REG:inst14|DFF_16bit:inst19|inst2	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst18|inst2	clk	clk	0.000	0.000	0.923
20	0.660	CPU_16bit:inst|BranchPrediction:inst15|PC...rage:inst9|RF_1x6_1O:inst3|RFC:inst|inst6	CPU_16bit:inst|BranchPrediction:inst15|PC...rage:inst9|RF_1x6_1O:inst2|RFC:inst|inst6	clk	clk	0.000	0.000	0.926
21	0.660	CPU_16bit:inst|ID_REG:inst14|DFF_16bit:inst19|inst5	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst18|inst5	clk	clk	0.000	0.000	0.926
22	0.661	CPU_16bit:inst|ID_REG:inst14|DFF_16bit:inst19|inst11	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst18|inst11	clk	clk	0.000	0.000	0.927
23	0.662	CPU_16bit:inst|ID_REG:inst14|DFF_3bit:inst2|inst2	CPU_16bit:inst|REG_EX:inst40|DFF_3bit:inst4|inst2_OTERM604	clk	clk	0.000	0.000	0.928
24	0.663	CPU_16bit:inst|BranchPrediction:inst15|PC...age:inst9|RF_1x6_1O:inst3|RFC:inst3|inst6	CPU_16bit:inst|BranchPrediction:inst15|PC...age:inst9|RF_1x6_1O:inst2|RFC:inst3|inst6	clk	clk	0.000	0.000	0.929
25	0.664	CPU_16bit:inst|ID_REG:inst14|DFF_16bit:inst19|inst4	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst18|inst4	clk	clk	0.000	0.000	0.930
26	0.665	CPU_16bit:inst|ID_REG:inst14|DFF_16bit:inst19|inst7	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst18|inst7	clk	clk	0.000	0.000	0.931
27	0.666	CPU_16bit:inst|BranchPrediction:inst15|PC...age:inst9|RF_1x6_1O:inst3|RFC:inst4|inst6	CPU_16bit:inst|BranchPrediction:inst15|PC...age:inst9|RF_1x6_1O:inst2|RFC:inst4|inst6	clk	clk	0.000	0.000	0.932
28	0.666	CPU_16bit:inst|BranchPrediction:inst15|PC...age:inst9|RF_1x6_1O:inst3|RFC:inst7|inst6	CPU_16bit:inst|BranchPrediction:inst15|PC...age:inst9|RF_1x6_1O:inst2|RFC:inst7|inst6	clk	clk	0.000	0.000	0.932
29	0.667	CPU_16bit:inst|EX_MEM:inst16|inst12	CPU_16bit:inst|MEM_WB:inst17|inst5	clk	clk	0.000	0.000	0.933
30	0.670	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst25|inst4_OTERM60	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst25|inst4	clk	clk	0.000	0.000	0.936
31	0.670	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst25|inst5_OTERM72	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst25|inst5	clk	clk	0.000	0.000	0.936
32	0.673	CPU_16bit:inst|REG_EX:inst40|DFF_16bit:inst3|inst9	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst3|inst9	clk	clk	0.000	0.000	0.939
33	0.674	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst25|inst11_OTERM262	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst25|inst11	clk	clk	0.000	0.000	0.940
34	0.678	CPU_16bit:inst|EX_MEM:inst16|DFF_16bit:inst24|inst11	CPU_16bit:inst|MEM_WB:inst17|DFF_16bit:inst|inst11	clk	clk	0.000	0.000	0.944

KIỂM TRA ĐỊNH THỜI



KIỂM TRA CÔNG SUẤT

PowerPlay Power Analyzer Summary

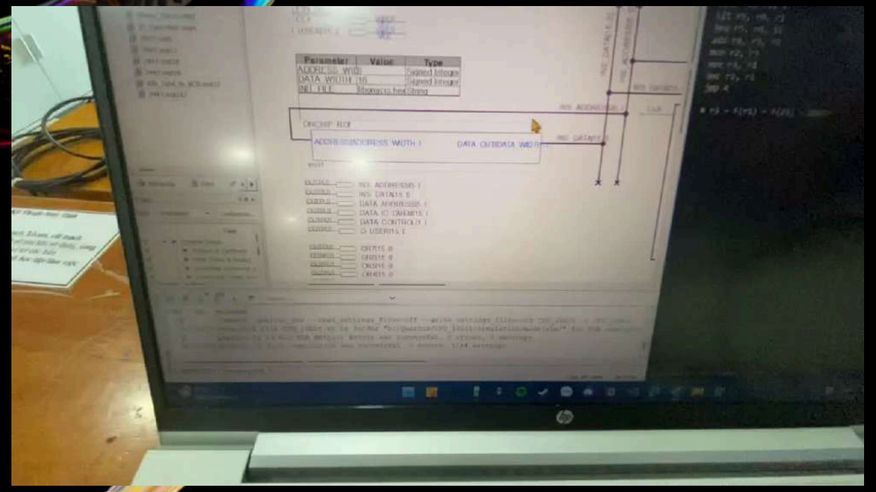
PowerPlay Power Analyzer Status	Successful - Thu Dec 18 12:45:15 2025
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	CPU_16bit
Top-level Entity Name	Datapath
Family	Cyclone II
Device	EP2C35F672C6
Power Models	Final
Total Thermal Power Dissipation	261.75 mW
Core Dynamic Thermal Power Dissipation	36.62 mW
Core Static Thermal Power Dissipation	80.43 mW
I/O Thermal Power Dissipation	144.70 mW
Power Estimation Confidence	Low: user provided insufficient toggle rate data

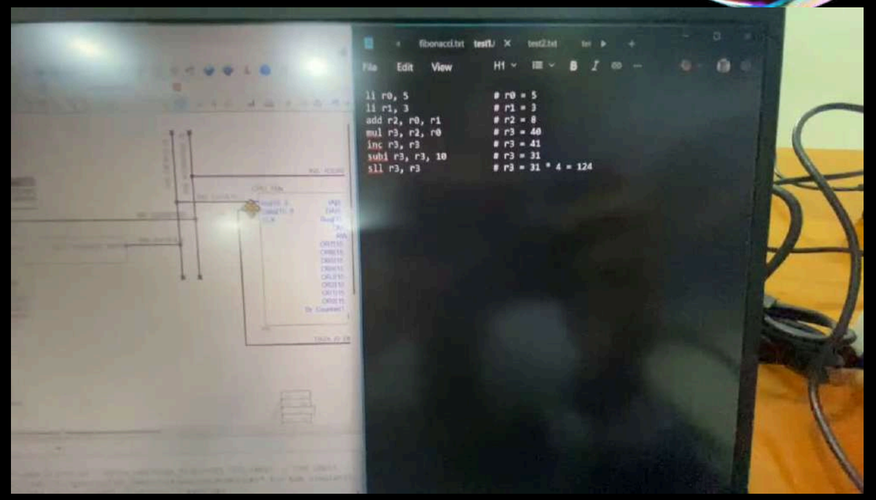
TỔNG HỢP KẾT QUẢ

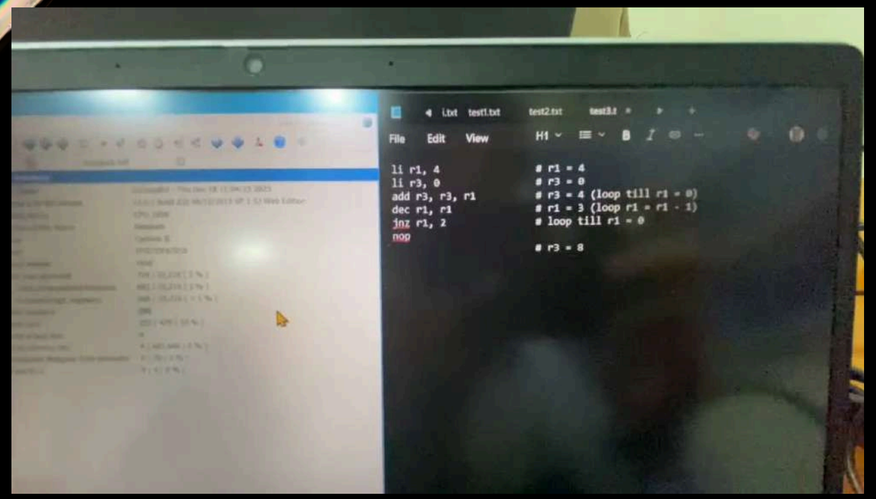
Tiêu chí	CPU 16-bit	Nios II/s (Standard)
Tài nguyên	- Logic elements: **3,109** - Registers: **541**	LEs: ≈ **1,400 - 1,800**
Tần số tối đa	- F_{max} ≈ **76.95** MHz - T_{min} ≈ **13.00** ns	≈ **80 - 100** MHz (trên Cyclone II)
Định thời	- Setup slack: **1.704** ns - Hold slack: **0.319** ns	…
Công suất	- Total Thermal Power Dissipation: **261.75** mW - Core Dynamic Thermal Power Dissipation: **36.62** mW - Core Static Thermal Power Dissipation: **80.43** mW - I/O Thermal Power Dissipation: **144.70** mW	- Core Dynamic Thermal Power Dissipation: **10 - 15 mW**

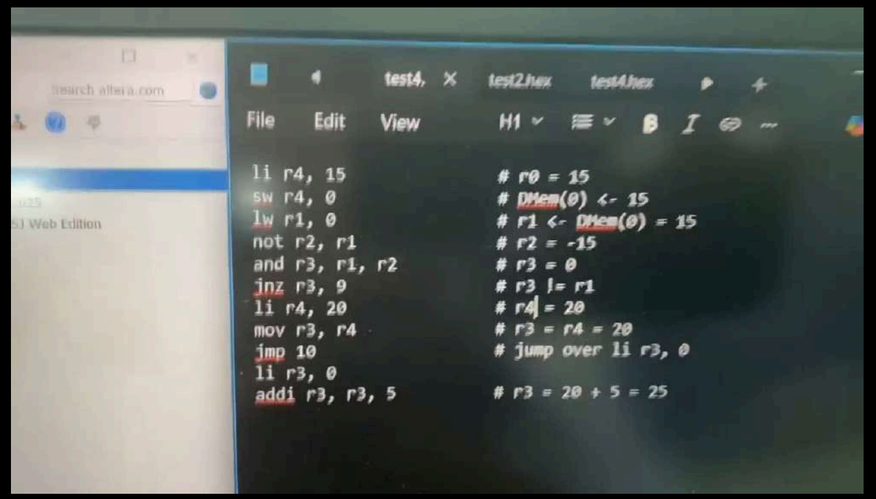
Hiệu năng (Testbench: Tìm số Fibonacci thứ 23)	- Thời gian thực thi: ≈ **2.083** µs
	- Testbench:
	li r1, 23
	li r2, 1
	li r3, 1
	subi r1, r1, 2
	slt r5, r0, r1
	beq r5, r0, 11
	add r4, r3, r2
	mov r2, r3
	mov r3, r4
	dec r1, r1
	jmp 4
		- Thời gian thực thi: ≈ **2.5 - 3.5** µs

MÔ PHỎNG FPGA









Nhóm 1 - CE118.Q11.VMTN

THANK YOU

for your time and attention

GitHub Project: