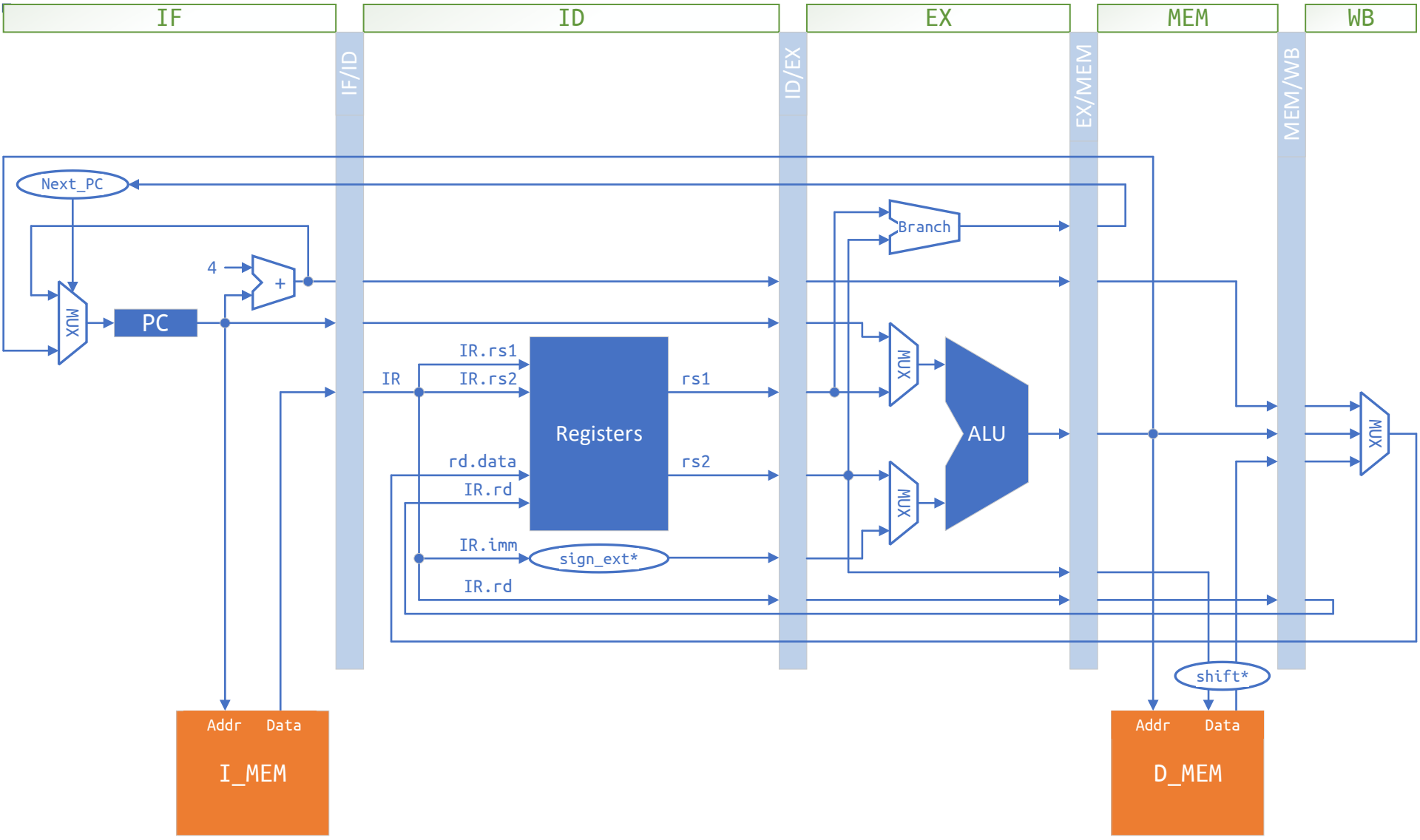
FRISC-V

Hazardi

Put podataka v1.0

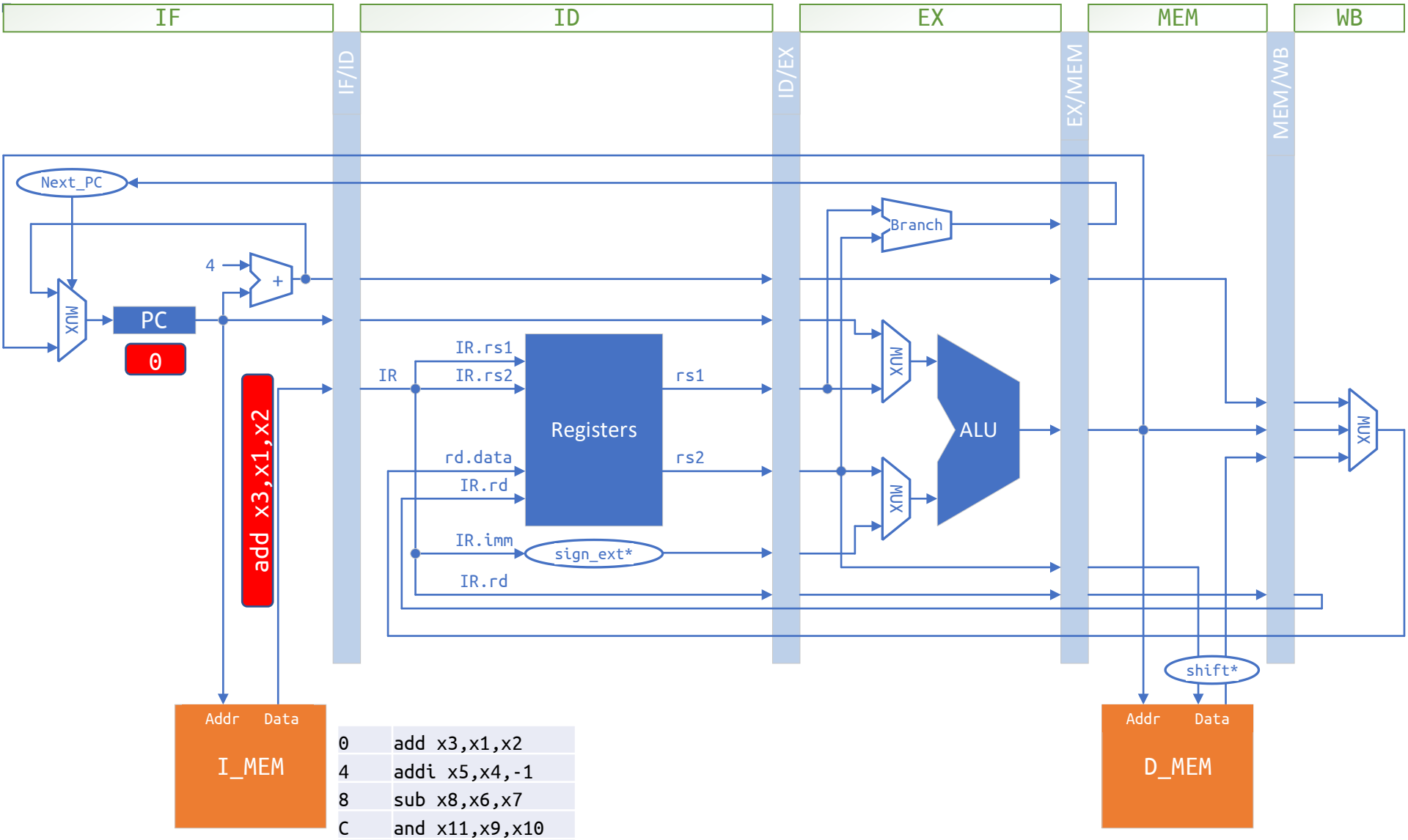


Primjer niza naredaba #1


```
0       add x3,x1,x2
4       addi x5,x4,-1
8       sub x8,x6,x7
C       and x11,x9,x10
10      or x14,x12,x13
```

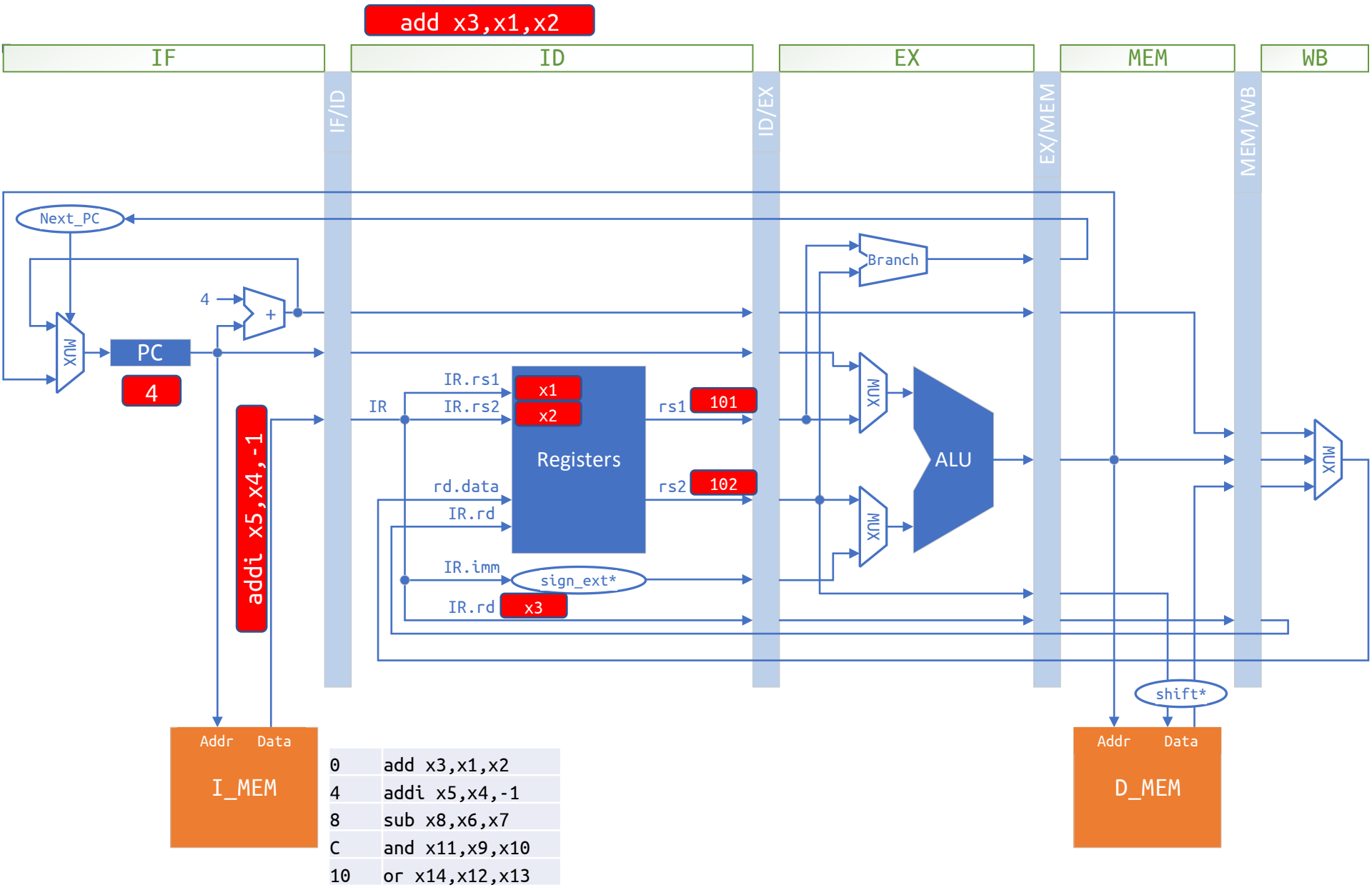
Pogledajmo kako se izvodi ovaj program u našoj mikroarhitekturi…
Zadajmo da su početne vrijednosti registara npr: x1=101, x2=102, x3=103..... x31=131

Period = 1



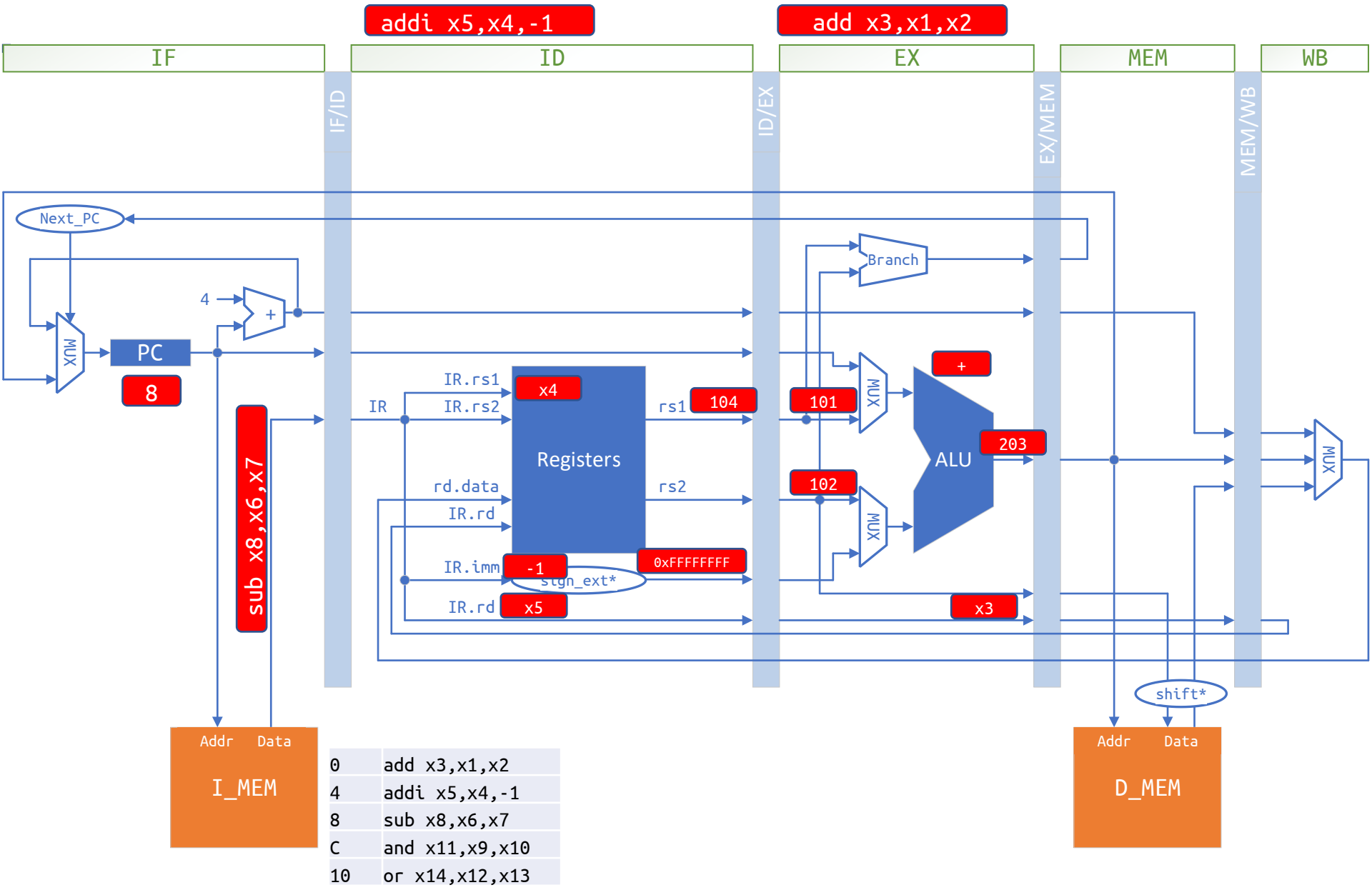
0	add x3,x1,x2
4	addi x5,x4,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 2



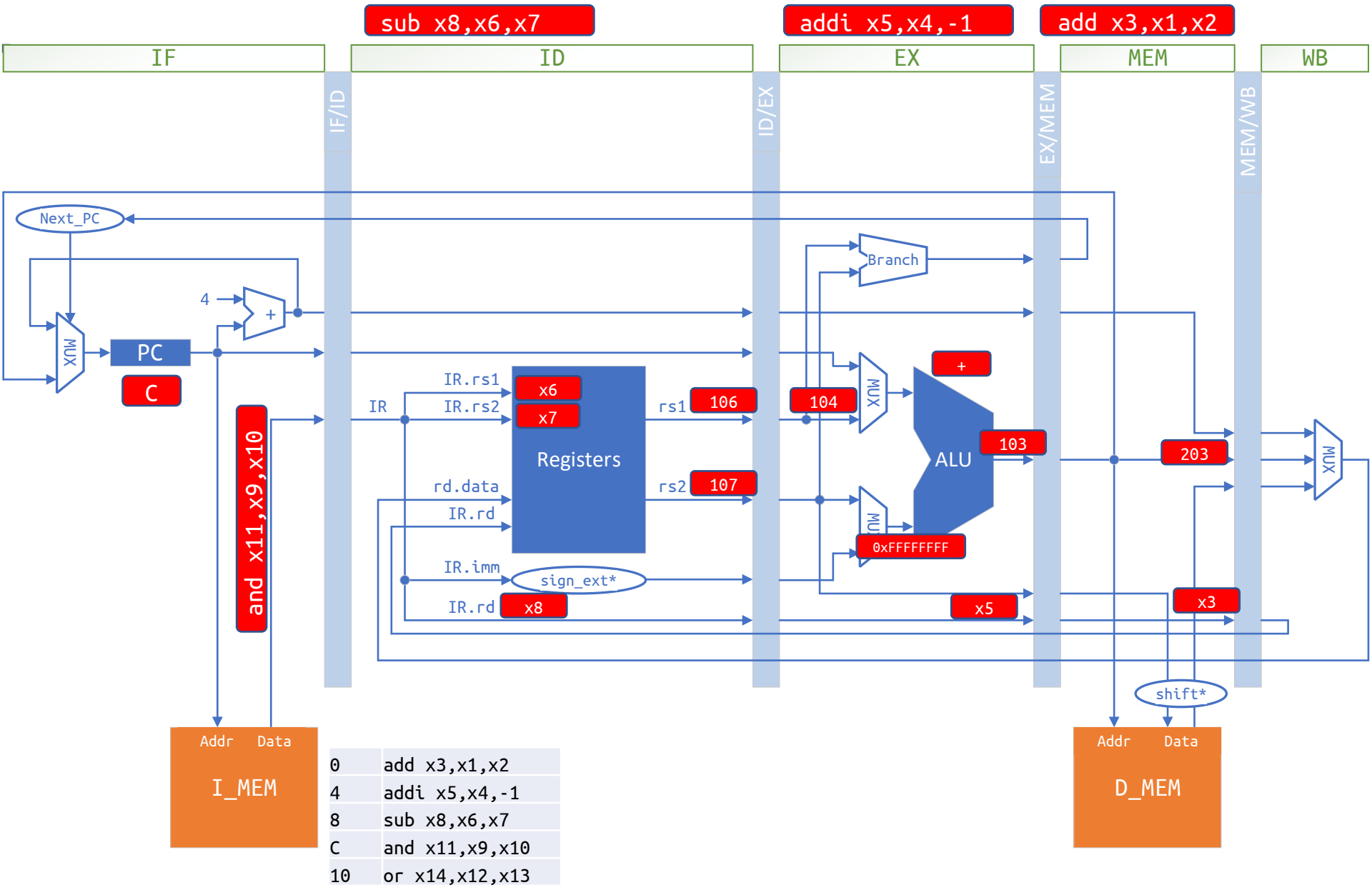
0	add x3,x1,x2
4	addi x5,x4,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 3



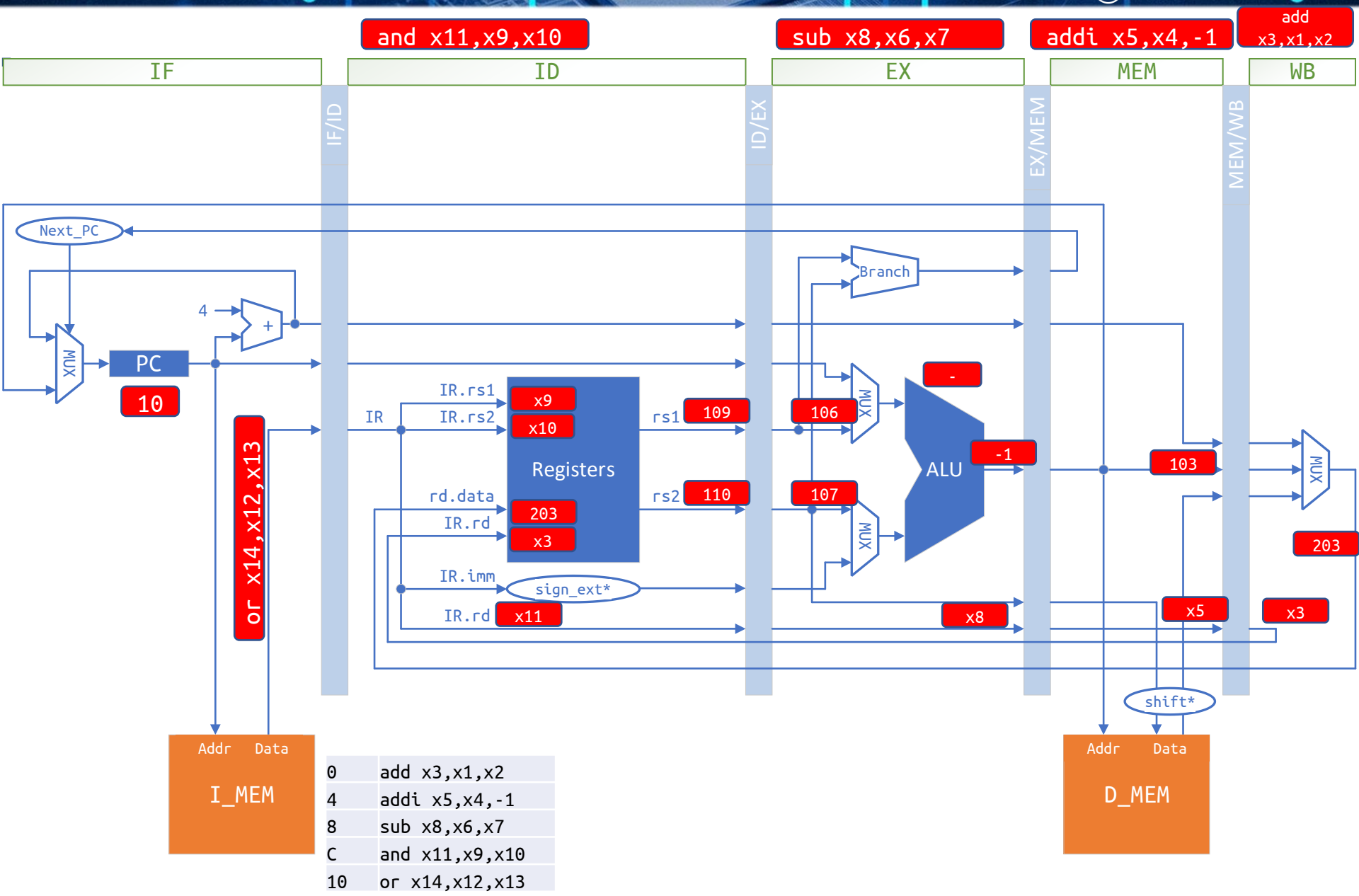
0	add x3,x1,x2
4	addi x5,x4,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 4



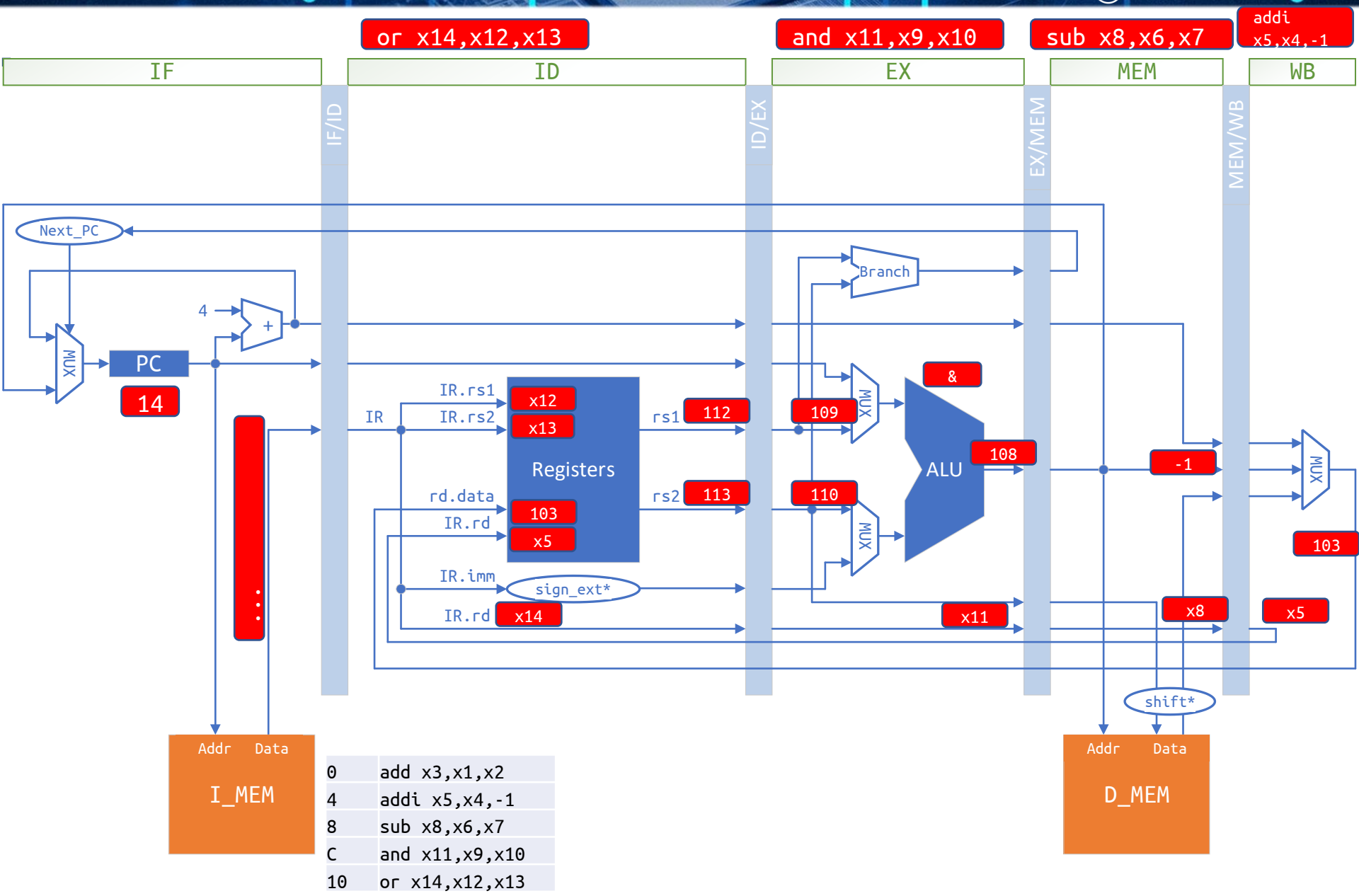
0	add x3,x1,x2
4	addi x5,x4,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 5



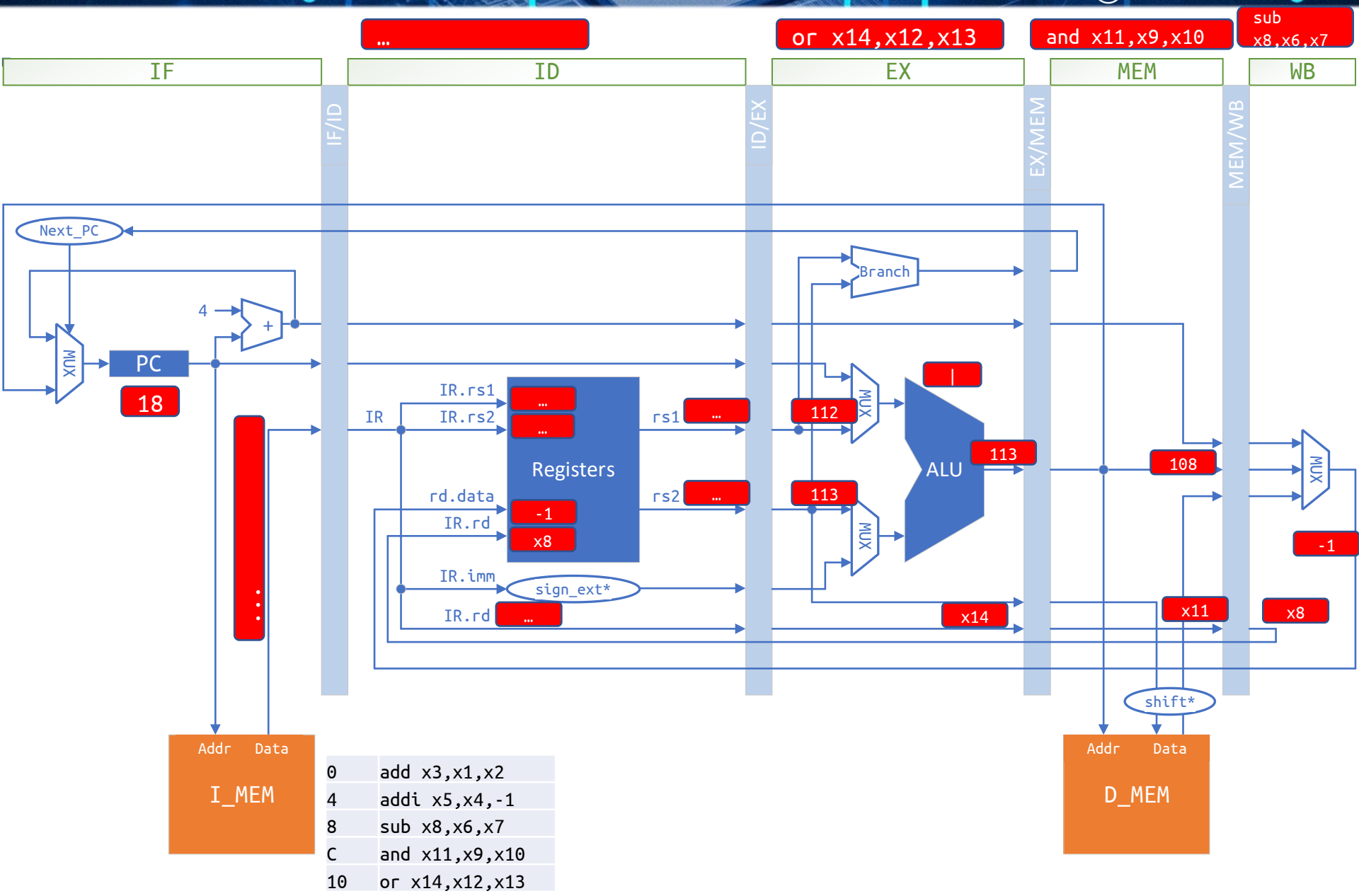
0	add x3,x1,x2
4	addi x5,x4,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 6

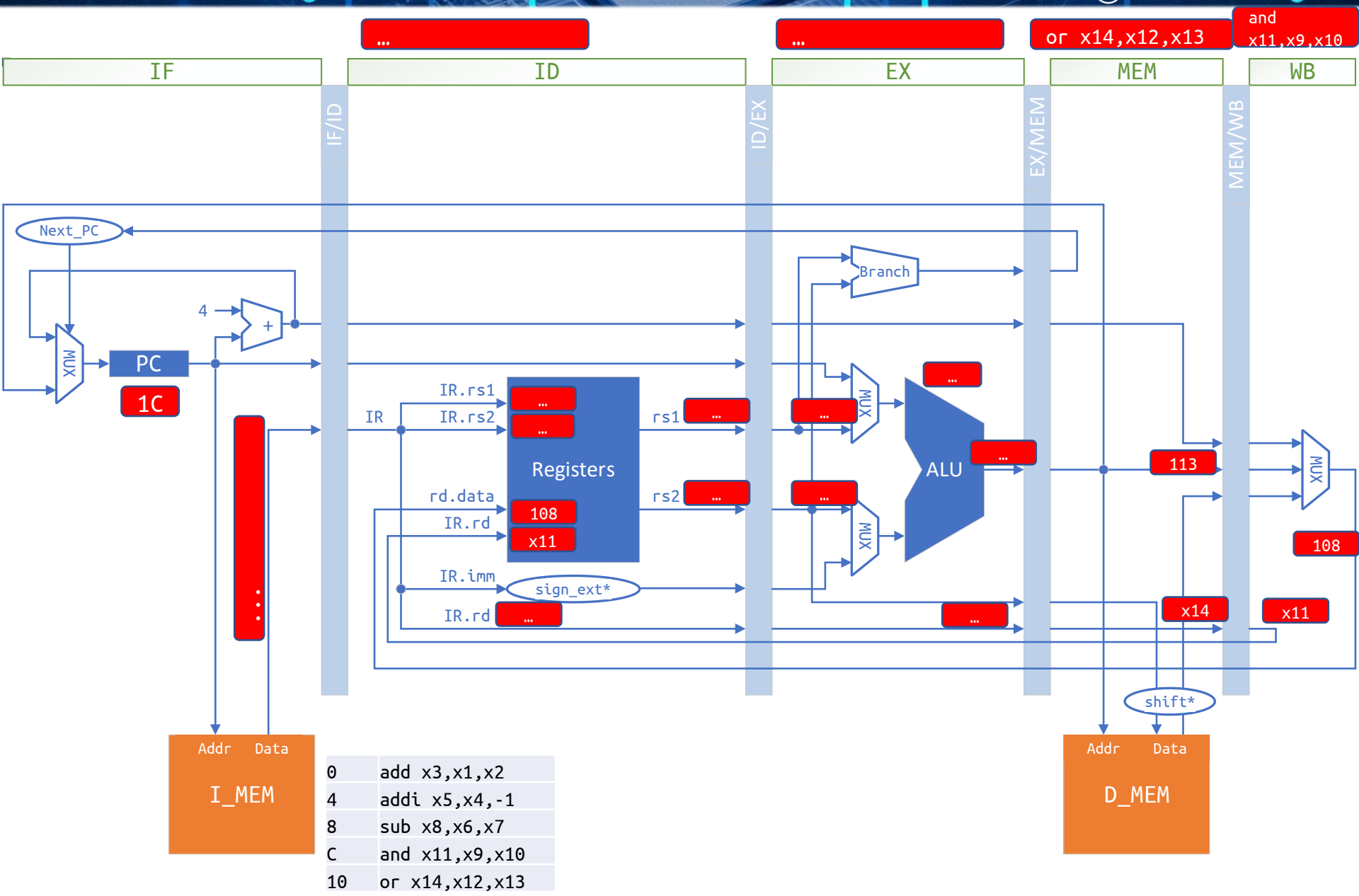


0	add x3,x1,x2
4	addi x5,x4,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 7

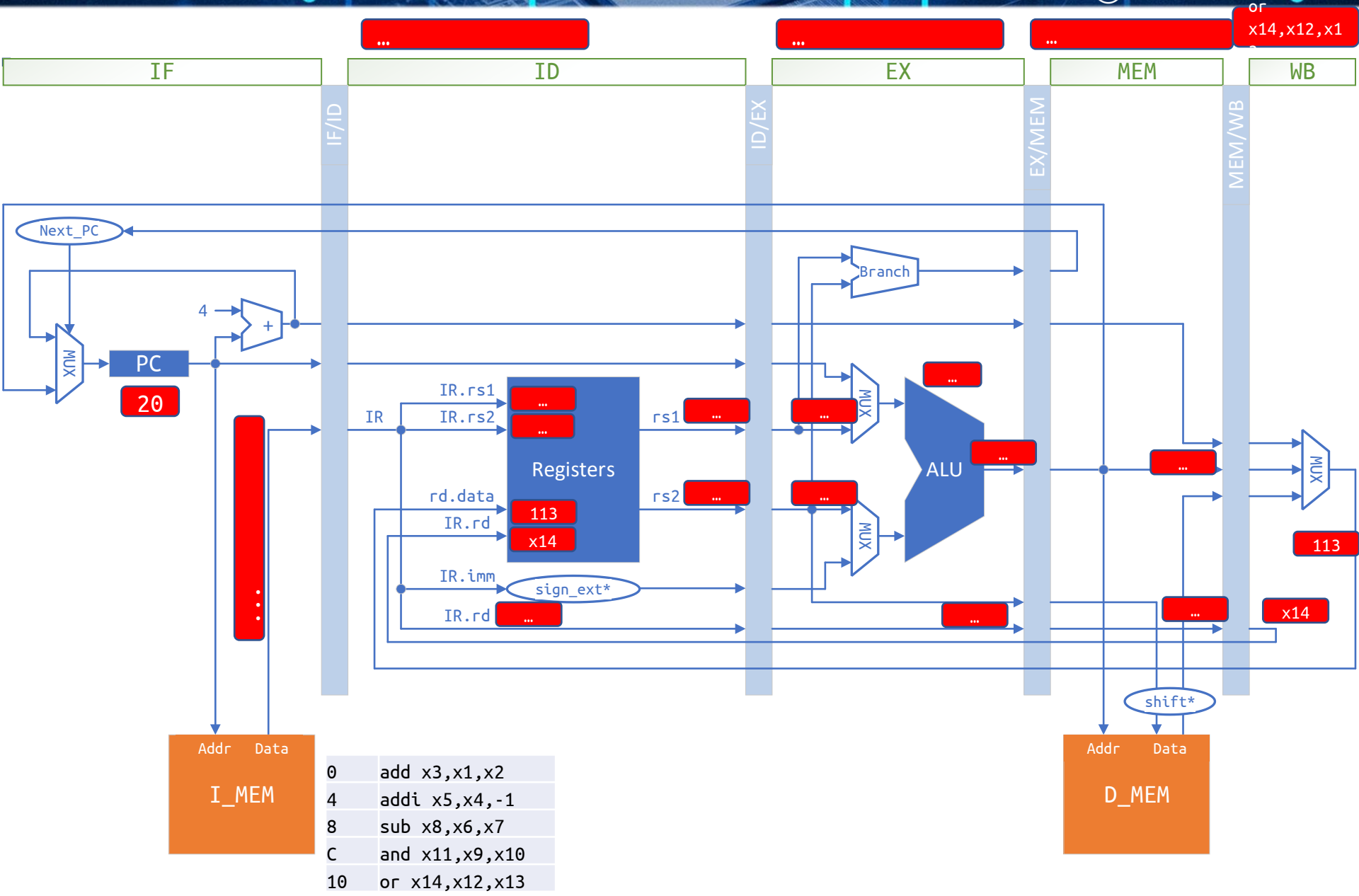

...
or x14,x12,x13
and x11,x9,x10
sub x8,x6,x7
IF
ID
EX
MEM
WB
IF/ID
ID/EX
EX/MEM
MEM/WB
Next_PC
MUX
PC
18
4
+
IR
IR.rs1
IR.rs2
rd.data
IR.rd
IR.imm
IR.rd
Registers
-1
x8
sign_ext*
rs1
rs2
Branch
112
113
ALU
x14
108
x11
x8
-1
shift*
Addr
Data
I_MEM
D_MEM
0 add x3,x1,x2
4 addi x5,x4,-1
8 sub x8,x6,x7
C and x11,x9,x10
10 or x14,x12,x13

Period = 8



0	add x3,x1,x2
4	addi x5,x4,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 9



0	add x3,x1,x2
4	addi x5,x4,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Primjer niza naredaba #1


```
0       add x3,x1,x2
4       addi x5,x4,-1
8       sub x8,x6,x7
C       and x11,x9,x10
10      or x14,x12,x13
```

Period	IF	ID	EX	MEM	WB
1	(PC=0) add				
2	(PC=4) addi	add			
3	(PC=8) sub	addi	add		
4	(PC=C) and	sub	addi	add	
5	(PC=10) or	and	sub	addi	add
6		or	and	sub	addi
7			or	and	sub
8				or	and
9					or

Ove naredbe se izutetno efikasno izvode.

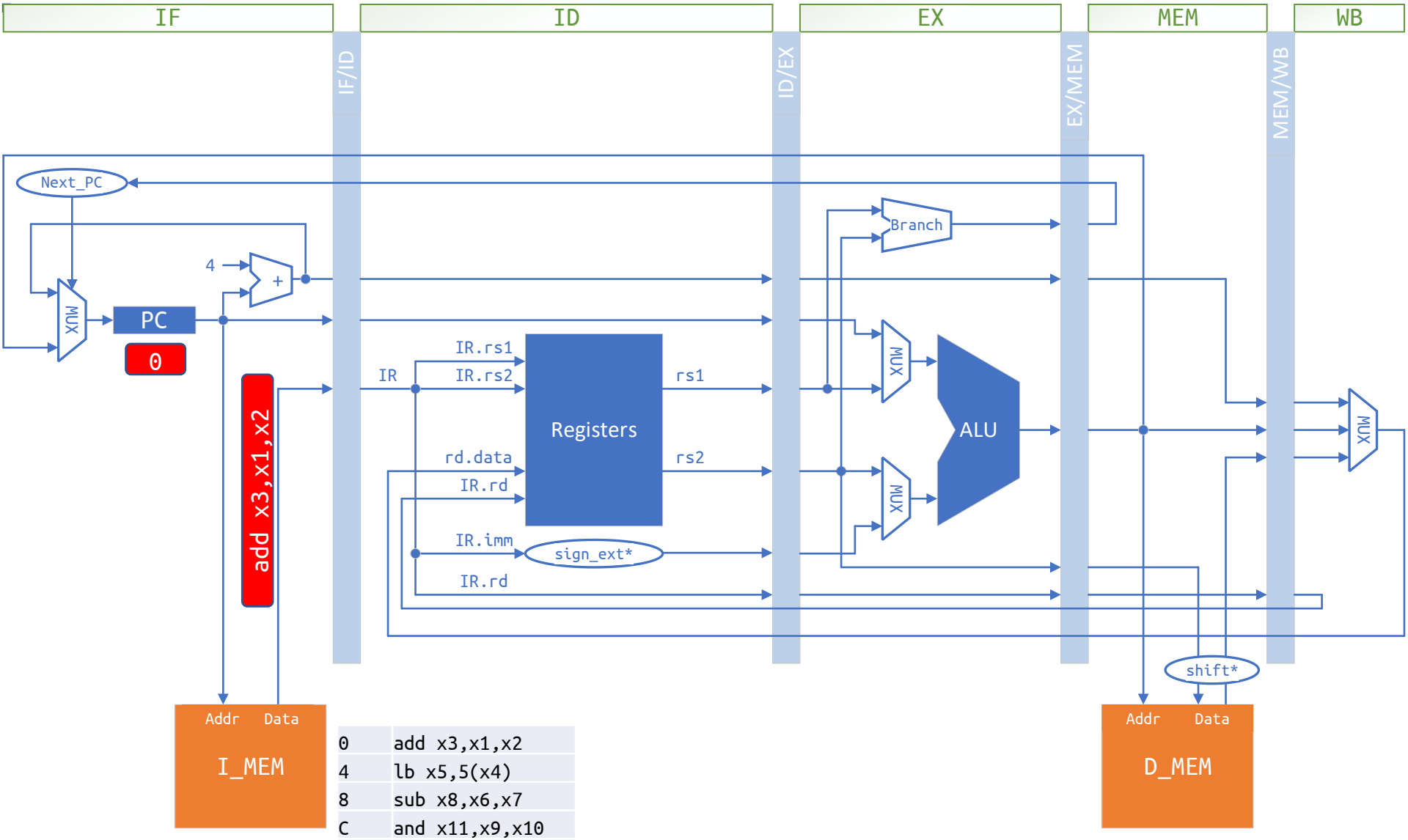
U prosjeku za izvođenje treba 1 period po naredbi ☺

Primjer niza naredaba #2

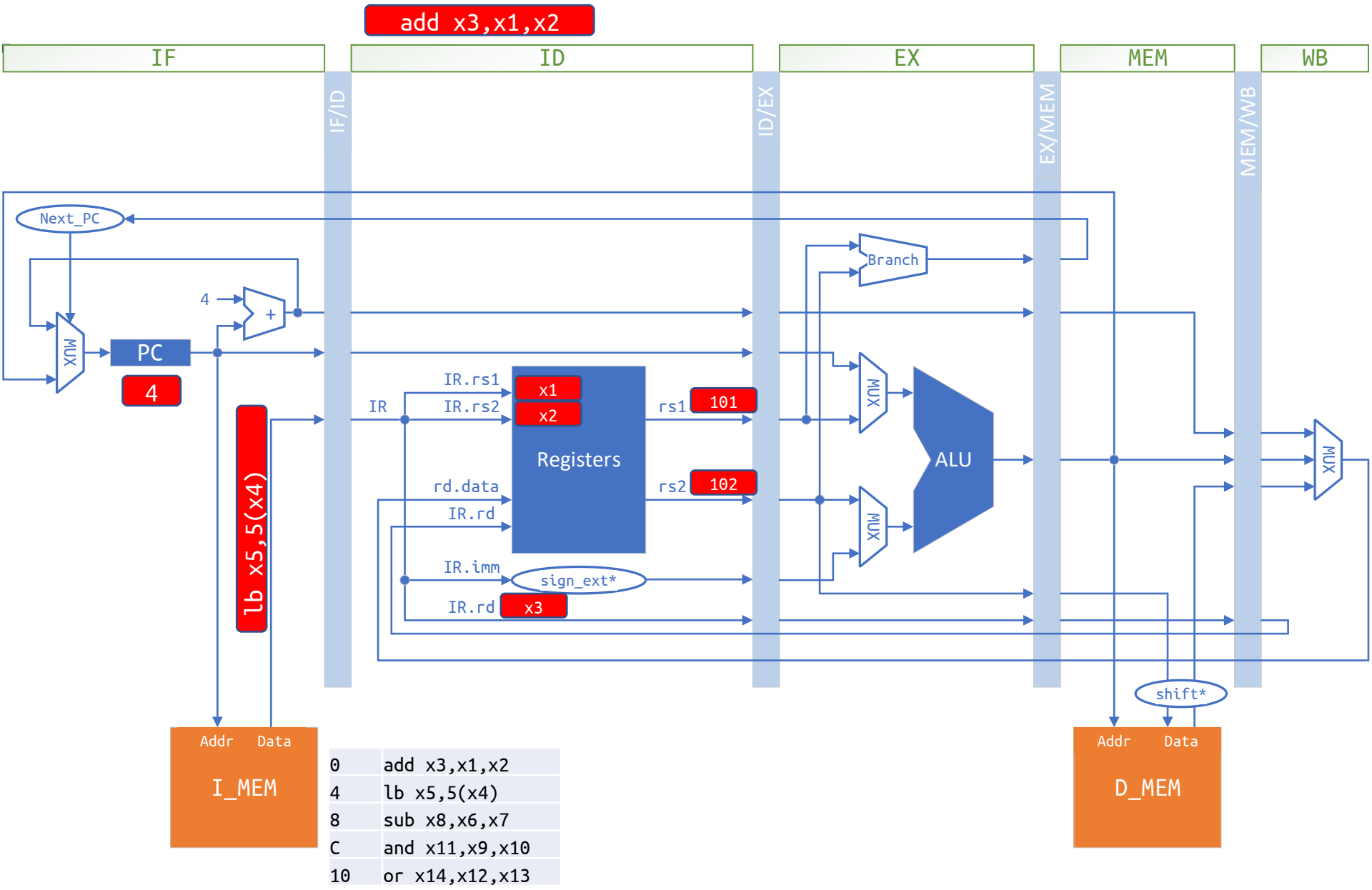

```
0       add x3,x1,x2
4       lb x5,5(x4)
8       sub x8,x6,x7
C       and x11,x9,x10
10      or x14,x12,x13
```

Pogledajmo kako se izvodi ovaj program u našoj mikroarhitekturi…
Zadajmo da su početne vrijednosti registara npr: x1=101, x2=102, x3=103..... x31=131
Zadajmo da je MEM[108]=0xAA; MEM[109]=0xBB ; MEM[110]=0xCC ; MEM[111]=0xDD

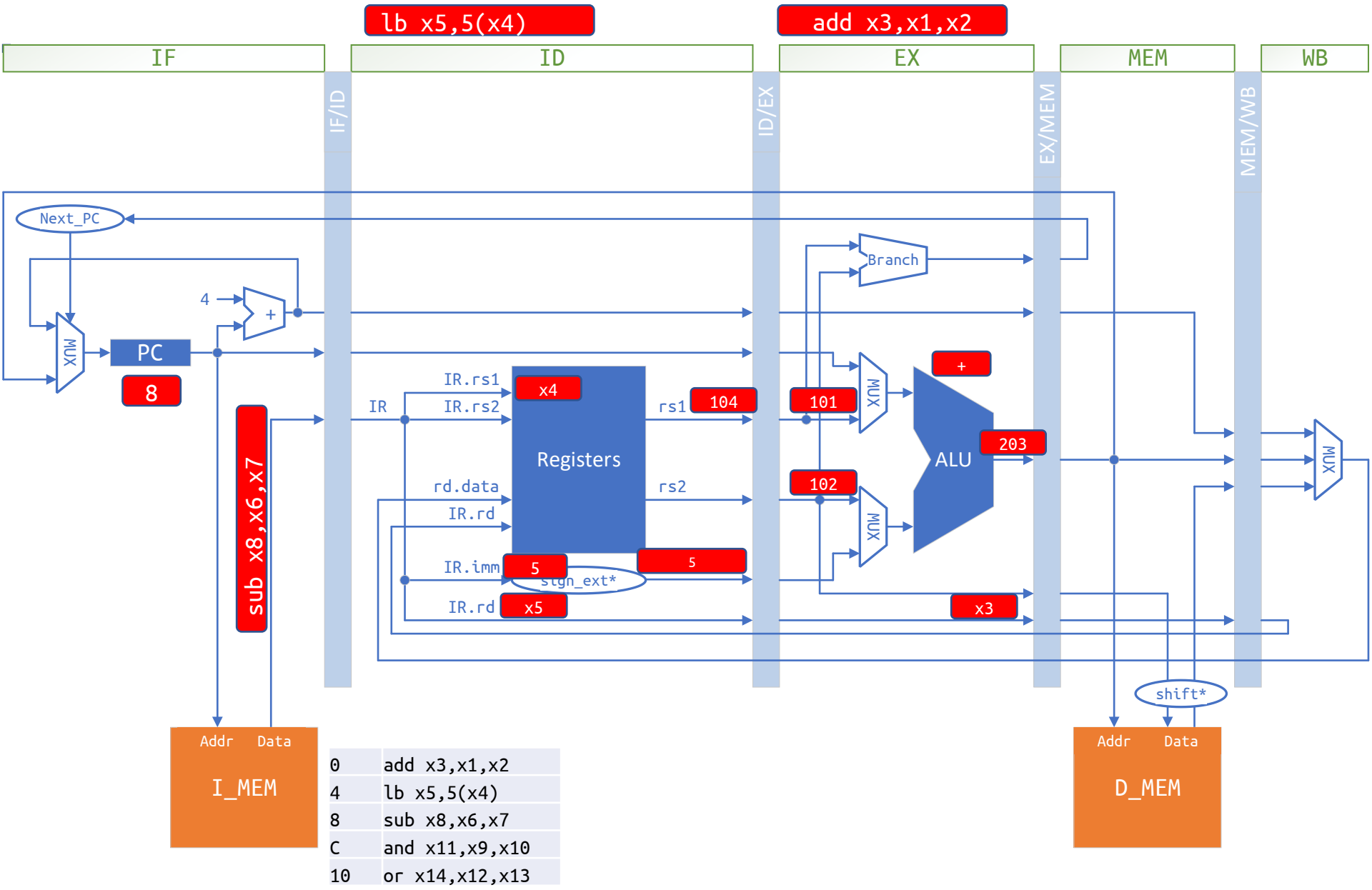
Period = 1


IF
ID
EX
MEM
WB
IF/ID
ID/EX
EX/MEM
MEM/WB
Next_PC
4
+
MUX
PC
0
add x3,x1,x2
IR
IR.rs1
IR.rs2
Registers
rs1
rs2
rd.data
IR.rd
IR.imm
sign_ext*
IR.rd
Branch
MUX
MUX
ALU
MUX
shift*
Addr
Data
I_MEM
Addr
Data
D_MEM
0 add x3,x1,x2
4 lb x5,5(x4)
8 sub x8,x6,x7
C and x11,x9,x10
10 or x14,x12,x13

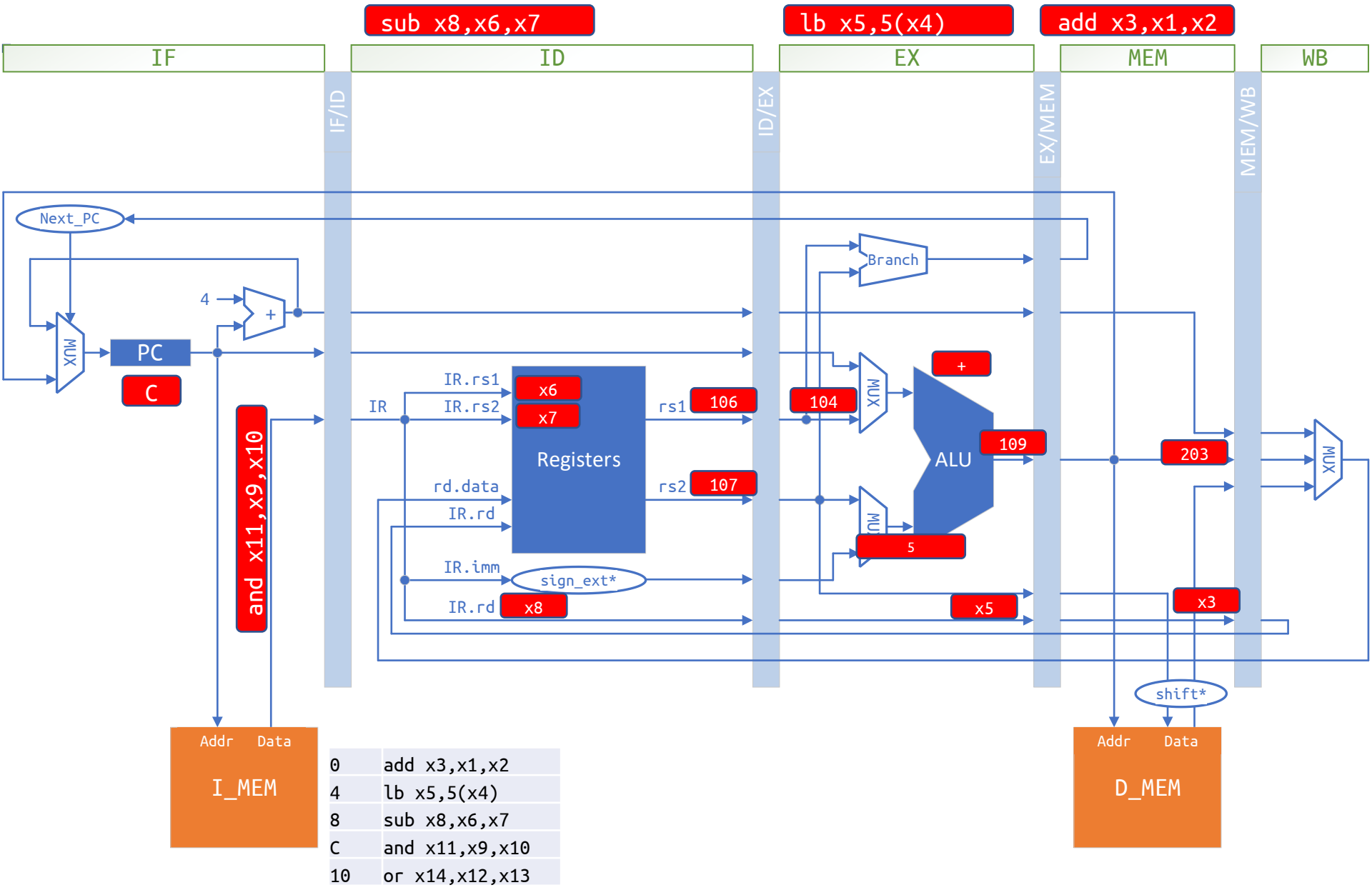
Period = 2


add x3,x1,x2
IF
ID
EX
MEM
WB
IF/ID
ID/EX
EX/MEM
MEM/WB
Next_PC
MUX
PC
4
4
+
lb x5,5(x4)
IR
IR.rs1
IR.rs2
x1
x2
Registers
rs1
101
rs2
102
rd.data
IR.rd
IR.imm
sign_ext*
IR.rd
x3
Branch
MUX
MUX
MUX
ALU
MUX
shift*
Addr
Data
I_MEM
Addr
Data
D_MEM
0 add x3,x1,x2
4 lb x5,5(x4)
8 sub x8,x6,x7
C and x11,x9,x10
10 or x14,x12,x13

Period = 3

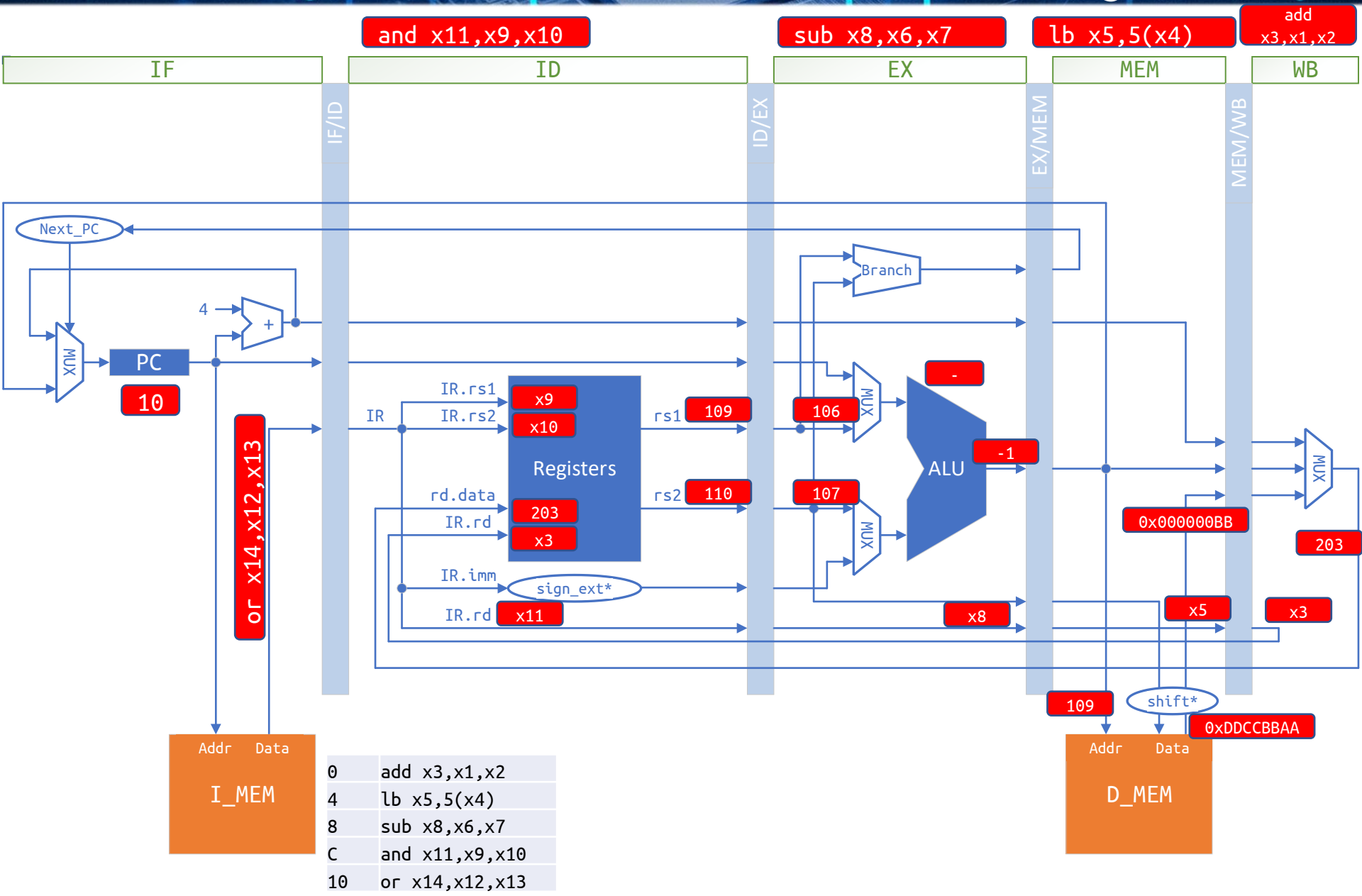

lb x5,5(x4)
add x3,x1,x2
IF
ID
EX
MEM
WB
IF/ID
ID/EX
EX/MEM
MEM/WB
Next_PC
MUX
PC
8
4
+
sub x8,x6,x7
IR.rs1
IR.rs2
IR
x4
Registers
rs1
104
rs2
rd.data
IR.rd
IR.imm
5
sign_ext*
5
IR.rd
x5
Branch
101
102
MUX
+
ALU
203
x3
shift*
Addr
Data
I_MEM
Addr
Data
D_MEM
0 add x3,x1,x2
4 lb x5,5(x4)
8 sub x8,x6,x7
C and x11,x9,x10
10 or x14,x12,x13

Period = 4



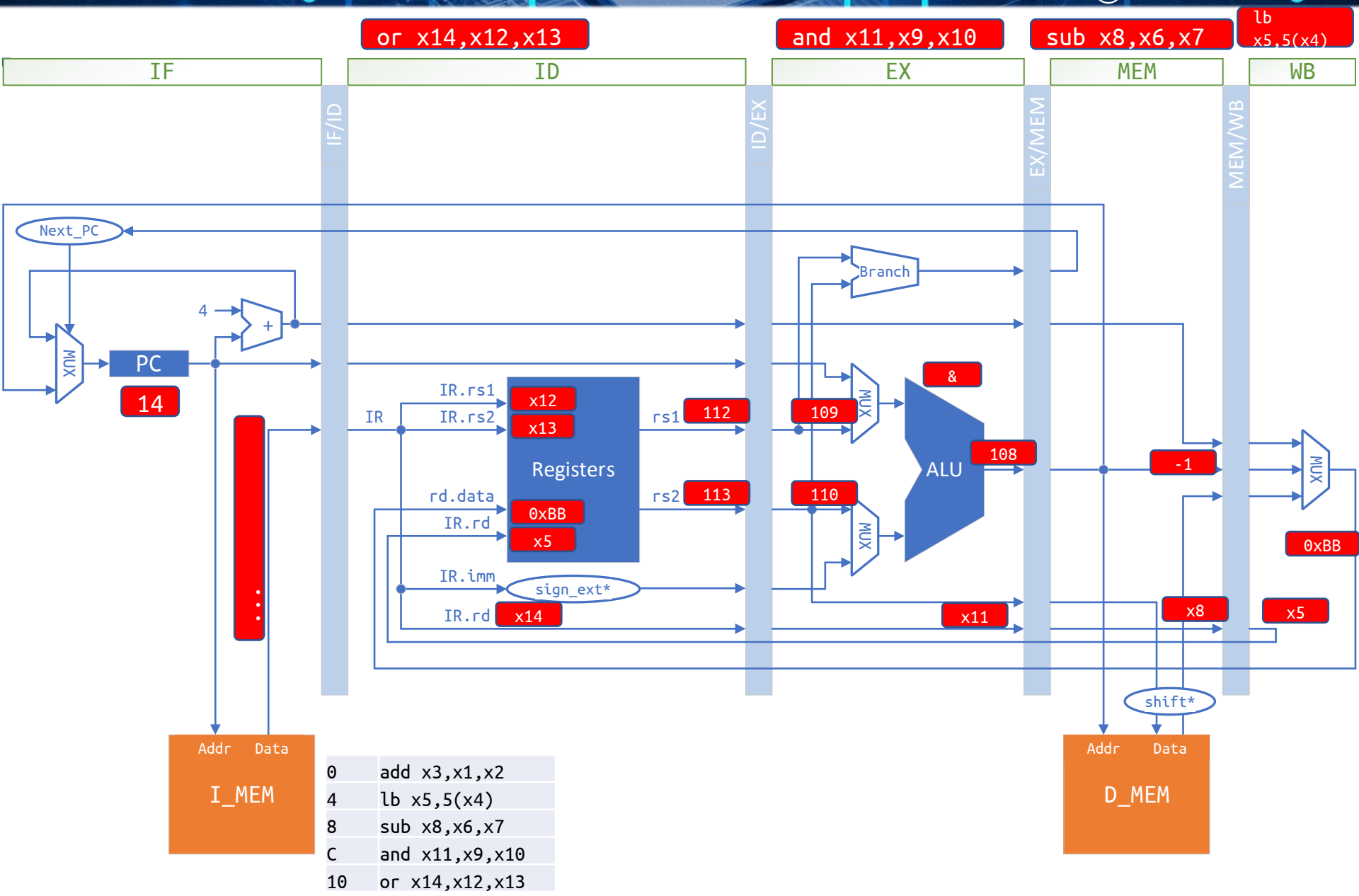
0	add x3,x1,x2
4	lb x5,5(x4)
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 5



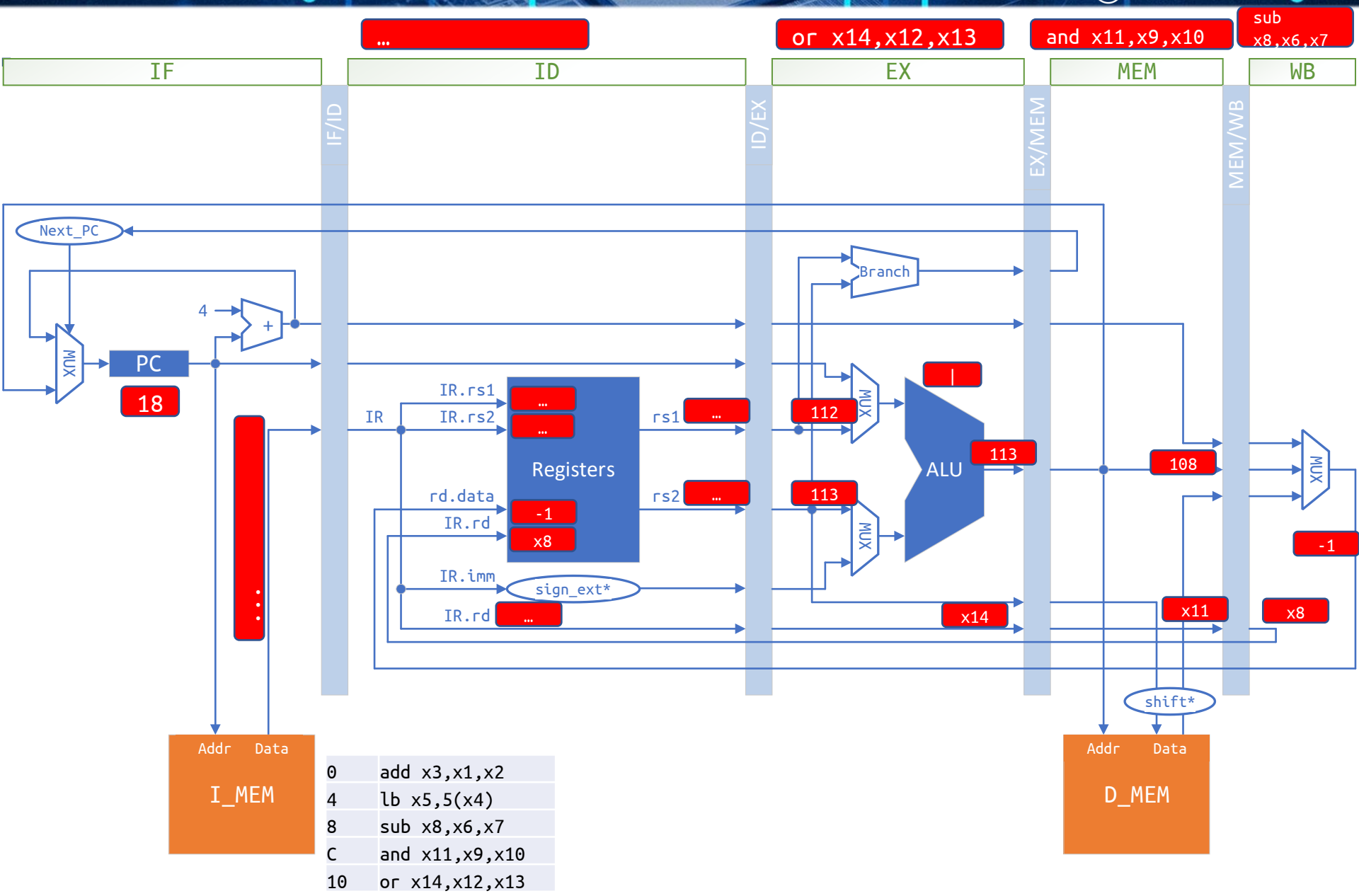
0	add x3,x1,x2
4	lb x5,5(x4)
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 6



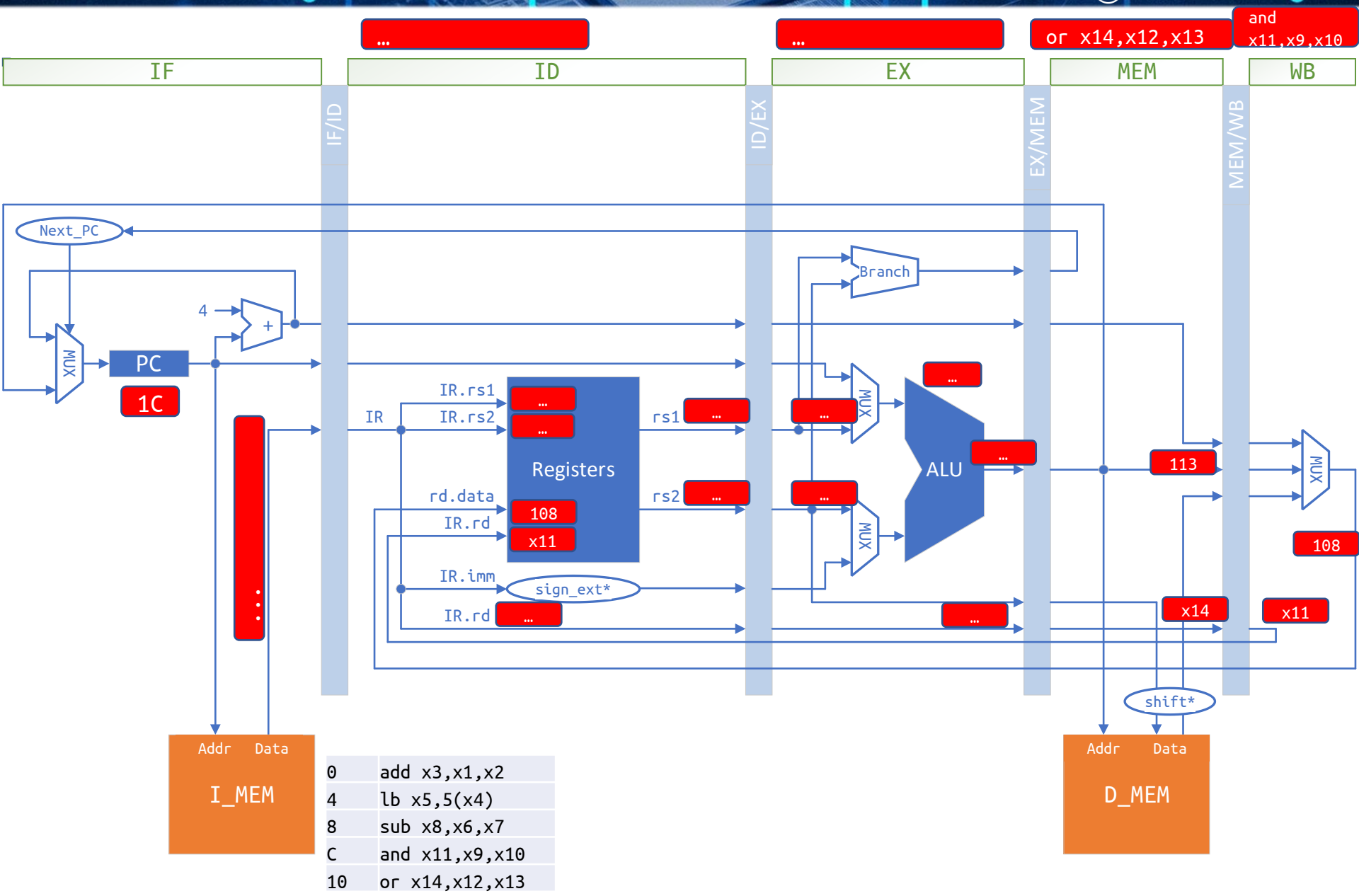
0	add x3,x1,x2
4	lb x5,5(x4)
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 7

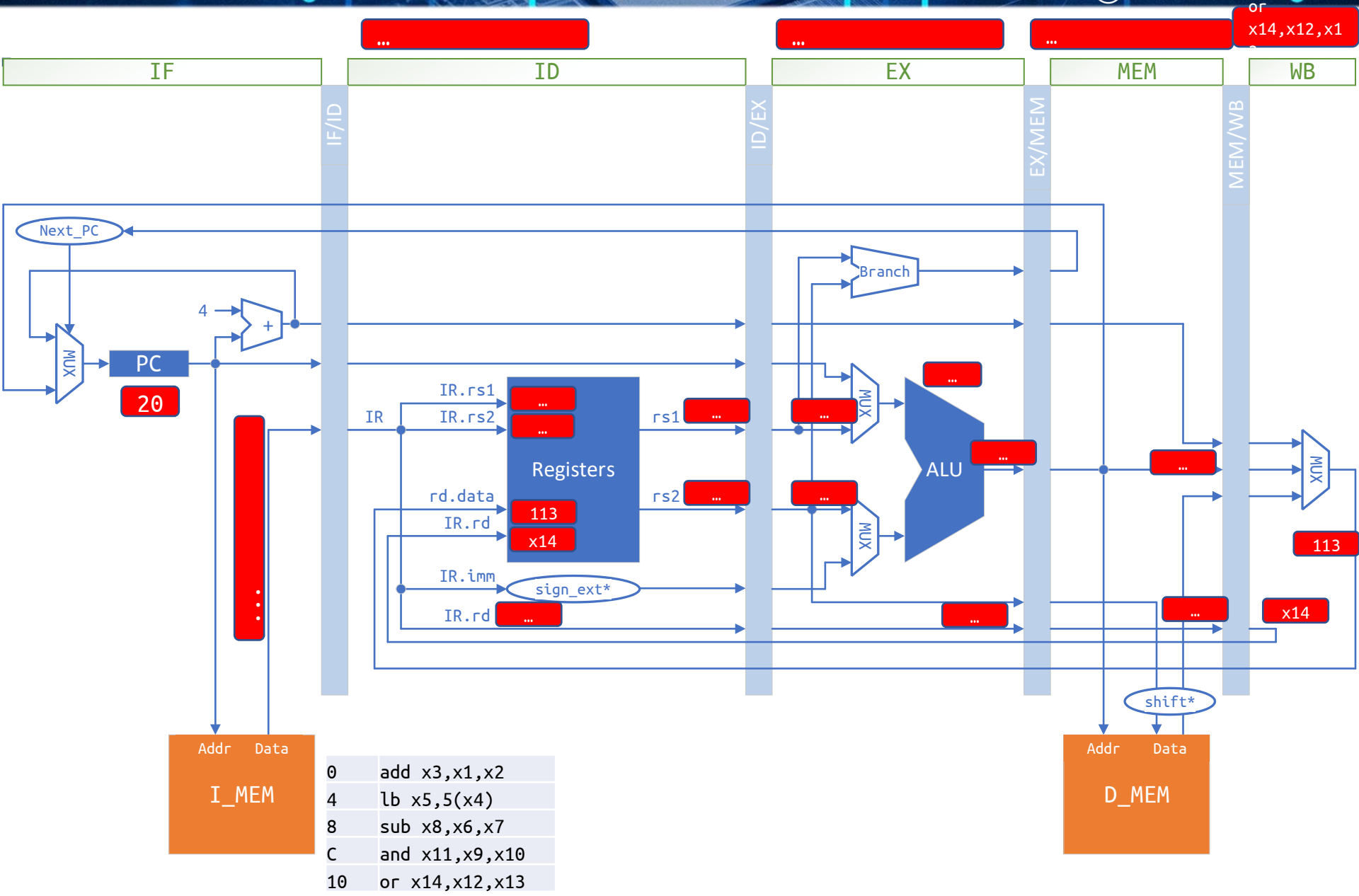



```
0     add x3,x1,x2
4     lb x5,5(x4)
8     sub x8,x6,x7
C     and x11,x9,x10
10    or x14,x12,x13
```

Period = 8


...
...
or x14,x12,x13
and x11,x9,x10
IF
ID
EX
MEM
WB
IF/ID
ID/EX
EX/MEM
MEM/WB
Next_PC
4
+
MUX
PC
1C
IR
IR.rs1
IR.rs2
Registers
rs1
rs2
rd.data
IR.rd
108
x11
IR.imm
sign_ext*
IR.rd
Branch
MUX
ALU
113
108
x14
x11
shift*
Addr
Data
I_MEM
Addr
Data
D_MEM
0 add x3,x1,x2
4 lb x5,5(x4)
8 sub x8,x6,x7
C and x11,x9,x10
10 or x14,x12,x13

Period = 9



0	add x3,x1,x2
4	lb x5,5(x4)
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Primjer niza naredaba #2


```
0     add x3,x1,x2
4     lb x5,5(x4)
8     sub x8,x6,x7
C     and x11,x9,x10
10    or x14,x12,x13
```

Period	IF	ID	EX	MEM	WB
1	(PC=0) add				
2	(PC=4) lb	add			
3	(PC=8) sub	lb	add		
4	(PC=C) and	sub	lb	add	
5	(PC=10) or	and	sub	lb	add
6		or	and	sub	lb
7			or	and	sub
8				or	and
9					or

Ove naredbe se ponovo izutetno efikasno izvode.

LOAD NAREDBA NE UZROKUJE HAZARD !!! (vrijedi isto i za STORE)

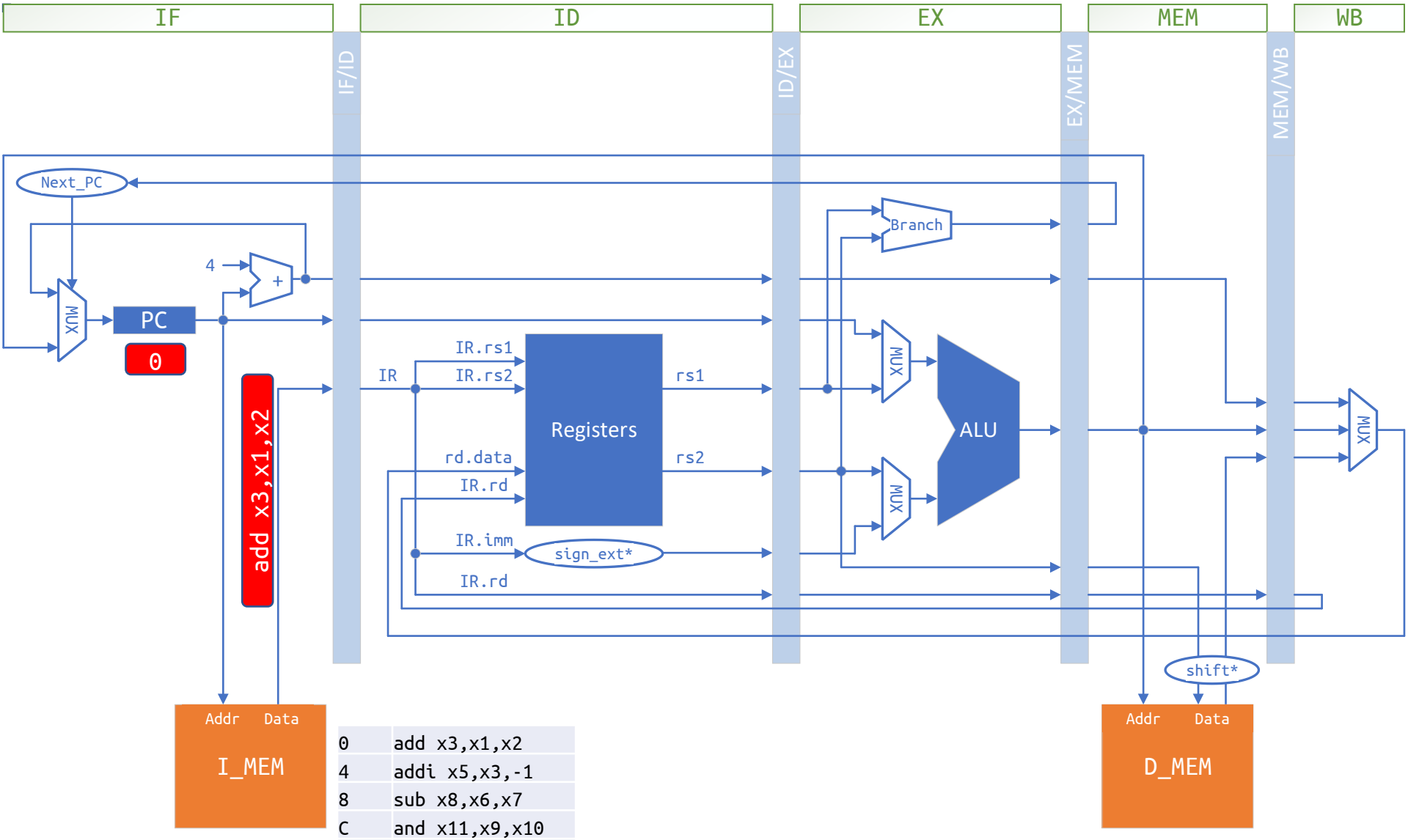
Zbog Harvard arhitekture NEMA STRUKTURNOG HAZARDA!!!

Primjer niza naredaba #3


```
0       add x3,x1,x2
4       addi x5,x3,-1
8       sub x8,x6,x7
C       and x11,x9,x10
10      or x14,x12,x13
```

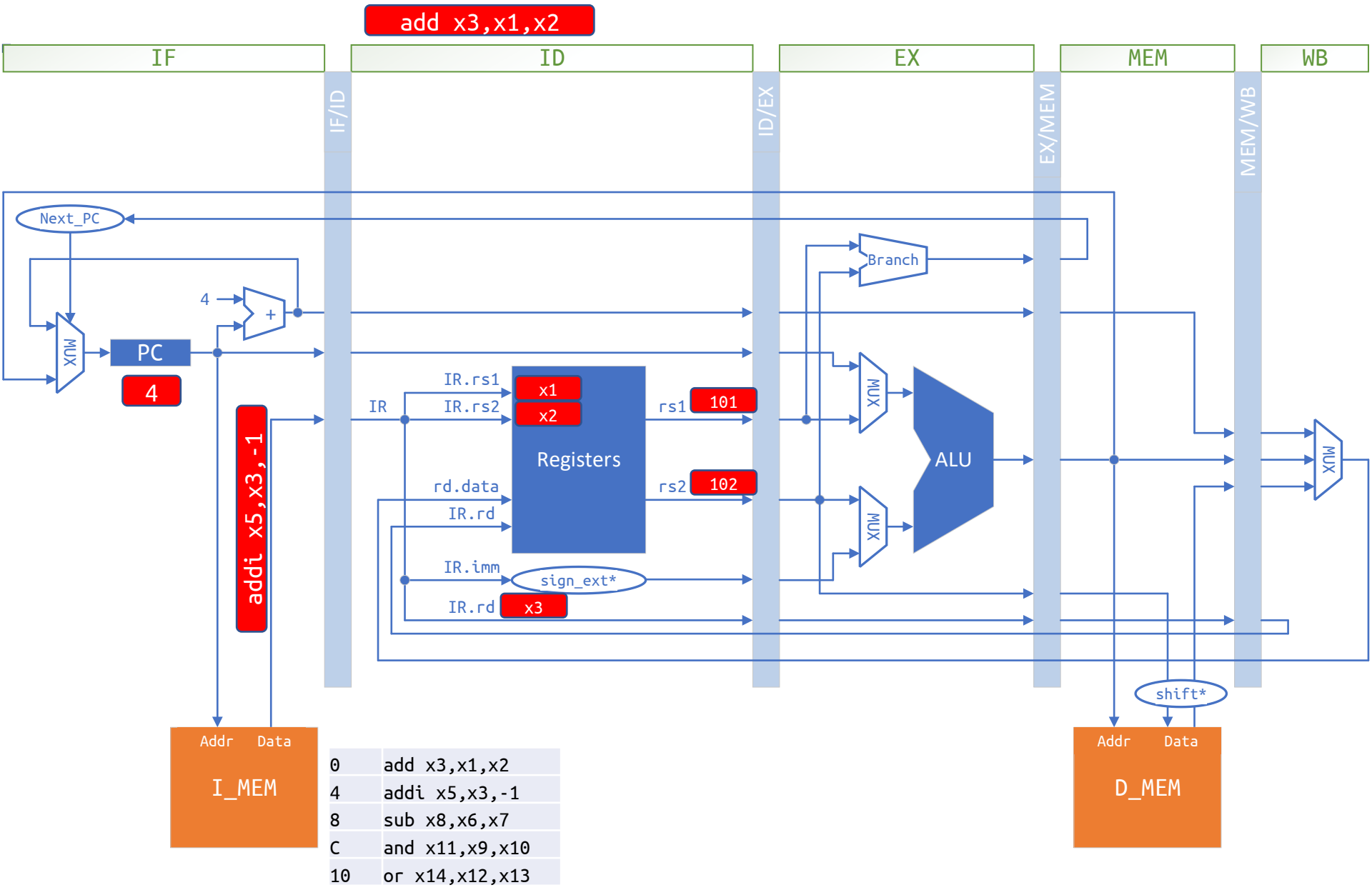
Pogledajmo kako se izvodi ovaj program u našoj mikroarhitekturi…

Period = 1



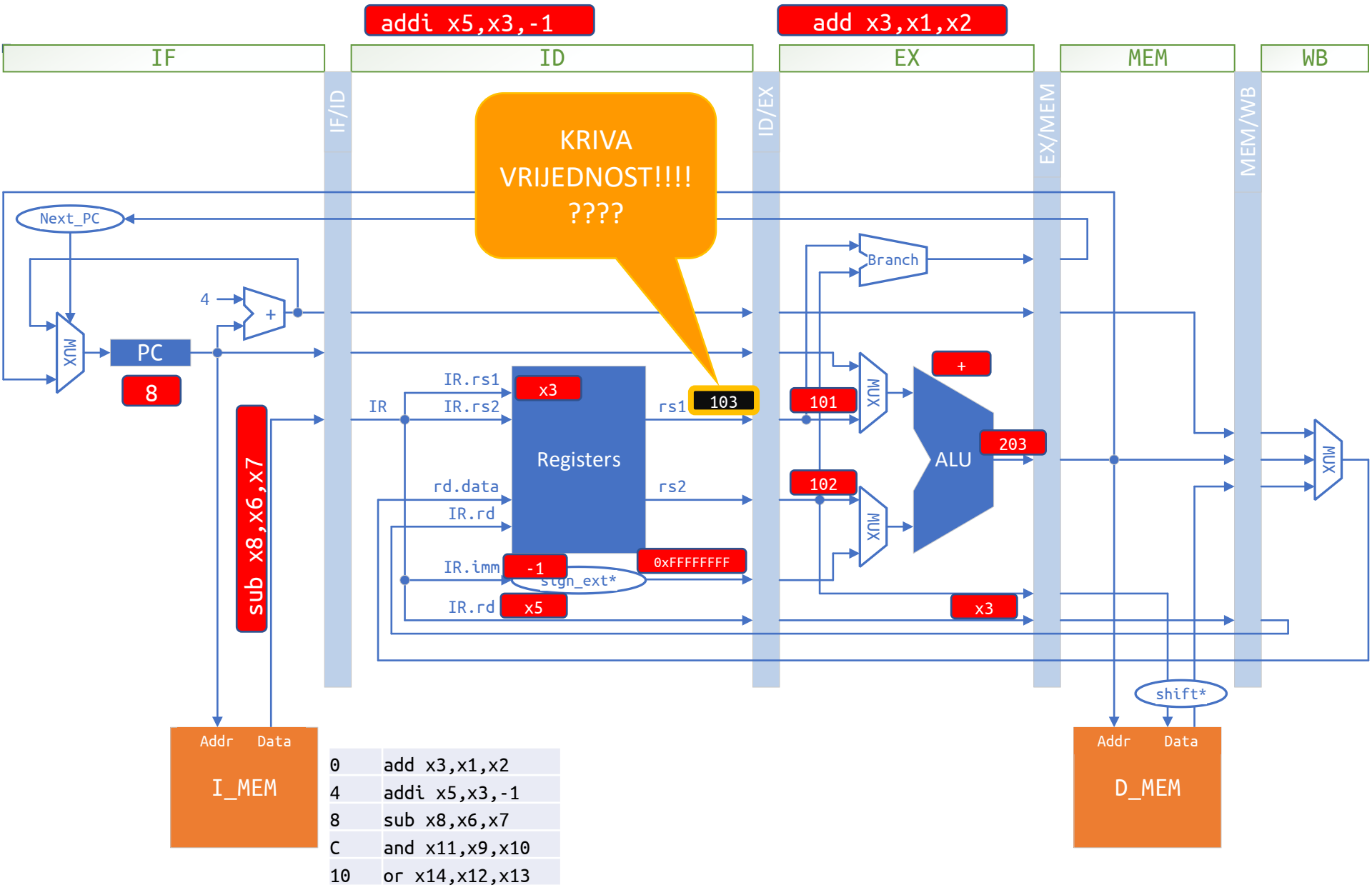
0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 2



0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 3



0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

PODATKOVNI HAZARD

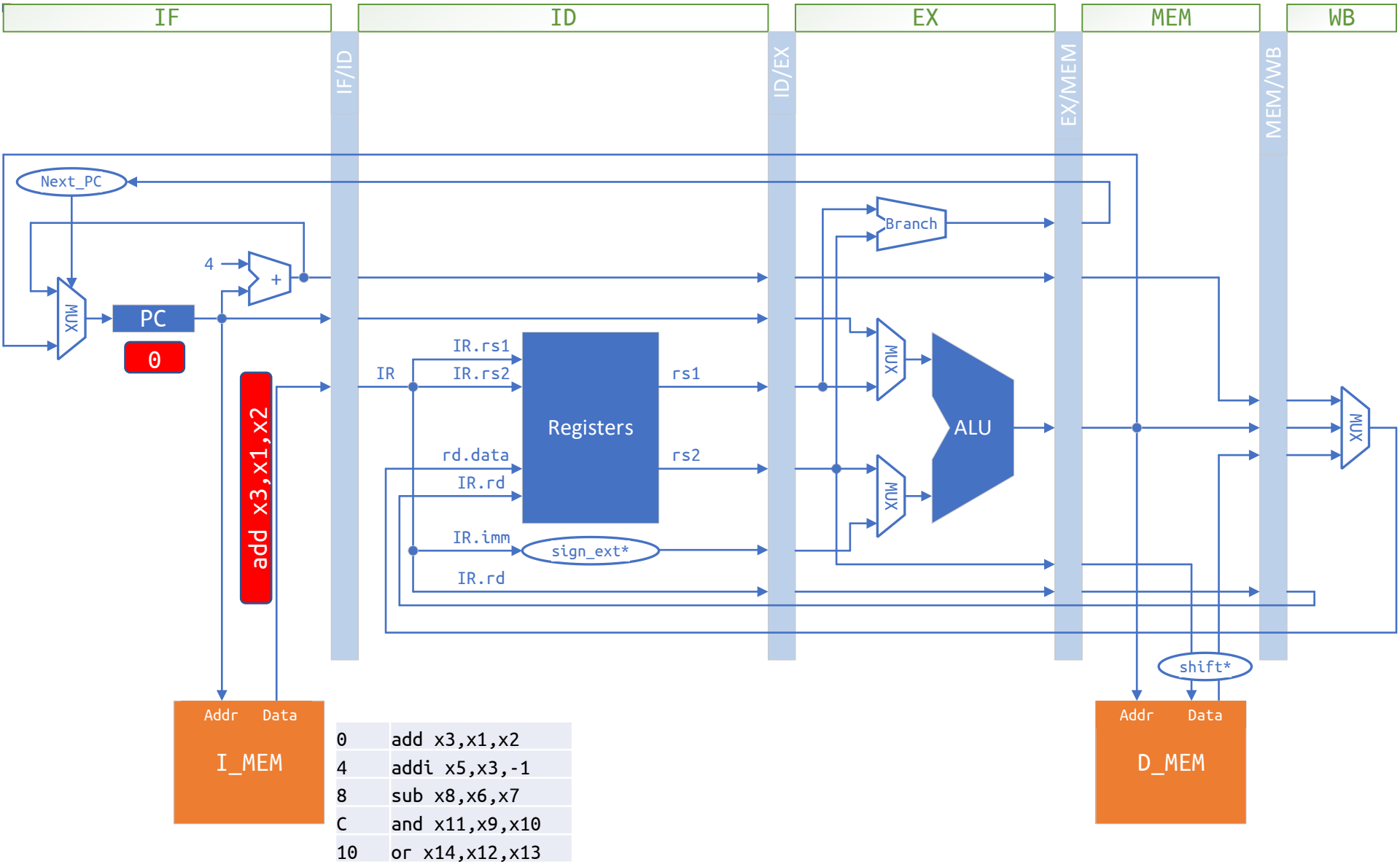
- Kasnija naredba u protočnoj strukturi treba iz registra rezultat ranije naredbe koji još uvijek nije spremljen u registar !!!
- Zašto dolazi do ovog hazarda:

  WB je na kraju protočne strukture

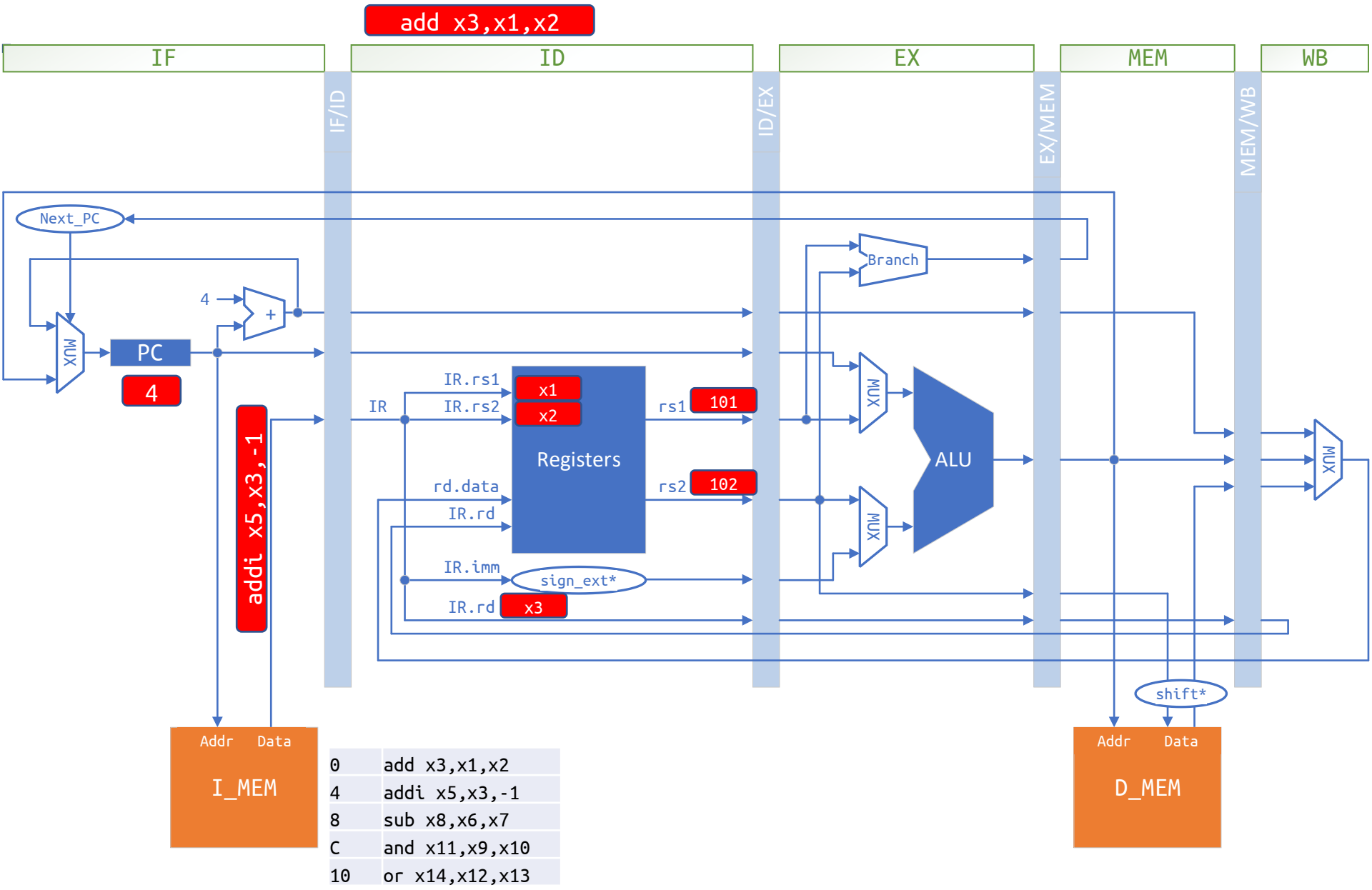
- Kako rješiti ovaj problem:
  - mjehurić

- Upravljačka logika kod dekodiranja svake naredbe provjerava da li je neki od dva operanda naredbe u ID fazi rezultat neke od naredaba koje su u EX ili MEM fazi te ne dozvoljava ulazak naredbe u EX fazu sve dok se ne pribave svi potrebni podatci

Period = 1


IF
ID
EX
MEM
WB
IF/ID
ID/EX
EX/MEM
MEM/WB
Next_PC
4
+
MUX
PC
0
add x3,x1,x2
IR
IR.rs1
IR.rs2
rd.data
IR.rd
IR.imm
IR.rd
Registers
rs1
rs2
sign_ext*
Branch
MUX
MUX
MUX
ALU
MUX
shift*
Addr
Data
I_MEM
Addr
Data
D_MEM
0 add x3,x1,x2
4 addi x5,x3,-1
8 sub x8,x6,x7
C and x11,x9,x10
10 or x14,x12,x13

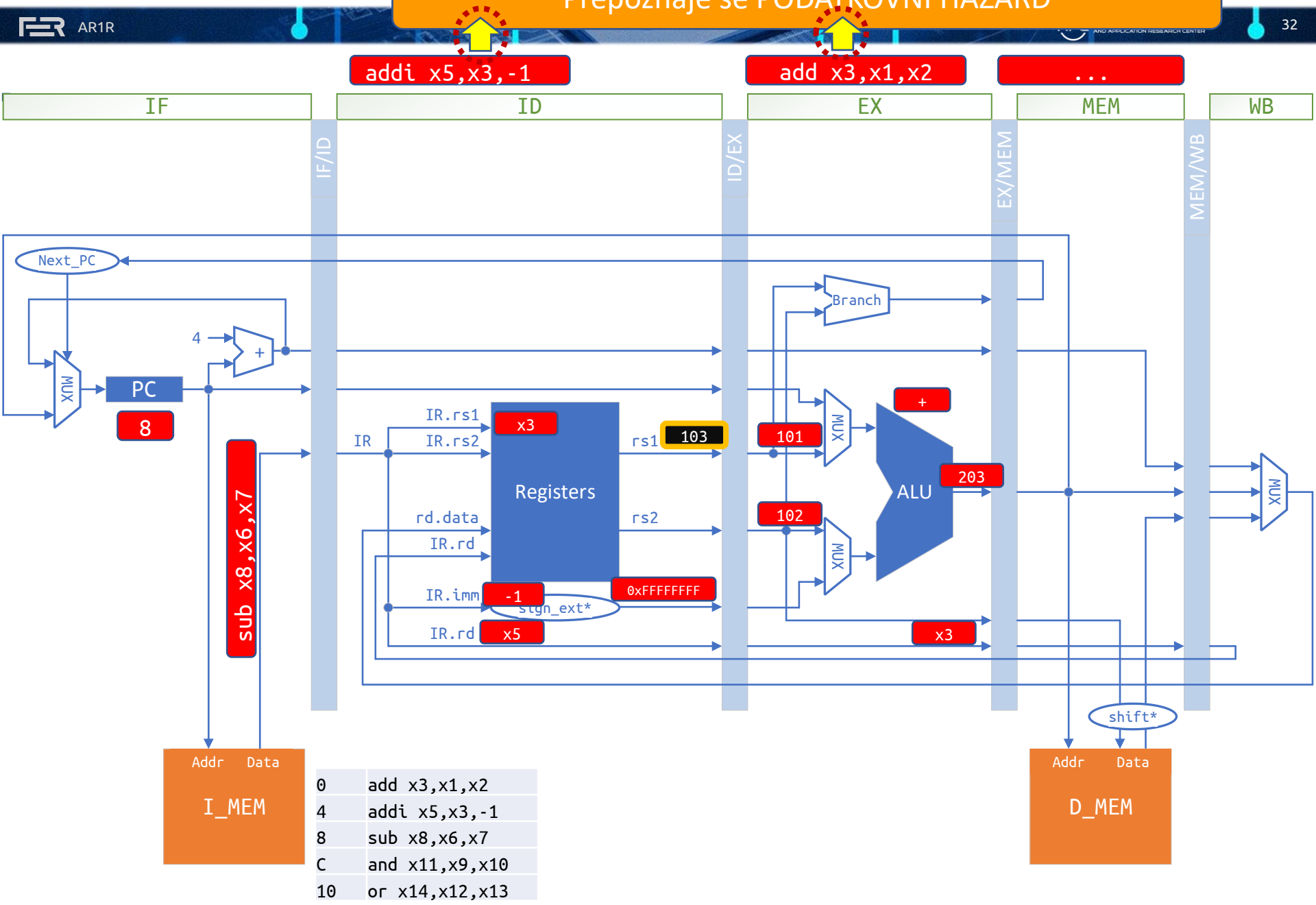
Period = 2



0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 3

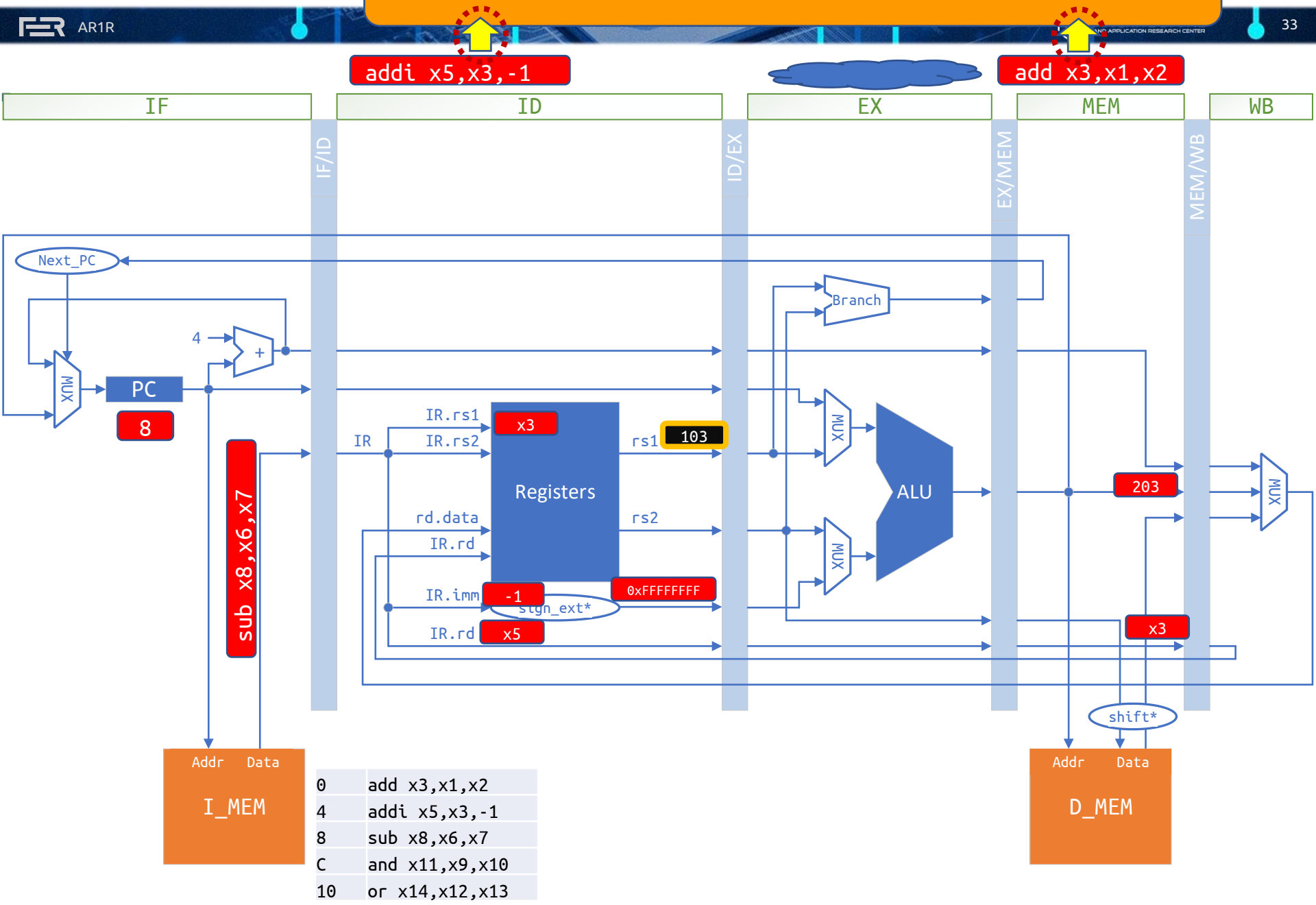
Prepoznaje se PODATKOVNI HAZARD



0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 4

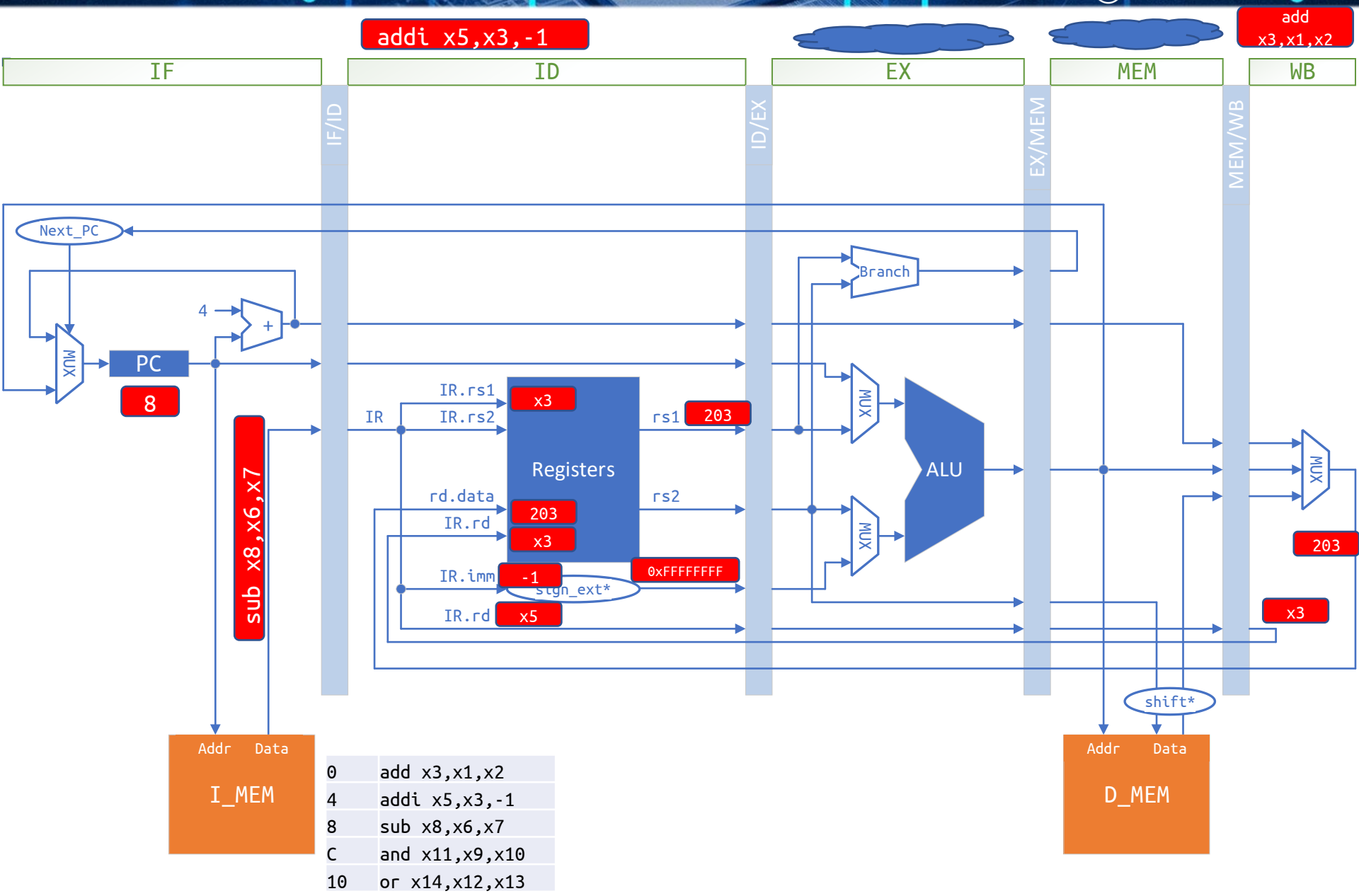
Zbog prethodnog hazarda ubacuje se mjehurić u EX, faze IF i ID se zadržavaju
Ali prepoznaje se ponovo PODATKOVNI HAZARD



0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 5

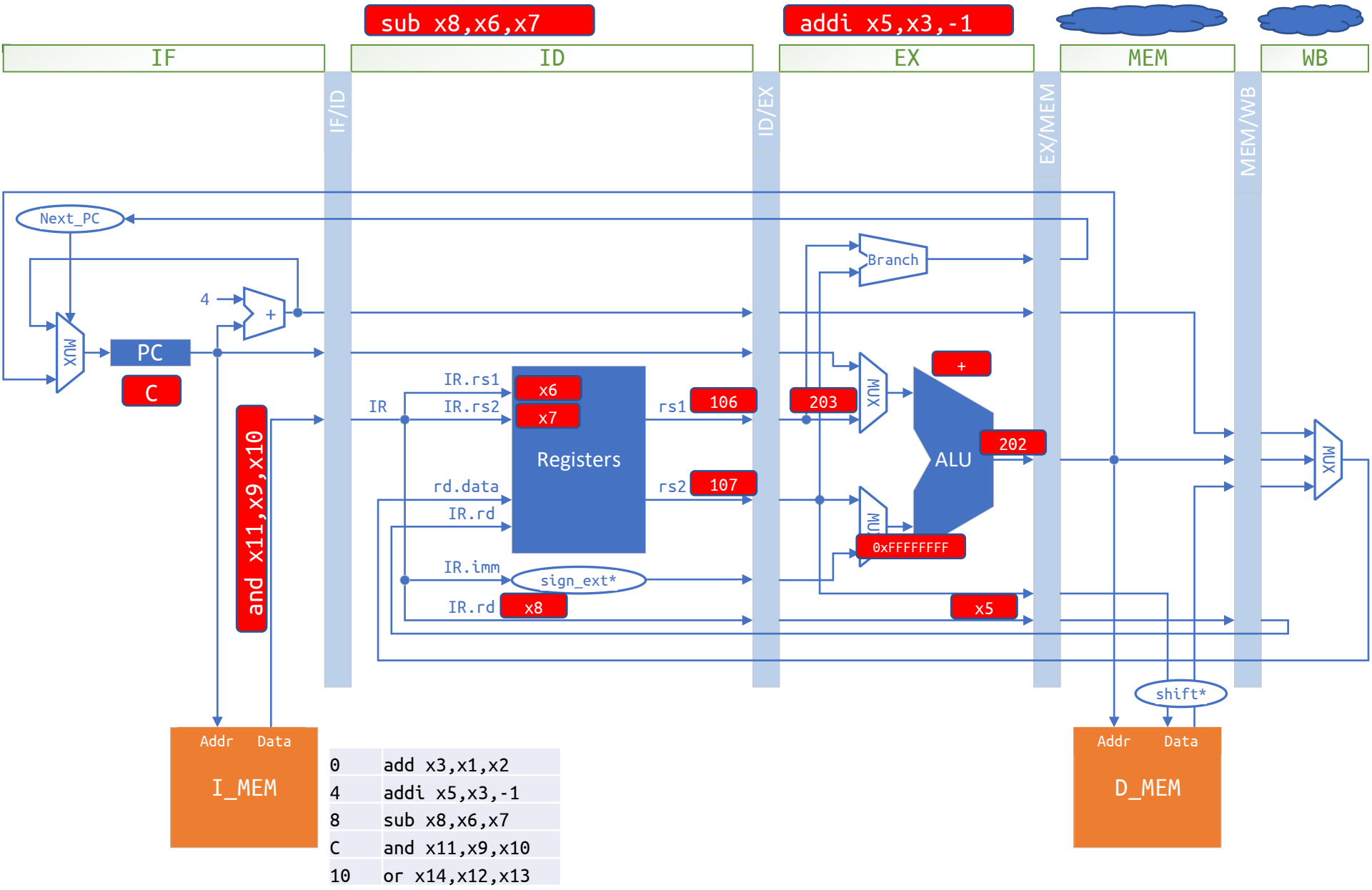
Zbog prethodnog hazarda ubacuje se mjehurić u EX, faze IF i ID se zadržavaju
Nema više PODATKOVNOG HAZARDA



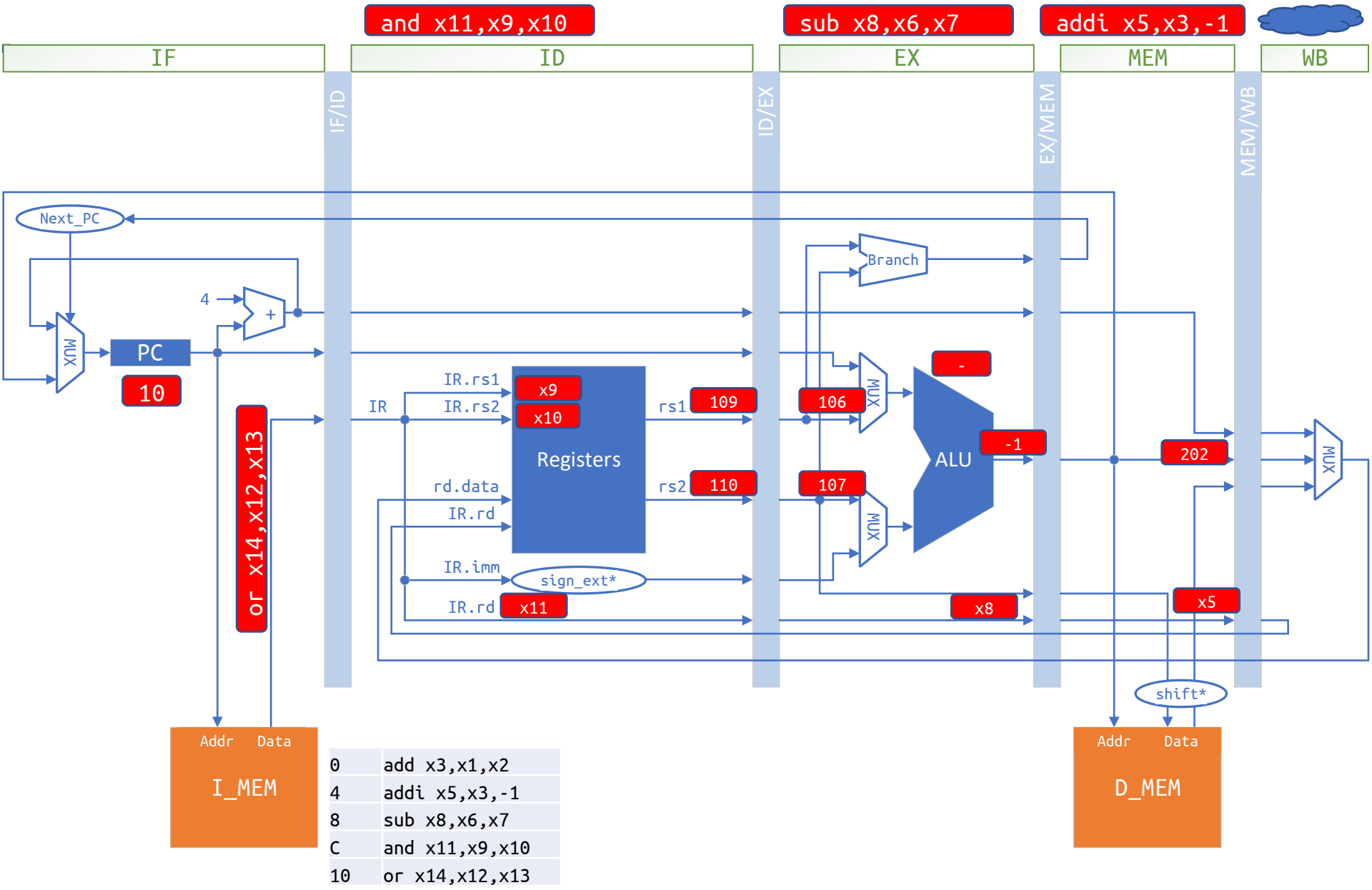
0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 6

Protočna struktura normalno nastavlja s radom....

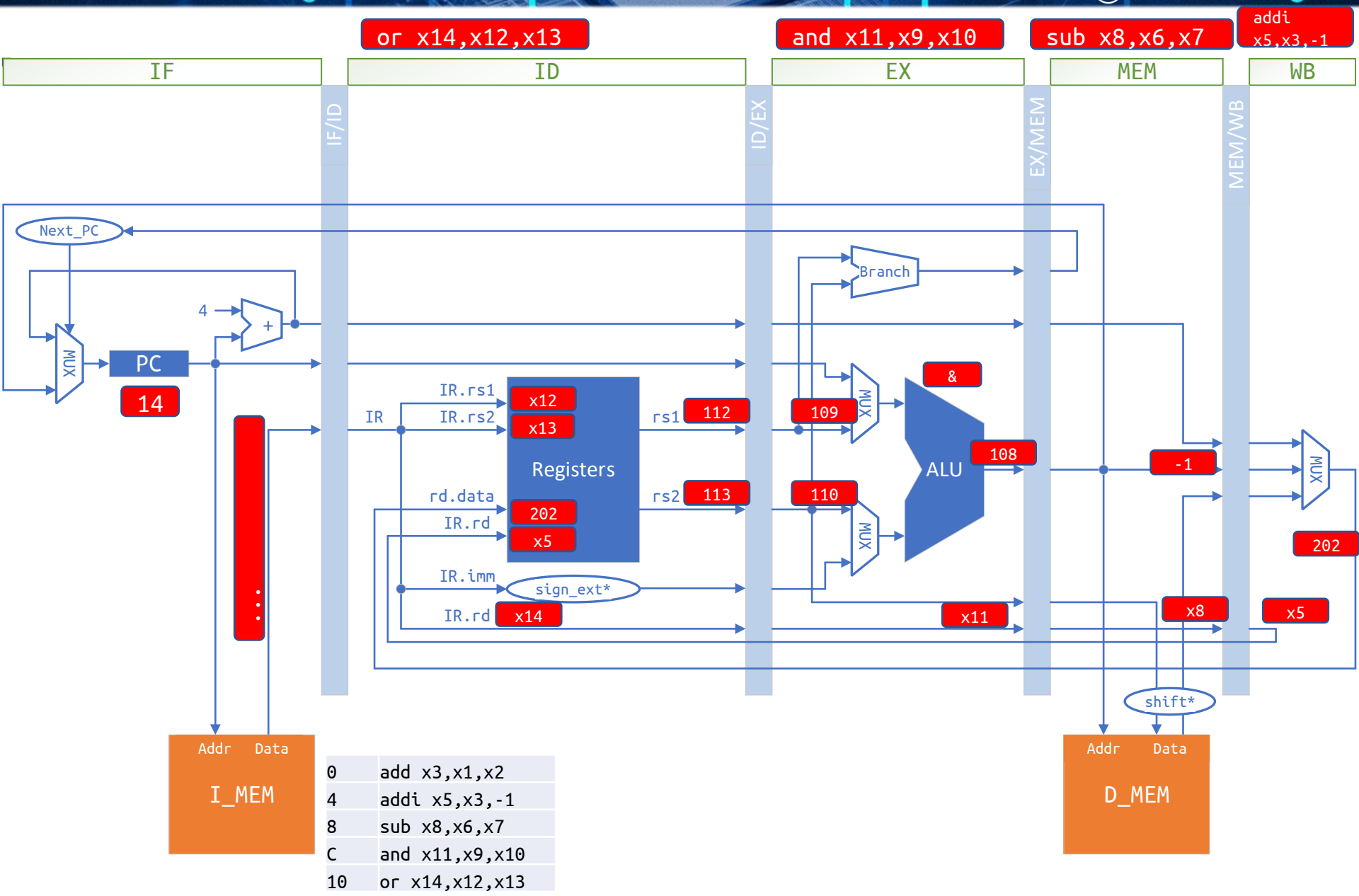


Period = 7



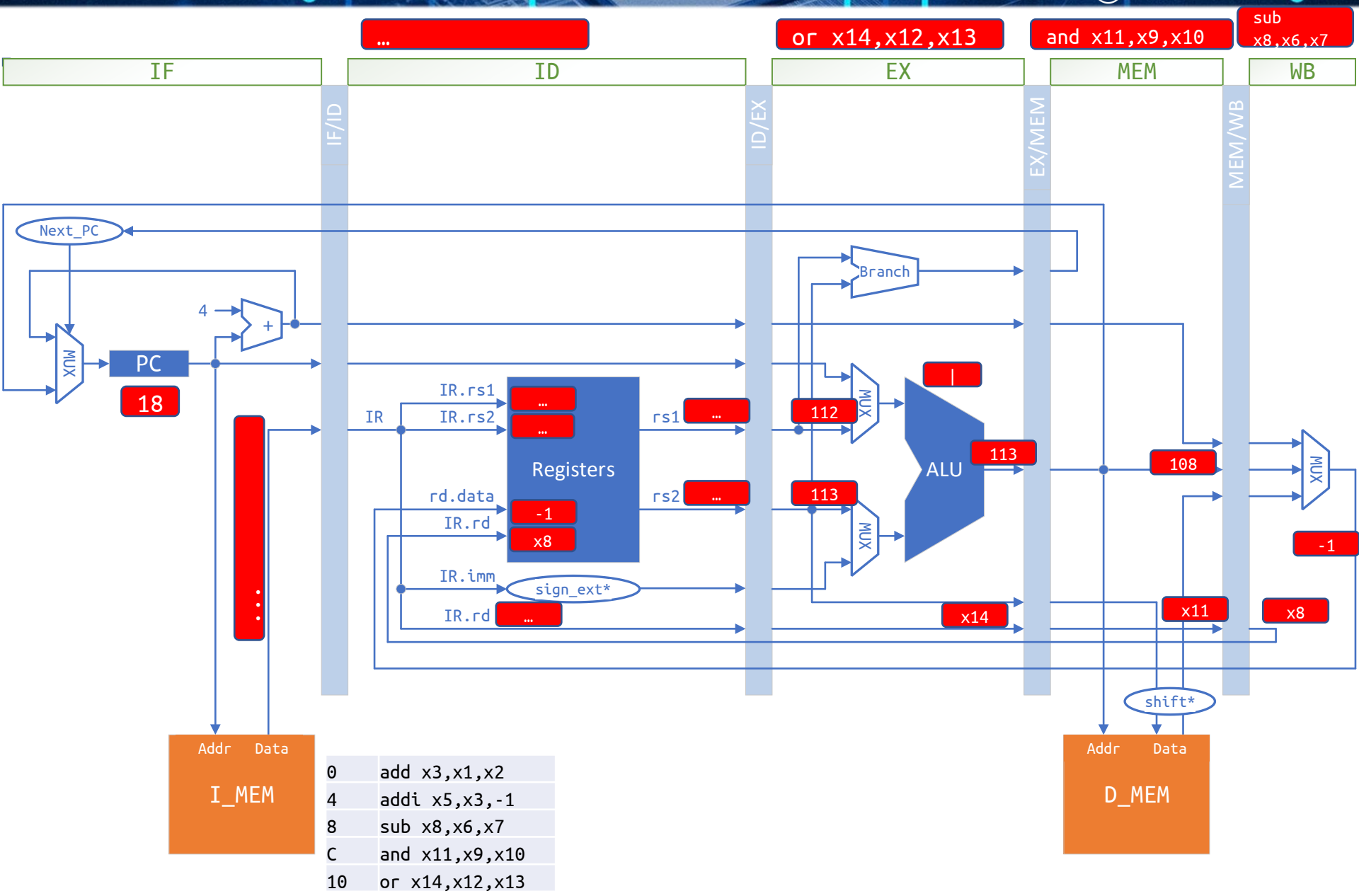
0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 8



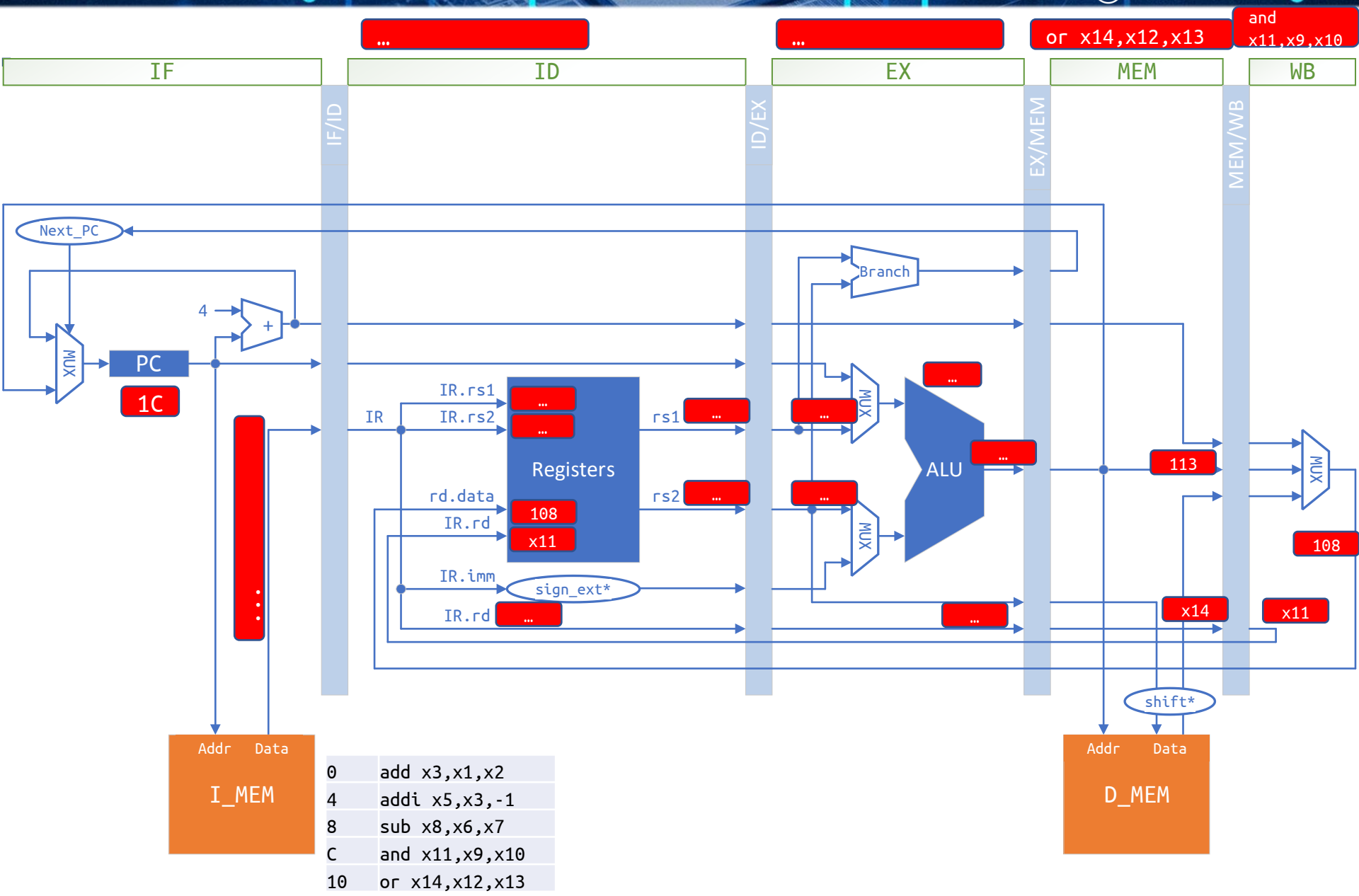
0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 9



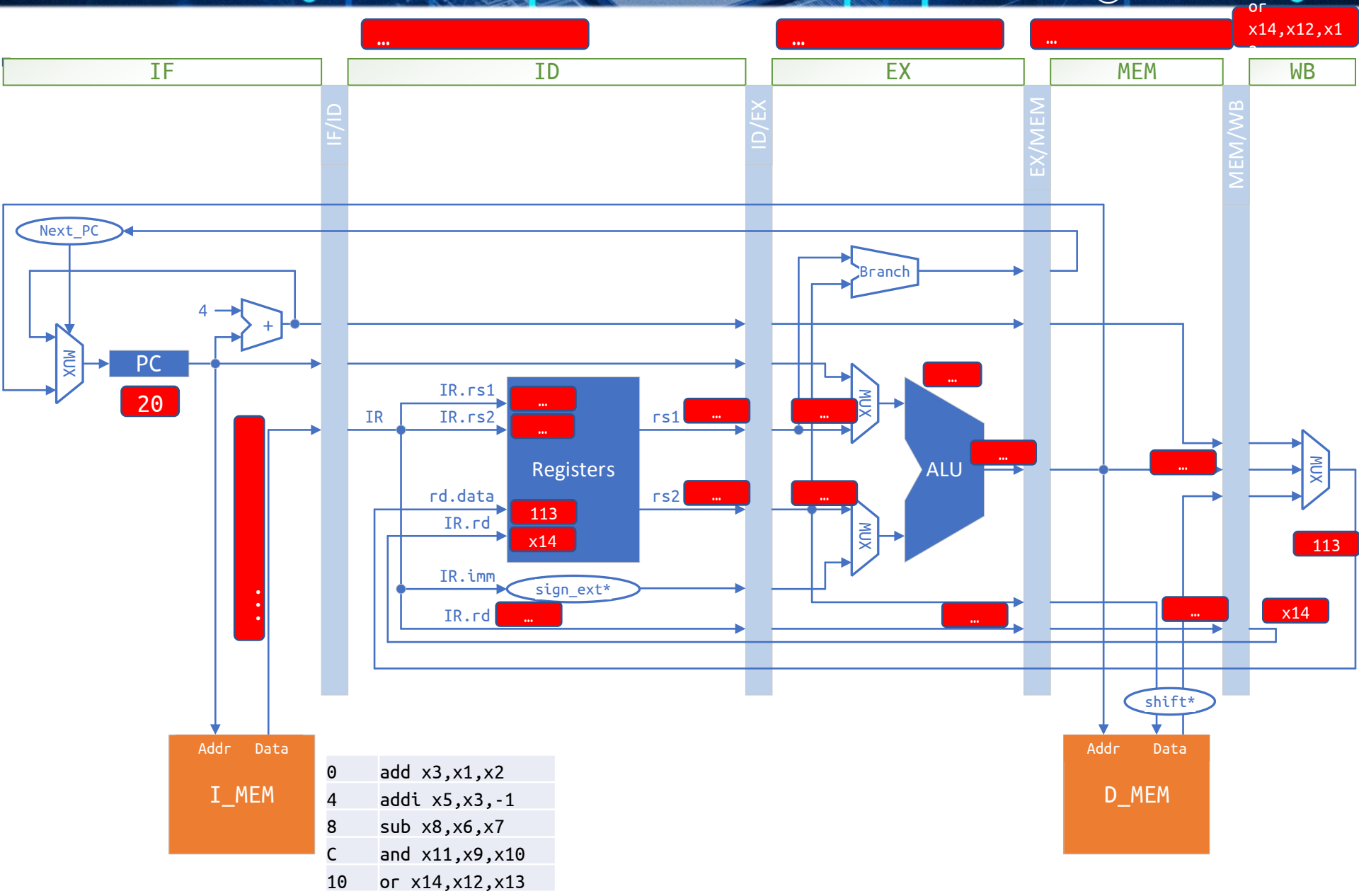

```
0     add x3,x1,x2
4     addi x5,x3,-1
8     sub x8,x6,x7
C     and x11,x9,x10
10    or x14,x12,x13
```

Period = 10



0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 11



0	add x3,x1,x2
4	addi x5,x3,-1
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Primjer niza naredaba #3


```
0       add x3,x1,x2
4       addi x5,x3,-1
8       sub x8,x6,x7
C       and x11,x9,x10
10      or x14,x12,x13
```

Period	IF	ID	EX	MEM	WB
1	(PC=0) add				
2	(PC=4) addi	add			
3	(PC=8) sub	addi	add		
4	(PC=8) sub	addi		add	
5	(PC=8) sub	addi			add
6	(PC=C) and	sub	addi		
7	(PC=10) or	and	sub	addi	
8		or	and	sub	addi
9			or	and	sub
10				or	and
11					or

Zbog podatkovne ovisnosti (PODATKOVNI HAZARD) upravljačka logika morala je ubaciti DVA mjehurića u protočnu strukturu

Ukupno trajanje odsječka:
add: 5 perioda (zbog punjenja protočne strukture)
addi:1 period + 2 perioda zbog mjehurića zbog podatkovne ovisnosti o naredbi add
sub+and+or: svaka po 1 period
UKUPNO = 11

Za vježbu

Provjerite koliko perioda traje svaki od ovih programskih odsječaka:

```
0       add x3,x1,x2
4       lw x3, 0(x1)
8       sub x8,x6,x7
C       and x11,x9,x10
10      or x14,x12,x13
```

```
0       add x3,x1,x2
4       lw x1, 0(x3)
8       sub x8,x6,x7
C       and x11,x9,x10
10      or x14,x12,x13
```

Za vježbu

Provjerite koliko perioda traje svaki od ovih programskih odsječaka:

```
0       add x3,x1,x2
4       sw x3, 0(x1)
8       sub x8,x6,x7
C       and x11,x9,x10
10      or x14,x12,x13
```

```
0       add x3,x1,x2
4       sw x1, 0(x3)
8       sub x8,x6,x7
C       and x11,x9,x10
10      or x14,x12,x13
```

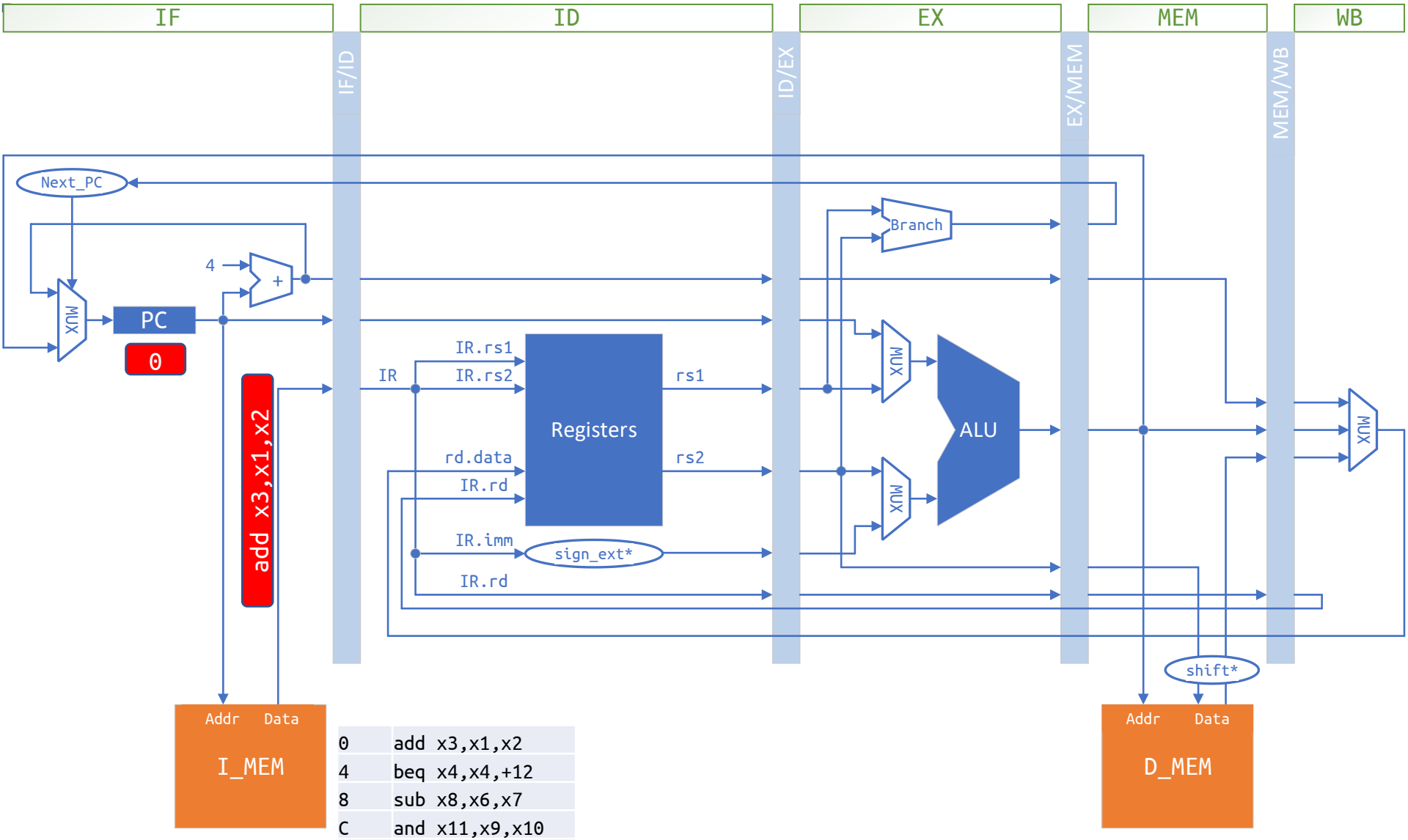
```
0       add x3,x1,x2
4       sub x1,x6,x5
8       sw x1, 0(x3)
C       and x11,x9,x10
10      or x14,x12,x13
```

Primjer niza naredaba #4


```
0        add x3,x1,x2
4        beq x4,x4,L1
8        sub x8,x6,x7
C        and x11,x9,x10
10  L1   or x14,x12,x13
```

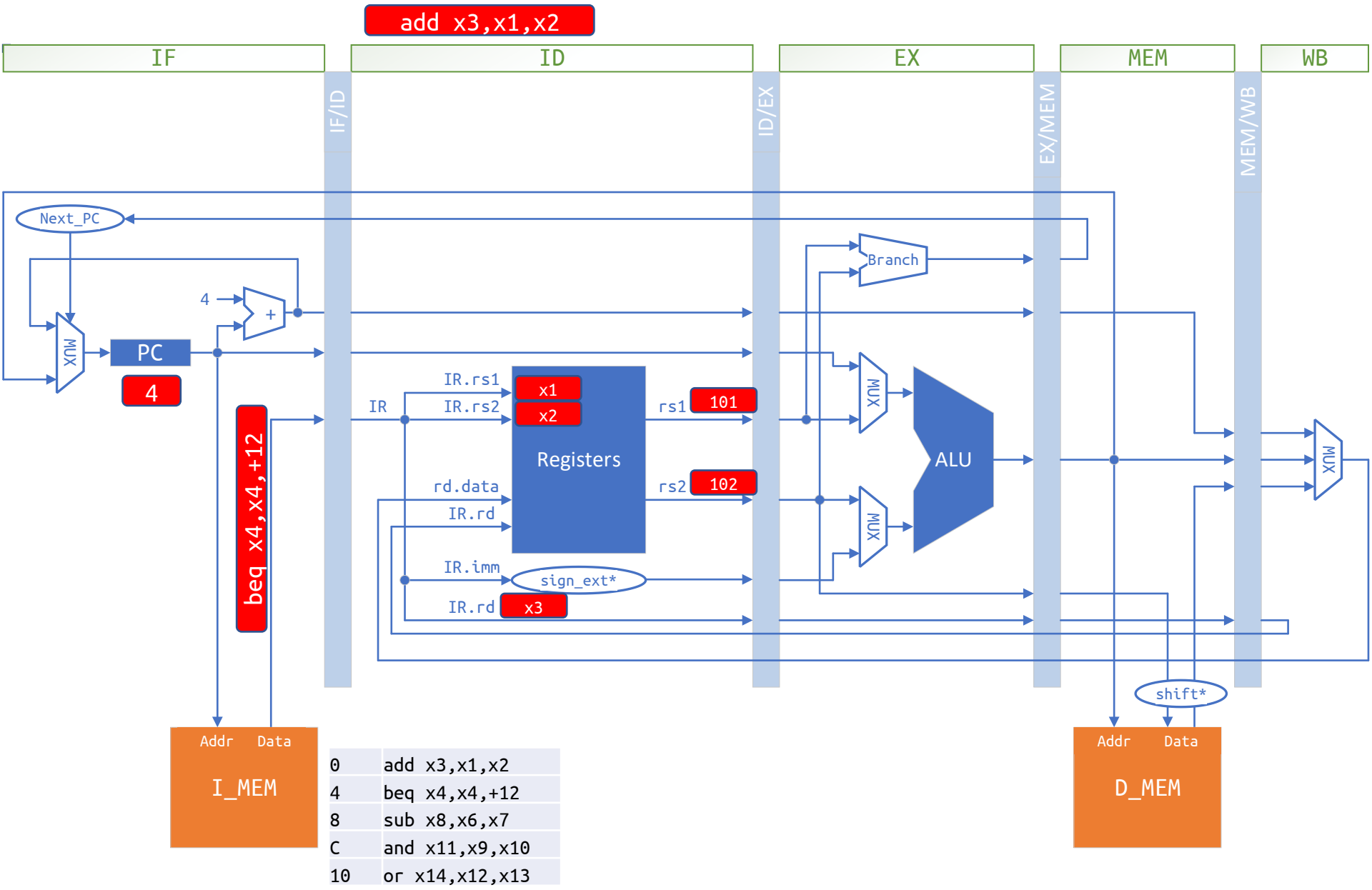
Pogledajmo kako se izvodi ovaj program u našoj mikroarhitekturi…

Period = 1



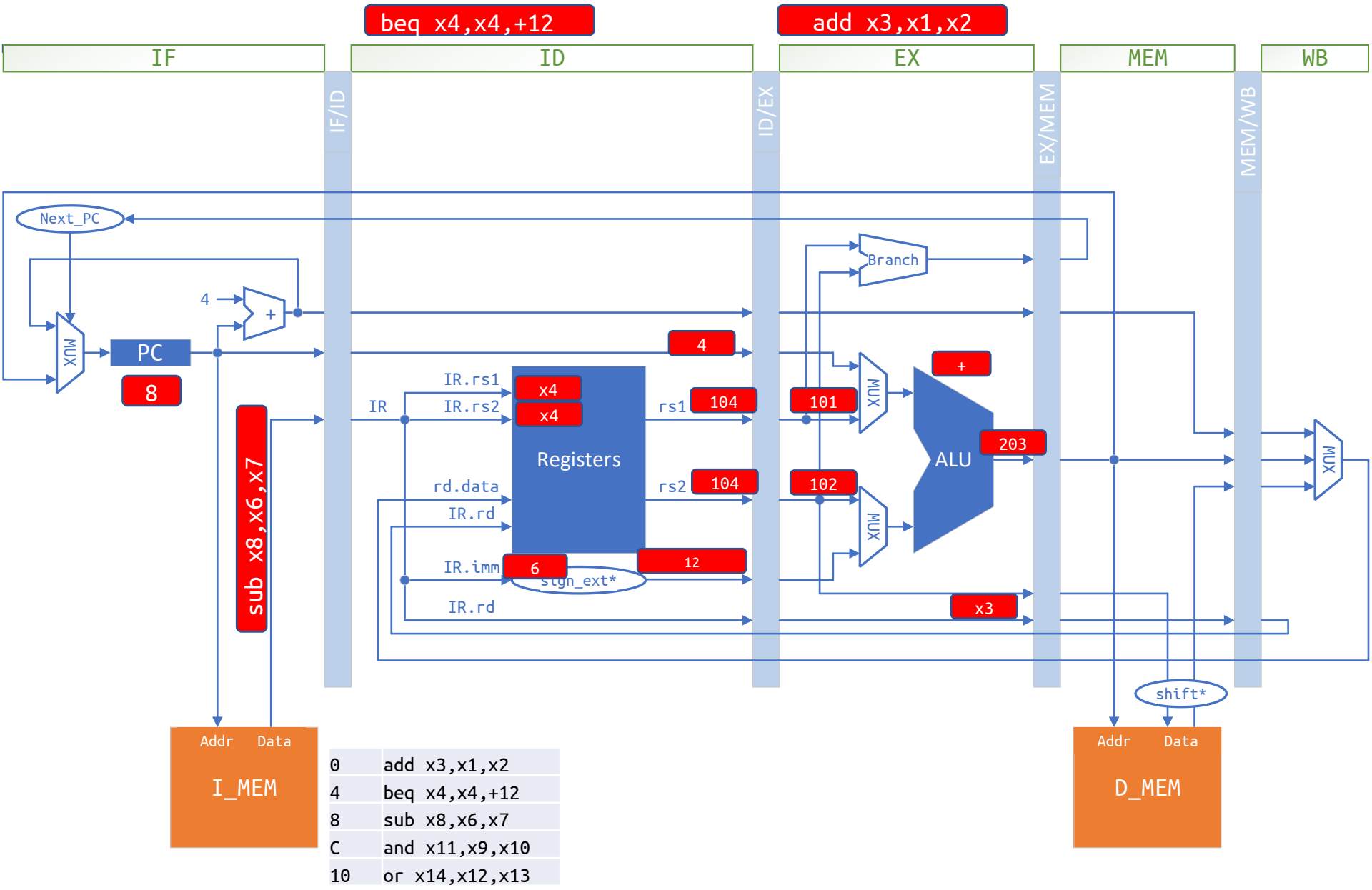
0	add x3,x1,x2
4	beq x4,x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 2



0	add x3,x1,x2
4	beq x4,x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

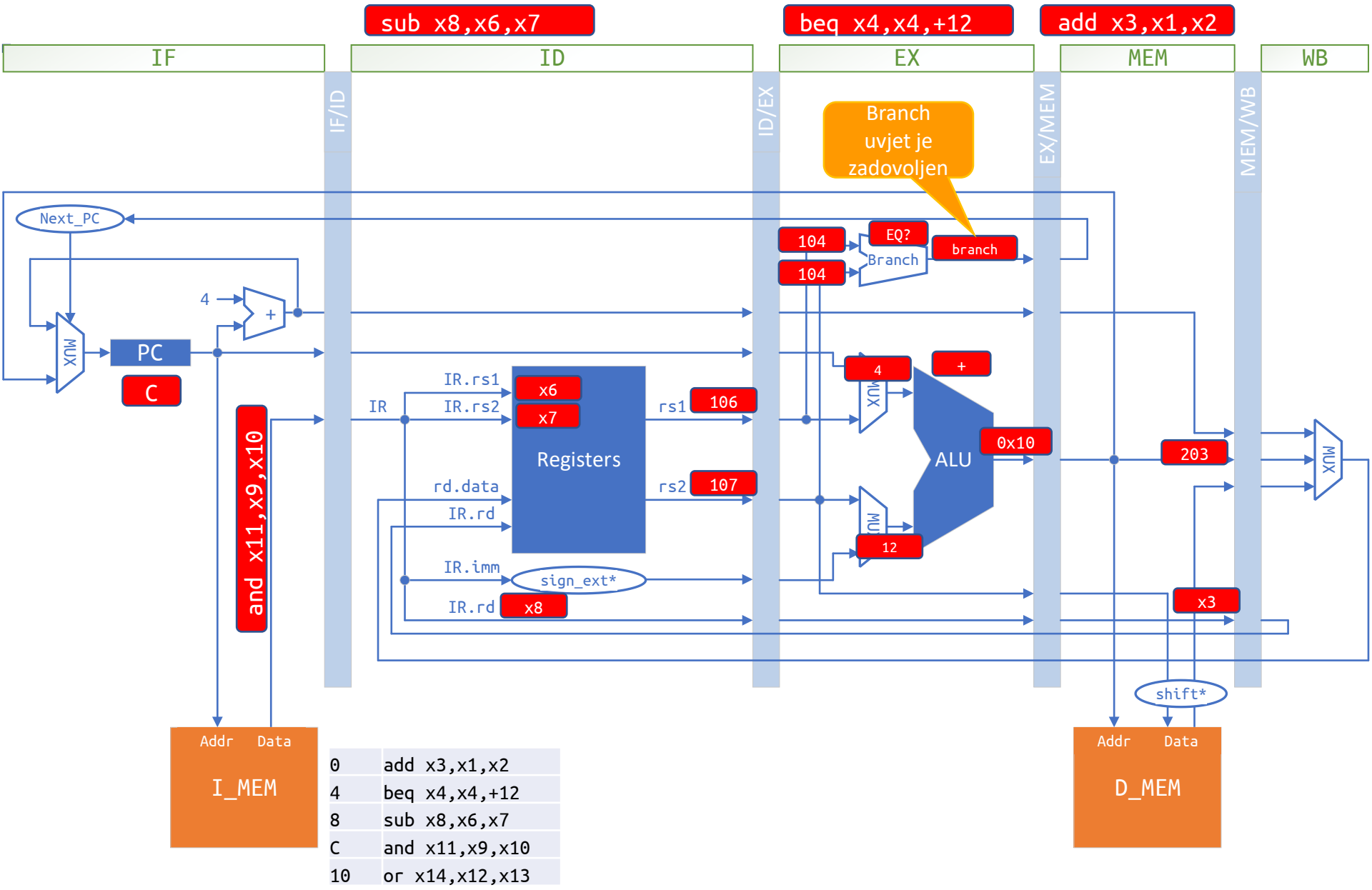
Period = 3



0	add x3,x1,x2
4	beq x4,x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

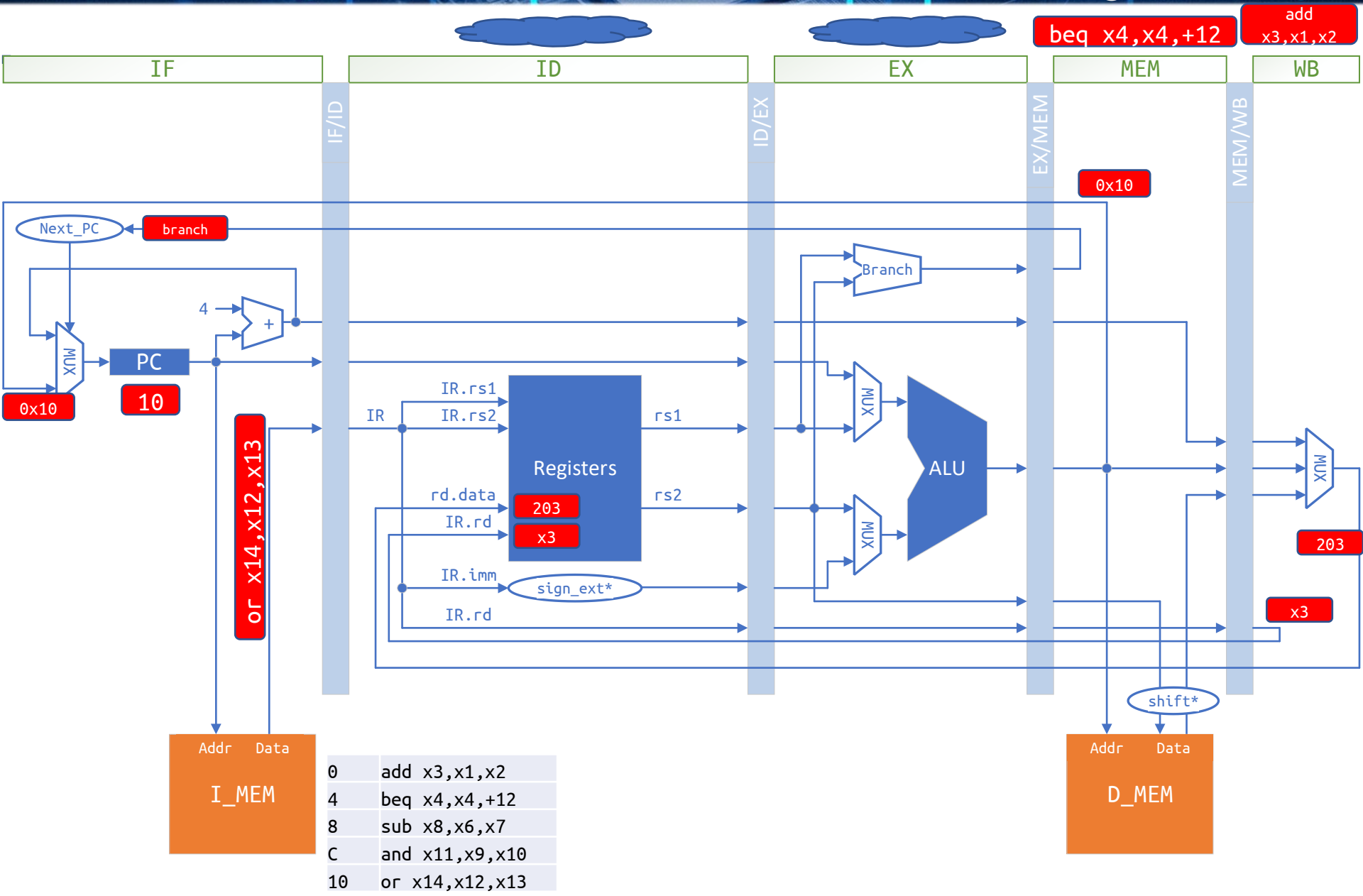
Period = 4

Prepoznaje se UPRAVLJAČKI HAZARD



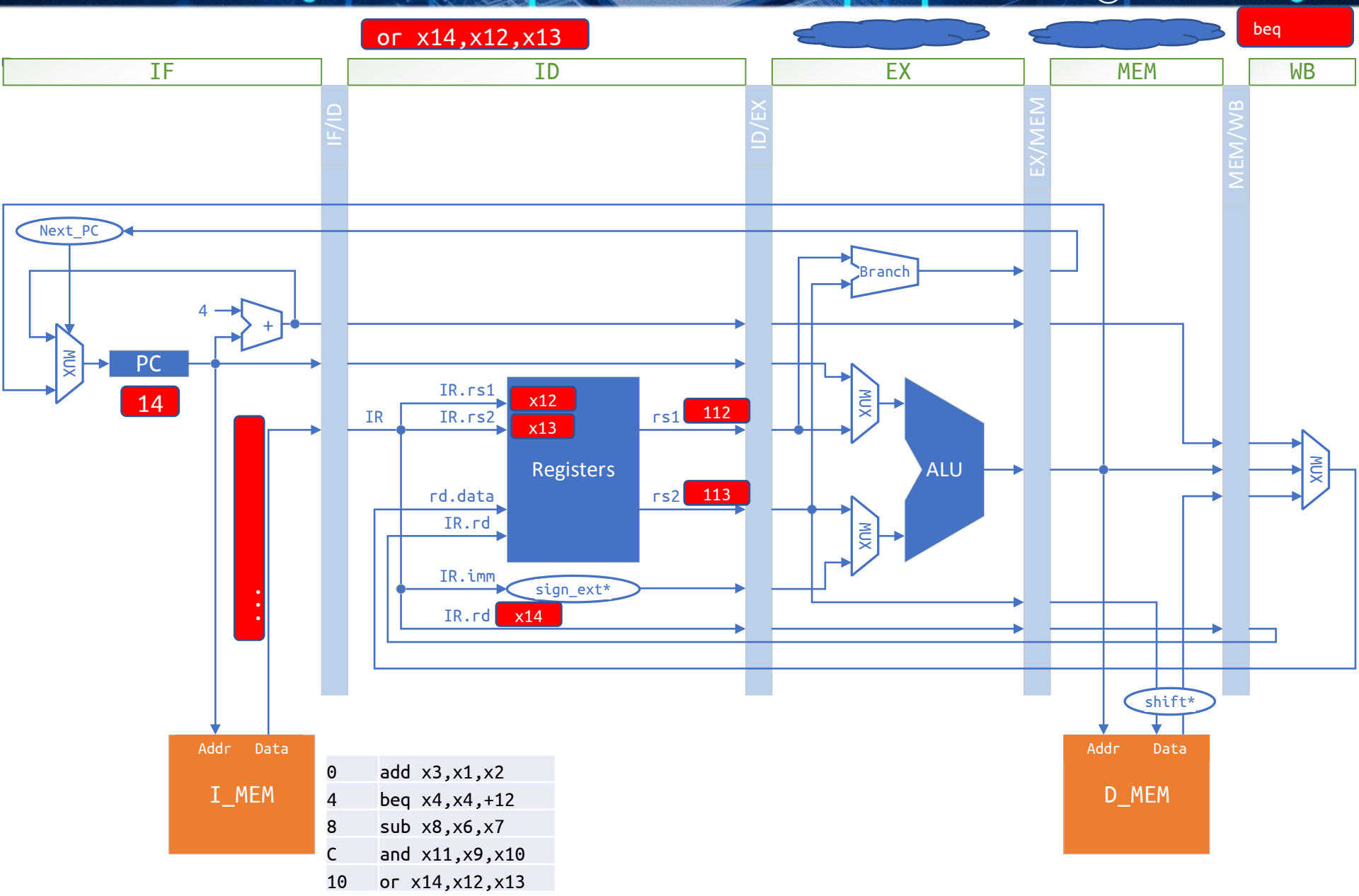

```
0     add x3,x1,x2
4     beq x4,x4,+12
8     sub x8,x6,x7
C     and x11,x9,x10
10    or x14,x12,x13
```

Period = 5



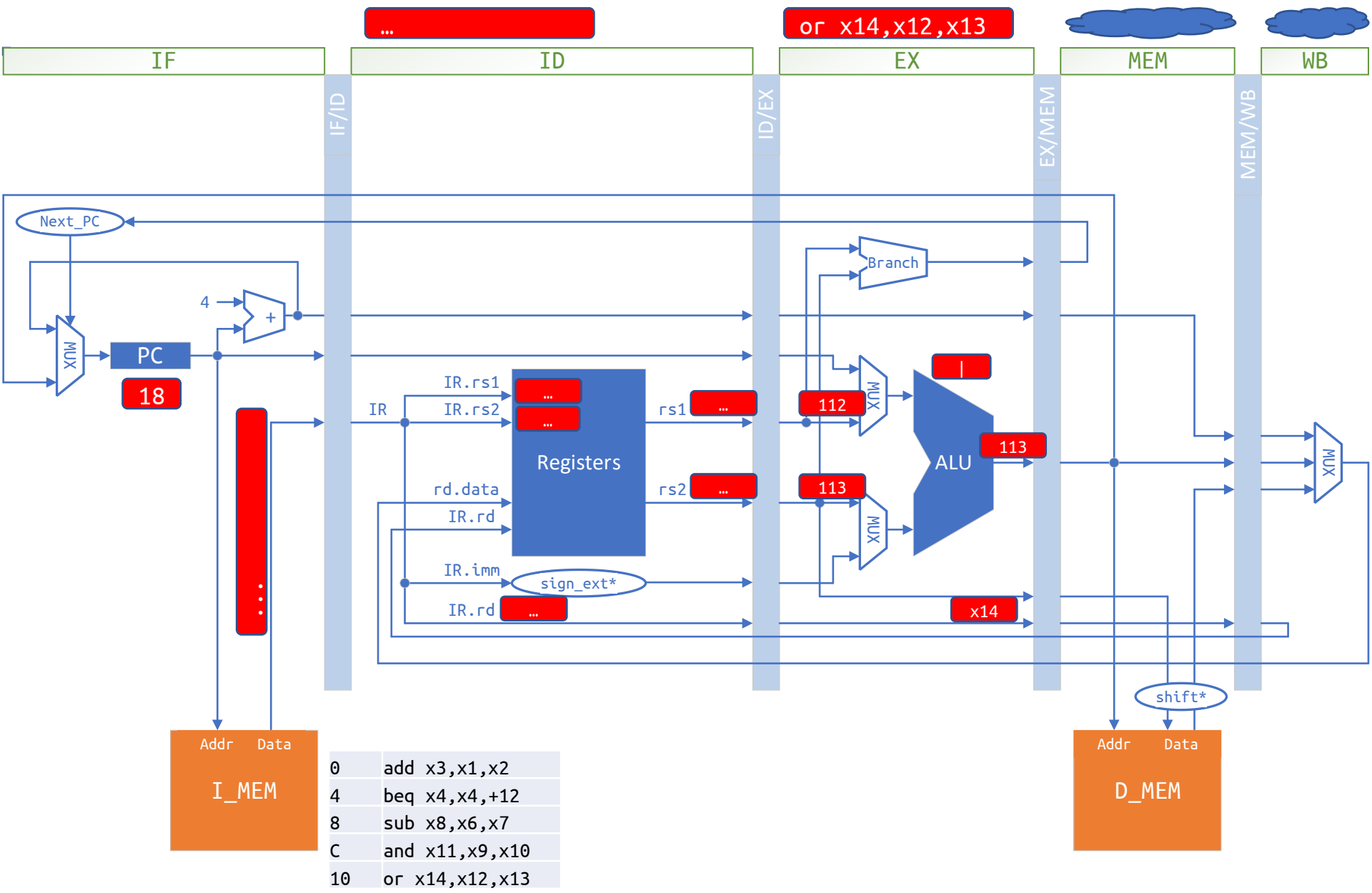
0	add x3,x1,x2
4	beq x4,x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 6



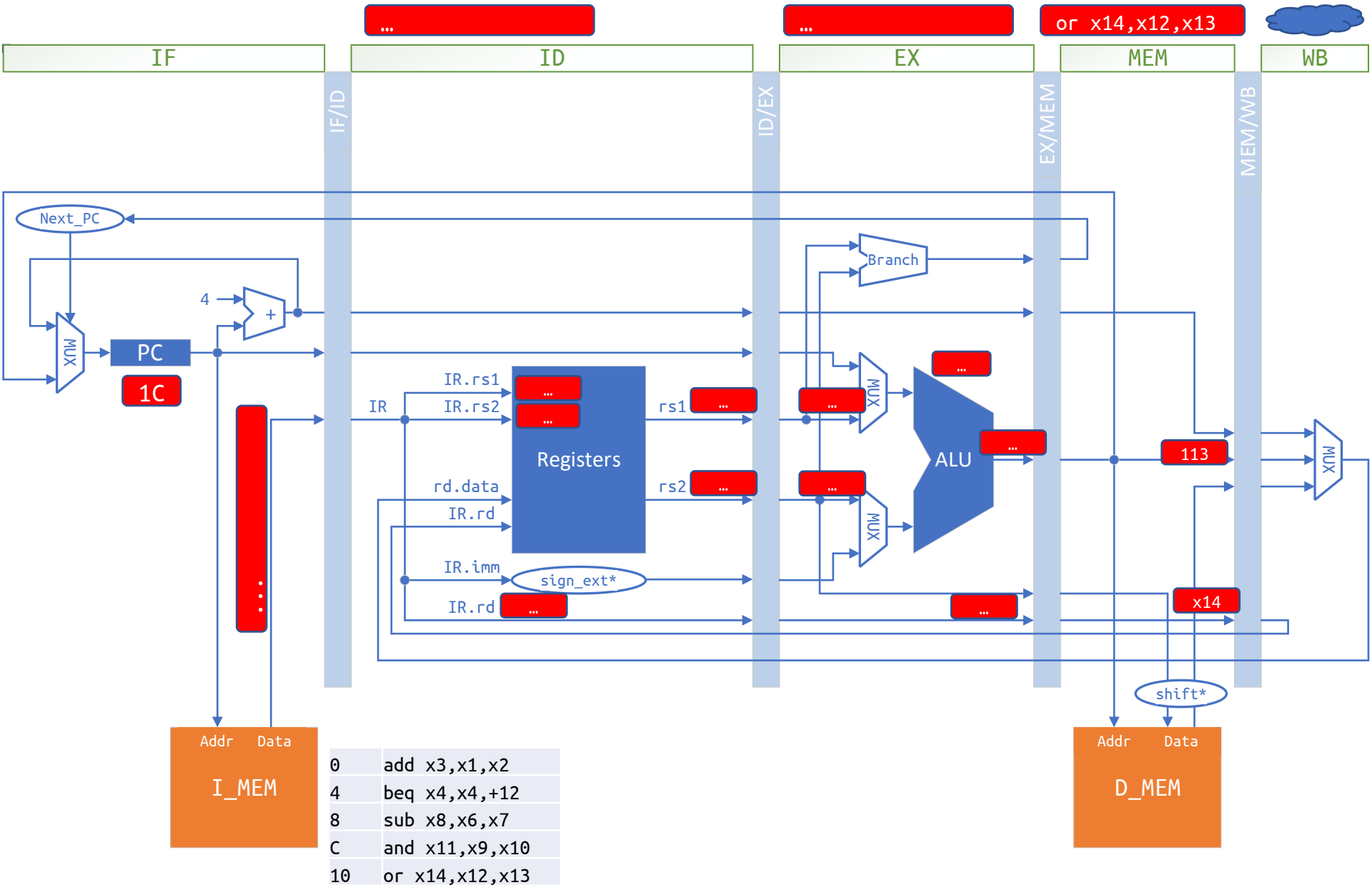
0	add x3,x1,x2
4	beq x4,x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 7



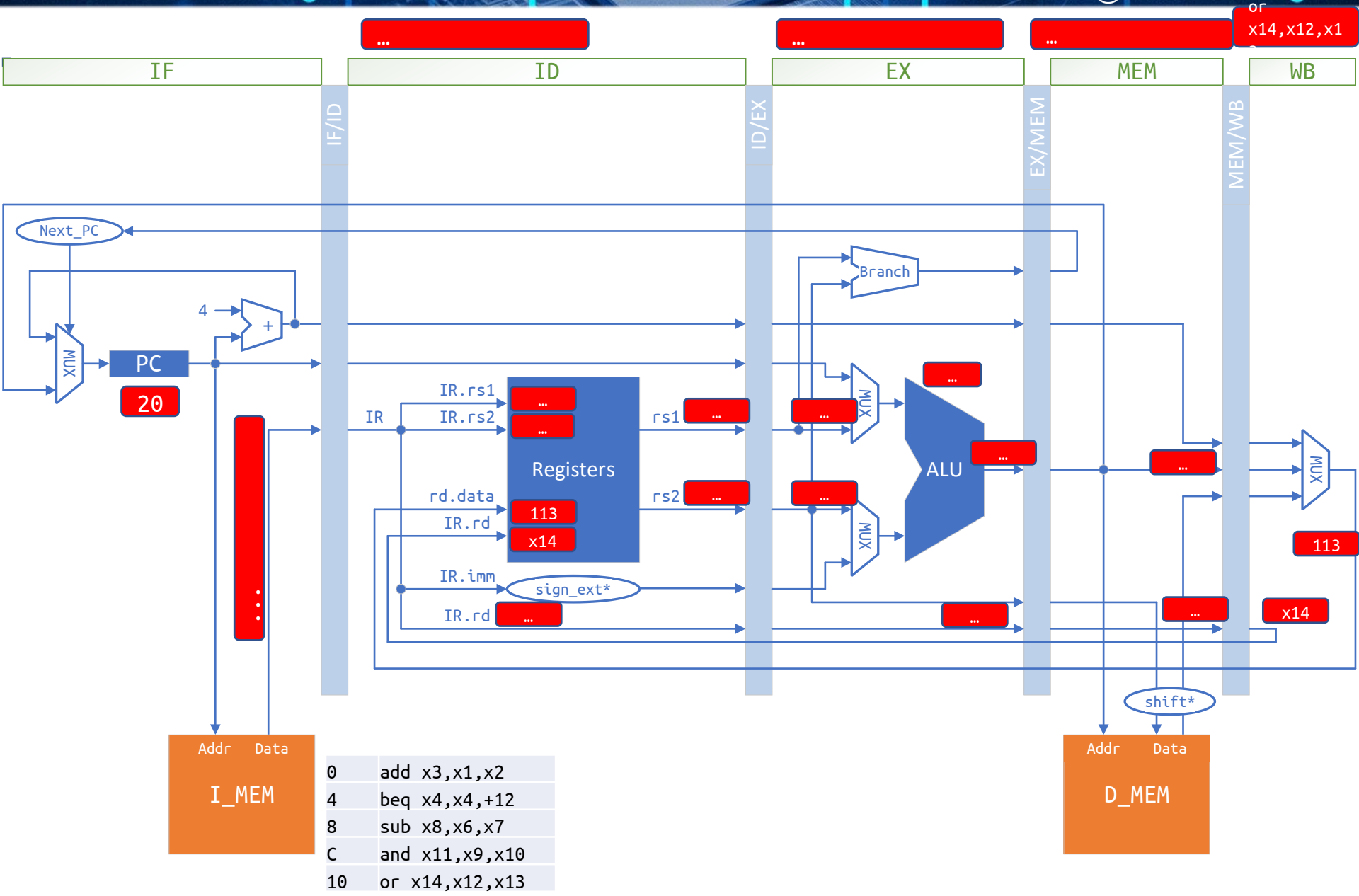
0	add x3,x1,x2
4	beq x4,x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 8



0	add x3,x1,x2
4	beq x4,x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 9



0	add x3,x1,x2
4	beq x4,x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Primjer niza naredaba #4


```
0        add x3,x1,x2
4        beq x4,x4,L1
8        sub x8,x6,x7
C        and x11,x9,x10
10  L1   or x14,x12,x13
```

Period	IF	ID	EX	MEM	WB
1	(PC=0) add				
2	(PC=4) beq	add			
3	(PC=8) sub	beq	add		
4	(PC=C) and	sub	beq	add	
5	(PC=10) or			beq	add
6		or			beq
7			or		
8				or	
9					or

Zbog ispunjenog uvjeta grananja (UPRAVLJAČKI HAZARD) upravljačka logika mora ubaciti DVA mjehurića u protočnu strukturu

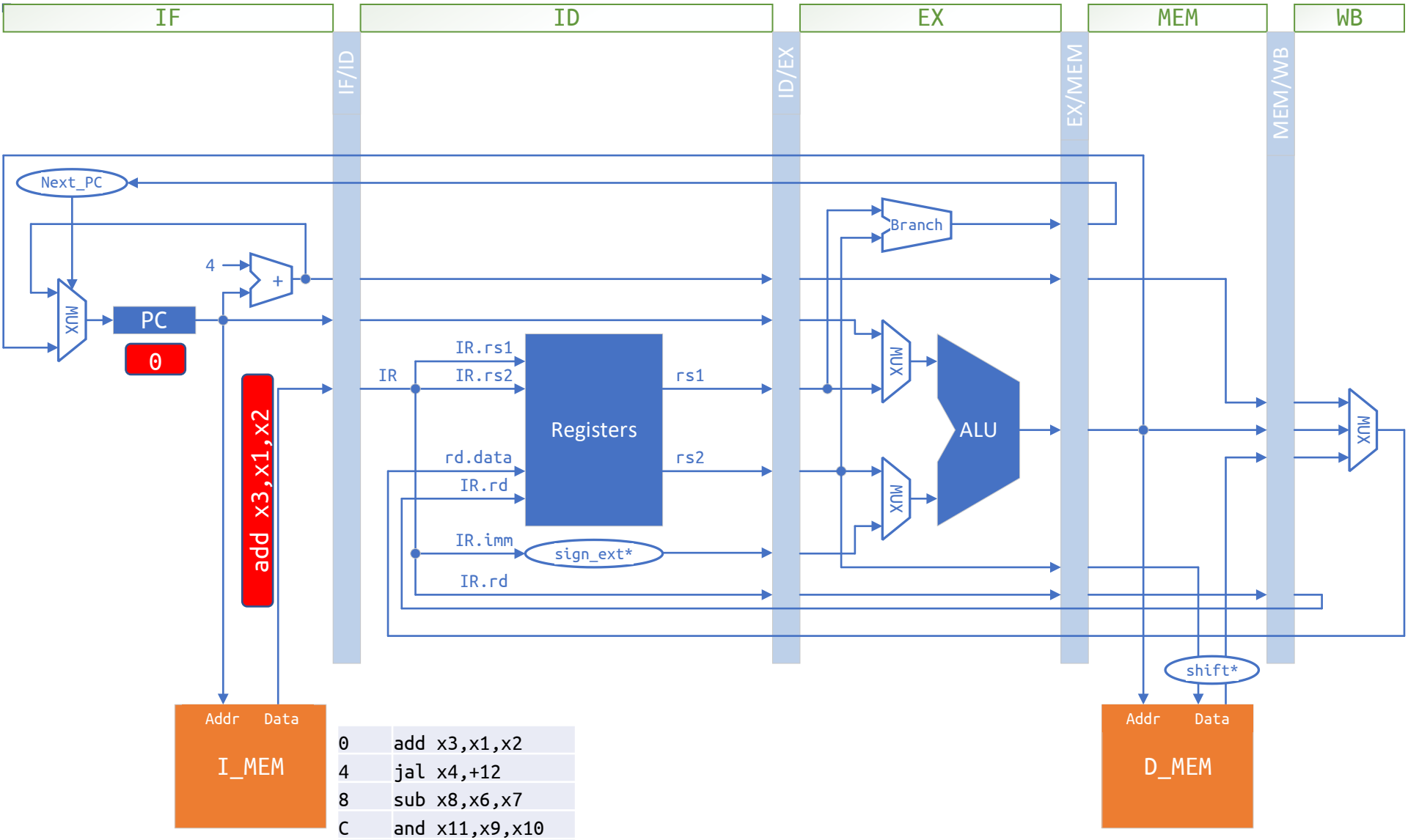
Ukupno trajanje odsječka:
add: 5 perioda (zbog punjenja protočne strukture)
beq: 1 period + 2 perioda zbog mjehurića ako je uvjet zadovoljen
or: 1 period
UKUPNO = 9

Primjer niza naredaba #5


```
0        add x3,x1,x2
4        jal x4, L1
8        sub x8,x6,x7
C        and x11,x9,x10
10  L1   or x14,x12,x13
```

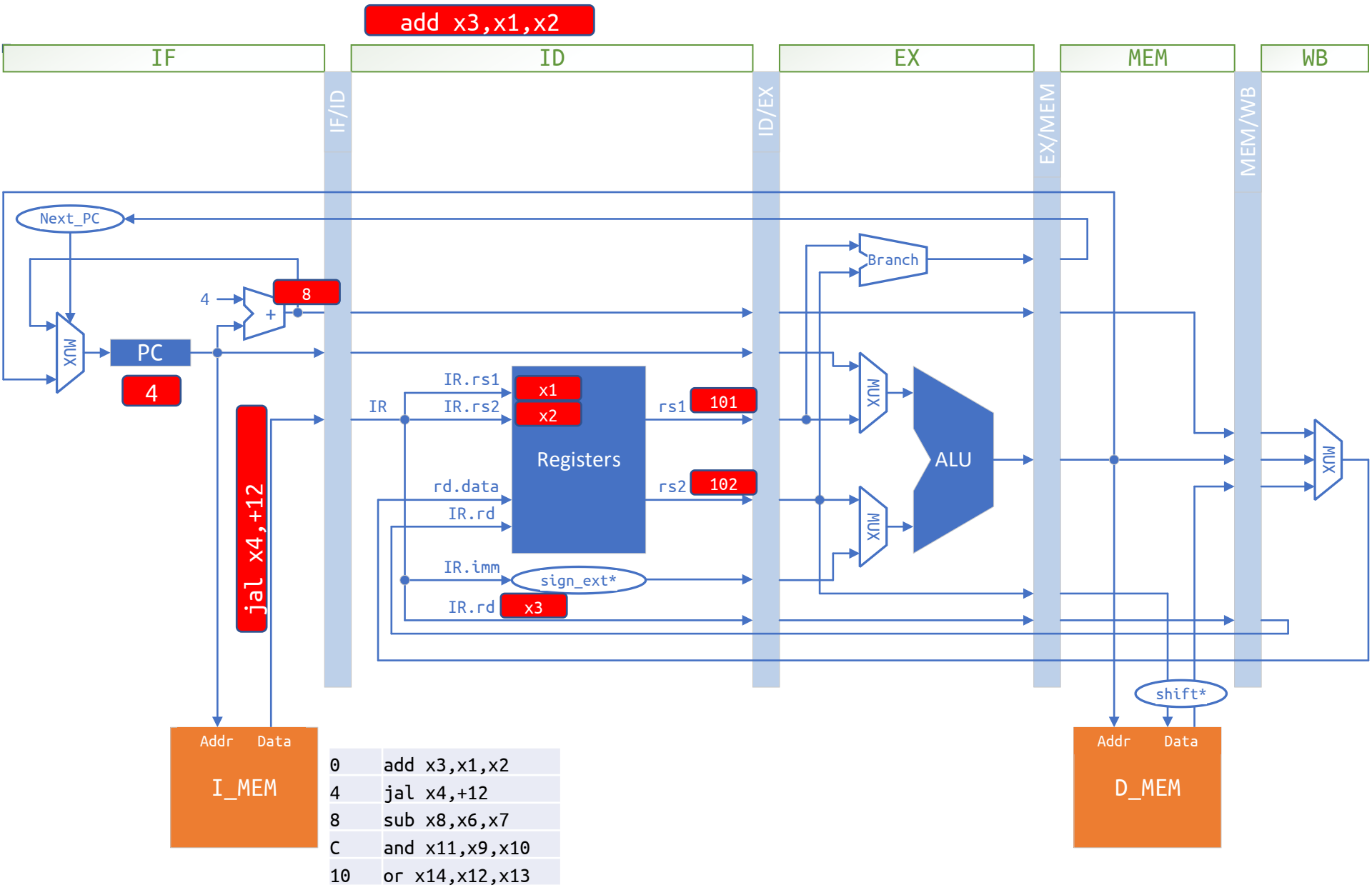
Pogledajmo kako se izvodi ovaj program u našoj mikroarhitekturi…

Period = 1



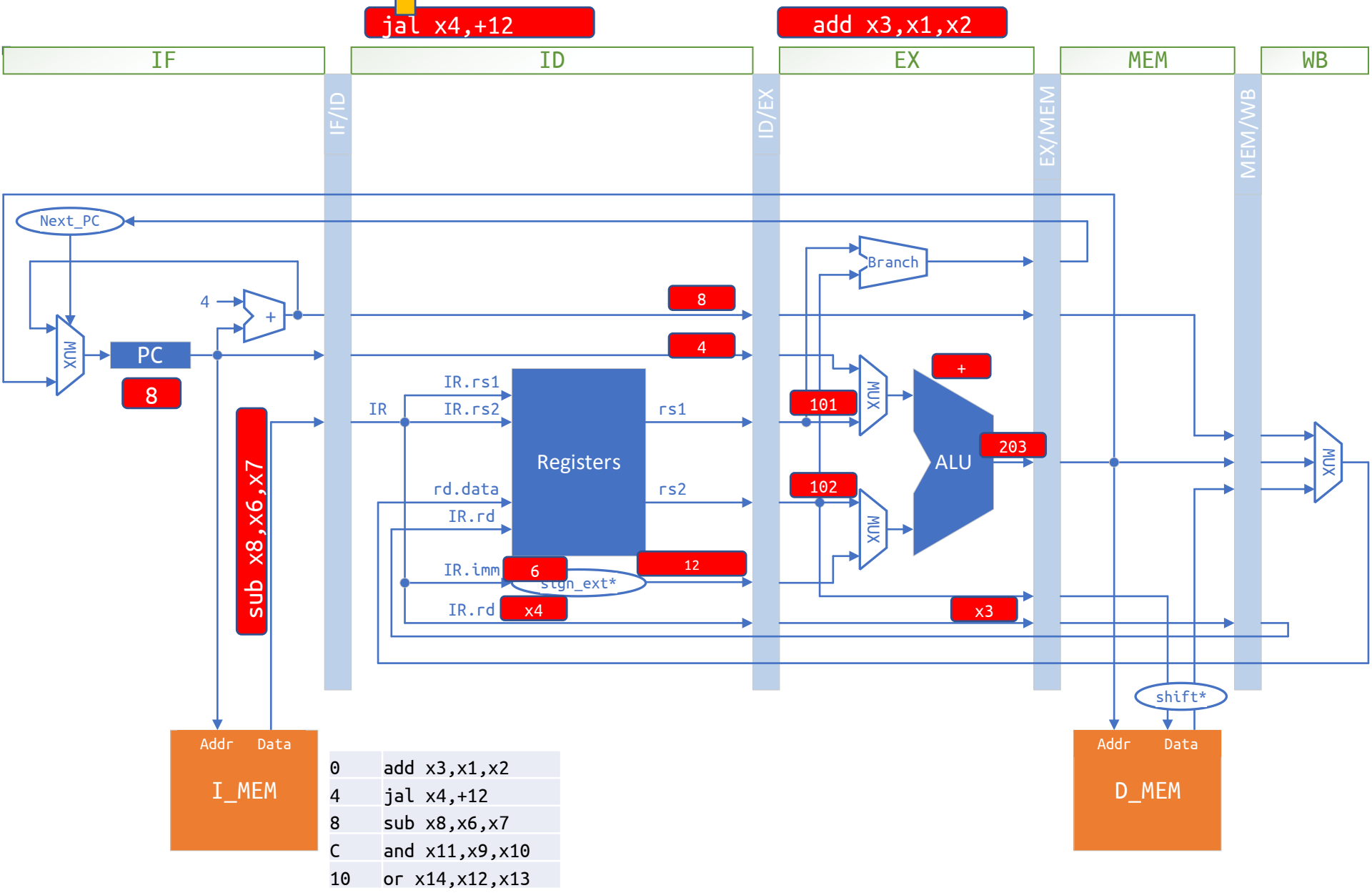
0	add x3,x1,x2
4	jal x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 2


add x3,x1,x2
IF
ID
EX
MEM
WB
IF/ID
ID/EX
EX/MEM
MEM/WB
Next_PC
MUX
PC
4
4
+
8
jal x4,+12
IR
IR.rs1
IR.rs2
x1
x2
Registers
rs1
101
rs2
102
rd.data
IR.rd
IR.imm
sign_ext*
IR.rd
x3
Branch
MUX
MUX
MUX
ALU
MUX
shift*
Addr
Data
I_MEM
Addr
Data
D_MEM
0 add x3,x1,x2
4 jal x4,+12
8 sub x8,x6,x7
C and x11,x9,x10
10 or x14,x12,x13

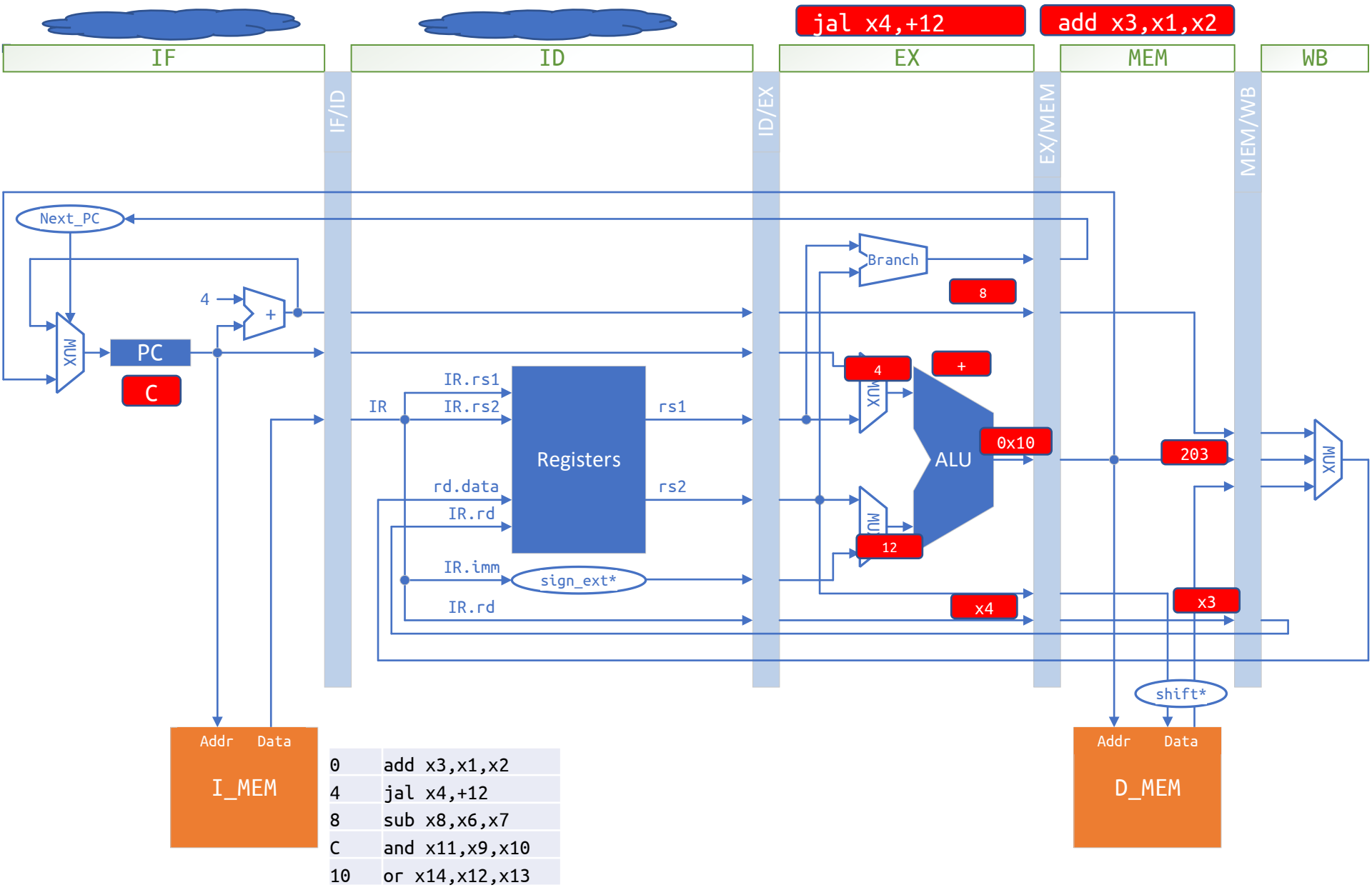
Period = 3

Prepoznaje se UPRAVLJAČKI HAZARD

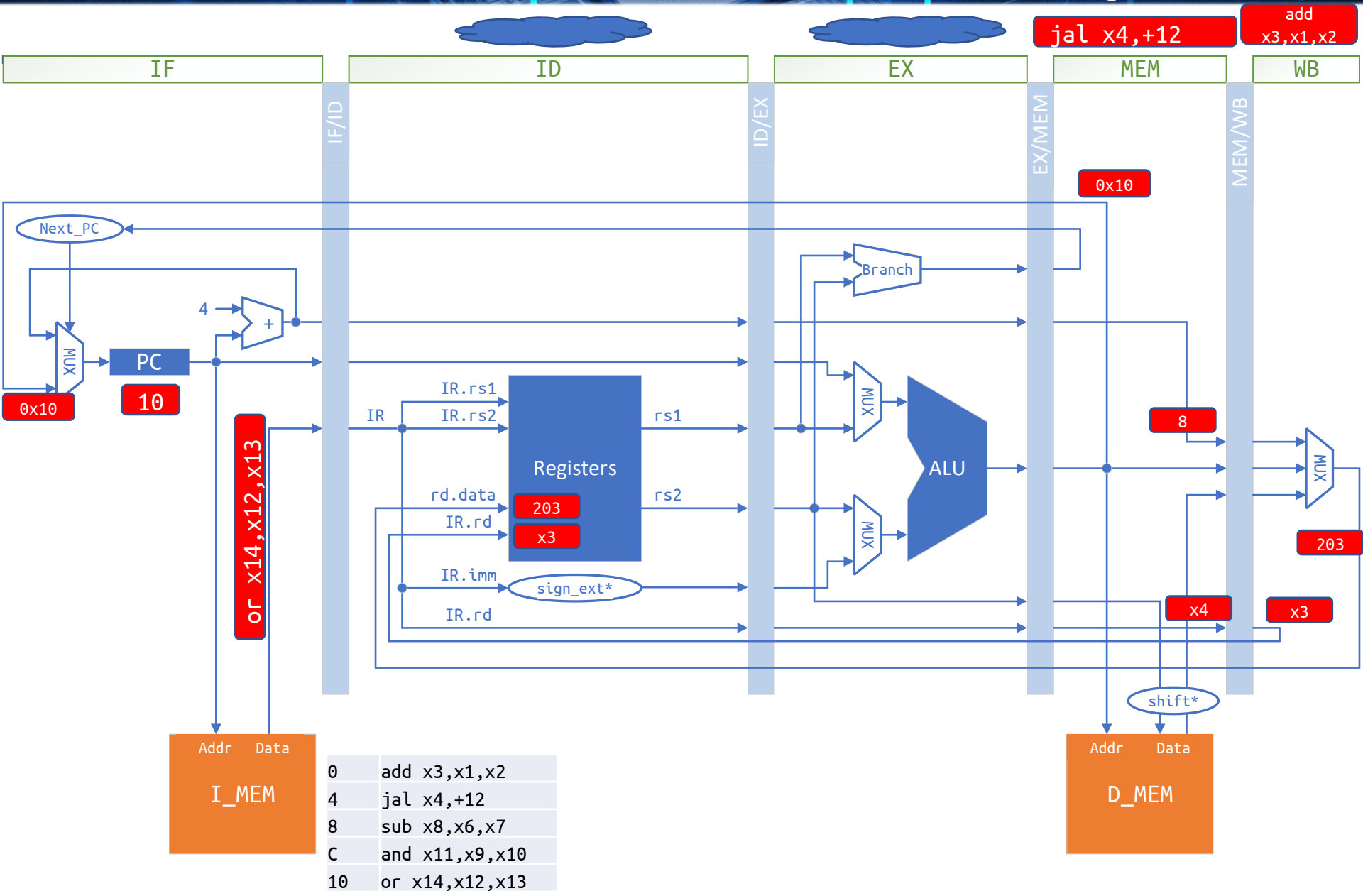


0	add x3,x1,x2
4	jal x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 4

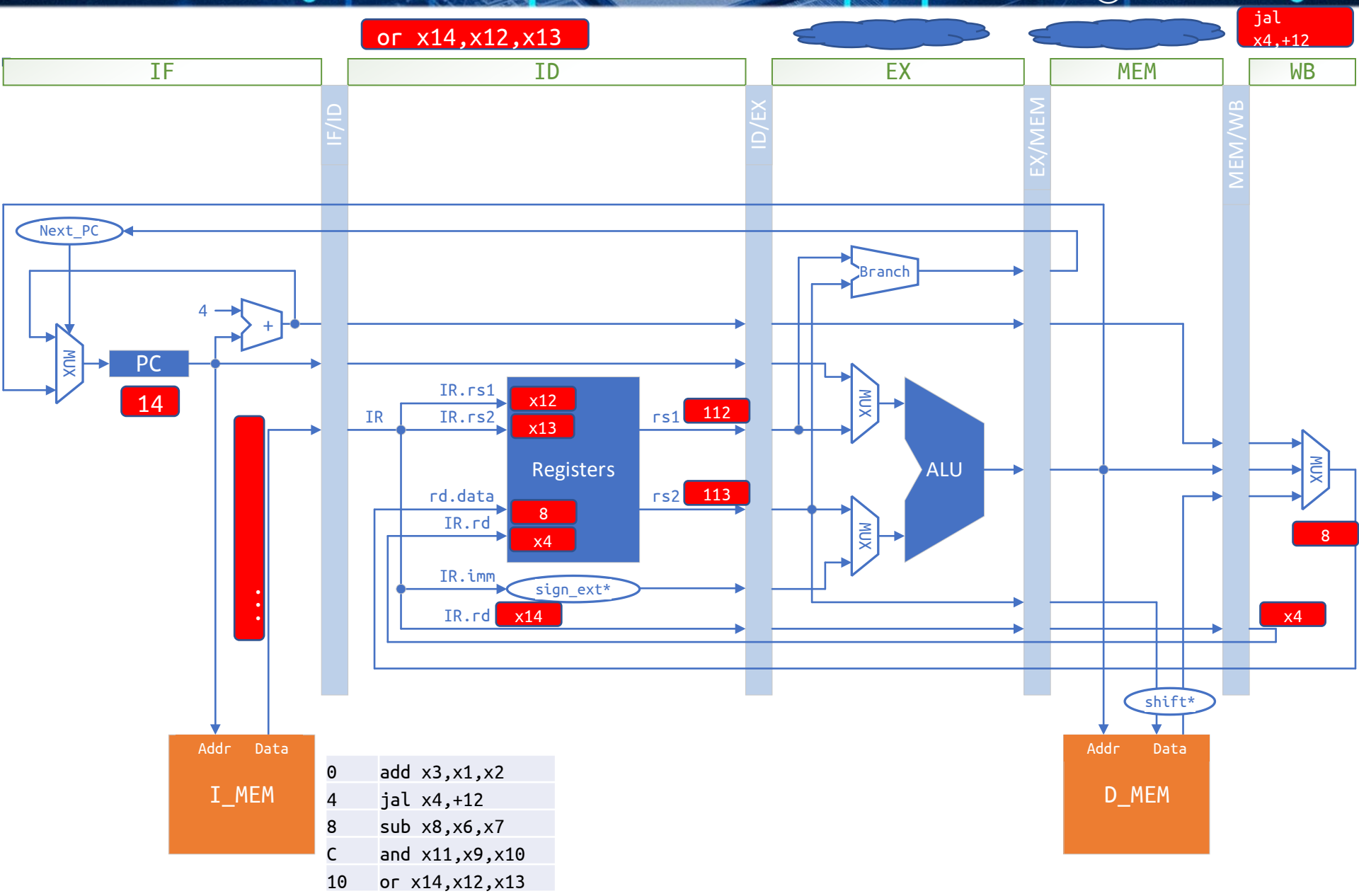

jal x4,+12
add x3,x1,x2
IF
ID
EX
MEM
WB
IF/ID
ID/EX
EX/MEM
MEM/WB
Next_PC
MUX
PC
C
4
+
IR
IR.rs1
IR.rs2
rd.data
IR.rd
IR.imm
IR.rd
Registers
rs1
rs2
sign_ext*
Branch
8
4
+
MUX
ALU
0x10
MUX
12
x4
203
x3
MUX
shift*
Addr
Data
I_MEM
Addr
Data
D_MEM
0 add x3,x1,x2
4 jal x4,+12
8 sub x8,x6,x7
C and x11,x9,x10
10 or x14,x12,x13

Period = 5

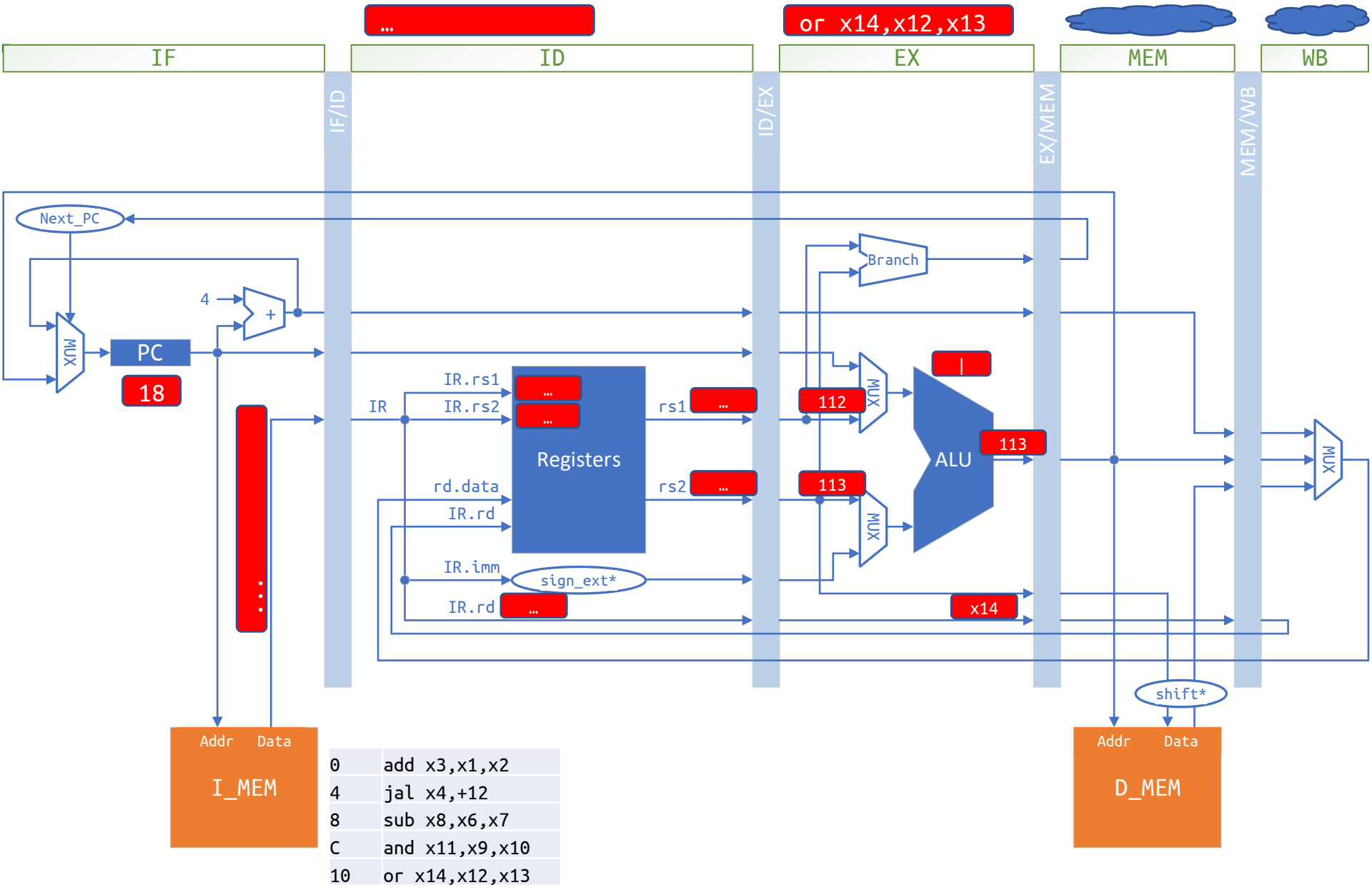


0	add x3,x1,x2
4	jal x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 6

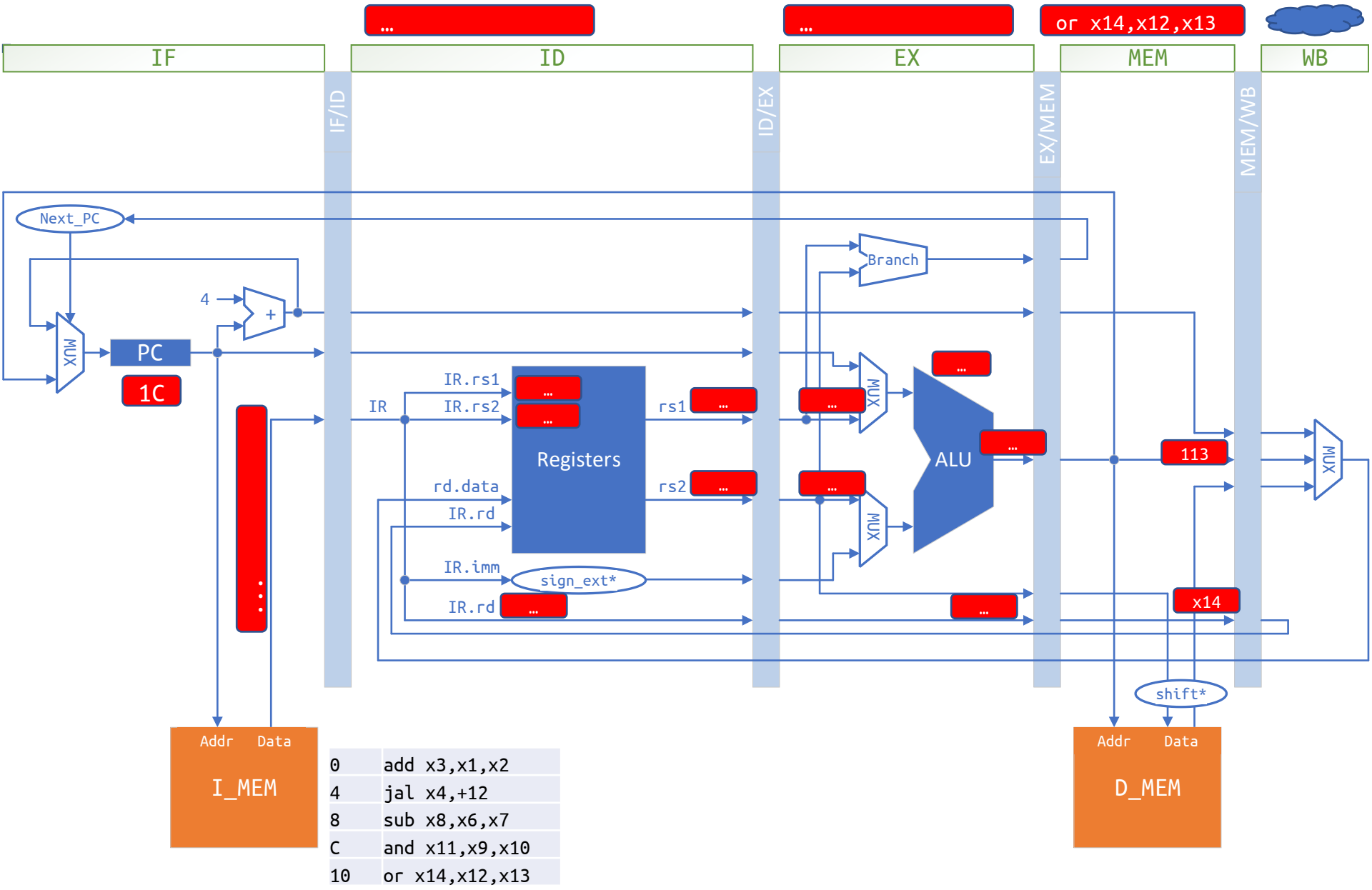

or x14,x12,x13
jal x4,+12
IF
ID
EX
MEM
WB
IF/ID
ID/EX
EX/MEM
MEM/WB
Next_PC
4
+
MUX
PC
14
IR
IR.rs1
IR.rs2
x12
x13
Registers
rs1
112
rs2
113
rd.data
IR.rd
8
x4
IR.imm
sign_ext*
IR.rd
x14
Branch
MUX
ALU
shift*
8
x4
Addr
Data
I_MEM
D_MEM
0 add x3,x1,x2
4 jal x4,+12
8 sub x8,x6,x7
C and x11,x9,x10
10 or x14,x12,x13

Period = 7

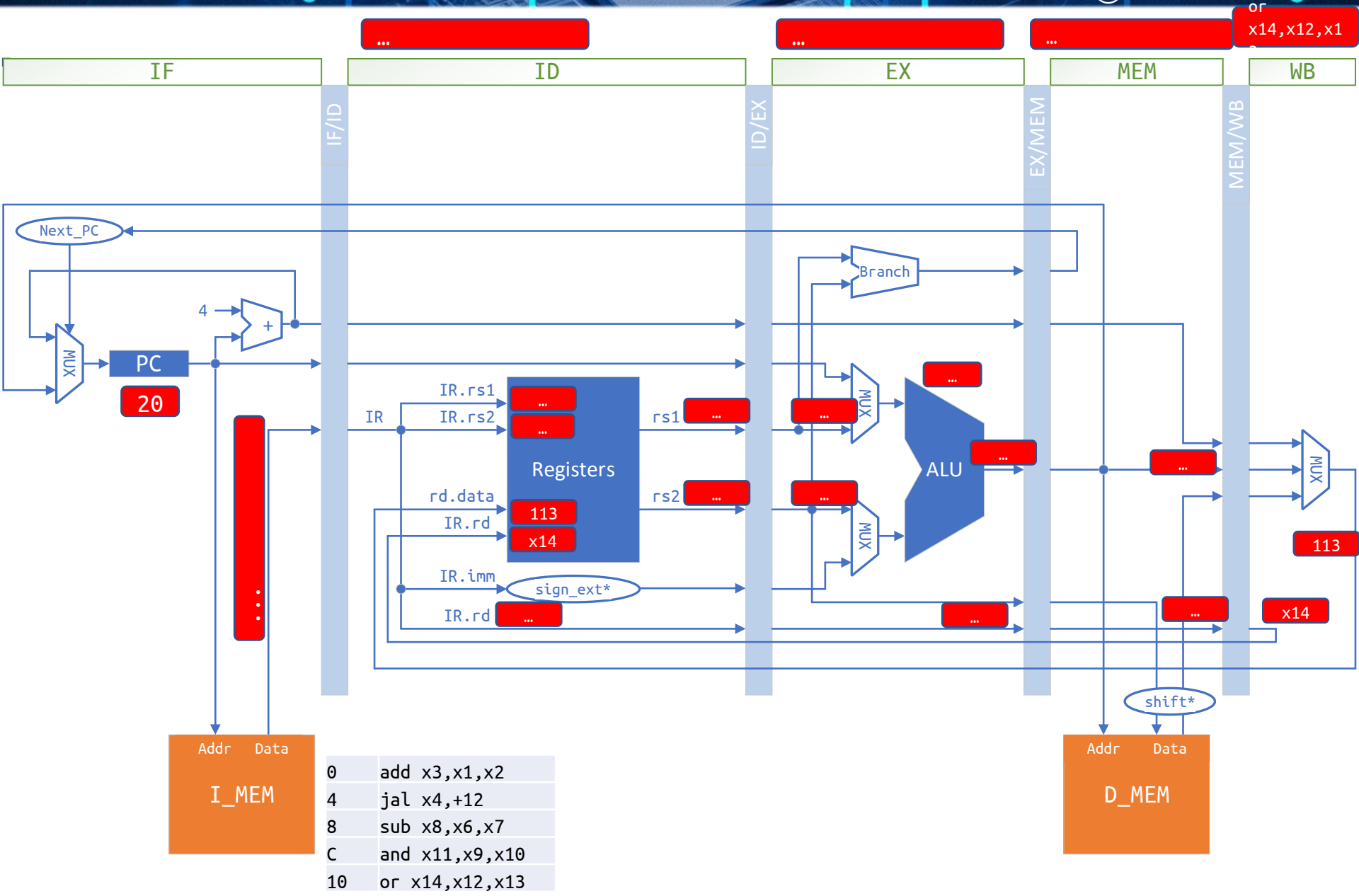


0	add x3,x1,x2
4	jal x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Period = 8


...
...
or x14,x12,x13
IF
ID
EX
MEM
WB
IF/ID
ID/EX
EX/MEM
MEM/WB
Next_PC
4
+
MUX
PC
1C
IR
IR.rs1
IR.rs2
rd.data
IR.rd
IR.imm
IR.rd
Registers
rs1
rs2
sign_ext*
Branch
MUX
ALU
113
x14
shift*
Addr
Data
I_MEM
Addr
Data
D_MEM
0 add x3,x1,x2
4 jal x4,+12
8 sub x8,x6,x7
C and x11,x9,x10
10 or x14,x12,x13

Period = 9



0	add x3,x1,x2
4	jal x4,+12
8	sub x8,x6,x7
C	and x11,x9,x10
10	or x14,x12,x13

Primjer niza naredaba #5


```
0          add x3,x1,x2
4          jal x4,L1
8          sub x8,x6,x7
C          and x11,x9,x10
10   L1    or x14,x12,x13
```

Period	IF	ID	EX	MEM	WB
1	(PC=0) add				
2	(PC=4) jal	add			
3	(PC=8) sub	jal	add		
4			jal	add	
5	(PC=10) or			jal	add
6		or			jal
7			or		
8				or	
9					or

Zbog bezuvjetog grananja (UPRAVLJAČKI HAZARD) upravljačka logika mora ubaciti DVA mjehurića u protočnu strukturu

Ukupno trajanje odsječka:
add: 5 perioda (zbog punjenja protočne strukture)
jal: 1 period + 2 perioda zbog mjehurića
or: 1 period
UKUPNO = 9

FRISC-V hazardi

- Strukturni hazard
  - Ne postoji zbog Harvard arhitekture
- Podatkovni hazard
  - Ako naredba treba sadržaj nekog registra koji je rezultat neke prethodne naredbe
  - Može zahtjevati 1 ili 2 dodatna perioda
- Upravljački hazard
  - Kod bezuvjetnih skokova (jal, jalr) uvijek zahtjeva 2 dodatna perioda
  - Kod uvjetnih skokova (b*xx*) zahtjeva 2 dodatna perioda samo ako je uvjet zadovoljen, inače nema dodatnih perioda