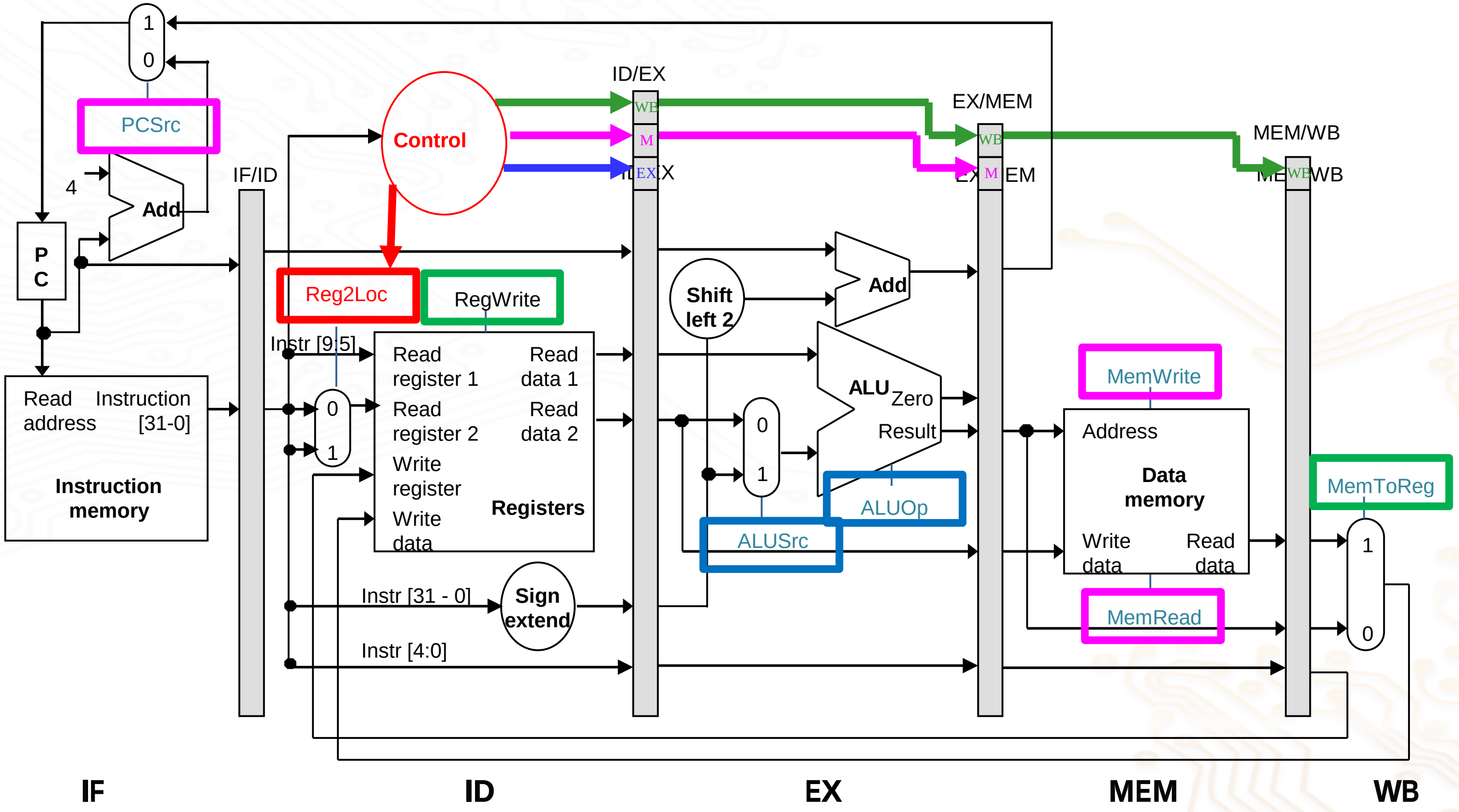
CE/CZ 3001:
Advanced Computer Architecture

Pipelining: An Example Walkthrough

Dr Smitha K. G.
School of Computer Science
and Engineering

Recap – Pipelined datapath and control



Pipelining in action: An example

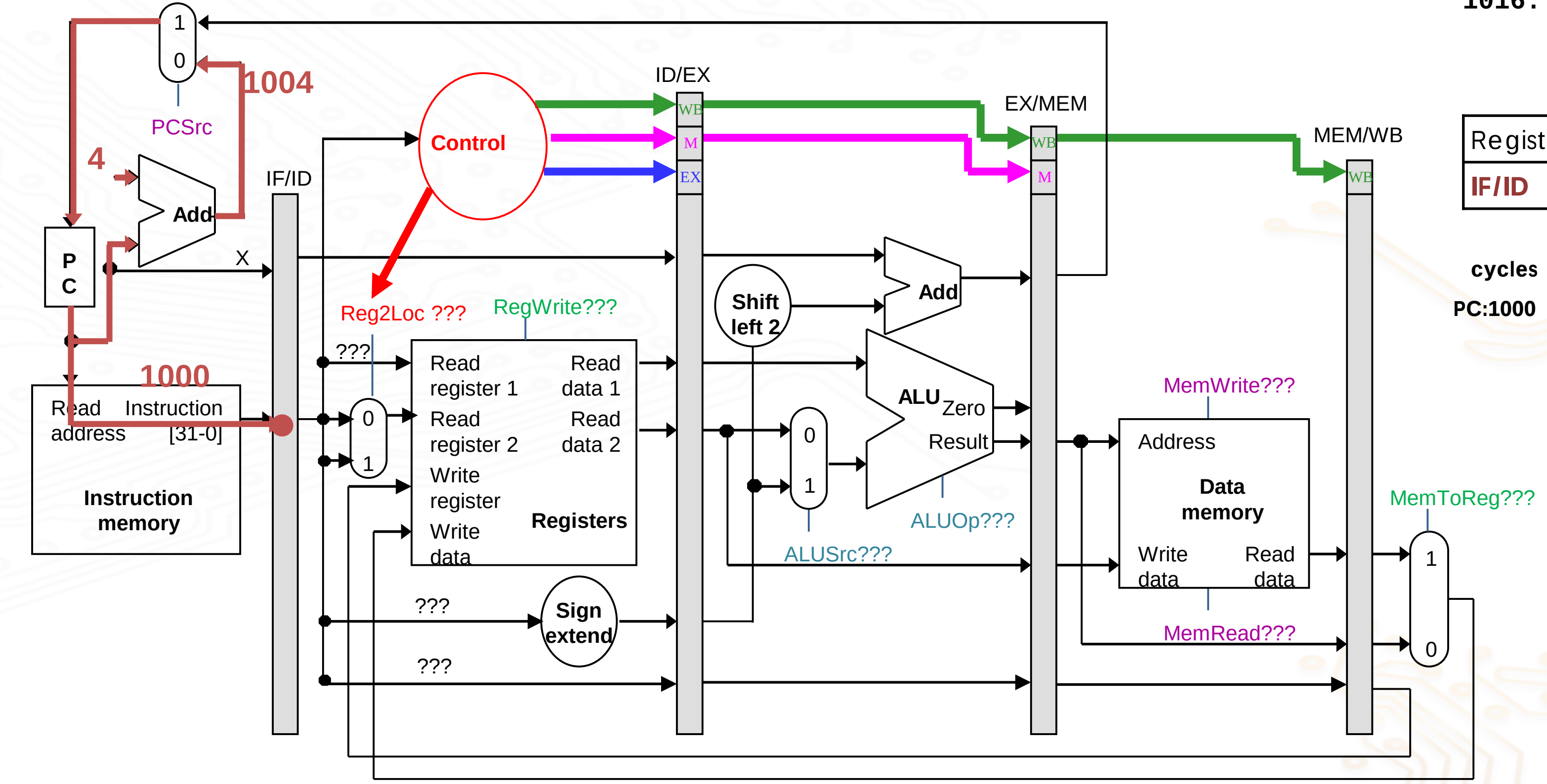
```
1000:   LDUR X8, [X29,#8]          LDUR X8,(8+129)  => X8  = 99
1004:   SUB  X2, X4, X5            X2 = 104 – 105  => X2  = -1
1008:   AND  X9, X10, X11          X9 = 110 & 111  => X9  = 110
1012:   ORR  X16,X17, X18          X16 = 117 | 118 => X16 = 22F
1016:   ADD  X13,X14, X31          X13 = 114 + 0 => X13 = 114
```

- Execution starts at address 1000. PC = 1000
- Each register contains its number plus 100 (X31 has zero)
  - **For instance, register X8 contains 108, register X29 contains 129, and so forth**
- Every data memory location contains 99

Cycle 1

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF: LDUR X8,[X29,#8]   ID: ???   EX: ???   MEM: ???   WB: ???

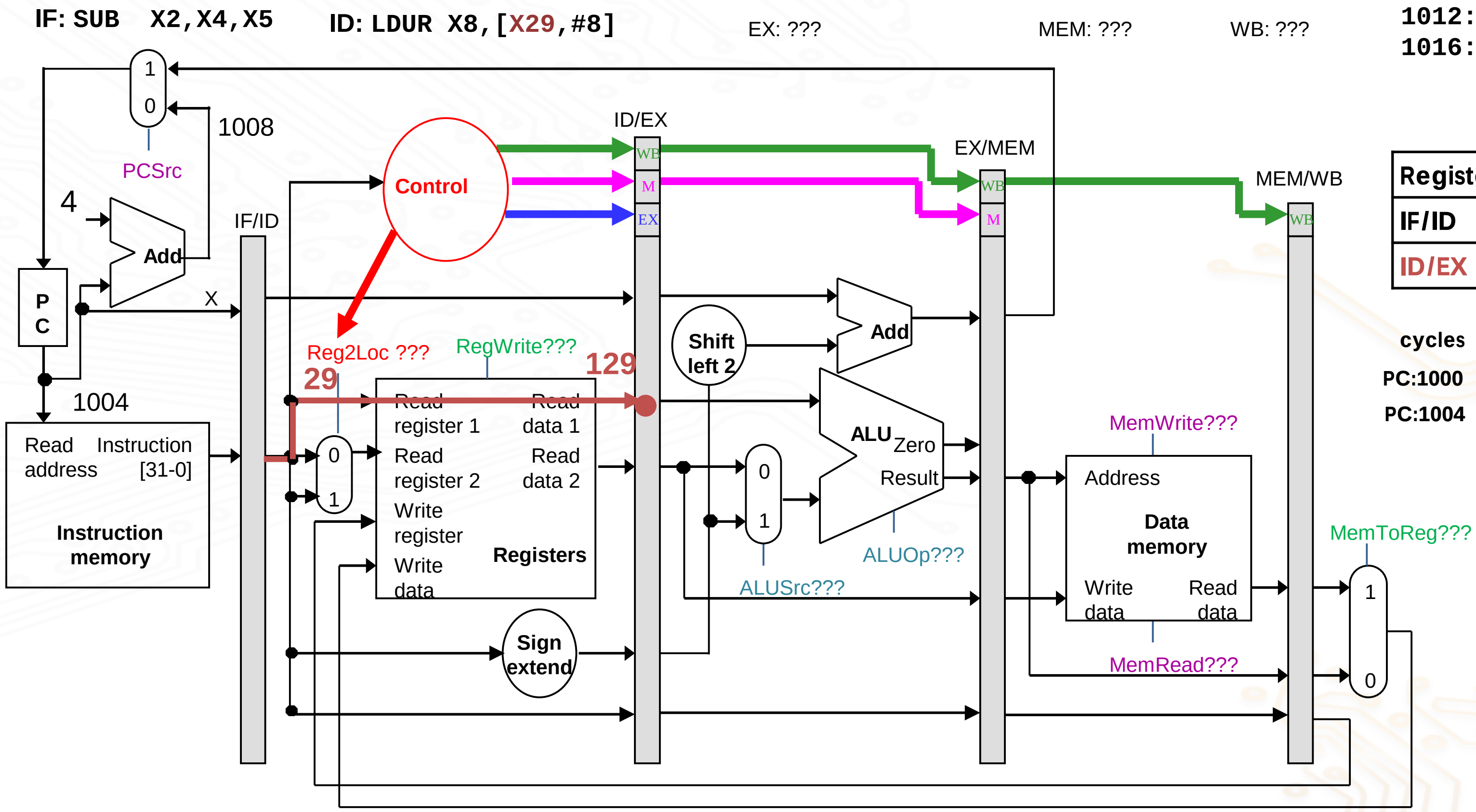


Register	Values
IF/ID	Instruction = LDUR

cycles	1	2	3	4	5	6	7	8	9
PC:1000	F								

Cycle 2: Decode stage

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

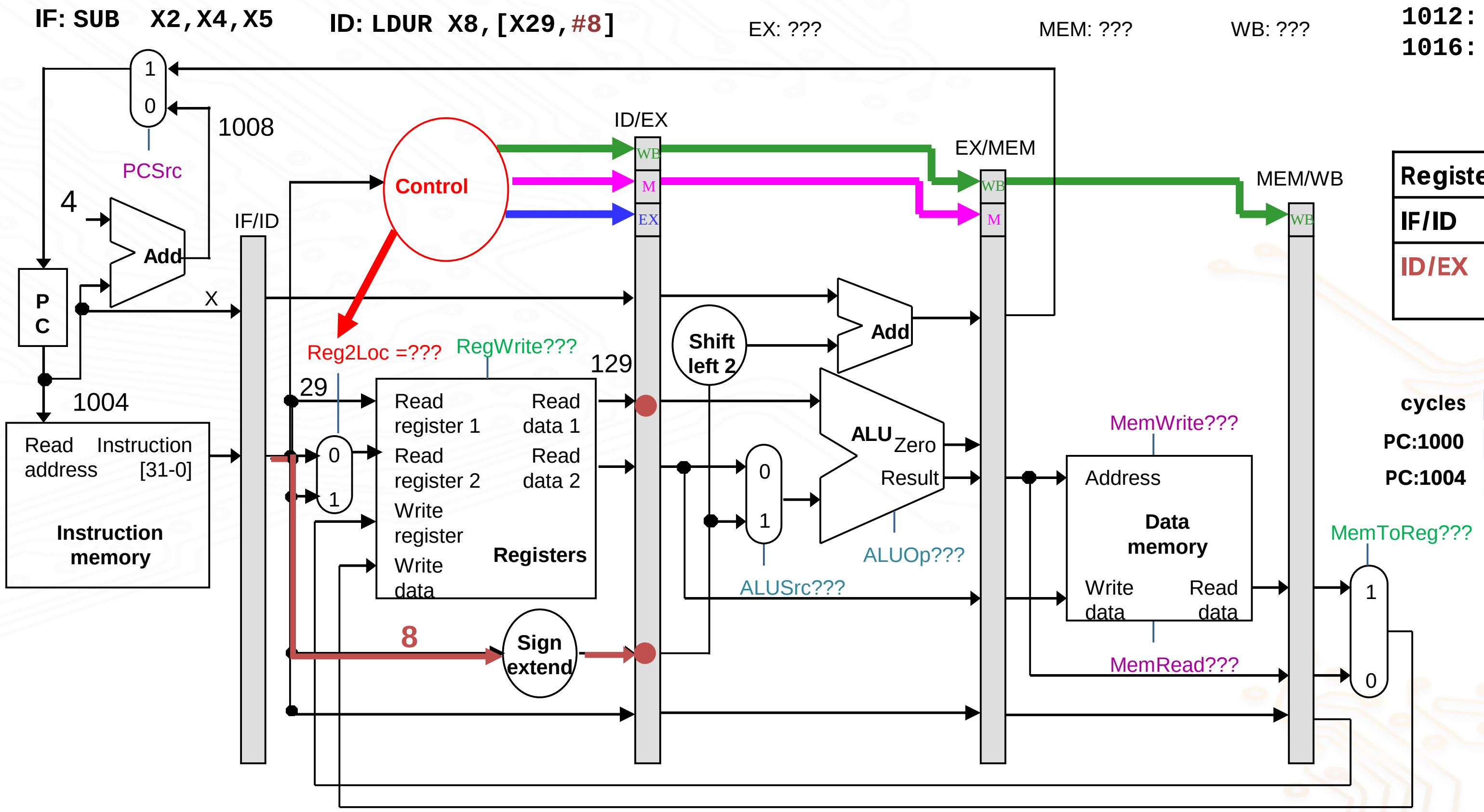


Register	Values
IF/ID	Instruction = LDUR
ID/EX	Read data1 = 129

cycles	1	2	3	4	5	6	7	8	9	
PC:1000	F	D								
PC:1004										

Cycle 2: Decode stage

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```



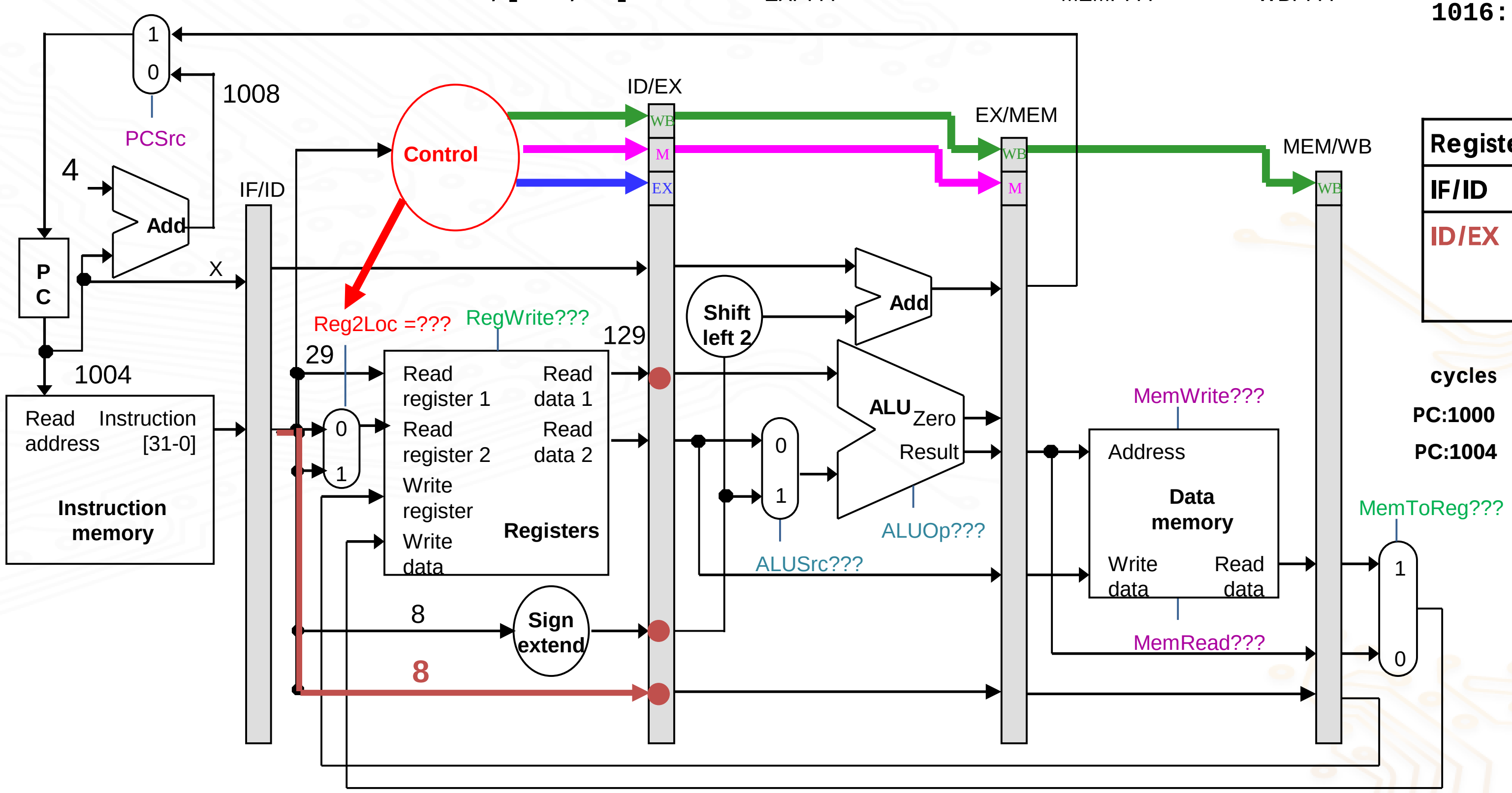
Register	Values
IF/ID	Instruction = LDUR
ID/EX	Read data1 = 129 signextend = 8

cycles	1	2	3	4	5	6	7	8	9	
PC:1000	F	D								
PC:1004										

Cycle 2: Decode stage

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF: SUB X2,X4,X5   ID: LDUR X8,[X29,#8]   EX: ???   MEM: ???   WB: ???

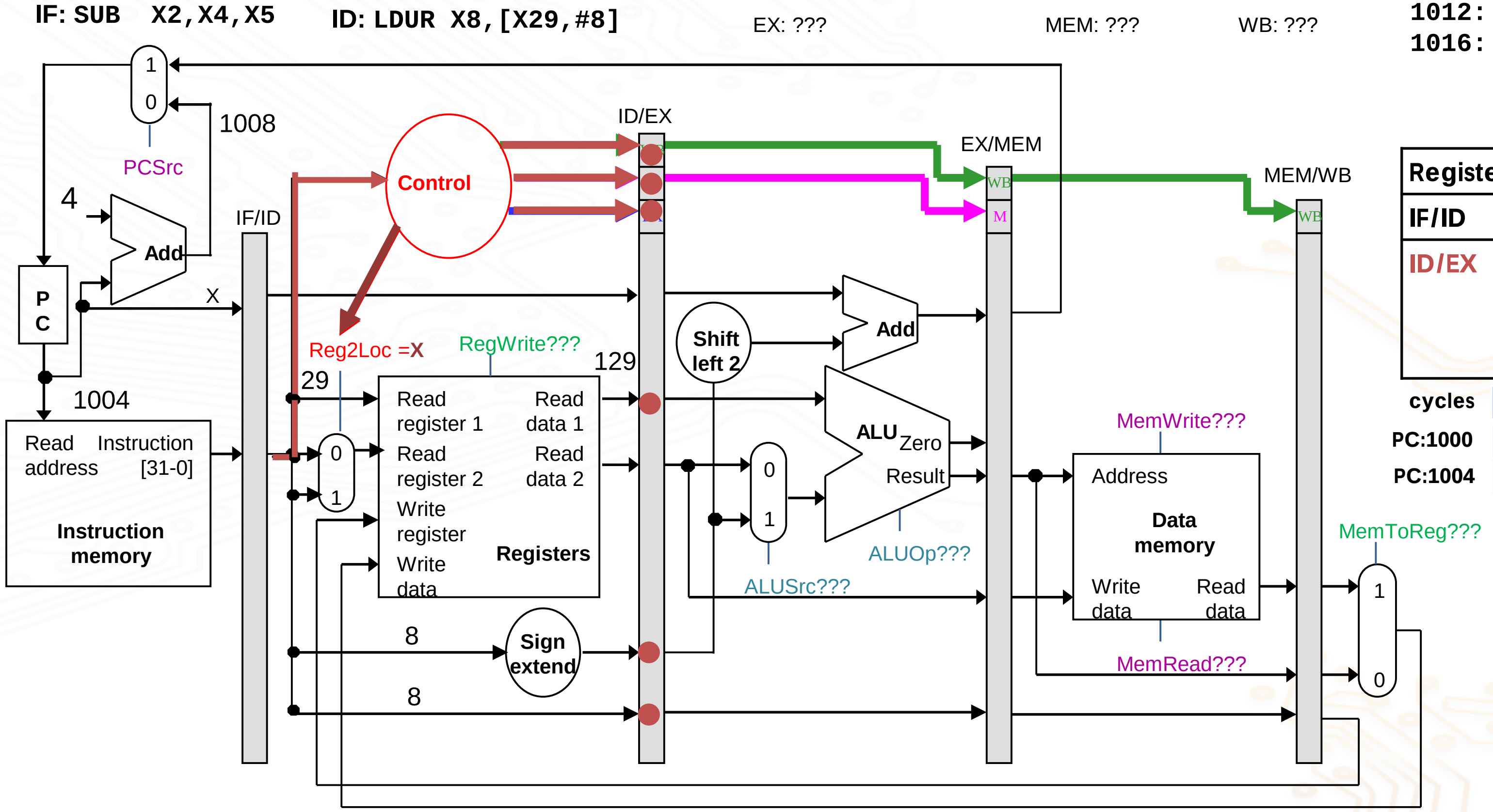


Register	Values
IF/ID	Instruction = LDUR
ID/EX	Read data1 = 129 signextend = 8 WB-Reg addr = 8

cycles	1	2	3	4	5	6	7	8	9	
PC:1000	F	D								
PC:1004										

Cycle 2: Decode stage

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

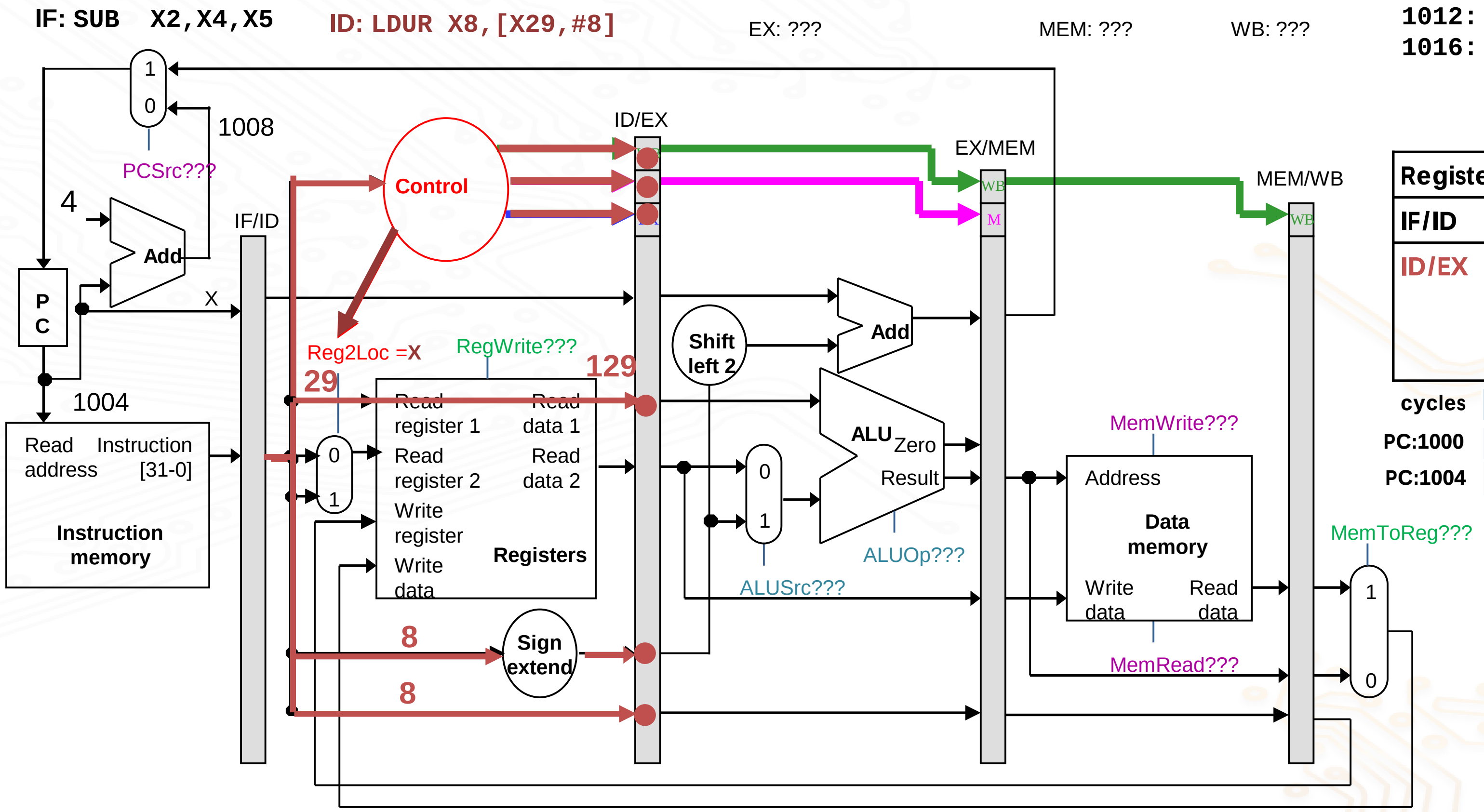


Register	Values
IF/ID	Instruction = LDUR
ID/EX	Read data1 = 129 signextend = 8 WB-Reg addr = X8 control

cycles	1	2	3	4	5	6	7	8	9	
PC:1000	F	D								
PC:1004										

Cycle 2: Decode stage- Complete

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

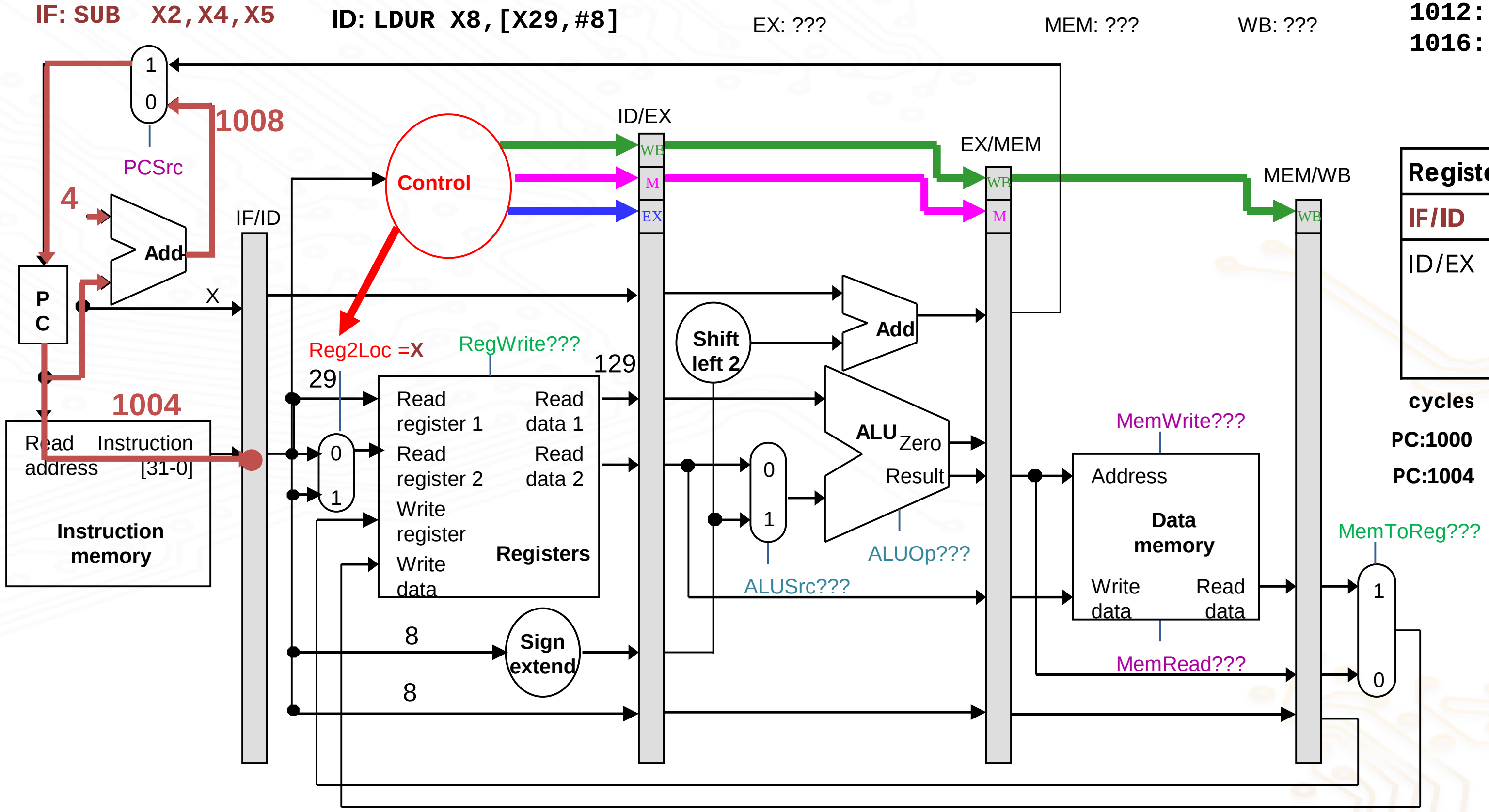


Register	Values
IF/ID	Instruction = LDUR
ID/EX	Read data1 = 129 signextend = 8 WB-Reg addr =X8 control

cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D							
PC:1004									

Cycle 2: Fetch Stage

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```



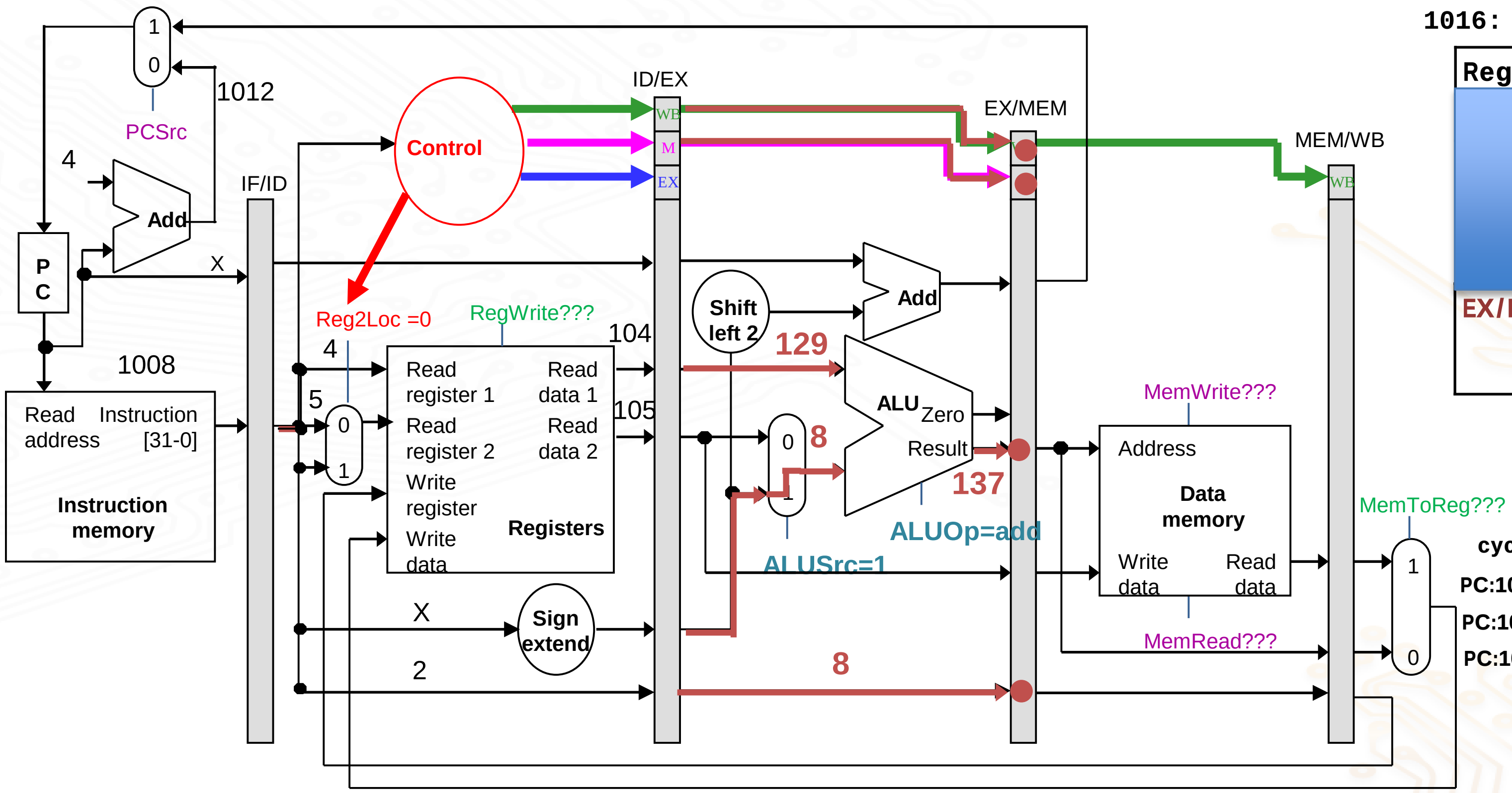
Register	Values
IF/ID	Instruction = SUB
ID/EX	Read data1 = 129 signextend = 8 WB-Reg addr = X8 control

cycles	1	2	3	4	5		7	8	9	
PC:1000	F	D								
PC:1004		F								

Cycle 3: Execute Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF: AND X9,X10,X11   ID: SUB X2,X4,X5   EX:LDUR X8,[X29, #8]   MEM: ???   WB: ???



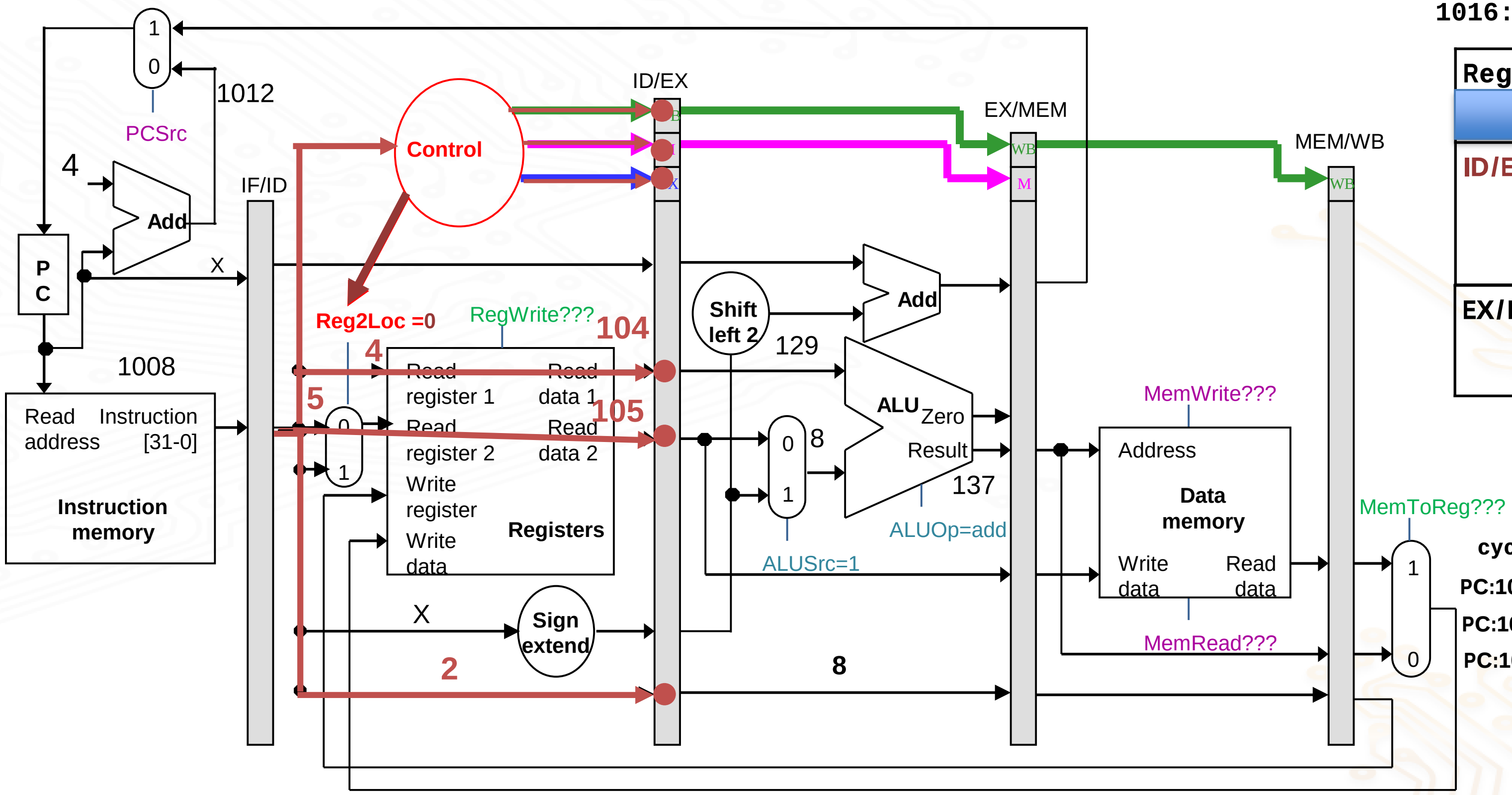
Register	Values
EX/MEM	ALUout = 137 WB-Reg addr = X8 Control

cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D	E						
PC:1004		F							
PC:1008									

Cycle 3: Decode Stage(complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF: AND X9,X10,X11 ID: SUB X2,X4,X5 EX:LDUR X8,[X29, #8] MEM: ??? WB: ???



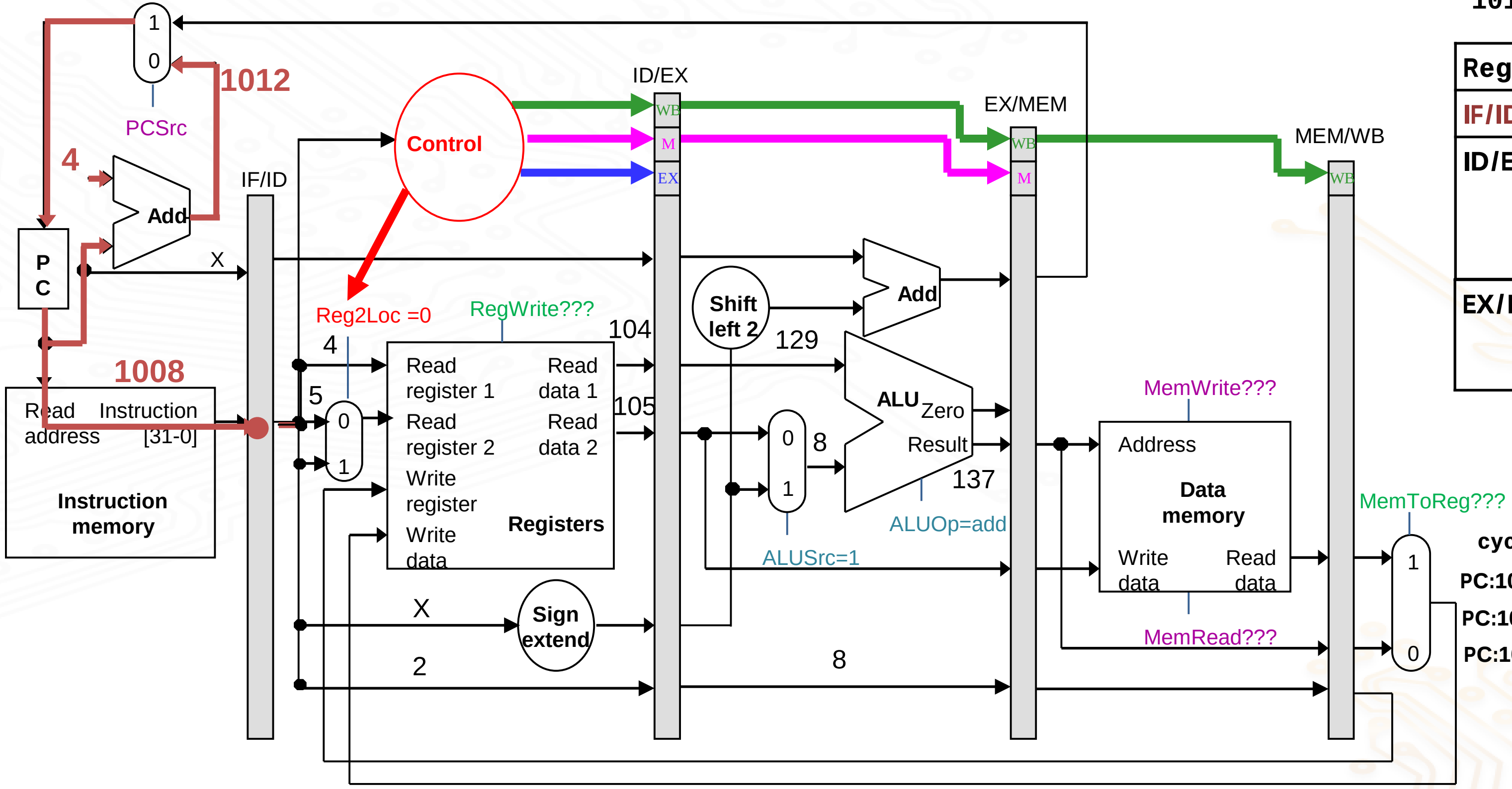
Register	Values
ID/EX	Read data1 = 104 Read data1 = 105 WB-Reg addr = X2 control
EX/MEM	ALUout = 137 WB-Reg addr = X8 Control

cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D	E						
PC:1004		F	D						
PC:1008									

Cycle 3: Fetch Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF: AND X9,X10,X11   ID: SUB X2,X4,X5   EX: LDUR X8,[X29, #8]   MEM: ???   WB: ???



Register	Values
IF/ID	Instruction = AND
ID/EX	Read data1 = 104 Read data1 = 105 WB-Reg addr = X2 control
EX/MEM	ALUout = 137 WB-Reg addr = X8 Control

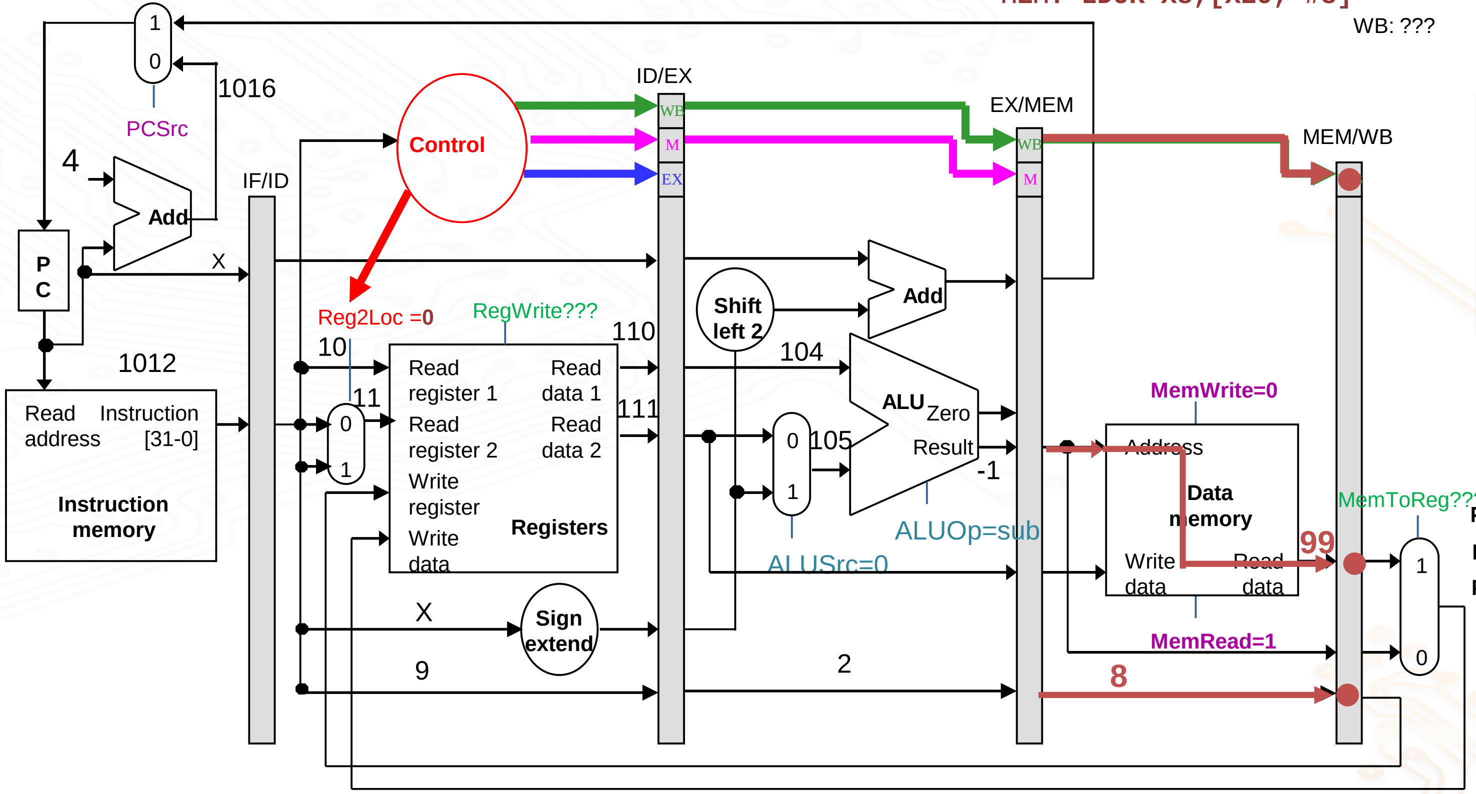
cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D	E						
PC:1004		F	D						
PC:1008			F						

Cycle 4: Memory Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF:ORR X16,X17, X18 ID: AND X9,X10,X11 EX: SUB X2,X4,X5 MEM: LDUR X8,[X29, #8]

WB: ???



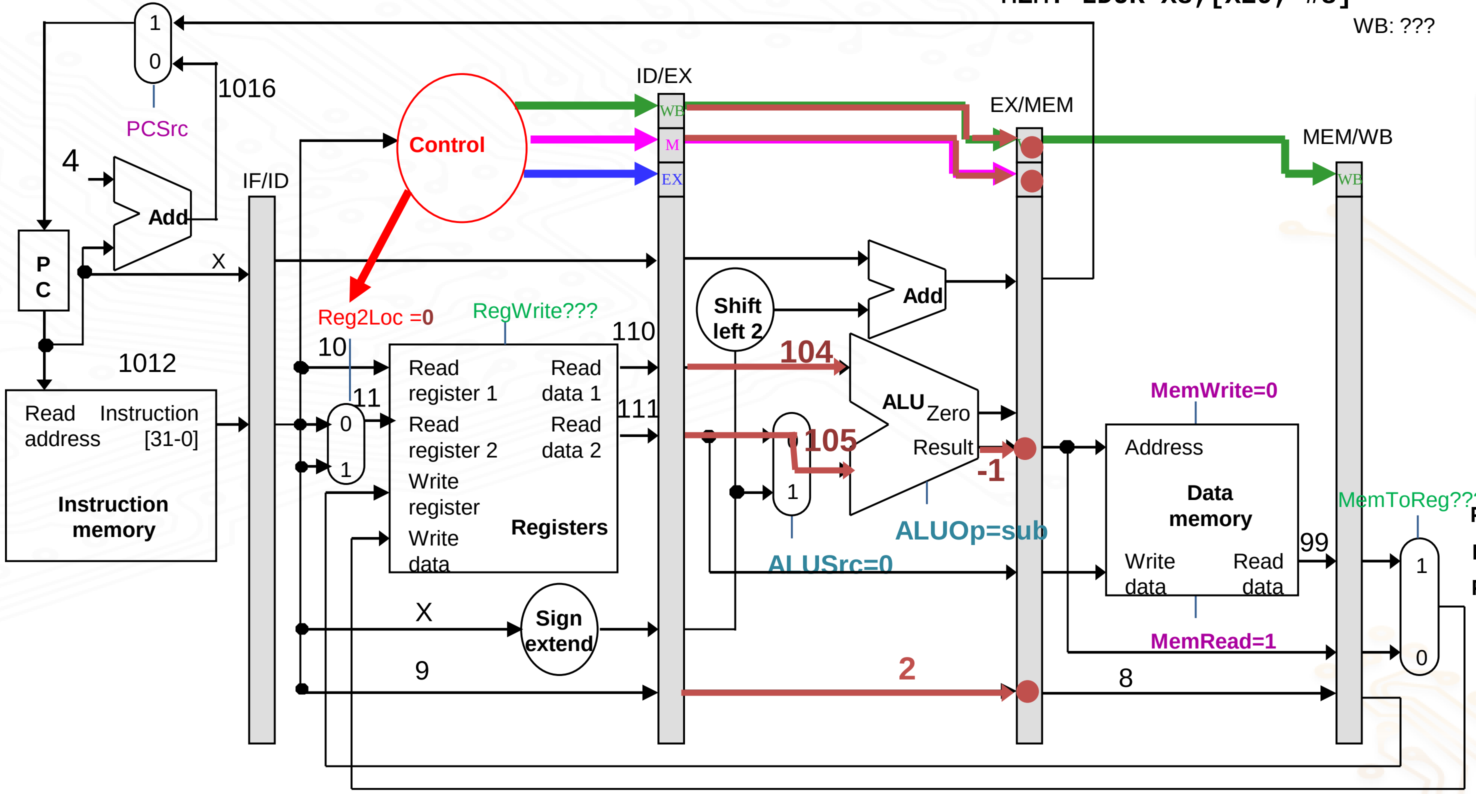
Register	Values
MEM/WB	ReadData M = 99 WB-Reg addr = X8 Control

cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D	E	M					
PC:1004		F	D						
PC:1008			F						

Cycle 4: Execute Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF:ORR X16,X17, X18 ID: AND X9,X10,X11 EX: SUB X2,X4,X5 MEM: LDUR X8,[X29, #8]



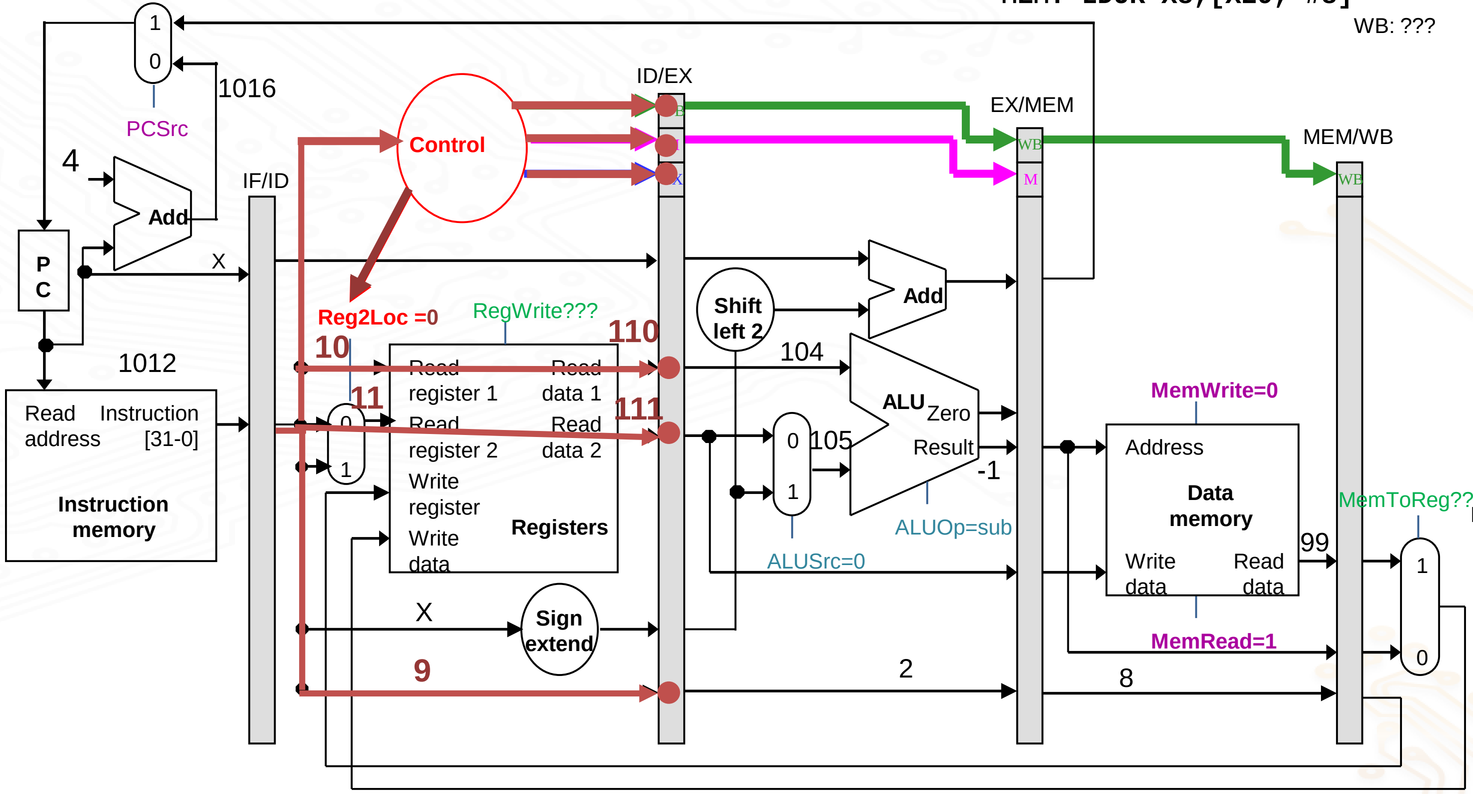
Register	Values
EX/MEM	ALUout = -1 WB-Reg addr = X2 Control
MEM/WB	ReadData M = 99 WB-Reg addr = X8

cycles	1	2	3	4	5	6	7	8	9	
PC:1000	F	D	E	M						
PC:1004		F	D	E						
PC:1008			F							

Cycle 4: Decode Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF:ORR X16,X17, X18   ID: AND X9,X10,X11   EX: SUB X2,X4,X5   MEM: LDUR X8,[X29, #8]   WB: ???

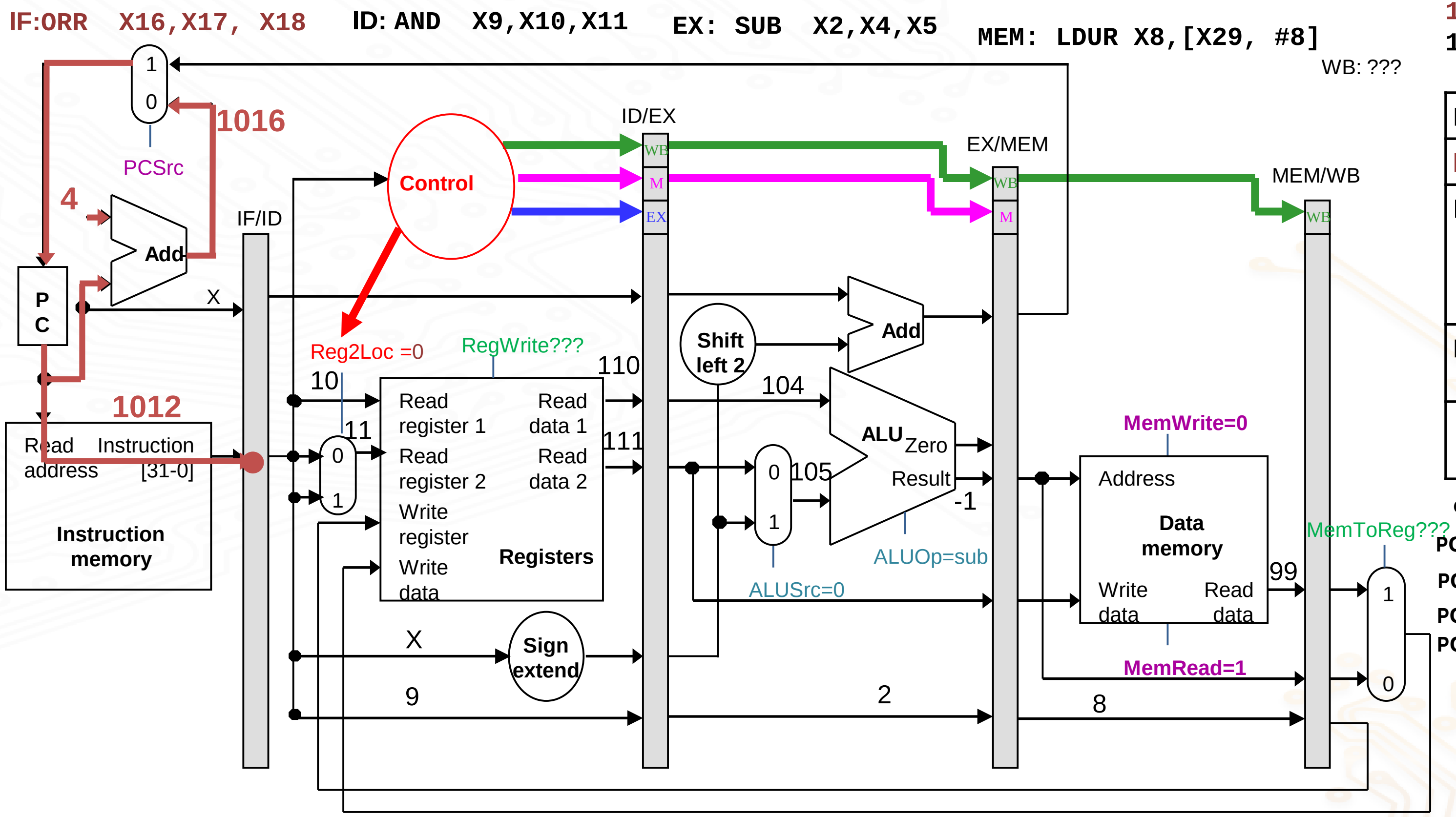


Register	Values
ID/EX	Read data1 = 110 Read data2 = 111 WB-Reg addr = X9 Control
EX/MEM	ALUout = -1 WB-Reg addr = X2
MEM/WB	ReadData M = 99 WB-Reg addr = X8

cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D	E	M					
PC:1004		F	D	E					
PC:1008			F	D					

Cycle 4: Fetch Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2,  X4,  X5
1008:  AND  X9,  X10,  X11
1012:  ORR  X16,X17,  X18
1016:  ADD  X13,X14,  X31
```



Register	Values
IF/ID	Instruction = ORR
ID/EX	Read data1 = 110 Read data2 = 111 WB-Reg addr = X9 Control
EX/MEM	ALUout = -1 WB-Reg addr = X2
MEM/WB	ReadData M = 99 WB-Reg addr = X8

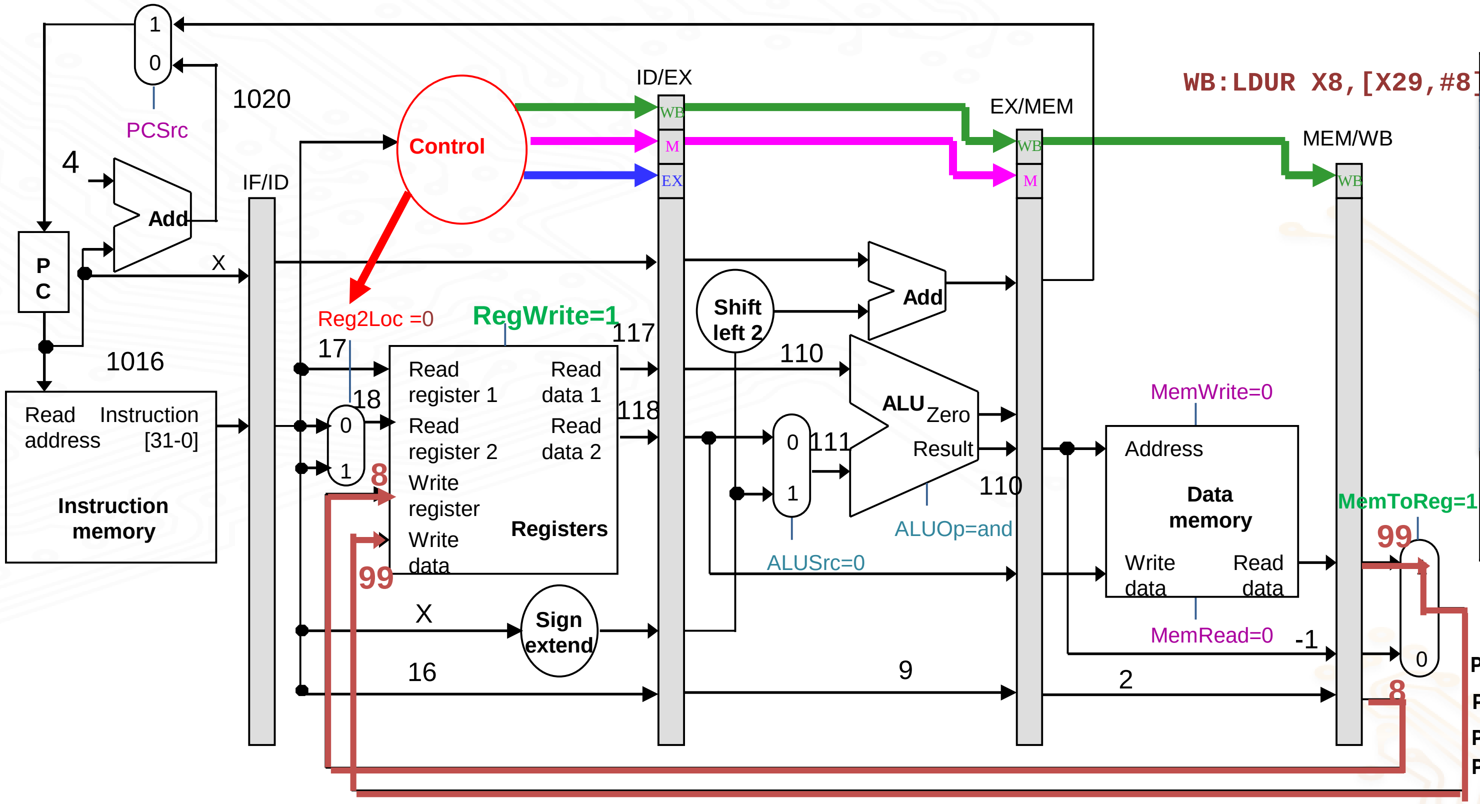
cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D	E	M					
PC:1004		F	D	E					
PC:1008			F	D					
PC:1012				F					

Cycle 5: Writeback Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF: ADD X13,X14,X31 ID: ORR X16,X17,X18 EX: AND X9,X10,X11 MEM: SUB X2,X4,X5

WB:LDUR X8,[X29,#8]



Register	Values
Regfile (WB)	Write register address = X8 Write data= 99

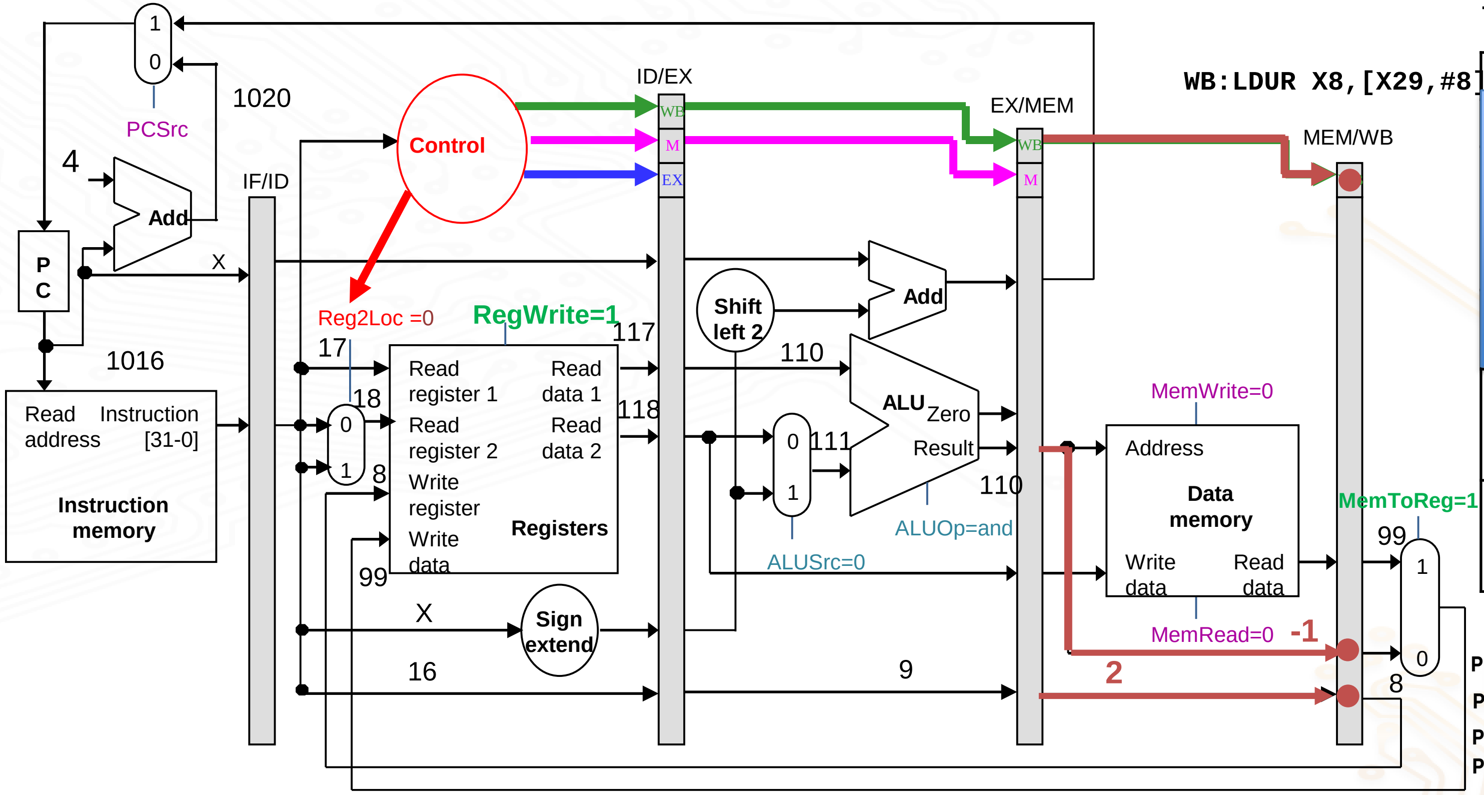
cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D	E	M	W				
PC:1004		F	D	E					
PC:1008			F	D					
PC:1012				F					

Cycle 5: Memory Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF: ADD X13,X14,X31 ID: ORR X16,X17,X18 EX: AND X9,X10,X11 MEM: SUB X2,X4,X5

WB:LDUR X8,[X29,#8]



Register	Values
MEM/WB	ReadData M = -1 WB-Reg addr = X2 Control
Regfile (WB)	Write register address = X8 Write data= 99

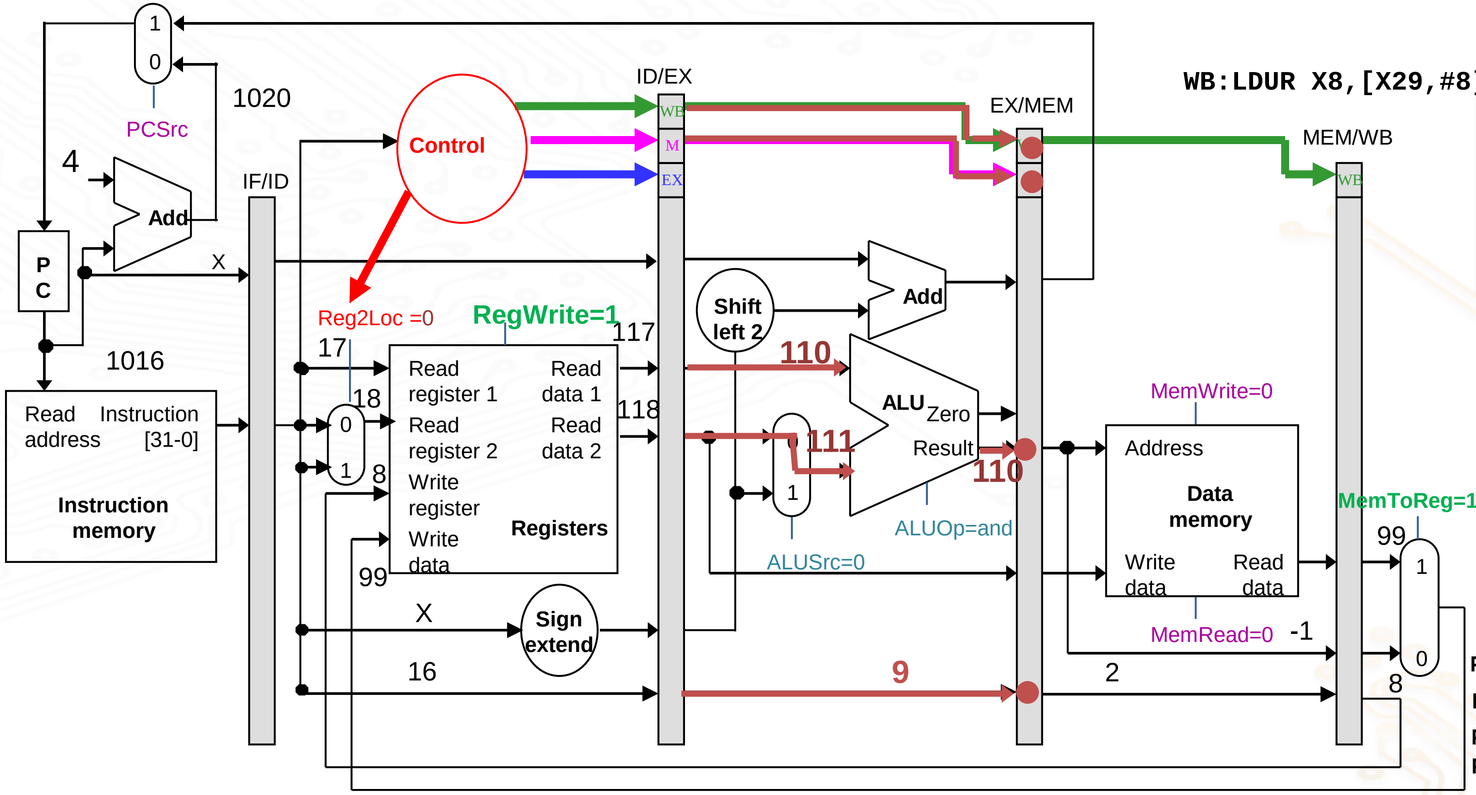
cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D	E	M	W				
PC:1004		F	D	E	M				
PC:1008			F	D					
PC:1012				F					

Cycle 5: Execute Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF: ADD X13,X14,X31 ID: ORR X16,X17,X18 EX: AND X9,X10,X11 MEM: SUB X2,X4,X5

WB:LDUR X8,[X29,#8]



Register	Values
EX/MEM	ALUout = 110 WB-Reg addr = X9 control
MEM/WB	ReadData M = -1 WB-Reg addr = X2
Regfile (WB)	Write register address = X8 Write data= 99

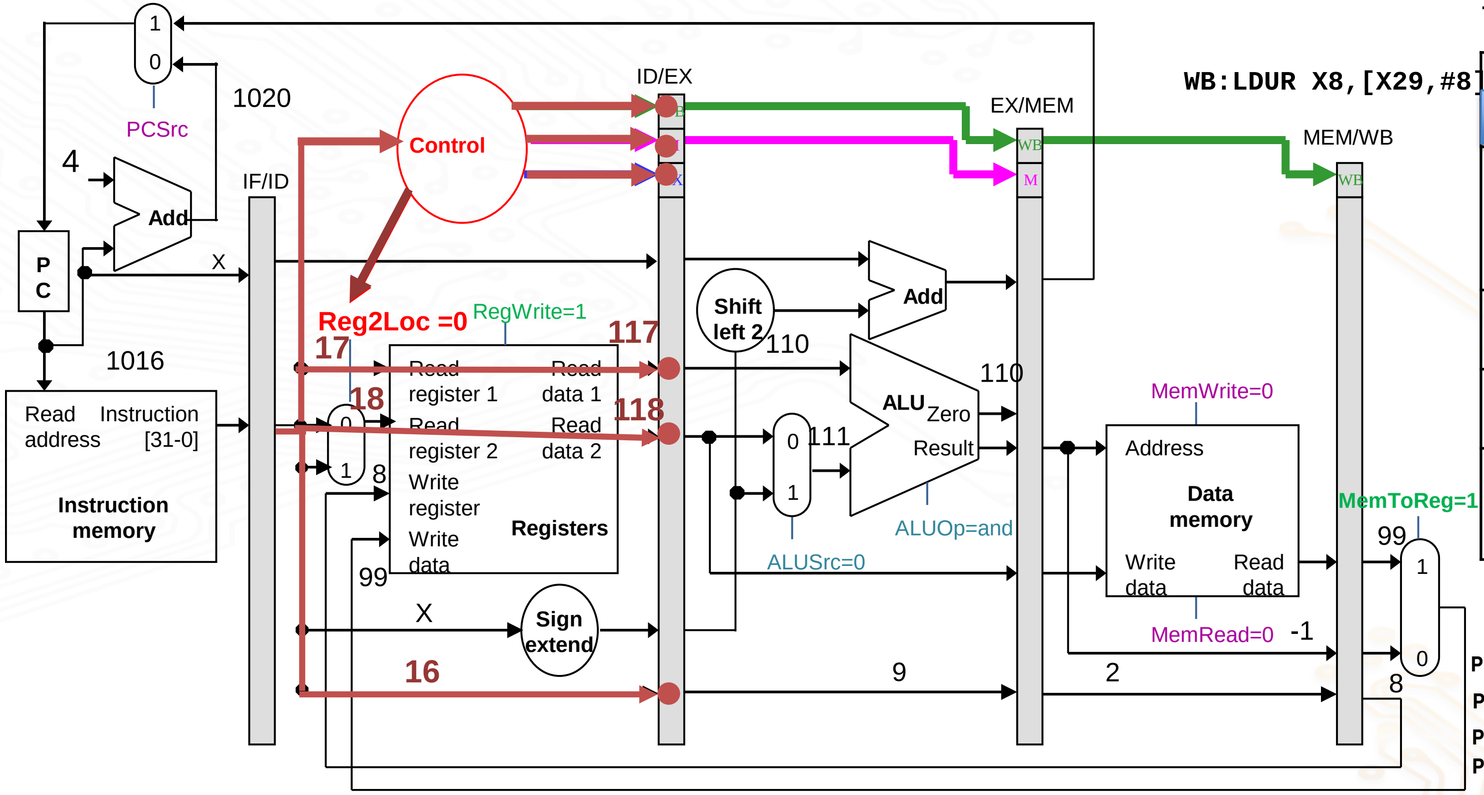
cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D	E	M	W				
PC:1004		F	D	E	M				
PC:1008			F	D	E				
PC:1012				F					

Cycle 5: Decode Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2,  X4,  X5
1008:  AND  X9,  X10,  X11
1012:  ORR  X16,X17,  X18
1016:  ADD  X13,X14,  X31
```

IF: ADD X13,X14,X31 ID: ORR X16,X17,X18 EX: AND X9,X10,X11 MEM: SUB X2,X4,X5

WB:LDUR X8,[X29,#8]



Register	Values
ID/EX	Read data1 = 117 Read data2 = 118 WB-Reg addr = X16 Control
EX/MEM	ALUout = 110 WB-Reg addr = X9
MEM/WB	ReadData M = -1 WB-Reg addr = X2
Regfile (WB)	Write register address = X8 Write data= 99

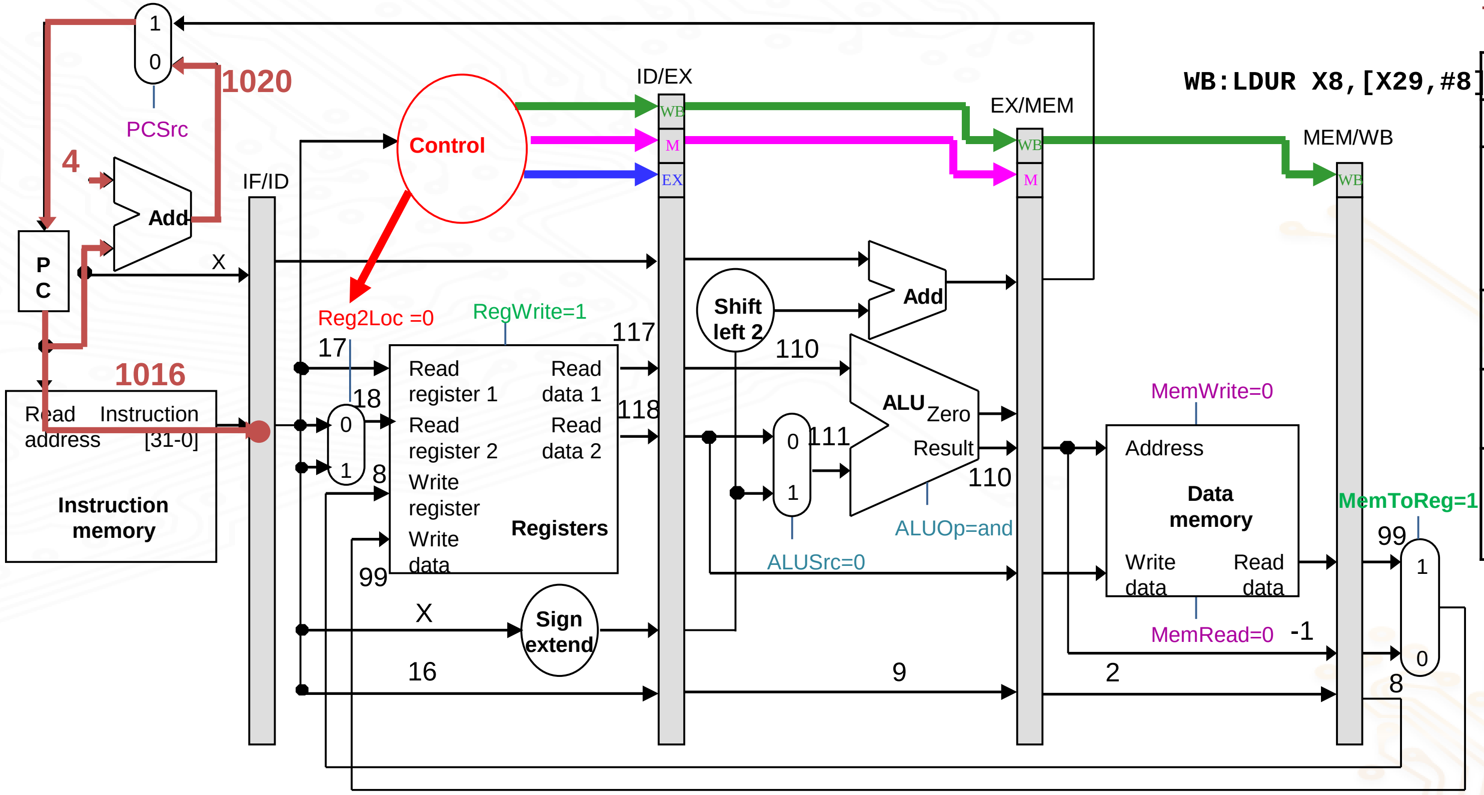
cycles	1	2	3	4	5	6	7	8	9
PC:1000	F	D	E	M	W				
PC:1004		F	D	E	M				
PC:1008			F	D	E				
PC:1012				F	D				

Cycle 5: Fetch Stage (complete)

```
1000:  LDUR X8, [X29,#8]
1004:  SUB  X2, X4, X5
1008:  AND  X9, X10, X11
1012:  ORR  X16,X17, X18
1016:  ADD  X13,X14, X31
```

IF: ADD X13,X14,X31 ID: ORR X16,X17,X18 EX: AND X9,X10,X11 MEM: SUB X2,X4,X5

WB:LDUR X8,[X29,#8]



Register	Values
IF/ID	Instruction = ADD
ID/EX	Read data1 = 117 Read data2 = 118 WB-Reg addr = X16 Control
EX/MEM	ALUout = 110 WB-Reg addr = X9
MEM/WB	ReadData M = -1 WB-Reg addr = X2
Regfile (WB)	Write register address = X8 Write data = 99

cycles	1	2	3	4	5	6	7
PC:1000	F	D	E	M	W		
PC:1004		F	D	E	M		
PC:1008			F	D	E		
PC:1012				F	D		
PC:1016					F		

Pipelining performance benefit recap

	1	2	3	4	5	6	7	8	9
`LDUR X8, [X29,#8]`	IF	ID	EX	MEM	WB				
`SUB  X2, X4, X5`		IF	ID	EX	MEM	WB			
`AND  X9, X10, X11`			IF	ID	EX	MEM	WB		
`ORR  X16,X17, X18`				IF	ID	EX	MEM	WB	
`ADD  X13,X14, X31`					IF	ID	EX	MEM	WB

- After cycle 5, we complete 1 instruction per cycle ($\text{CPI}_{\text{once-full}}$ -> 1)
- Pipelining does *not* improve instruction latency
  - Instruction latency often slightly worse than single-cycle datapath
- Pipelining improves instruction throughput
  - ideal speedup = pipeline depth

Does an N-stage pipeline always guarantee Nx speedup?