实验五 流水线CPU设计

2022春季

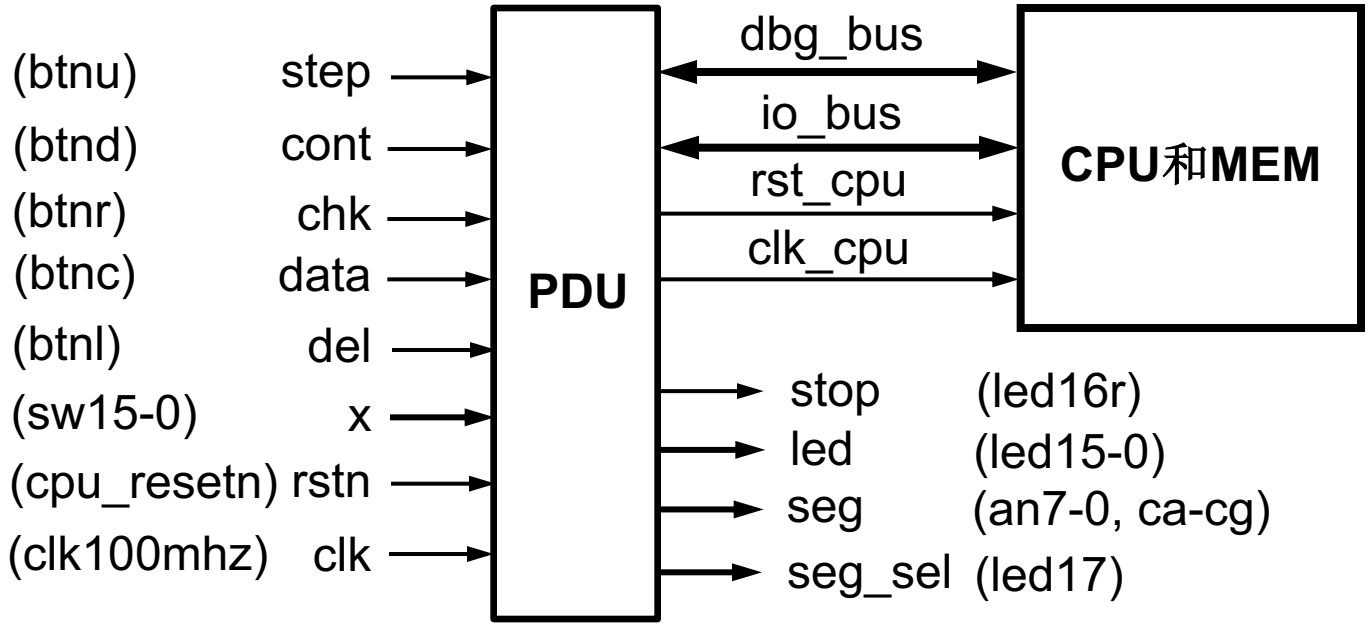
zjx@ustc.edu.cn

实验目标

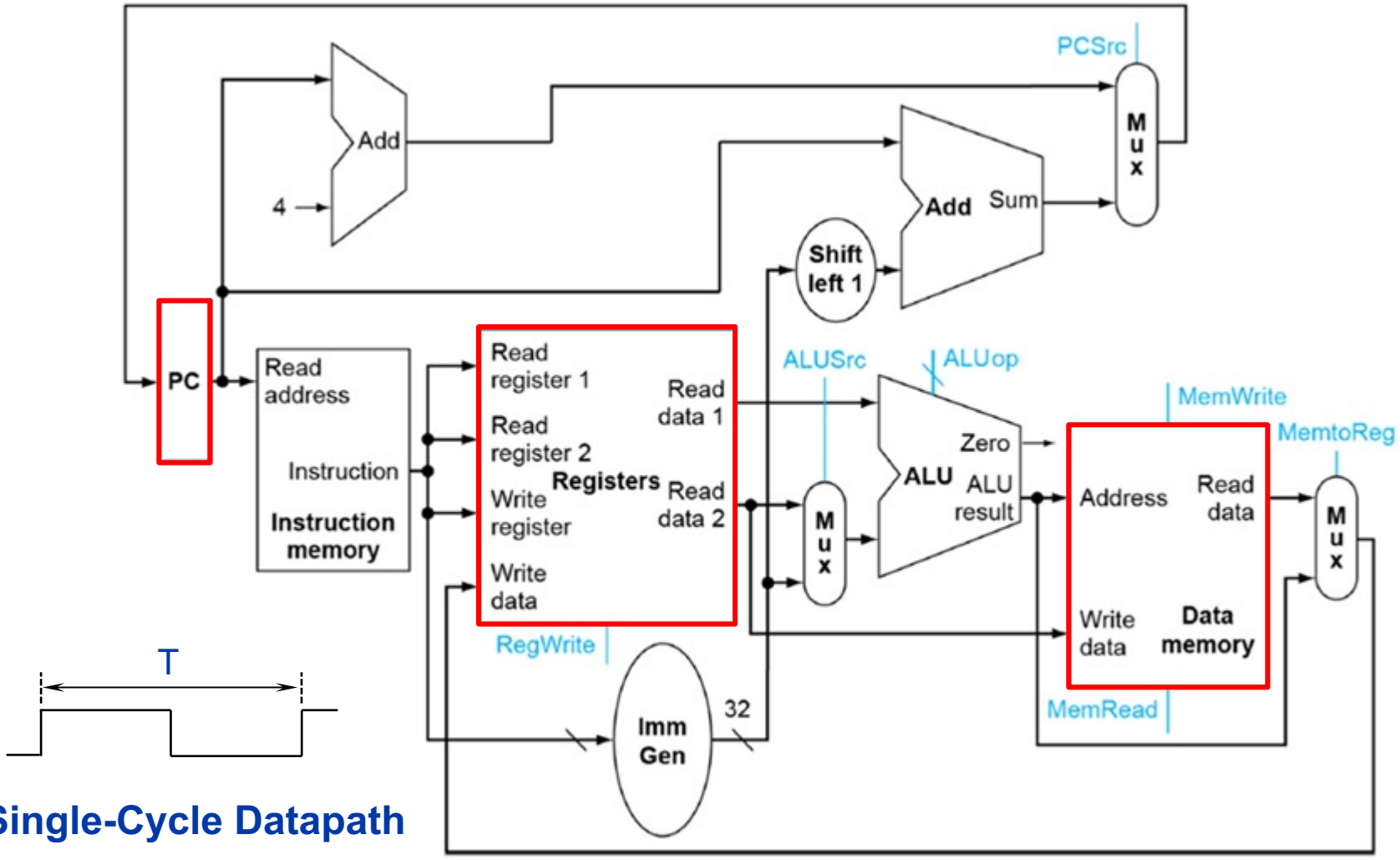
- 理解流水线CPU的结构和工作原理
- 掌握流水线CPU的设计和调试方法，特别是流水线中的数据相关和控制相关的处理
- 熟练掌握数据通路和控制器的设计和描述方法

实验内容

- **设计5级流水线的RISC-V CPU，可执行以下10条指令**
  - add, addi, sub, auipc, lw, sw, beq, blt, jal, jalr
- **配合外设和调试单元PDU，实现对CPU的下载测试**

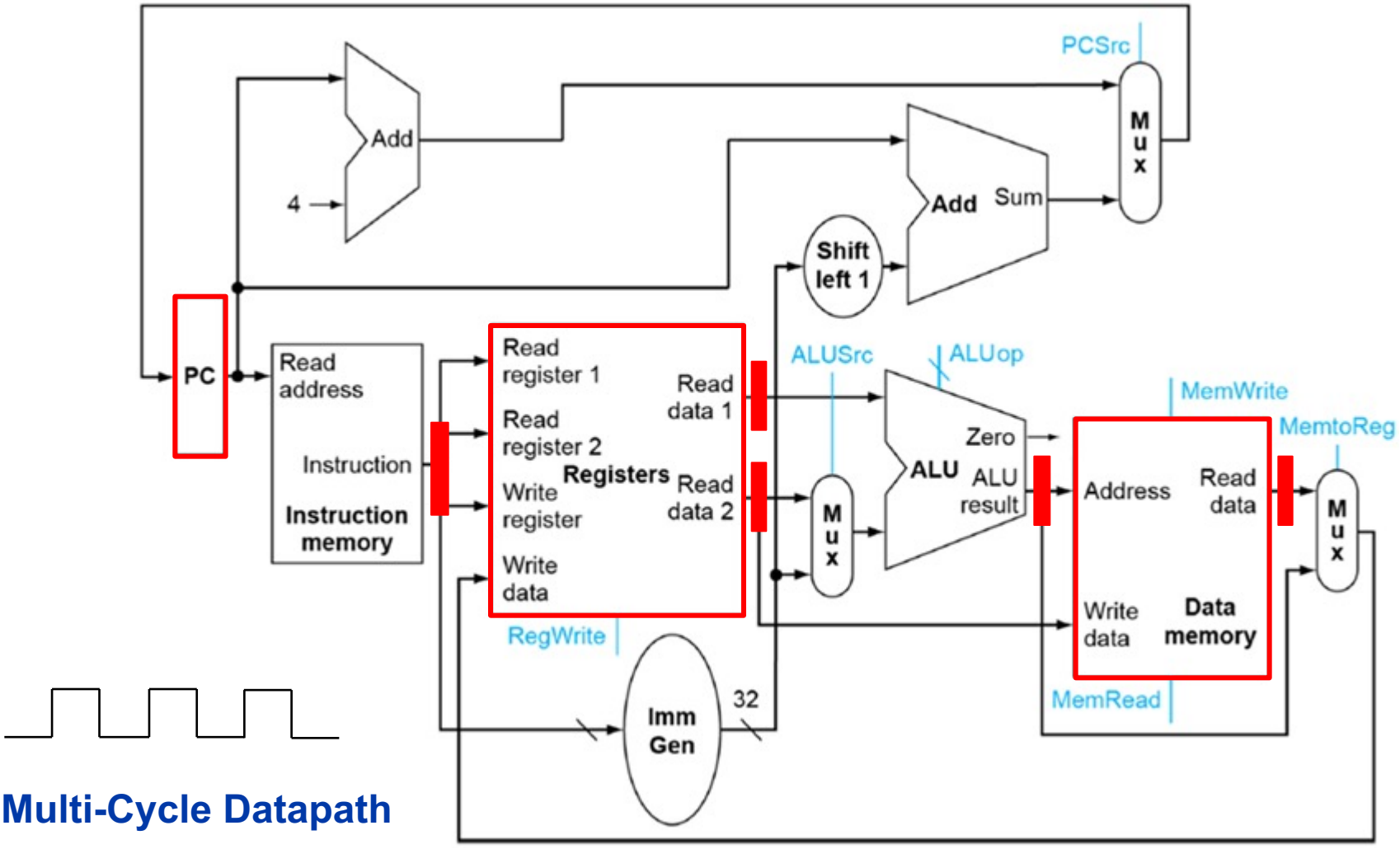


单周期CPU数据通路



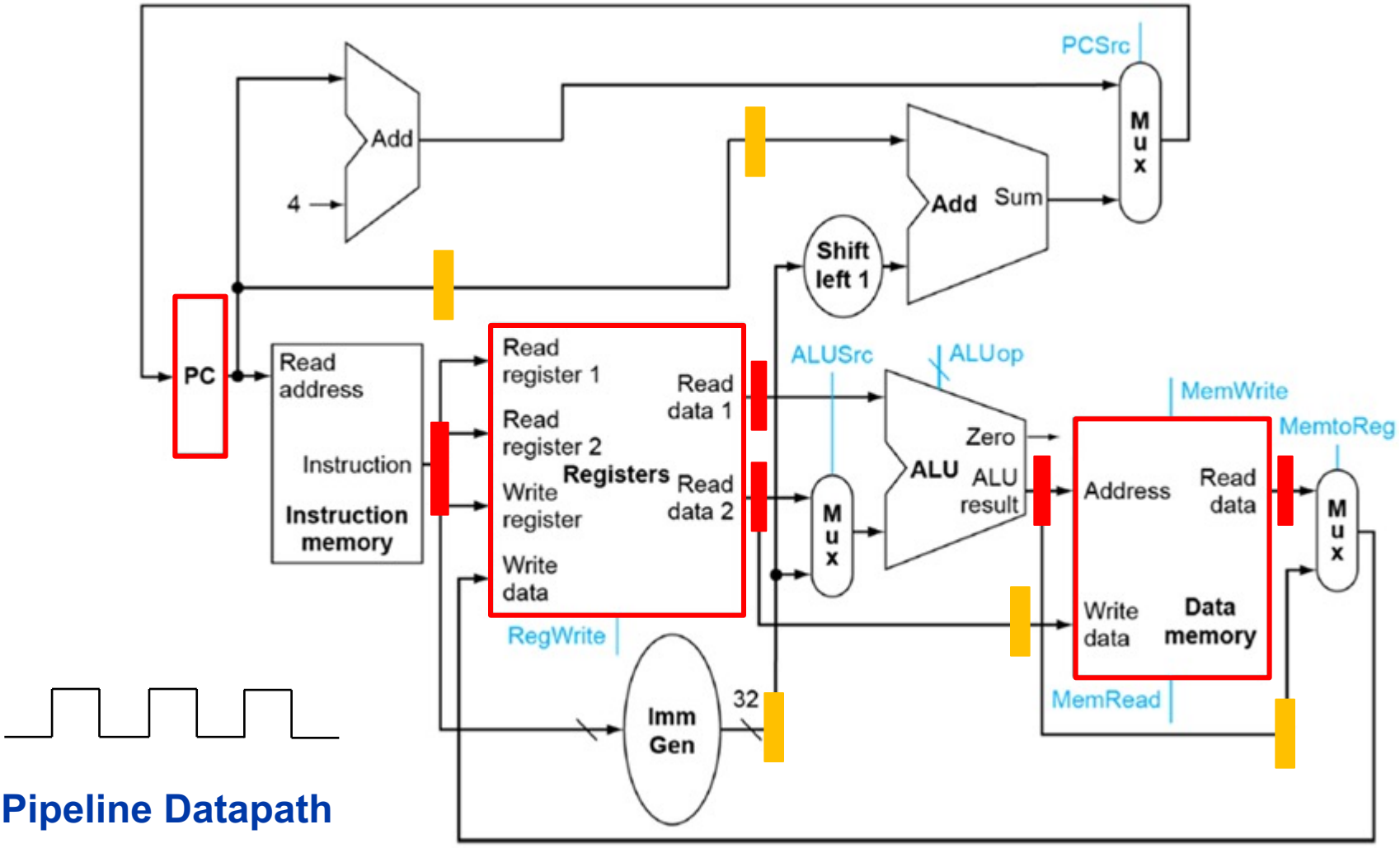
Single-Cycle Datapath

多周期CPU数据通路



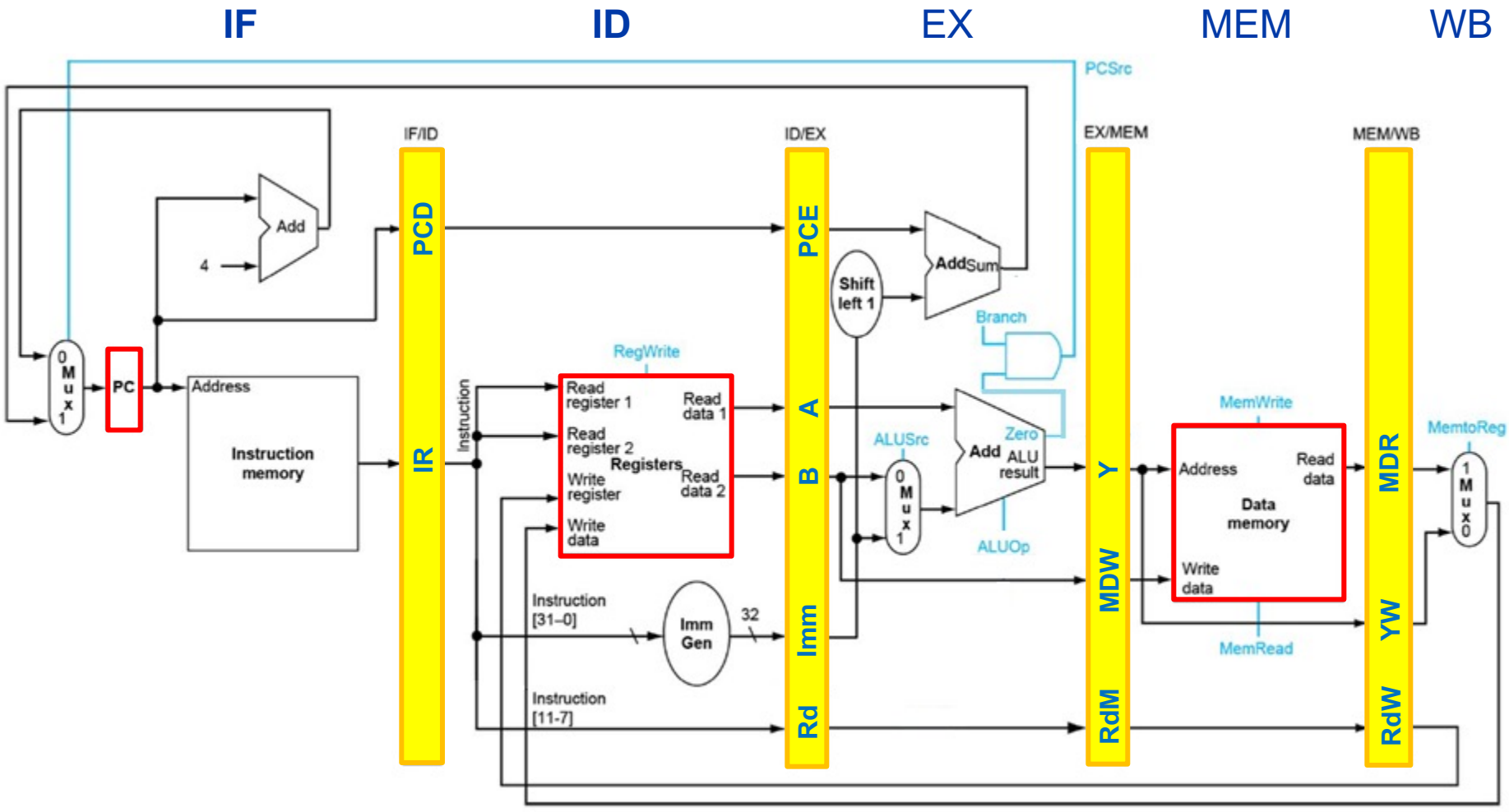
Multi-Cycle Datapath

流水线CPU数据通路

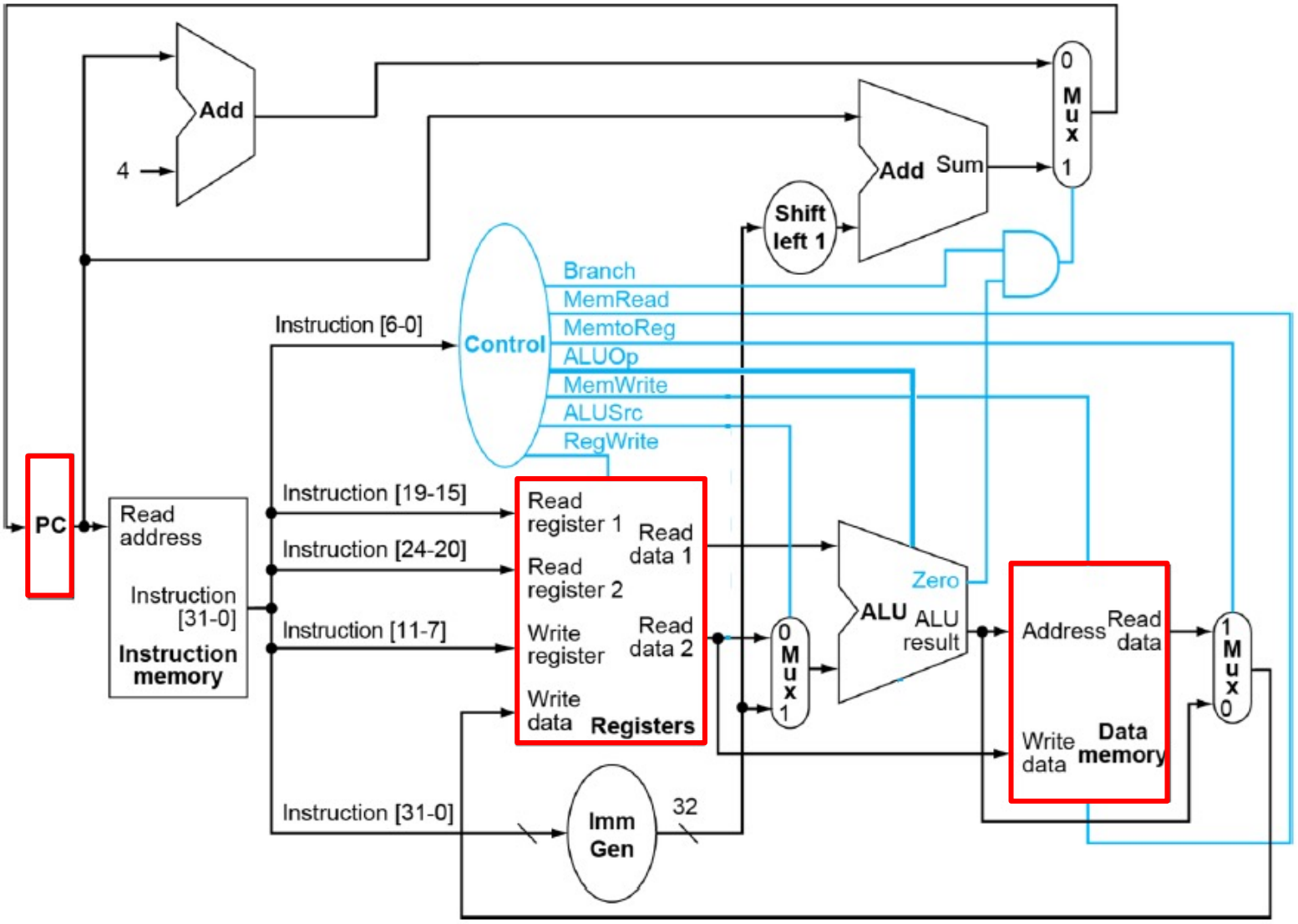


Pipeline Datapath

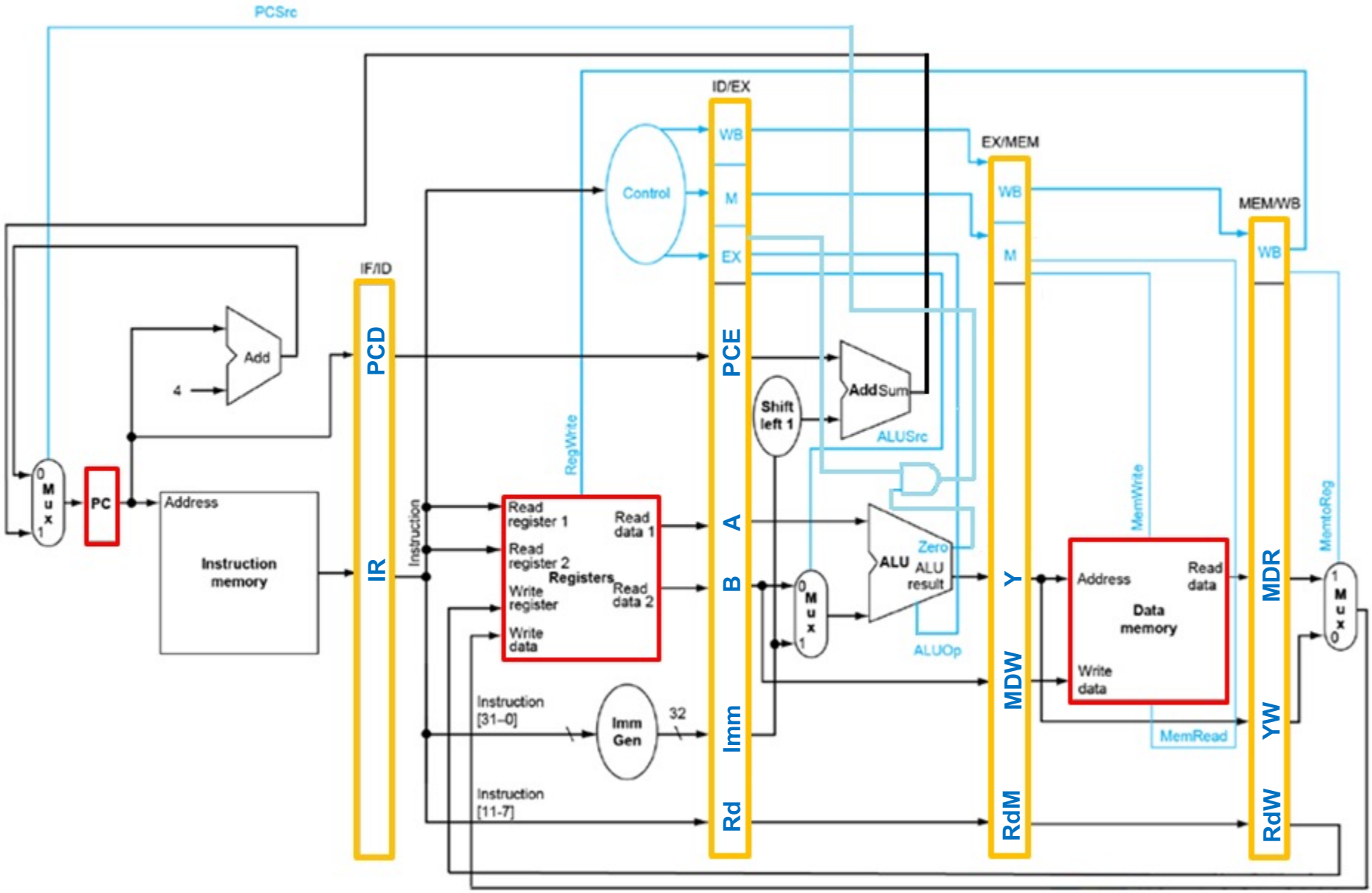
流水线CPU数据通路


IF
ID
EX
MEM
WB
PCSrc
IF/ID
ID/EX
EX/MEM
MEM/WB
Add
4
M u x
0
1
PC
Address
Instruction memory
PCD
IR
Instruction
RegWrite
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
PCE
A
B
Imm
Rd
Instruction [31–0]
Imm Gen
32
Instruction [11-7]
Shift left 1
AddSum
Branch
ALUSrc
Add
Zero
ALU result
ALUOp
Y
MDW
RdM
MemWrite
Address
Read data
Data memory
Write data
MemRead
MDR
YW
RdW
MemtoReg

单周期CPU数据通路+控制器



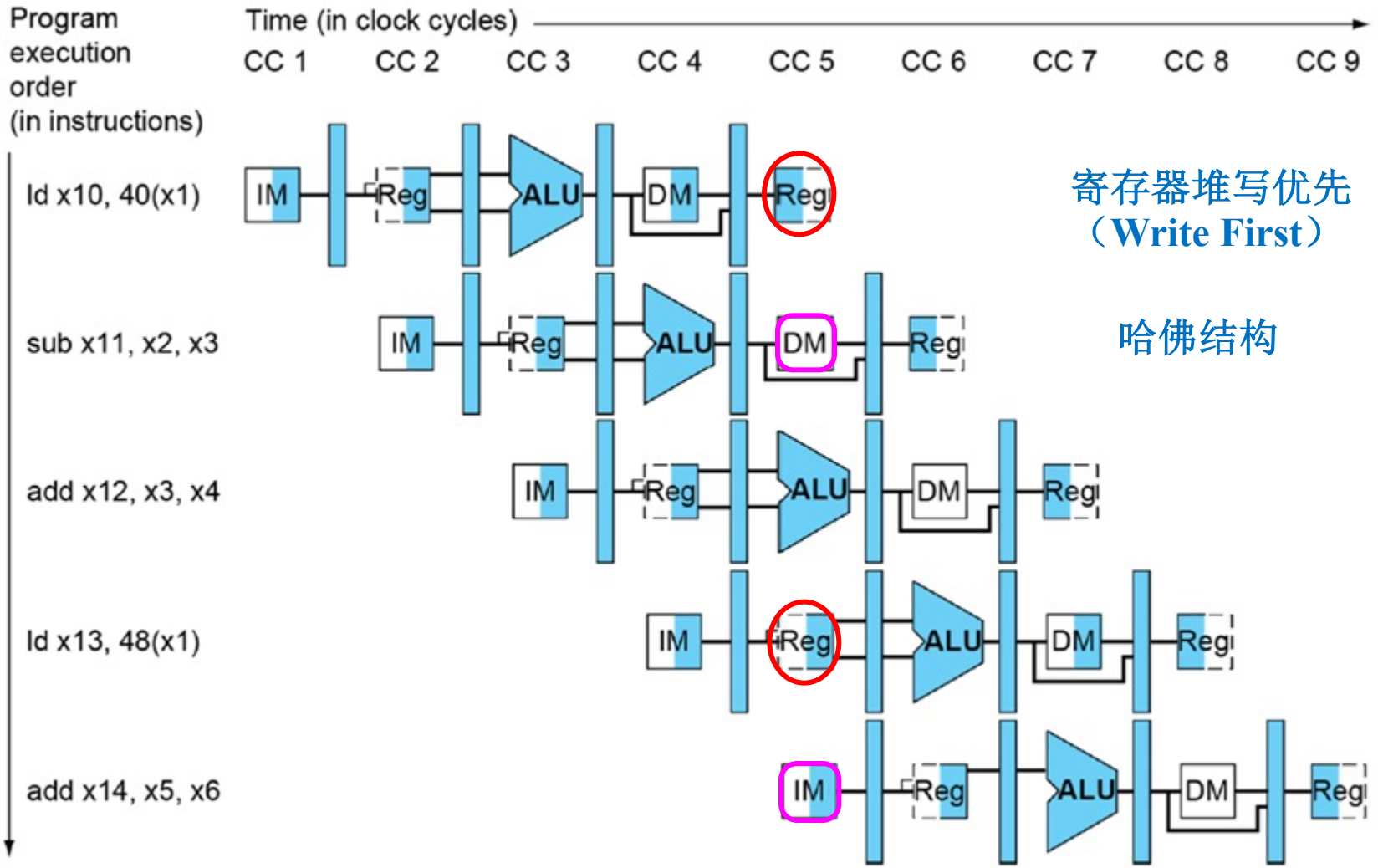
流水线CPU数据通路+控制器



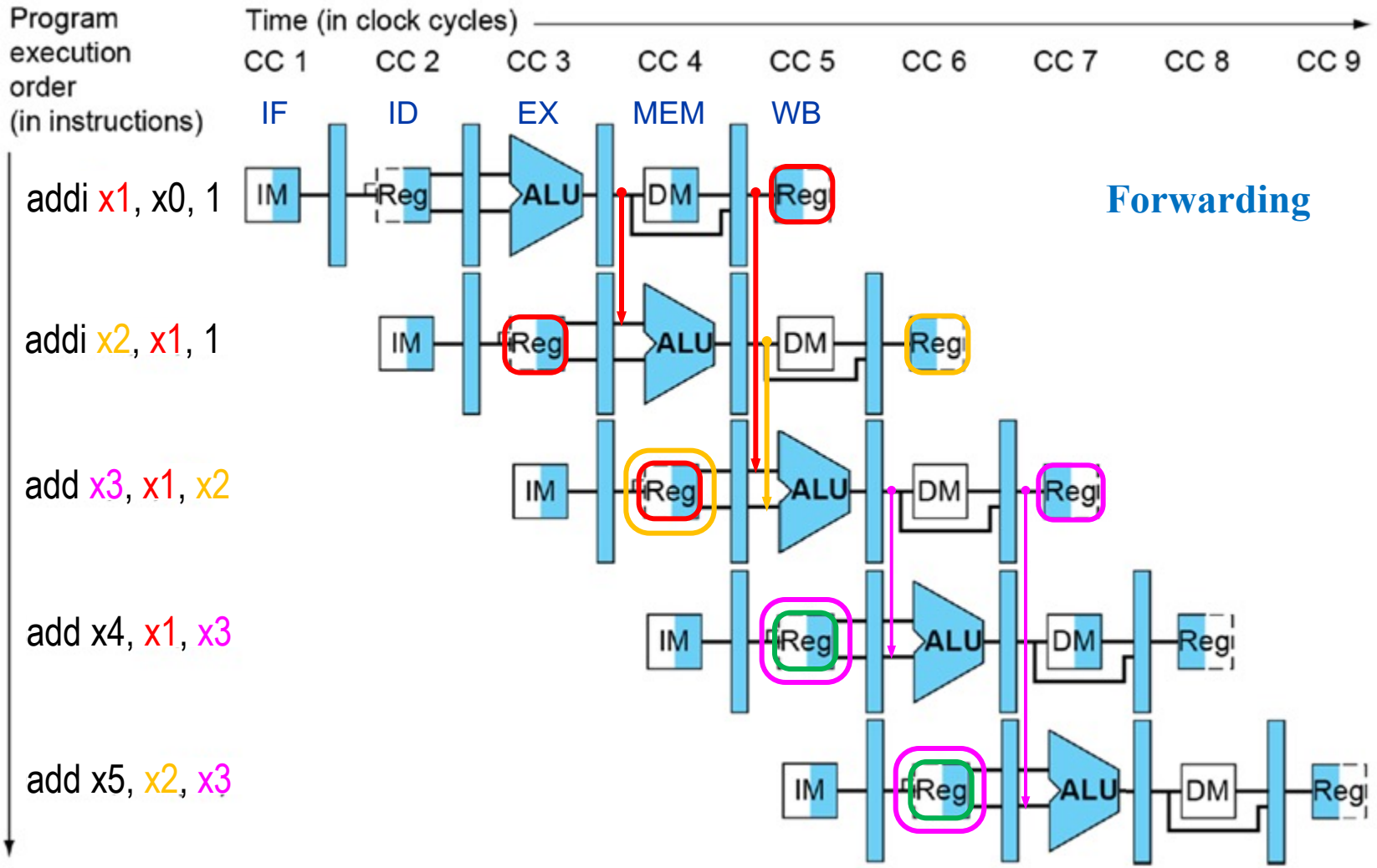
流水线相关及其处理

- **结构相关：当多条指令执行时竞争使用同一资源时**
  - 存储器相关处理：哈佛结构（指令和数据存储器分开）
  - 寄存器堆相关处理：同一寄存器读写时，写优先（Write First）
- **数据相关：当一条指令需要等待前面指令的执行结果时**
  - 数据定向（Forwarding）：将执行结果提前传递至之前流水段
  - 加载-使用相关（Load-use hazard）：阻止紧随Load已进入流水线的指令流动（Stall），向后续流水段插入空操作（Bubble）
- **控制相关：当遇到转移指令且不能继续顺序执行时**
  - 清除（Flush）紧随转移指令已进入流水线的指令
  - 从转移目标处取指令后执行

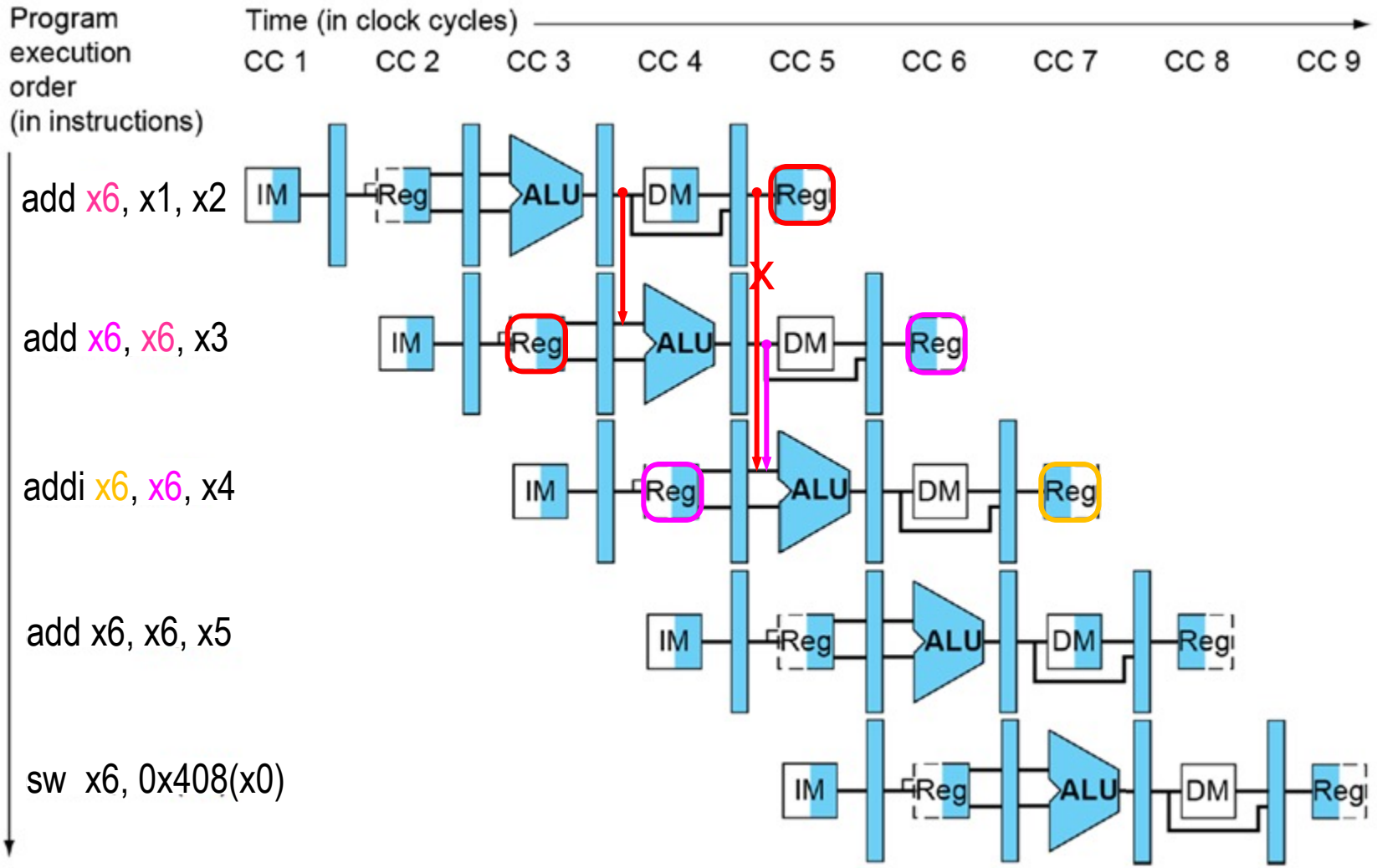
流水线相关：结构相关


Program execution order (in instructions)
Time (in clock cycles)
CC 1
CC 2
CC 3
CC 4
CC 5
CC 6
CC 7
CC 8
CC 9
ld x10, 40(x1)
sub x11, x2, x3
add x12, x3, x4
ld x13, 48(x1)
add x14, x5, x6
IM
Reg
ALU
DM
Reg
寄存器堆写优先
（Write First）
哈佛结构

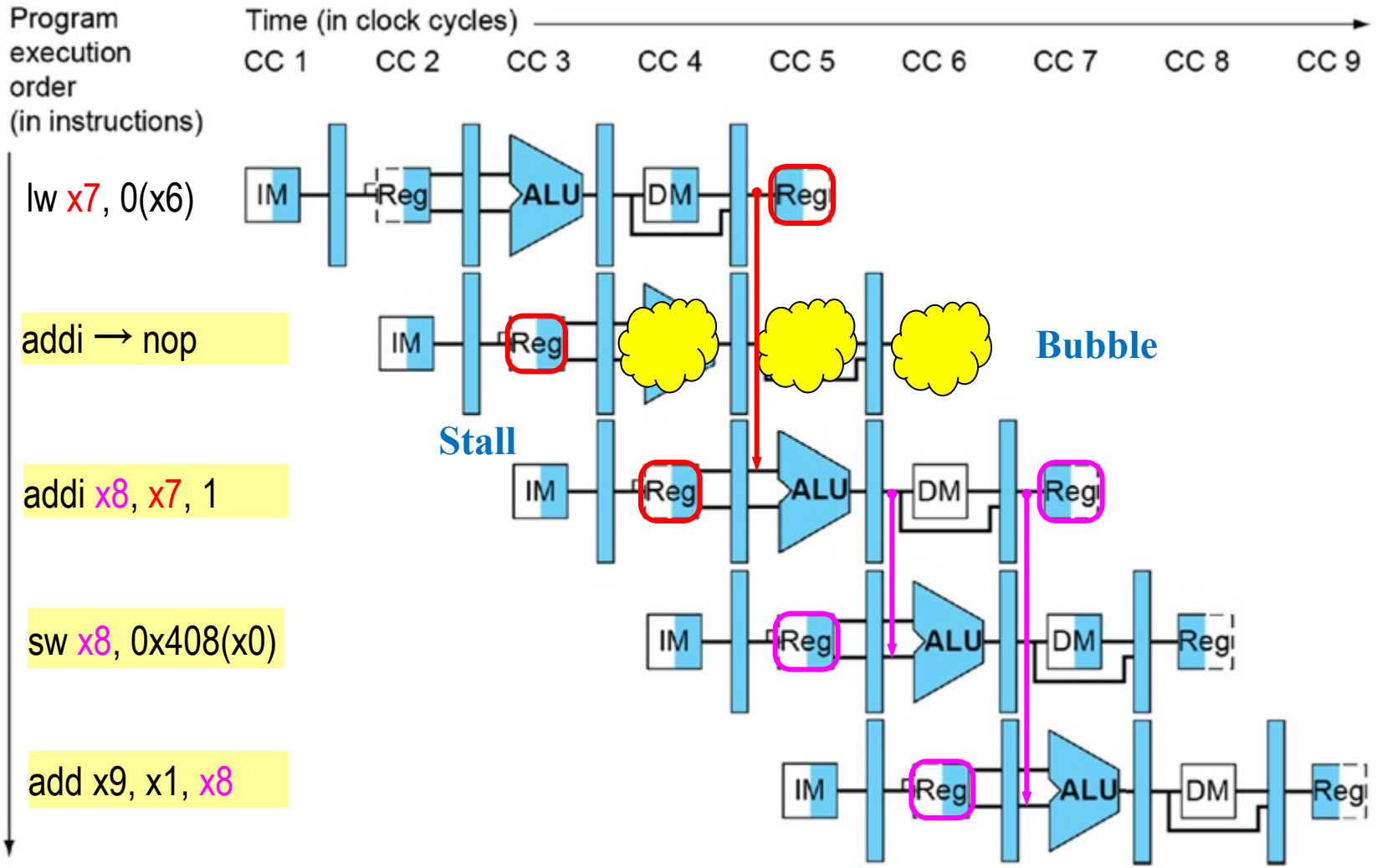
流水线相关：数据相关


Program execution order (in instructions)
Time (in clock cycles)
CC 1
CC 2
CC 3
CC 4
CC 5
CC 6
CC 7
CC 8
CC 9
IF
ID
EX
MEM
WB
addi x1, x0, 1
addi x2, x1, 1
add x3, x1, x2
add x4, x1, x3
add x5, x2, x3
IM
Reg
ALU
DM
Reg
Forwarding

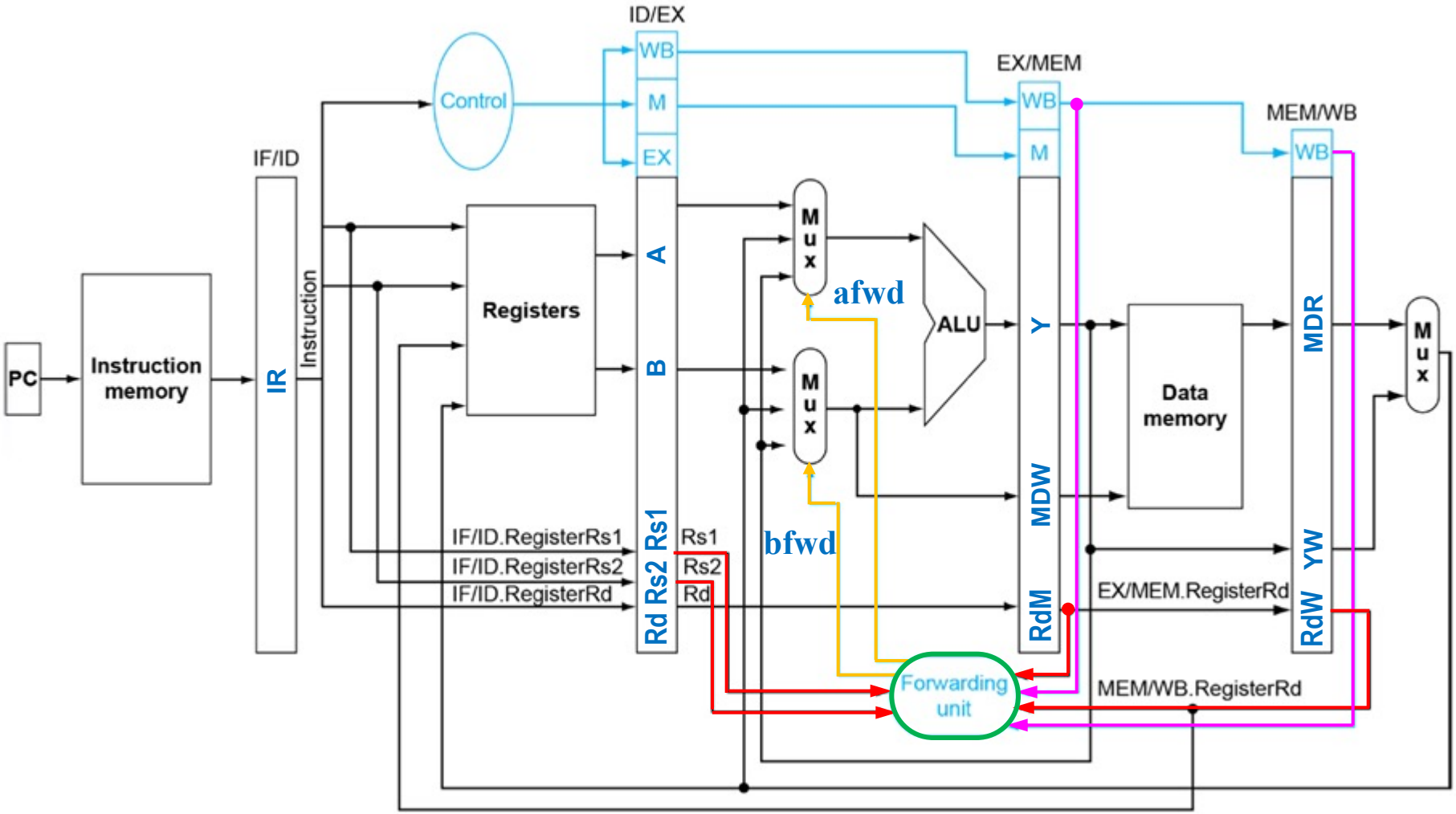
流水线相关：数据相关 (续1)



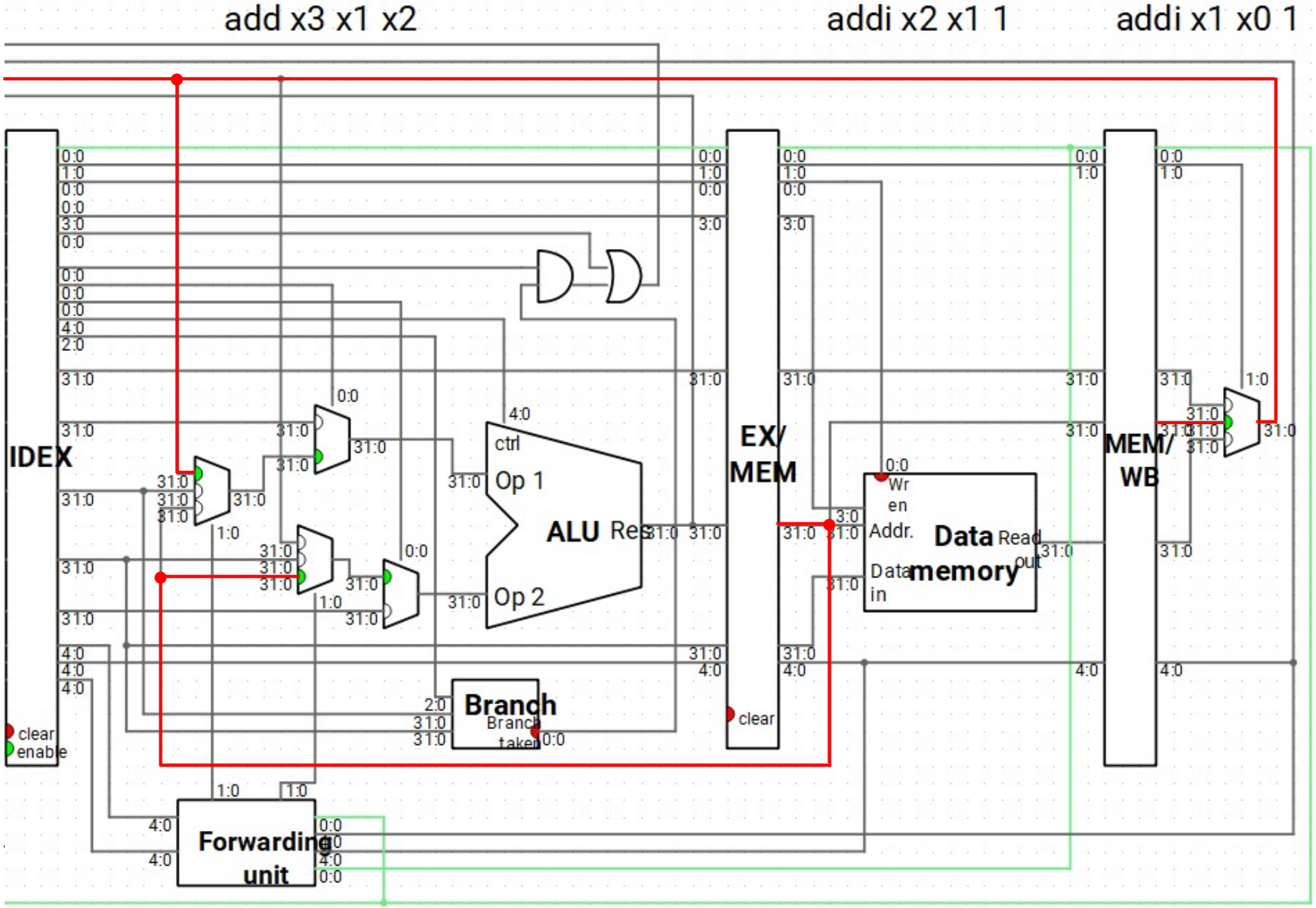
数据相关 (续2)


Program execution order (in instructions)
Time (in clock cycles)
CC 1
CC 2
CC 3
CC 4
CC 5
CC 6
CC 7
CC 8
CC 9
lw x7, 0(x6)
addi → nop
addi x8, x7, 1
sw x8, 0x408(x0)
add x9, x1, x8
IM
Reg
ALU
DM
Reg
Bubble
Stall

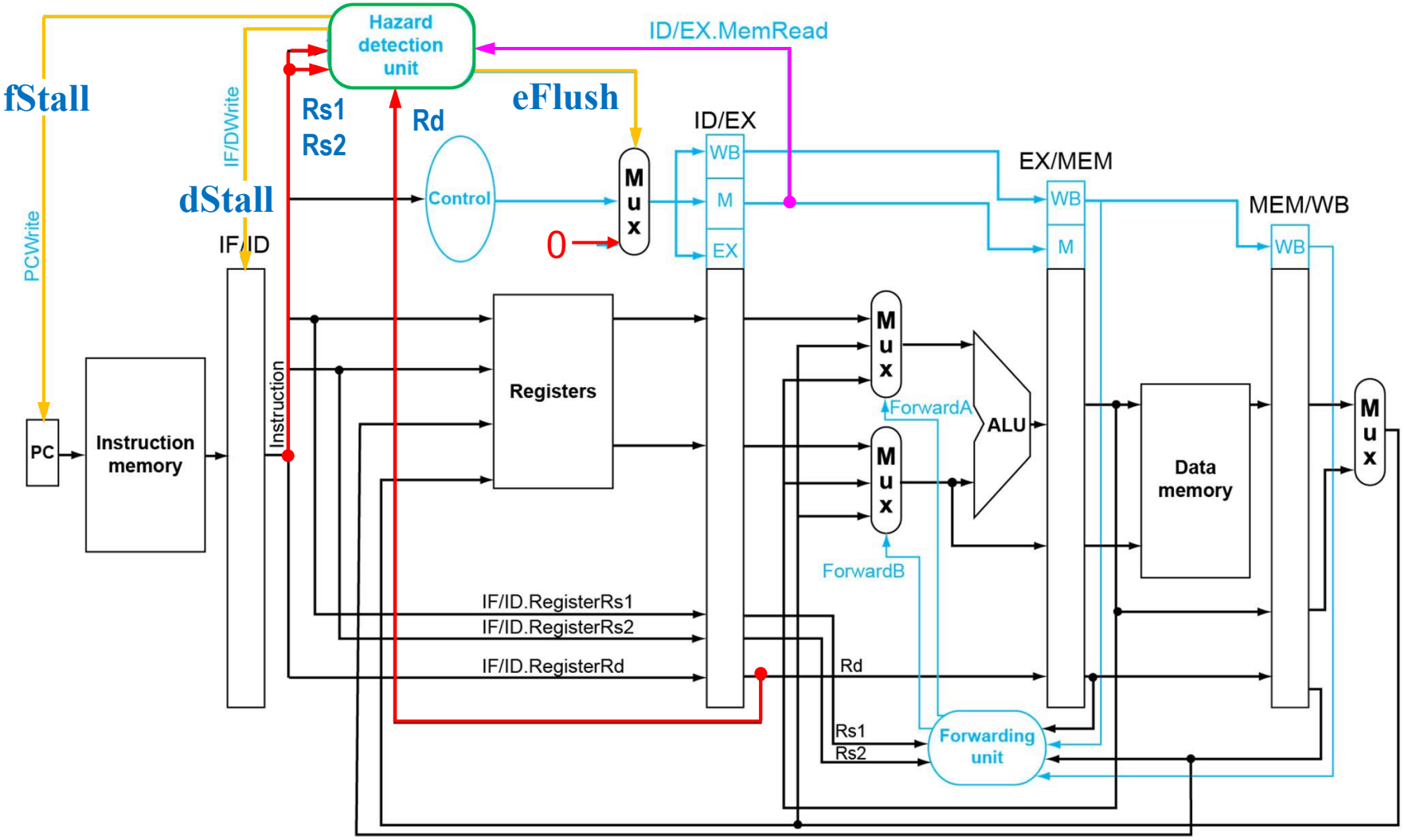
Forwarding


ID/EX
EX/MEM
MEM/WB
WB
M
EX
Control
IF/ID
PC
Instruction memory
IR
Instruction
Registers
A
B
Rd Rs2 Rs1
Mux
afwd
bfwd
ALU
Y
MDW
RdM
Data memory
MDR
YW
RdW
IF/ID.RegisterRs1
IF/ID.RegisterRs2
IF/ID.RegisterRd
Rs1
Rs2
Rd
EX/MEM.RegisterRd
MEM/WB.RegisterRd
Forwarding unit

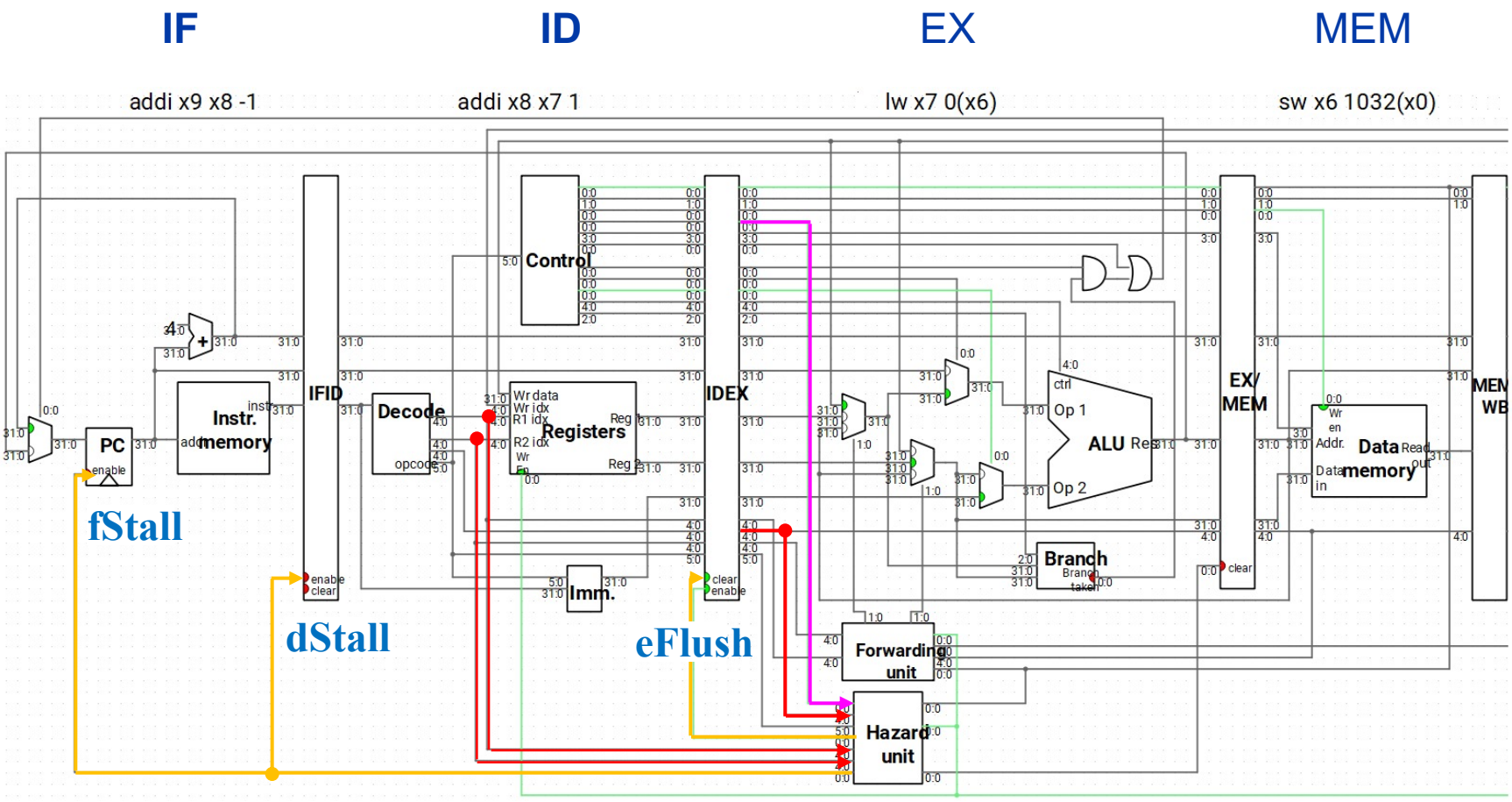
Forwarding：Ripes



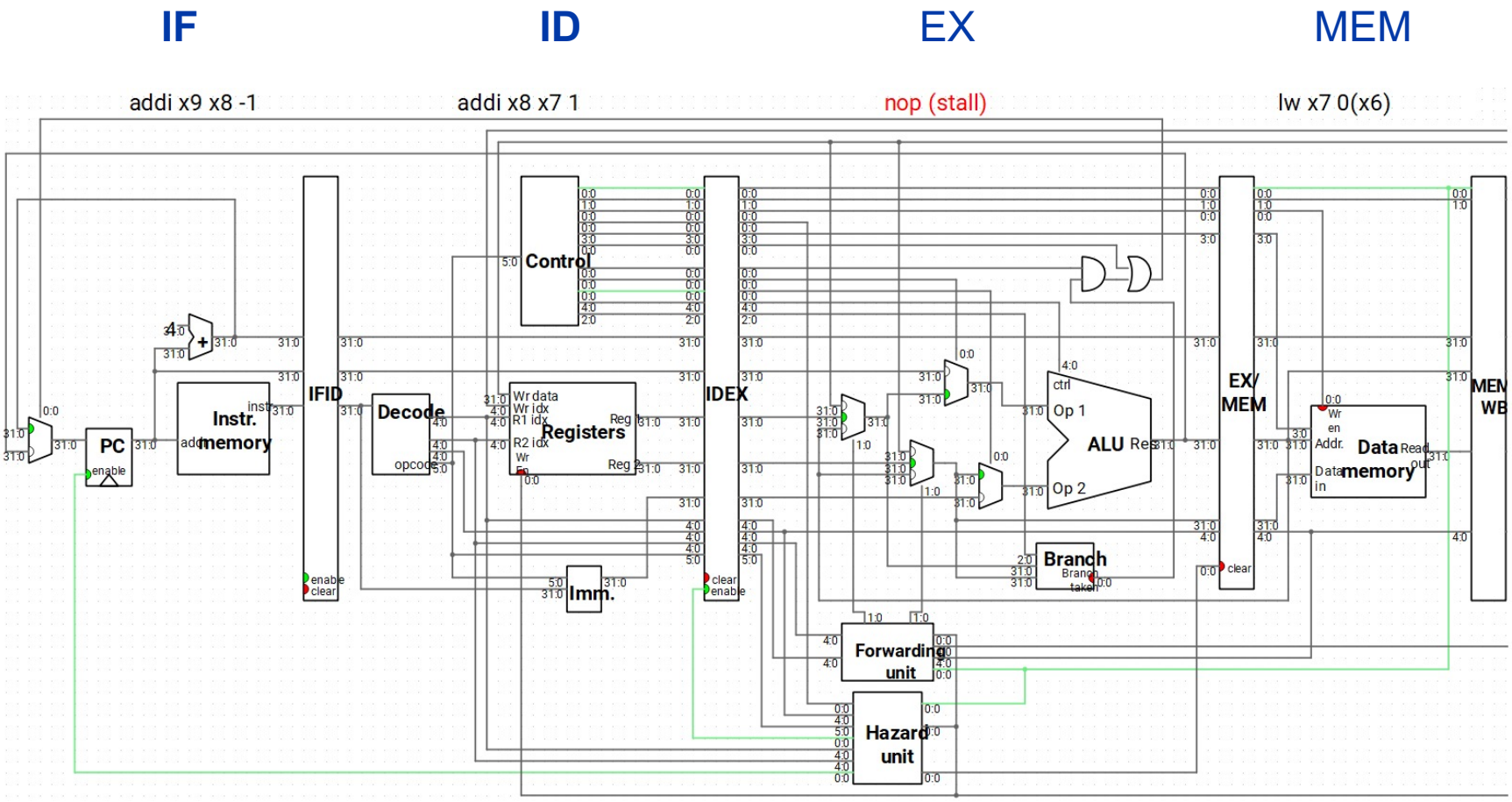
Load-Use Hazard



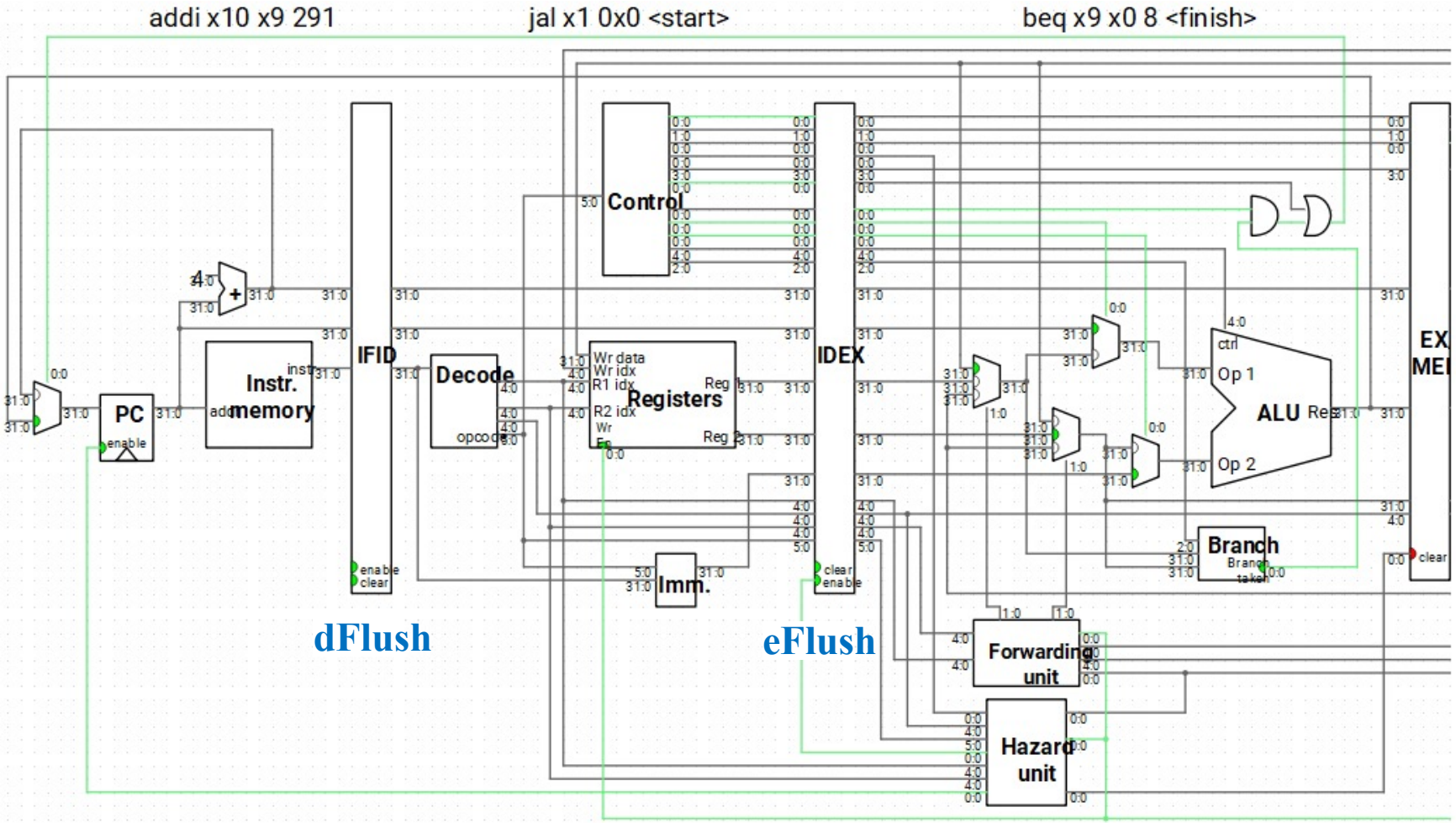
Load-Use Hazard：Ripes



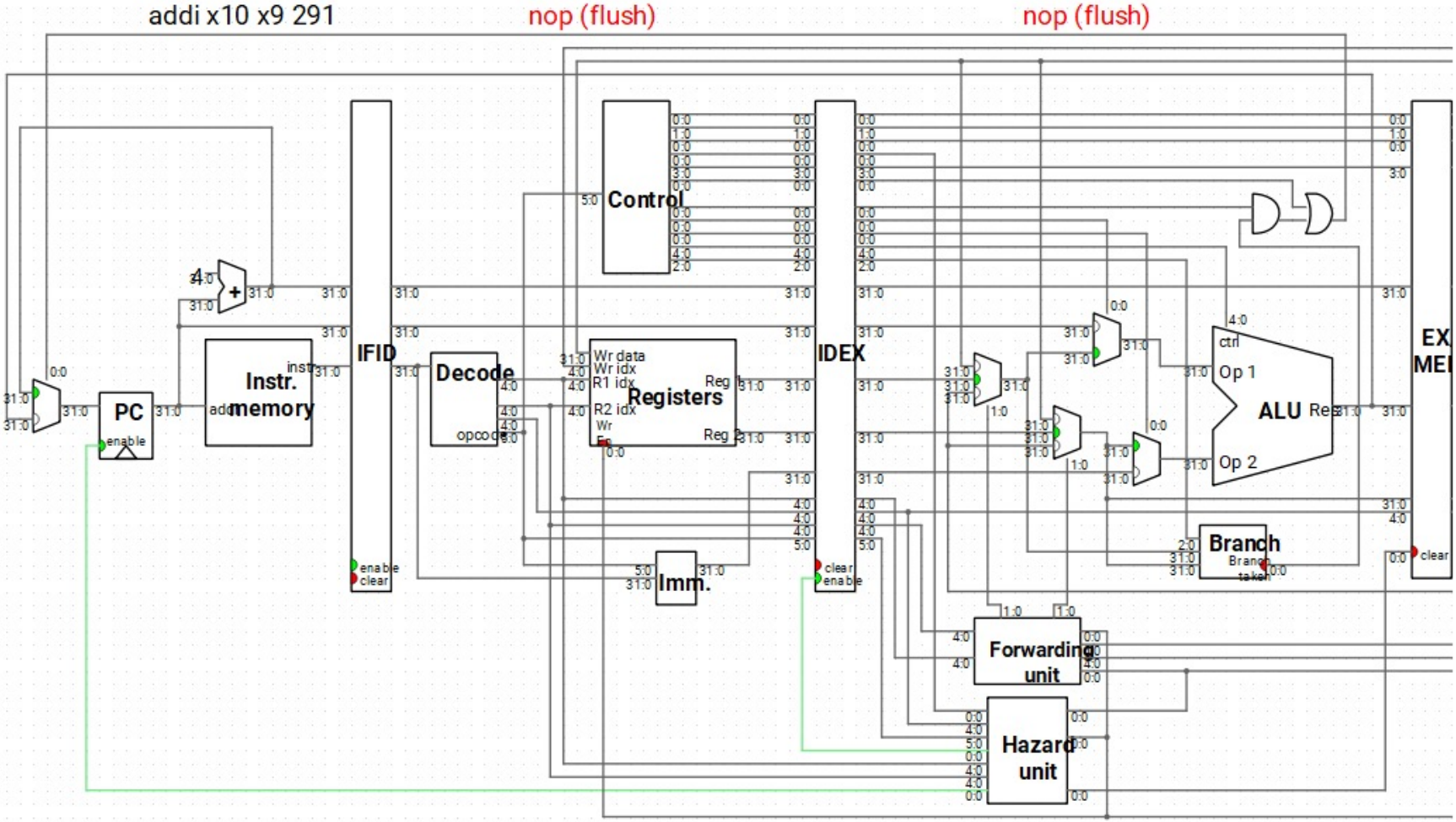
Load-Use Hazard：Ripes



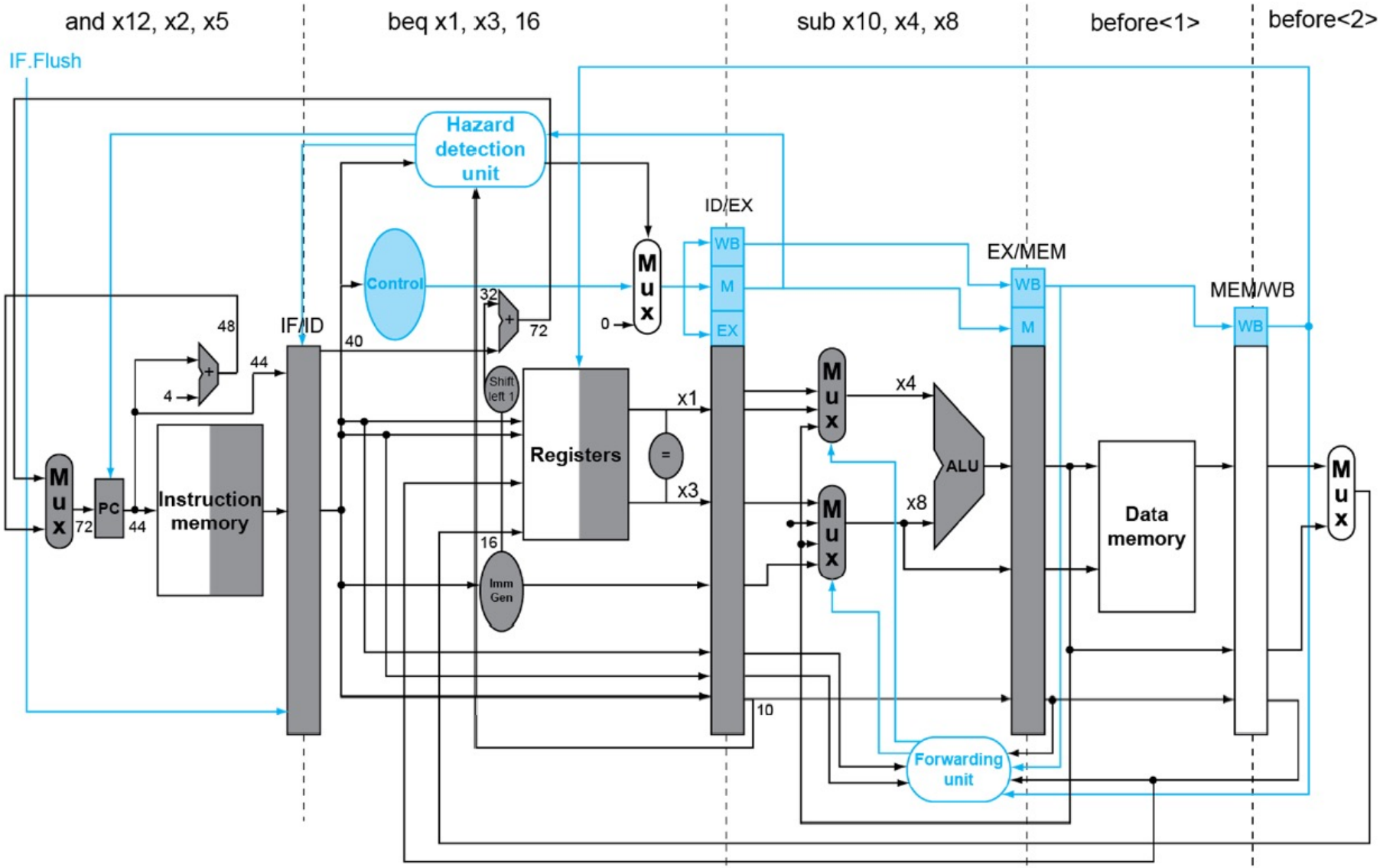
Branch Hazard：Ripes



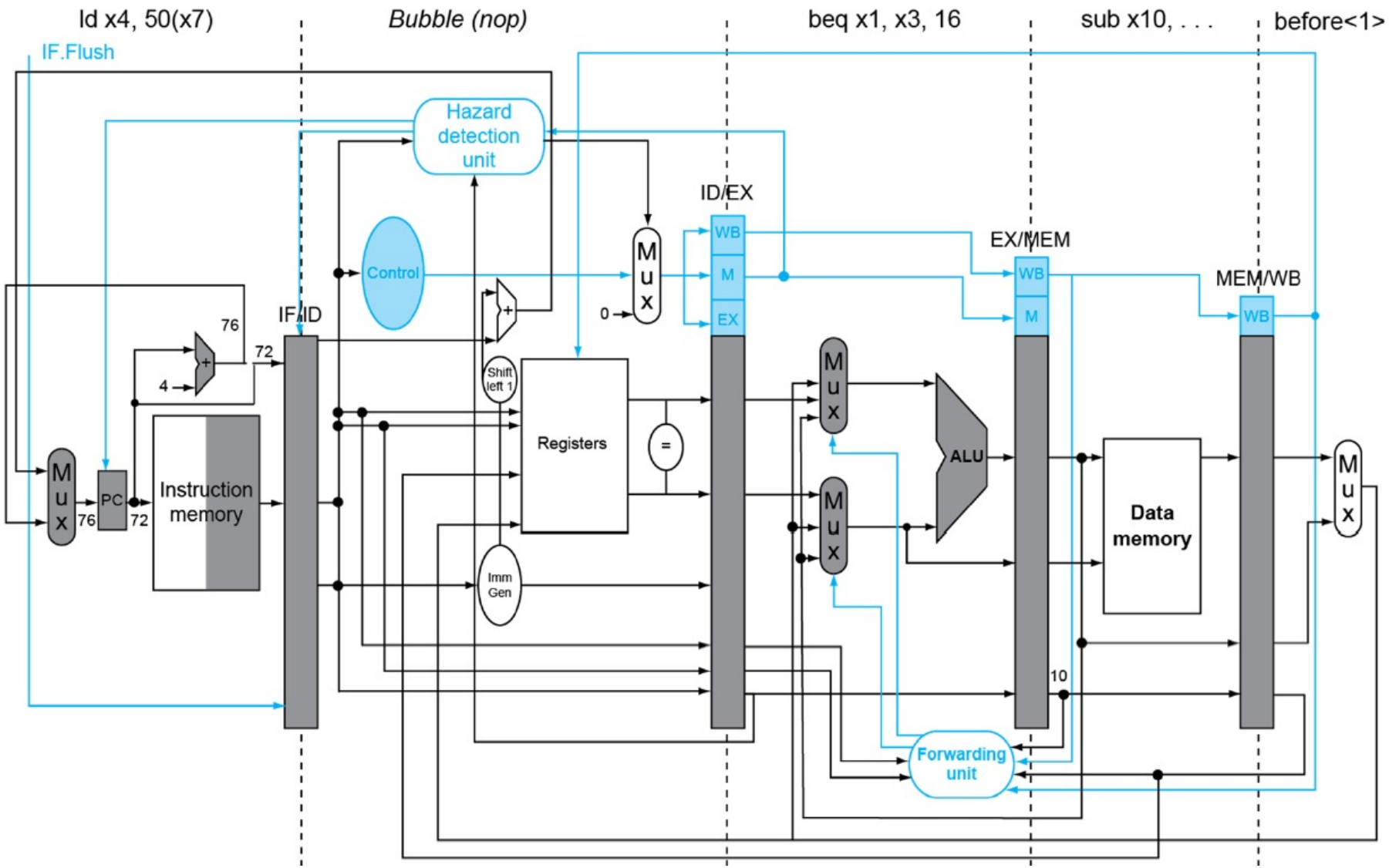
Branch Hazard：Ripes



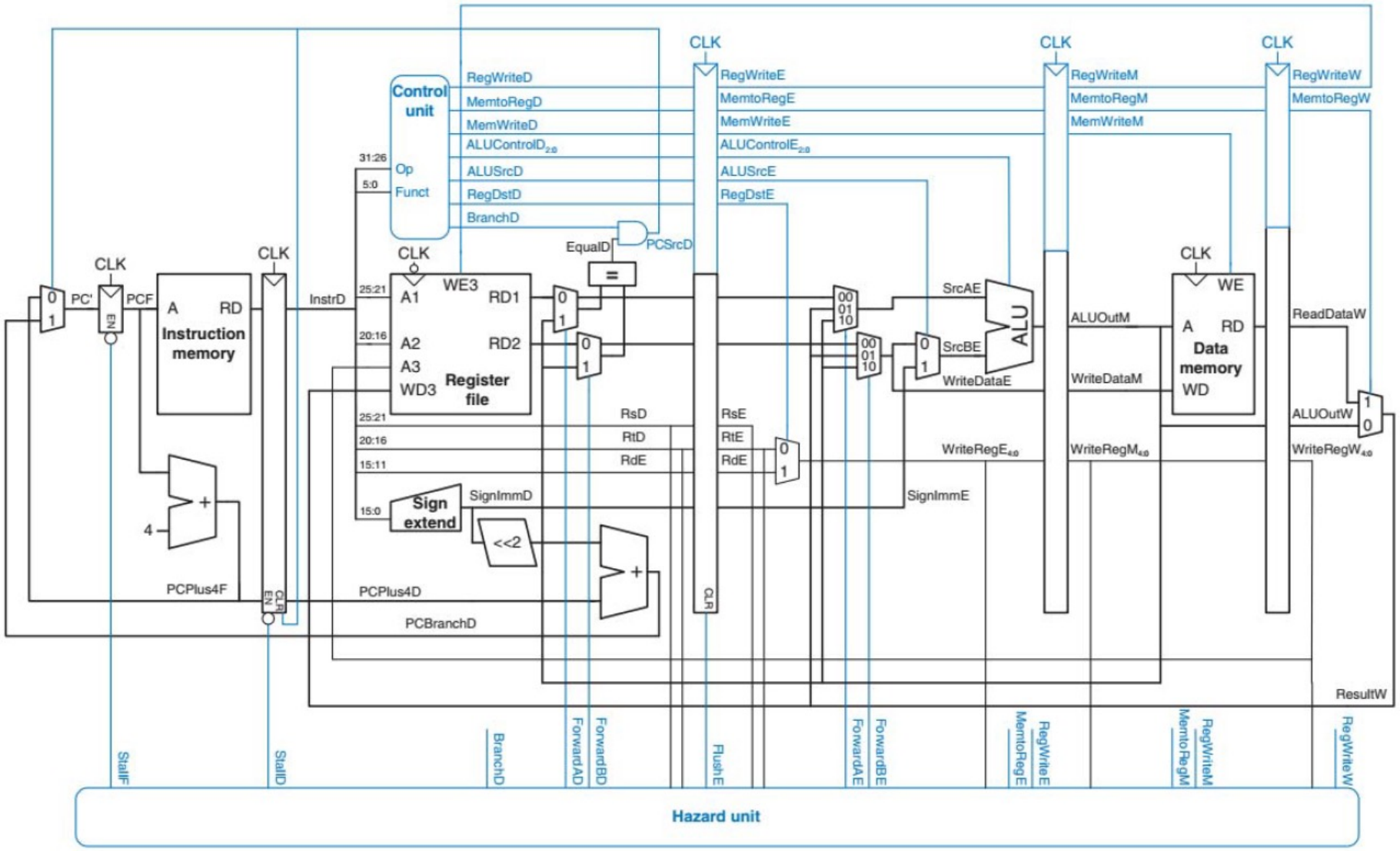
Branch Hazard



Branch Hazard

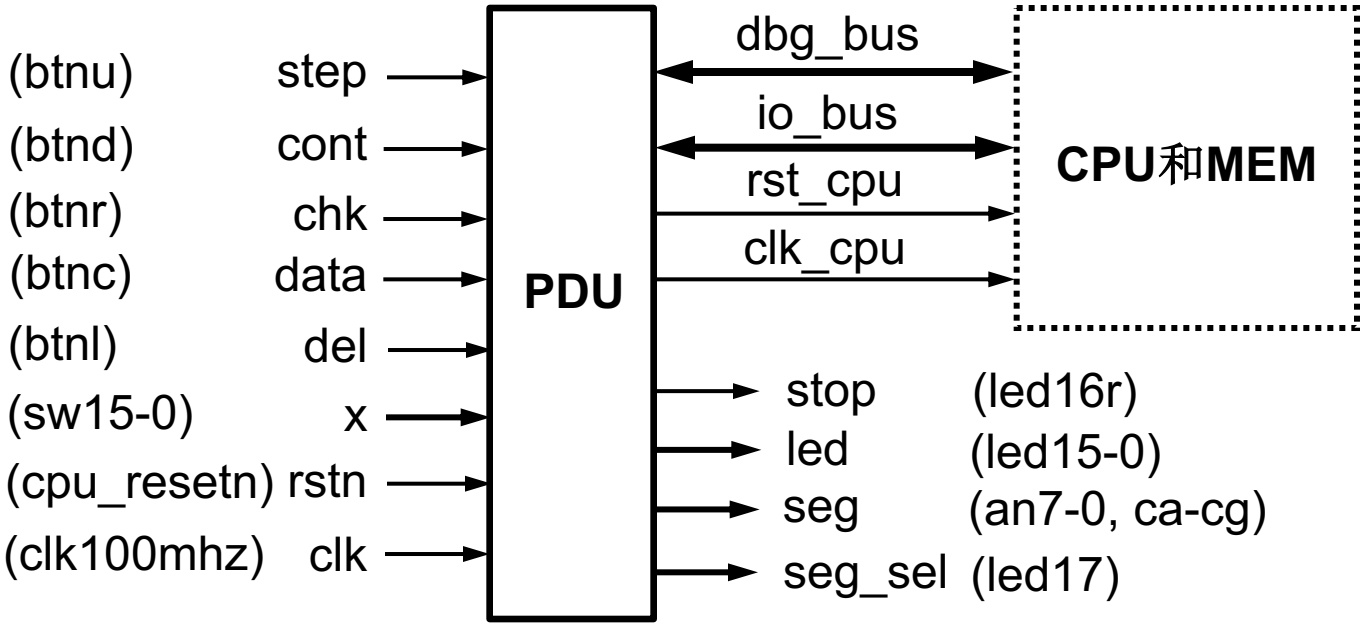


流水线CPU：MIPS


CLK
Control unit
RegWriteD
MemtoRegD
MemWriteD
ALUControlD2:0
ALUSrcD
RegDstD
BranchD
Op
Funct
31:26
5:0
RegWriteE
MemtoRegE
MemWriteE
ALUControlE2:0
ALUSrcE
RegDstE
RegWriteM
MemtoRegM
MemWriteM
RegWriteW
MemtoRegW
EqualD
PCSrcD
PC'
PCF
A
RD
Instruction memory
InstrD
25:21
20:16
15:11
15:0
WE3
A1
A2
A3
WD3
RD1
RD2
Register file
RsD
RtD
RdE
RsE
RtE
RdE
Sign extend
SignImmD
SignImmE
<<2
4
PCPlus4F
PCPlus4D
PCBranchD
SrcAE
SrcBE
ALU
WriteDataE
ALUOutM
WriteDataM
WE
Data memory
WD
ReadDataW
ALUOutW
WriteRegE4:0
WriteRegM4:0
WriteRegW4:0
ResultW
StallF
StallD
BranchD
ForwardAD
ForwardBD
FlushE
ForwardAE
ForwardBE
MemtoRegE
RegWriteE
MemtoRegM
RegWriteM
RegWriteW
Hazard unit

外设和调试单元

- **PDU：Peripherals and Debug Unit**
  - **控制CPU运行方式，查看数据通路状态**
  - **管理外设 (开关sw、指示灯led、数码管seg、计数器cnt等)，实现基本输入/输出**



CPU运行调试

- **控制CPU运行方式**
  - step: 单步运行，按动step，CPU执行一个时钟周期停止(stop = 1)
  - cont: 连续运行，利用x和del编辑断点地址 (brk_addr)，按动cont，CPU连续运行 (stop = 0)，直至chk_pc = brk_addr后停止 (stop = 1)
- **查看数据通路状态**
  - 当CPU停止 (stop = 1) 时，利用x和del编辑查看地址 (chk_addr)，按动chk，chk_addr和数据通路状态(chk_data)分别显示在指示灯led和数码管seg上，再次单独按动chk，将顺序显示后续信息
- **调试信号DBG_BUS**
  - chk_pc：输入，32位，监测执行指令地址 = pcd
  - chk_addr：输出，16位，数据通路状态的编码地址
  - chk_data：输入，32位，数据通路状态的数据

查看数据通路状态

- **数据通路地址编码**
  - chk_addr：4位16进制数，最高位区分查看数据类型（寄存器堆RF、数据存储器DM、PC及其他)，余下位表示具体地址

chk_addr	chk_data
0 0xx	pcs
1 0yy	RF
2 zzz	DM

xx：流水段寄存器编号

yy：寄存器堆地址

zzz：数据存储器地址

x, y, z：十六进制数字

xx	chk_data	xx	chk_data
00	pcin	0A	ctrlm
01	pc	0B	y
02	pcd	0C	mdw
03	ir	0D	irm
04	ctrl	0E	ctrlw
05	pce	0F	mdr
06	a	10	yw
07	b	11	irw
08	imm		
09	ire		

CPU_P模块接口

```
module  cpu_p (
  input clk,
  input rst,

  //IO_BUS
  output [7:0]  io_addr,      //外设地址
  ouput [31:0]  io_dout,      //向外设输出的数据
  output  io_we,              //向外设输出数据时的写使能信号
  output  io_rd,              //从外设输入数据时的读使能信号
  input [31:0]  io_din,       //来自外设输入的数据

  //Debug_BUS
  output [31:0] chk_pc,       //监测执行指令地址 = pcd
  input [15:0] chk_addr,      //数据通路状态的编码地址
  output [31:0] chk_data     //数据通路状态的数据
);
```

PDU模块接口

```
module  pdu (
  input clk,            //clk100mhz
  input rstn,           //cpu_resetn

  input step,           //btnu
  input cont,           //btnd
  input chk,            //btnr
  input data,           //btnc
  input del,            //btnl
  input [15:0]  x,      //sw15-0

  output stop,                  //led16r
  output [15:0] led,            //led15-0
  output [7:0] an,              //an7-0
  output [6:0] seg,             //ca-cg
  output [2:0] seg_sel,         //led17

  output clk_cpu,       //cpu's clk
  output rst_cpu,       //cpu's rst

  //IO_BUS
  input [7:0] io_addr,
  input [31:0] io_dout,
  input io_we,
  input io_rd,
  output [31:0] io_din,

  //Debug_BUS
  input [31:0] chk_pc,
  output [15:0] chk_addr,
  input [31:0] chk_data
);
```

实验步骤

1. 设计无数据和控制相关处理的流水线CPU
2. 设计仅有数据相关处理的流水线CPU
3. 设计完整的有数据和控制相关处理的流水线CPU

The End