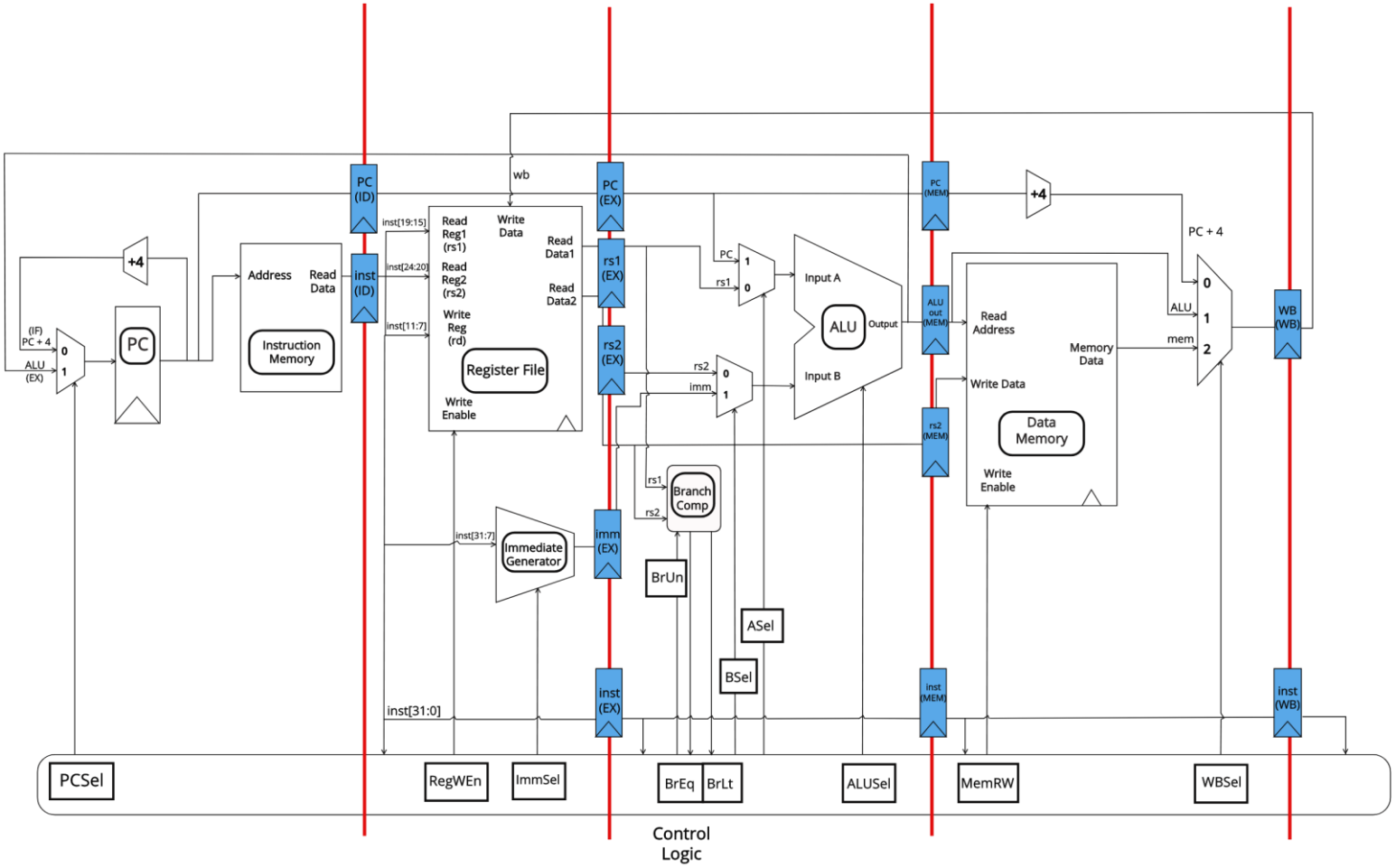
Pipelining Registers 流水线寄存器

（1）为了实现流水线，需要在五个阶段之间加流水线寄存器，在图中写出每个阶段的名称（IF, ID, EX, MEM, and WB)和各个流水线寄存器的名称.



（2）在各个阶段之间加流水线寄存器的目的是什么?

When we pipeline the datapath, the values from each stage need to be passed on at each clock cycle. Each stage in the pipeline only operates on a small set of values, but those values need to be correct with respect to the instruction that is currently being processed. Say we use load word (lw) as an example: if it is in the EX stage, then the EX stage should look like a snapshot of the single-cycle datapath. The values on the rs1, rs2, immediate, and PC values should be as if lw was the only instruction in the entire path. This also includes the control logic: the instruction is passed in at each stage, the appropriate control signals are generated for the stage of interest, and that stage can execute properly.

2. 性能分析

Register clk-to-q 30 ps Register setup 20 ps Mux 25 ps	Branch comp. 75 ps ALU 200 ps Memory read 250 ps	Memory write 200 ps RegFile read 150 ps RegFile setup 20 ps

（1）如上给定各个部件的延迟时间，如果是单周期最快的时钟周期是多少?

tclk ≥ tPC clk-to-q + tIMEM read + tRF read + tmux + tALU + tDMEM read + tmux + tRF setup
≥ 30 + 250 + 150 + 25 + 200 + 250 + 25 + 20
≥ 950 ps
1
950 ps
= 1.05 GHz

（2）如果流水线CPU，最快的时钟周期是多少?

IF : tPC clk-to-q + tIMEM read + tReg setup = 30 + 250 + 20 = 300 ps
ID : tReg clk-to-q + tRF read + tReg setup = 30 + 150 + 20 = 200 ps
EX : tReg clk-to-q + tmux + tALU + tReg setup + tmux = 30 + 25 + 200 + 20 + 25 = 300 ps 答案错误，应该没有最后的mux
MEM : tReg clk-to-q + tDMEM read + tmux + tReg setup = 30 + 250 + 25 + 20 = 325 ps
WB : tReg clk-to-q + tRF setup = 30 + 20 = 50 ps

max(IF, ID, EX, MEM, WB) = 325 ps

分支比较器的时间被ALU计算所掩盖

NOTE: For the EX stage, the branch comparator time is overshadowed by the ALU computation (The same would be true in the ID stage as well, but since there is no mentioned time for Immediate Generator, we assumed here it is trivial):
Branch comparator : tPC clk-to-q + tBranch comp. = 30 + 75 = 105 ps
ALU computation : tReg clk-to-q + tmux + tALU + tReg setup = 25 + 200 = 275 ps

（3）从单周期到流水线CPU，加速比是多少，为什么小于5?

950 ps /325, or a 2.9 times speedup. The speedup is less than 5 because of (1) the necessity of adding pipeline registers, which have clk-to-q and setup times, and (2) the need to set the clock to the maximum of the five stages, which take different amounts of time.
Note: because of hazards, which require additional logic to resolve, the actual speedup would likely be even less than 2.9 times.

（4） 如果某程序其中 25% loads， 10% stores, 11% branches, 2% jumps, 52%, R-type,用上面相同的部件实现多周期CPU，那么它的时钟周期是多少？，与单周期相比加速比多少，流水线与多周期相比，加速比多少？

Tm=325，CPI=(0.11 + 0.02) 3 +(0.52 + 0.10) 4 +(0.25) 5
= 4.12
IC*CPIsTs /IC*CPIs*Tm =950*1/4.12*325=
IC*CPIm*Tm /IC*CPIp*Tp=4.12*325/1*325=4.12

3. 多周期和流水线分成五个阶段:

1. Hardware to support an instruction *fetch*
2. Hardware to support an instruction *decode* (i.e. a register file read)
3. Hardware to support instruction *execution* (i.e. the ALU)
4. Hardware to support a *memory* load or store
5. Hardware to support the *write back* of the ALU operation back to the register file

假设上面每个阶段花费时间如下.

Fetch	Decode	Execute	Memory	Write Back
305 ps	275 ps	280 ps	305 ps	250 ps

（1）那么时钟周期是多少?

Solution:

- The clock period is defined by the longest time … here 305 ps.
- The clock rate is the inverse of the period … here, 3.28 GHz.
    - However, this is NOT the answer I am looking for.

（2）如果是流水线，假设没有阻塞和冒险现象，执行一条指令需要用多少时间？

Solution:

1 instruction would need to proceed through all 5 stages of the pipeline

- Thus, the total time would be 305 ps x 5 stages = 1,525 ps or 1.525×10^{-9} s.

（3）假设执行N条指令，这N条指令全部是add指令，那么流水CPU与多周期CPU相比，加速比是多少？

Solution:

For the multi-cycle approach:

- Each add instruction would take 4 clock cycles and each clock cycle would take 305 ps.
- Thus, the total time would be: 1220(N)

For the pipelined approach:

- For N instructions, we can apply the formula: （5+（N−1））*T
- Thus, the total time would be: ○ = 305(N)
  + (5-1)(305) ○ = 305N + 1220 ps

Thus, the overall speedup is: 1220(N) / [305(N) +

1220]