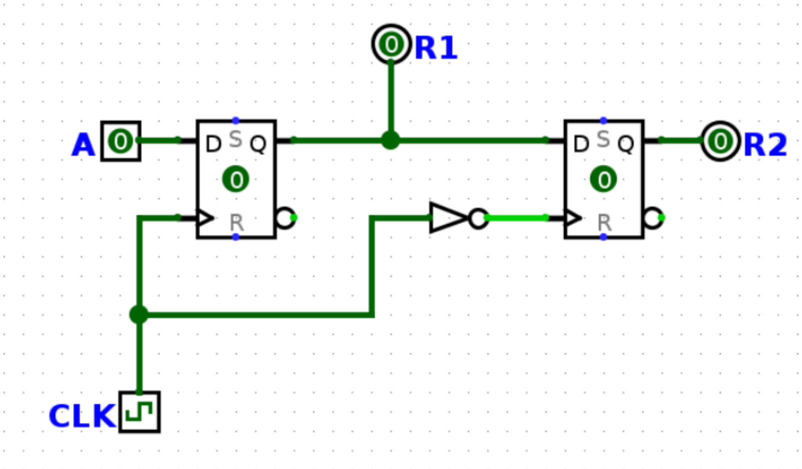
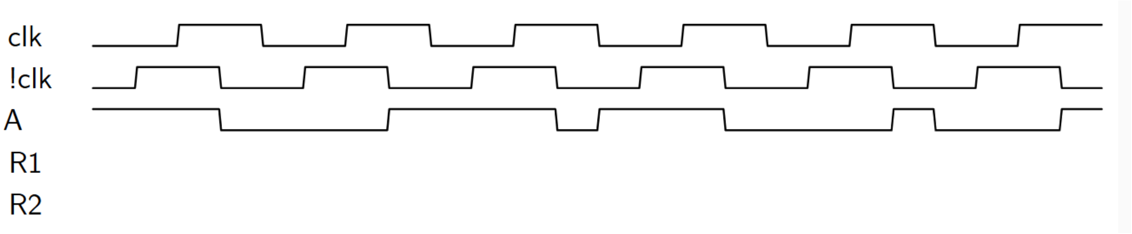
# 1 SDS Intro

1.1 Fill out the timing diagram. The clock period (rising edge to rising edge) is 8ps. For every register, clk-to-q delay is 2ps, setup time is 4ps, and hold time is 2ps. NOT gates have a 2ps propagation delay, which is already accounted for in the !clk signal given.

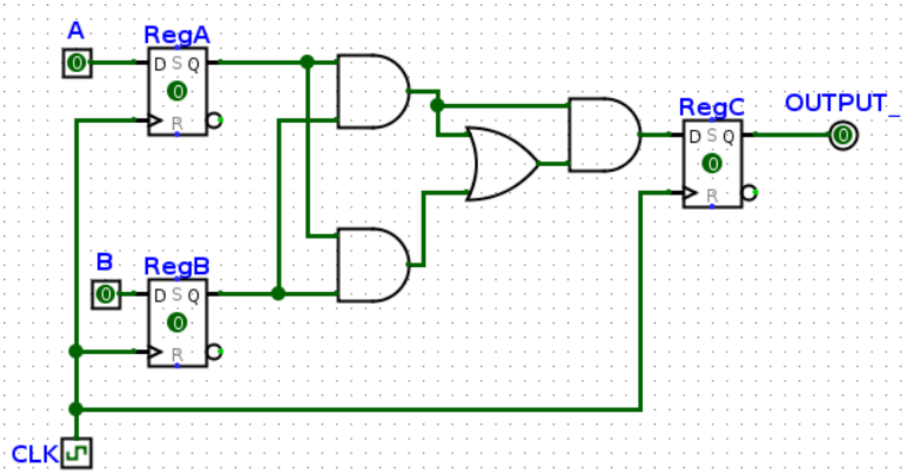




1.2 In the circuit below:

- RegA and RegB have setup, hold, and clk-to-q times of 4ns,
- All logic gates have a delay of 5ns
- RegC has a setup time of 6ns.

What is the maximum allowable hold time for RegC? What is the minimum acceptable clock cycle time for this circuit, and clock frequency does it correspond to?



# 2 Single-Cycle CPU

For this worksheet, we will be working with the single-cycle CPU datapath provided on the last page.

2.1 List all possible values that each control signal may take on for the single cycle datapath, then briefly describe what each value means for each signal.

(a) PCSel

0：PC+4

1：跳转ALU计算的地址

(b) RegWEn

0：不开启寄存器写入

1：开启寄存器写入

(c) ImmSel

不同类型的立即数生成

0-4：I，B，S，J，U

5-7：未使用

(d) BrEq

返回是否相等

0：不相等

1：相等

(e) BrLt

返回是否小于

0：不小于

1：小于

(f) ALUSel

控制计算模式

add，sub，xor，and，or等

(g) MemRW

控制DMEM的读写

0：不允许写

1：允许写

(h) WBSel

控制写回寄存器的数据

0：DMEM的数据

1：ALU计算的数据

2：PC+4

3：未使用

2.2 Fill out the following table with the control signals for each instruction based on the datapath on the last page.

- If the value of the signal does not affect the execution of an instruction, use the * (don't care) symbol to indicate this.
- If the value of the signal does affect the execution, but can be different depending on the program, list all possible values (for example, for a signal that may output 0 and 1, write 0/1).
- For ALUSel, write the ALU operation (`add`, `or`, `sll`, ...)

The first row has been filled out for you.

| | BrEq | BrLT | PCSel | Imm-Sel | BrUn | ASel | BSel | ALUSel | MemRW | Reg-WEn | WB-Sel |
|---|---|---|---|---|---|---|---|---|---|---|---|
| add | * | * | 0 (PC + 4) | * | * | 0 (Reg) | 0 (Reg) | add | 0 | 1 | 1 (ALU) |
| ori | * | * | 0 | I | * | 0 | 1 | or | 0 | 1 | 1 |
| lw | * | * | 0 | I | * | 0 | 1 | add | 0 | 1 | 0 |
| sw | * | * | 0 | S | * | 0 | 1 | add | 1 | 0 | * |
| beq | 0/1 | * | 0/1 | B | * | 1 | 1 | add | 0 | 0 | * |
| jal | * | * | 1 | J | * | 1 | 1 | add | 0 | 1 | 2 |
| blt | * | 0/1 | 0/1 | B | 0 | 1 | 1 | add | 0 | 0 | * |

# 3 Timing the Datapath

**Clocking review:**

- A **state element** is an element connected to the clock (denoted by a triangle at the bottom). The **input signal** to each state element must stabilize before each **rising edge**. For example, registers are state elements.
- The **critical path is the longest delay path between any two state elements in the circuit (with no other state elements along that path)**. The circuit cannot be clocked faster than this, since anything faster would mean that the correct value is not guaranteed to reach the state element in the alloted time. We can shorten the critical path by placing registers along it, thus reducing the amount of logic between state elements (i.e. registers);.

For this exercise, the times for each circuit element is given as follows:

| Clk-to-Q | RegFile Read | PC/RegFile Setup | Mux | Adder |
|---|---|---|---|---|
| 5ns | 35ns | 20ns | 15ns | 20ns |

| ALU | Branch Comp | Imm Gen | MEM Read | DMEM Setup |
|---|---|---|---|---|
| 100ns | 50ns | 45ns | 300ns | 200ns |

3.1 Mark an X for the datapath stages used by each instruction

| | IF | ID | EX | MEM | WB |
|---|---|---|---|---|---|
| `add` | X | X | X | | X |
| `ori` | X | X | X | | X |
| `lw` | X | X | X | X | X |
| `sw` | X | X | X | X | |
| `beq` | X | X | X | | |
| `jal` | X | X | X | | X |

3.2 Ignoring the length of a clock cycle, how long does it take to execute each instruction? Assume that the setup times to the RegFile and the PC are the same.

Hint: For each instruction, first identify which elements of the datapath are being used. What is the longest path between two state elements?

(a) jal

500ns

(b) lw

800ns

(c) sw 665ns

3.3 Which instruction(s) are responsible for the critical path?

lw

3.4 What is the highest clock frequency for this single cycle datapath?

1/800ns 1.25 MHz

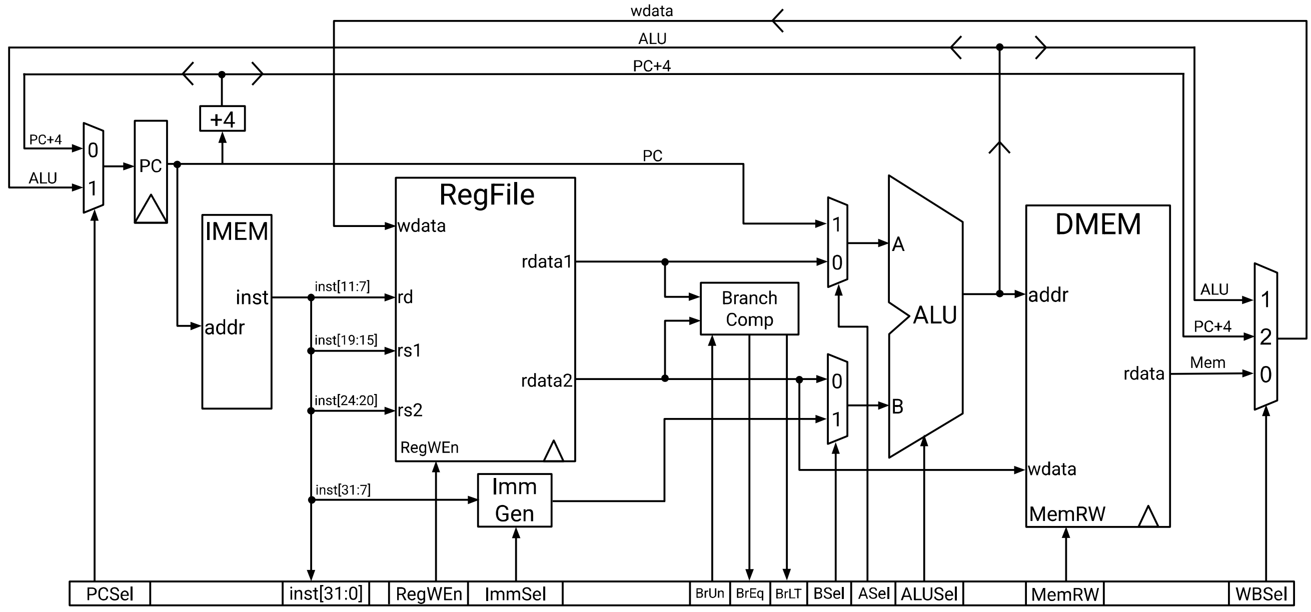
3.5 Why is the single-cycle datapath inefficient?

因为每条指令都要等上一条指令执行完才成开始执行，数据通路的很多部件处于闲置状态，没有充分利用

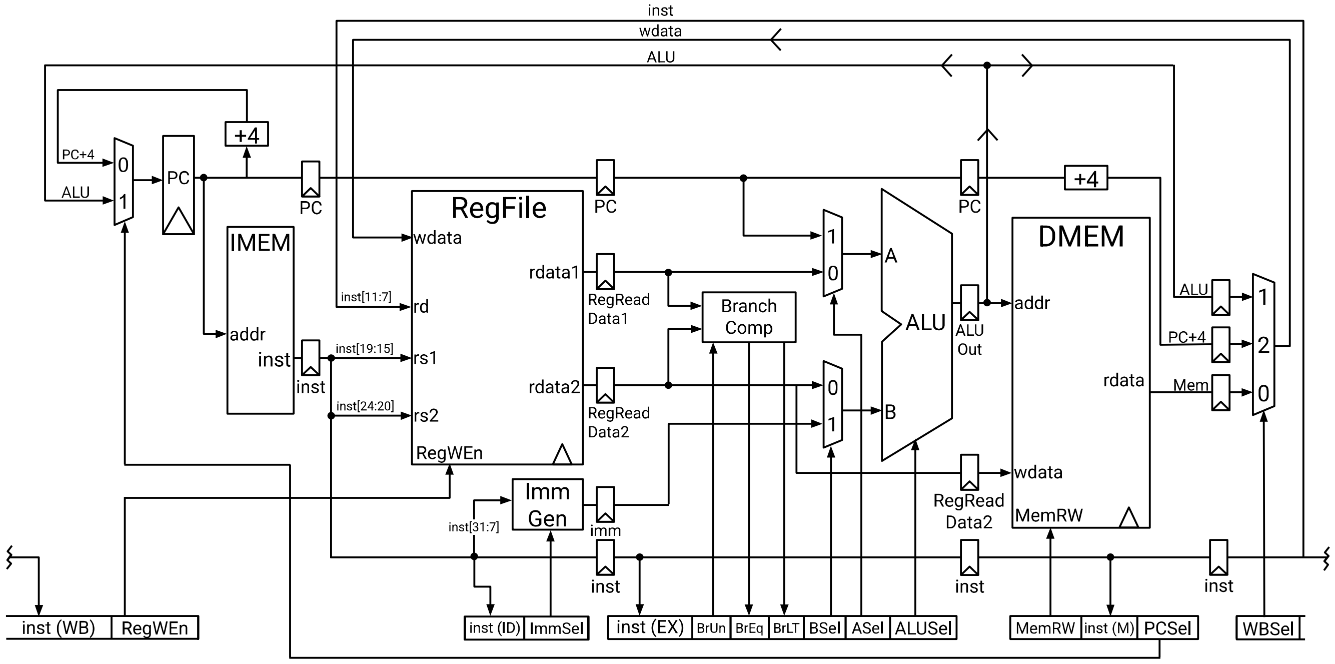
3.6 How can you improve its performance? What is the purpose of pipelining?

利用流水线机制进行改进，使得每个部件都可以充分利用，从而缩短时钟周期，提高效率

## Single-Cycle Datapath Diagram



## 5-Stage Datapath Diagram


inst
wdata
ALU
PC+4
ALU
0
1
PC
+4
PC
IMEM
addr
inst
inst
RegFile
wdata
inst[11:7]
rd
inst[19:15]
rs1
inst[24:20]
rs2
RegWEn
rdata1
rdata2
PC
RegRead
Data1
RegRead
Data2
Branch
Comp
Imm
Gen
inst[31:7]
imm
inst
1
0
0
1
A
B
ALU
PC
ALU
Out
+4
DMEM
addr
wdata
rdata
MemRW
RegRead
Data2
inst
ALU
PC+4
Mem
1
2
0
inst
inst (WB)
RegWEn
inst (ID)
ImmSel
inst (EX)
BrUn
BrEq
BrLT
BSel
ASel
ALUSel
MemRW
inst (M)
PCSel
WBSel