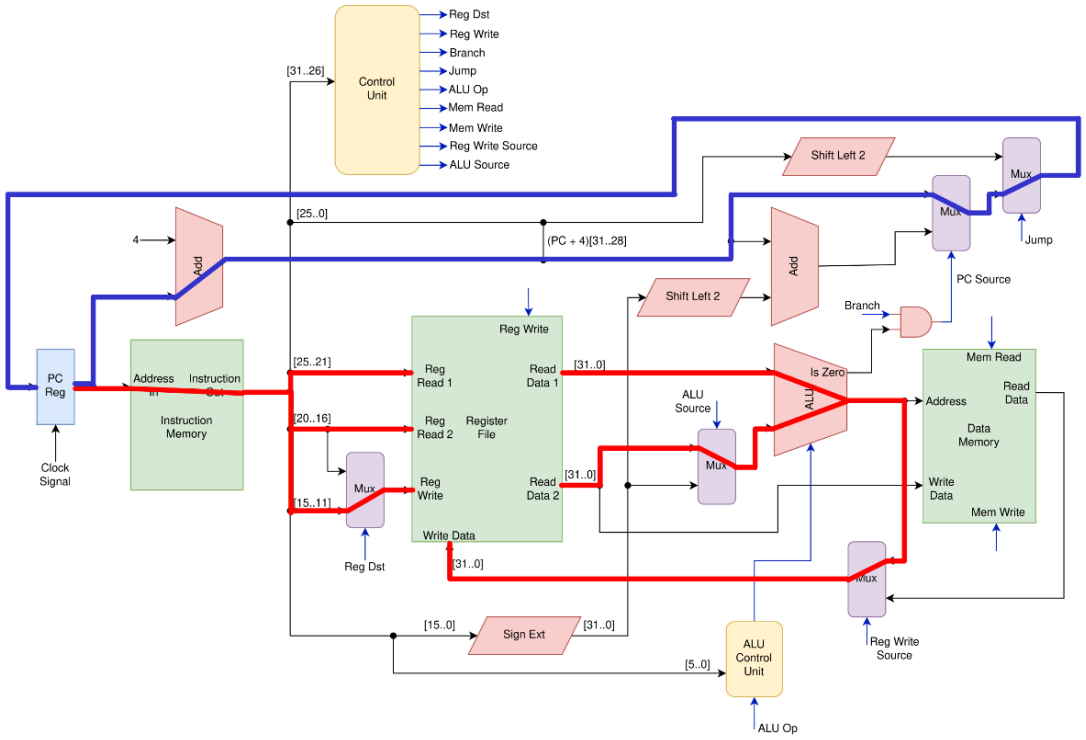
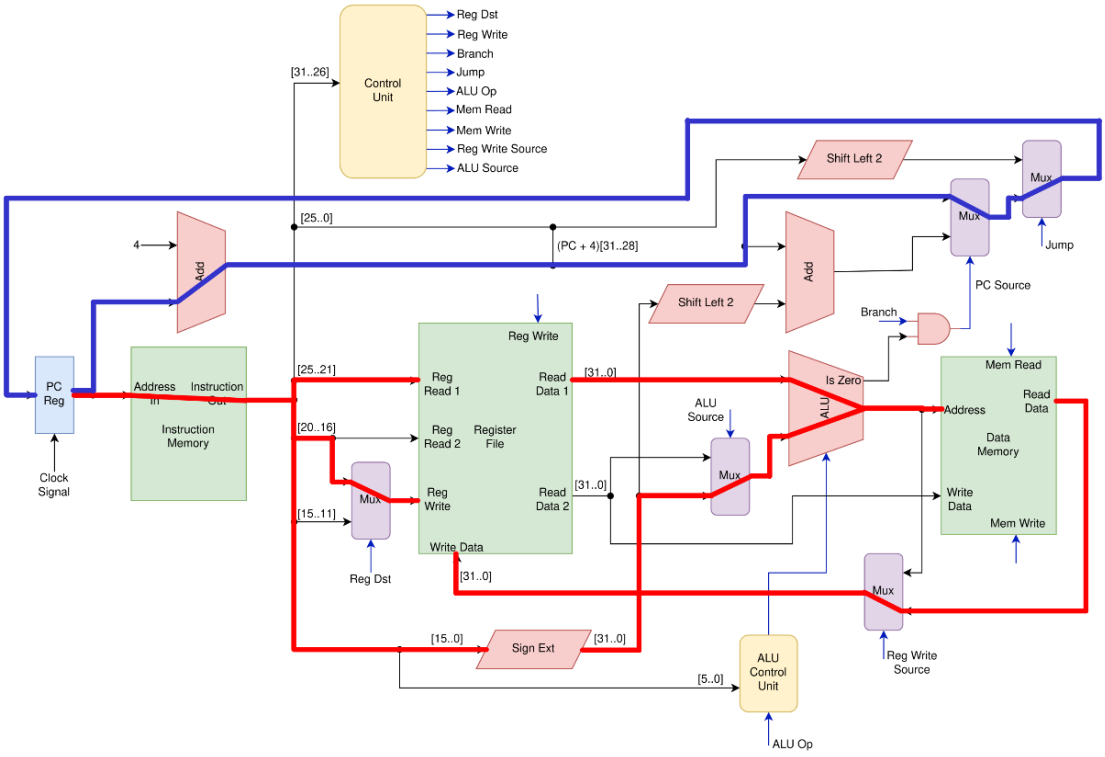
# CSE490/590, Spring 2023, Homework 2

1. Highlight the flow of data in the MIPS data path for the instructions:

   a) `add $s1, $s2, $s3`




   b) `lw $t3, 4($t4)`



# CSE490/590, Spring 2023, Homework 2

2. Consider the following sequence of simplified instructions:

| Instruction Address: | Instruction: |
|---|---|
| 100 | ADD |
| 104 | BEQ + 200 |
| 108 | ADD |
| 112 | ADD |
| 308 | ADD |

Assume the following:

- There are 5 pipeline stages — IF (Instruction Fetch), ID (Instruction Decode), EX (Execution), MEM (Memory Access), WB (Write Back).
- The second instruction, "BEQ +200", takes the branch and jumps to the instruction at the address 308.
- A branch instruction can determine the next PC only at the EX stage.
- The CPU always speculates that it will execute the instruction at (PC + 4) next.
- If a branch or jump needs to be taken, all instructions in the pipeline are killed.

Complete the time table below with the correct stages at each cycle for all instructions. Use *** for pipeline bubbles.

| Instruction Address | Instruction | Time: | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | t0 | t1 | t2 | t3 | t4 | t5 | t6 | t7 | t8 |
| 100 | ADD | IF | ID | EX | MEM | WB | | | | |
| 104 | BEQ + 200 | | IF | ID | EX | MEM | WB | | | |
| 108 | ADD | | | IF | ID | *** | *** | *** | | |
| 112 | ADD | | | | IF | *** | *** | *** | *** | |
| 308 | ADD | | | | | IF | ID | EX | MEM | WB |

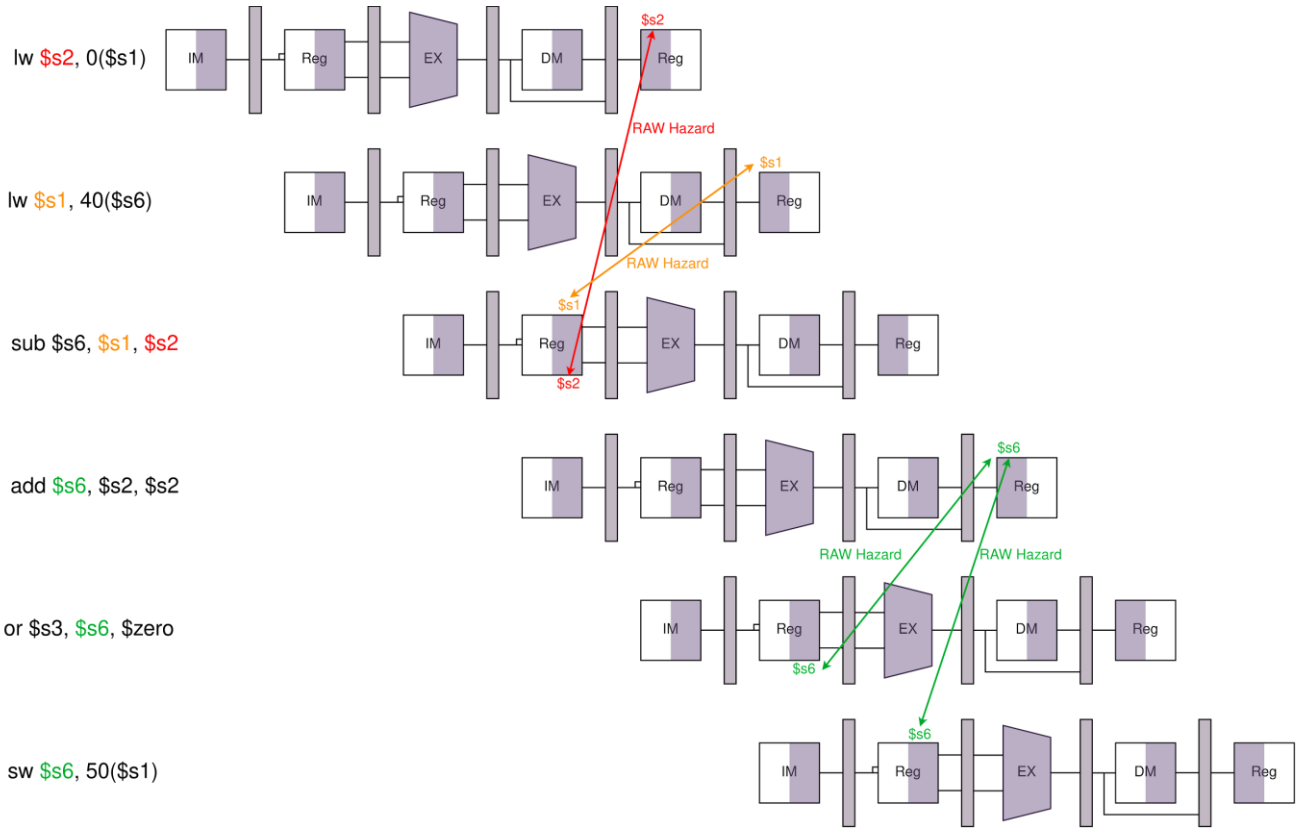
# CSE490/590, Spring 2023, Homework 2

3. In the following instruction sequence for a MIPS 5-stage pipelined datapath, list the data hazards:

```
lw $s2, 0($s1)
      lw $s1, 40($s6)
      sub $s6, $s1, $s2
      add $s6, $s2, $s2
or $s3, $s6, $zero
sw $s6, 50($s1)
```

Answer:

1. lw $s2, 0($s1)
2. lw $s1, 40($s6)
3. sub $s6, $s1, $s2
4. add $s6, $s2, $s2
5. or $s3, $s6, $zero
6. sw $s6, 50($s1)



# CSE490/590, Spring 2023, Homework 2

4. Consider the following instruction sequence.

```
add r5,r2,r1
lw r3,4(r5)
lw r2,0(r2)
or r3,r5,r3
sw r3,0(r5)
```

a) Show the pipeline diagram after inserting NOPs to overcome data dependencies

| Instruction: | T0 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 | T11 | T12 | T13 | T14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| add r5, r2, r1 | IF | ID | EX | MEM | WB | | | | | | | | | | |
| NOP | | - | - | - | - | - | | | | | | | | | |
| NOP | | | - | - | - | - | - | | | | | | | | |
| lw r3, 4(r5) | | | | IF | ID | EX | MEM | WB | | | | | | | |
| lw r2, 0(r2) | | | | | IF | ID | EX | MEM | WB | | | | | | |
| NOP | | | | | | - | - | - | - | - | | | | | |
| or r3, r5, r3 | | | | | | | IF | ID | EX | MEM | WB | | | | |
| NOP | | | | | | | | - | - | - | - | - | | | |
| NOP | | | | | | | | | - | - | - | - | - | | |
| sw r3, 0(r5) | | | | | | | | | | IF | ID | EX | MEM | WB | |
| | | | | | | | | | | | | | | | |

b) Show the pipeline diagram after inserting Data Forwarding Unit to overcome data dependencies

| Instruction: | T0 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 | T11 | T12 | T13 | T14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| add r5, r2, r1 | IF | ID | EX | MEM | WB | | | | | | | | | | |
| lw r3, 4(r5) | | IF | ID | EX | MEM | WB | | | | | | | | | |
| lw r2, 0(r2) | | | IF | ID | EX | MEM | WB | | | | | | | | |
| or r3, r5, r3 | | | | IF | ID | EX | MEM | WB | | | | | | | |
| sw r3, 0(r5) | | | | | IF | ID | EX | MEM | WB | | | | | | |
| | | | | | | | | | | | | | | | |

c) Compare performance of (a) vs (b): Option (b) is better in terms of performance because the sequence of instructions is completed in T8 time units (cycles) whereas option (a) take T13 time units. But option (b) has a tradeoff of using an extra hardware forwarding unit which increases area.

# CSE490/590, Spring 2023, Homework 2

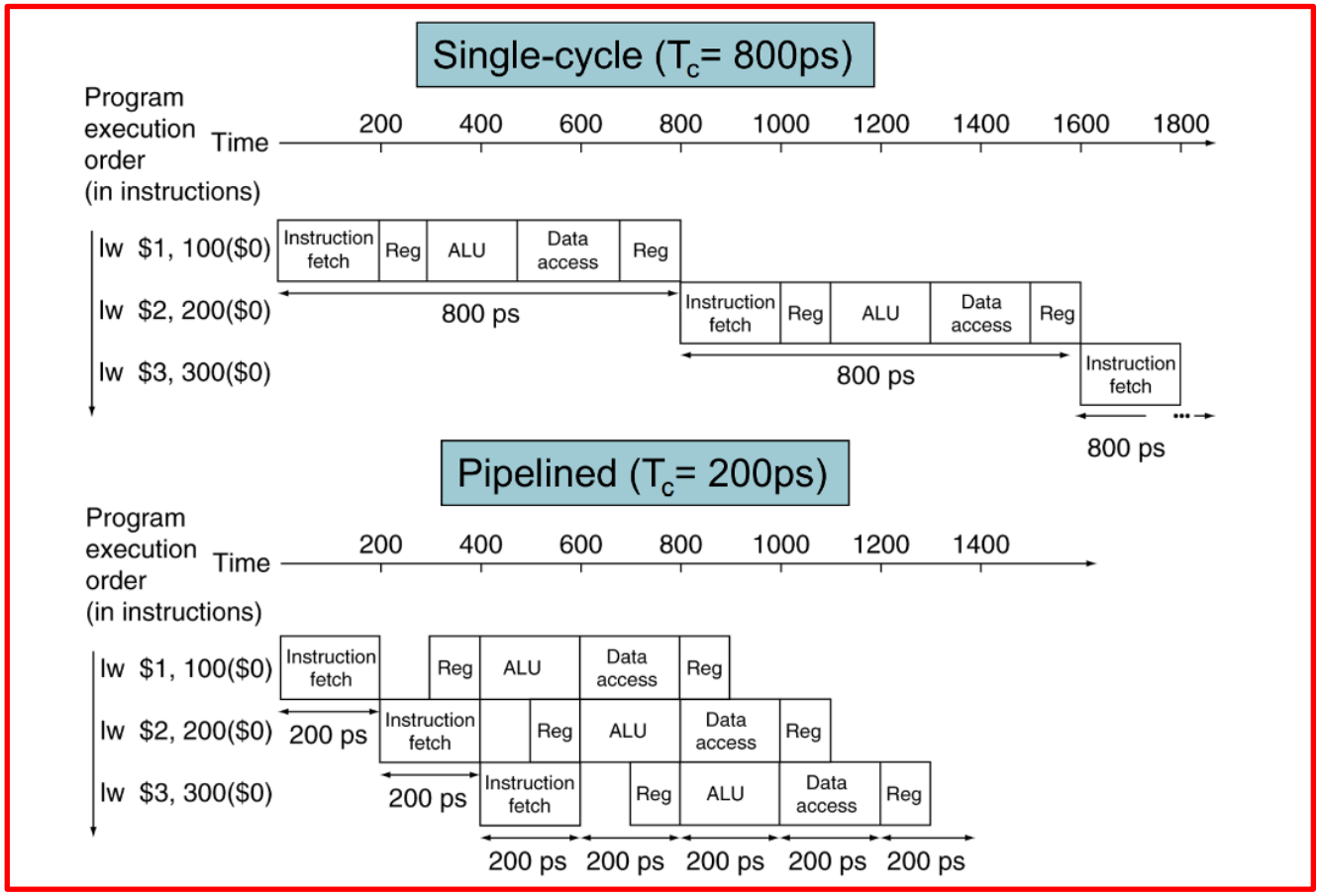
5. Assume the time for stages is
   - 100ps for register read or write
   - 200ps for other stages

   Compare pipelined datapath with single-cycle datapath for the following instruction sequence:

```
lw $1,100($0)
lw $2,200($0)
lw $3,300($0)
```

   The following table provides how much time is spent in each stage by a specific instruction:

| Instr | Instr fetch | Register read | ALU op | Memory access | Register write | Total time |
|---|---|---|---|---|---|---|
| lw | 200ps | 100 ps | 200ps | 200ps | 100 ps | 800ps |
| sw | 200ps | 100 ps | 200ps | 200ps | | 700ps |
| R-format | 200ps | 100 ps | 200ps | | 100 ps | 600ps |
| beq | 200ps | 100 ps | 200ps | | | 500ps |

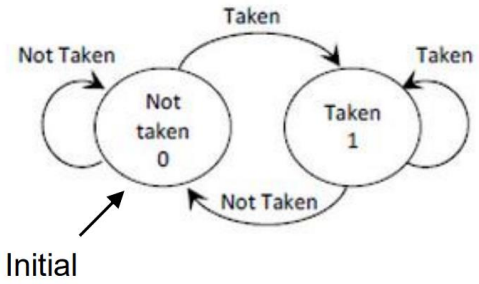



6. Consider the following instructions. Assume that the initial values for **R1**, **R2**, **R3, R4 and R5** are all 0:

# CSE490/590, Spring 2023, Homework 2

```
loop:    ADDI R2, R1, -2
         BNE R5, R2, target1
         ADDI R3, R3, 1
target1: ADDI R1, R1, 1
         ADDI R4, R1, -3
         BNE R5, R4, loop
```

Assume that we have a 1-bit branch predictor that stores the result of the last branch and makes the prediction based on the result. Show the results of all predictions throughout the execution. (Use T/N to represent Taken/ Not Taken)




| Branch | Prediction (T/N) | Actual Result (T/N) |
|---|---|---|
| 1st BNE (target1) | N | T |
| | | |
| | | |
| | | |

| Branch | Prediction(T/N) | Actual Result(T/N) |
|---|---|---|
| 1st BNEZ (target1) | N | T |
| 2 | T | T |
| 3 | T | T |
| 4 | T | T |
| 5 | T | N |
| 6 | N | N |

The first Branch in the table is filled for you. The Prediction was Not Taken (N) and the Actual Result was Taken (T). The Prediction of current branch depends on the Prediction and Actual Result of

previous branch. Here, N and T (prediction and actual results of branch 1) would produce a predicted result of T for branch 2.

**Calculation of Branch 2's Actual Result:**
(Note that the initial values for R1, R2, R3, R4 and R5 are all 0)

After branch 1 execution jumps to "target1". As the instructions execute, R1 is set to 1 and R4 is set to -2. The next branch instruction is reach, since R5 is not equal to R4, the branch is taken which causes execution to jump to "loop", thus the Actual Result is Taken (T) for branch 2. T and T from branch 2 gives T as the predicted result for branch 3. The rest of the table follows the same pattern.