1a) jr: add inputs to PCSrc mux & widen control bit-width

jal: take value from the “PC+4” ALU and add it as an input to the mux controlled by the RegDataSrc signal. Widen the signal to support the additional mux input. RegWrite should be enabled during this instruction. PCSrc should select the immediate field as the next address. A mux should be added to the write register input, which selects input either from the instruction or from a constant value (the return address register).

1b) jr: This instruction does not need to actually extend into the EX, MEM, or WB phases, it finishes its work by the end of the ID phase. It only needs to pass along its control signals so that it can update PC appropriately at the beginning of the MEM stage.

jal: pass along jump address and use muxes. All control signals must be preserved through jump registers. Pass along pc+4 to be saved in the register file ($ra).

2a) single-cycle: 400 + 200 + 300 + 500 + 200 = 1600ps

We need to be able to complete **all stages** in a single cycle for this implementation

2b) multi-cycle: 500ps

We need to be able to complete **the longest stage** in a single cycle here.

3a) Cycles: .25 * 1 cycle per load = .25 average load cycle latency

- All instructions take 1 cycle, and loads form 25% of the instructions.

Seconds: .25 * 1600ps per load = 400ps average load time latency

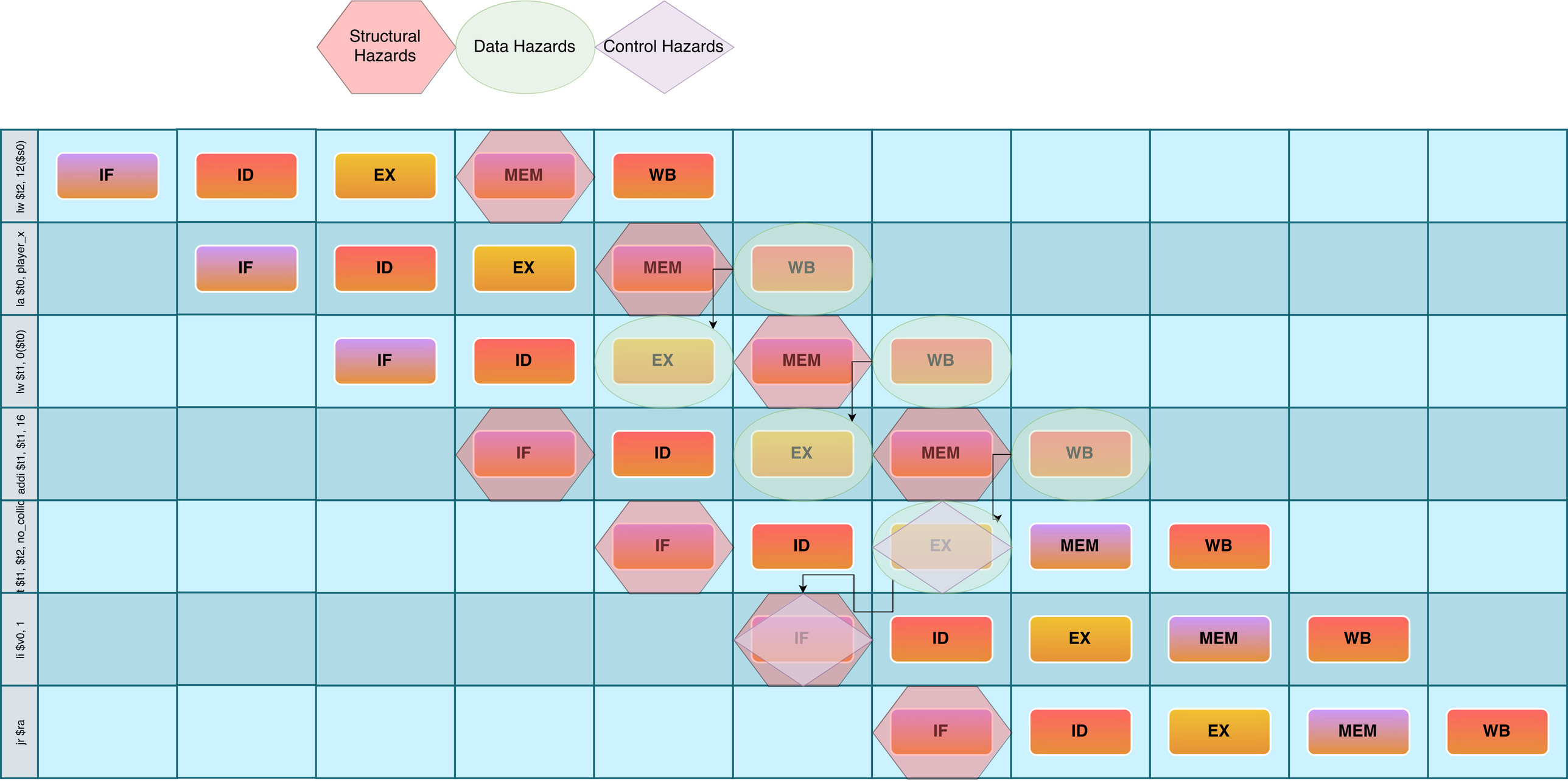
3b) Assuming all instructions must run through full 5 cycle pipeline:

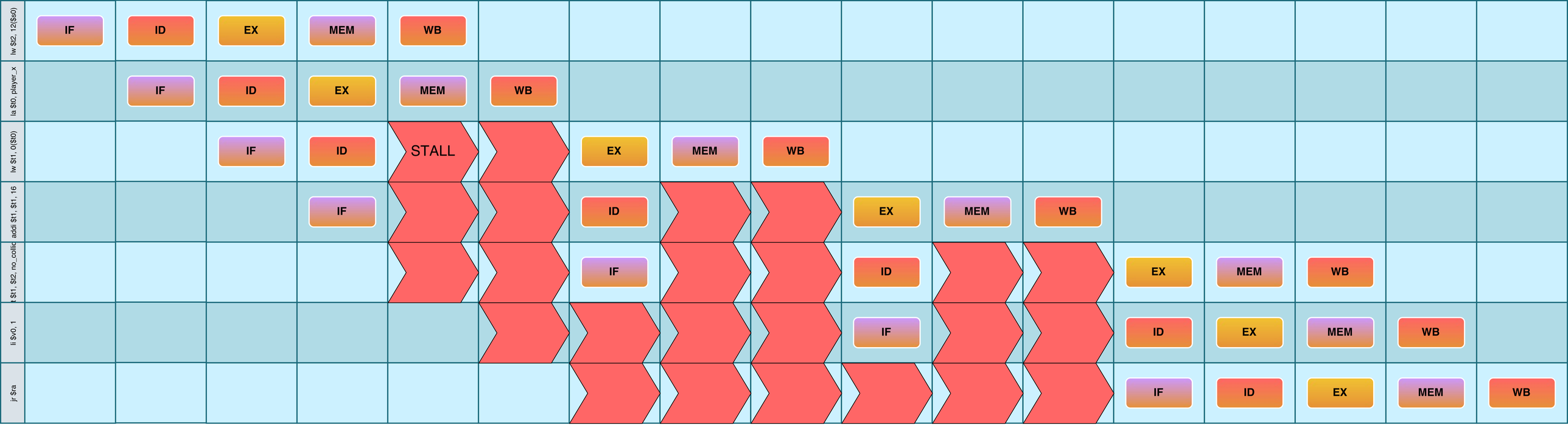
Cycles: .25 * 5 cycles per load = 1.25 average load cycle latency

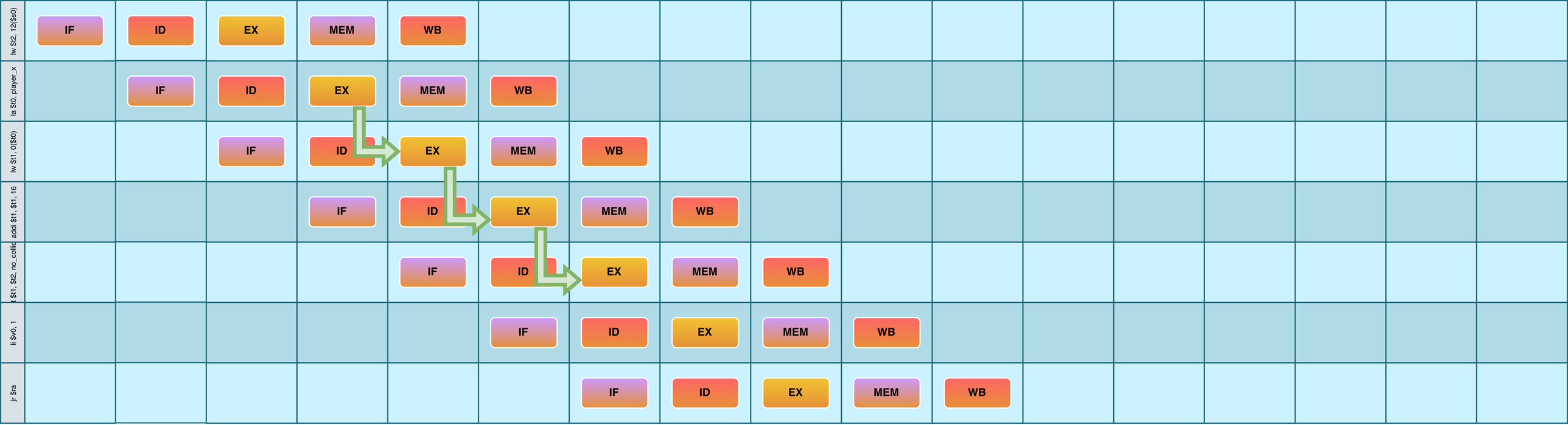
Seconds: .25 * 500ps * 5 stages = 625ps average load time latency

4) The MEM stage should be split, as it is the max stage length that determines the overall cycle latency. Once it is split into two 250ps stages, the cycle latency becomes 400ps, as the IF stage is the new max latency stage.

5) See pdfs.


Structural Hazards
Data Hazards
Control Hazards
lw $t2, 12($s0)
la $t0, player_x
lw $t1, 0($t0)
addi $t1, $t1, 16
t $t1, $t2, no_collic
li $v0, 1
jr $ra
IF
ID
EX
MEM
WB


lw $t2, 12($s0)
la $t0, player_x
lw $t1, 0($t0)
addi $t1, $t1, 16
bne $t1, $t2, no_collic
li $v0, 1
jr $ra
IF
ID
EX
MEM
WB
STALL


lw $t2, 12($s0)
la $t0, player_x
lw $t1, 0($t0)
addi $t1, $t1, 16
t $t1, $t2, no_collic
li $v0, 1
jr $ra
IF
ID
EX
MEM
WB