Problem 1. Pipelined Processor with Integer Multiplier

In this problem we will consider a 5-stage pipelined TinyRV2 processor with hardware stalling to resolve data hazards. We will then investigate augmenting the processor with a single-cycle integer multiplier and a 4-cycle unpipelined iterative multiplier unit.

The baseline design is the processor which includes a single-cycle integer multiplier and the alternative design would be the processor which includes a 4-cycle unpipelined iterative integer multiplier. We will evaluate the performance of both the baseline and alternative design by considering a simple scalar multiplication C kernel as shown below. We have modified the kernel to only handle integer inputs to be able to compute using our integer multiplier units.

```
1 void scalarMul_int( int src[ ], int a, int size )
2 {
3   for ( unsigned int i = 0; i < size; i ++ )
4     src[i] = a*src[i];
5 }
```

The simplified assembly instructions which correspond to the loop above are as shown below. Study the assembly instructions and make sure that you understand how it corresponds to the C program above. **Assume the following**:

- `x3` : holds the base address of the `src[ ]` array
- `x4` : holds the scalar constant a
- `x5` : holds the size or loop count of value 64

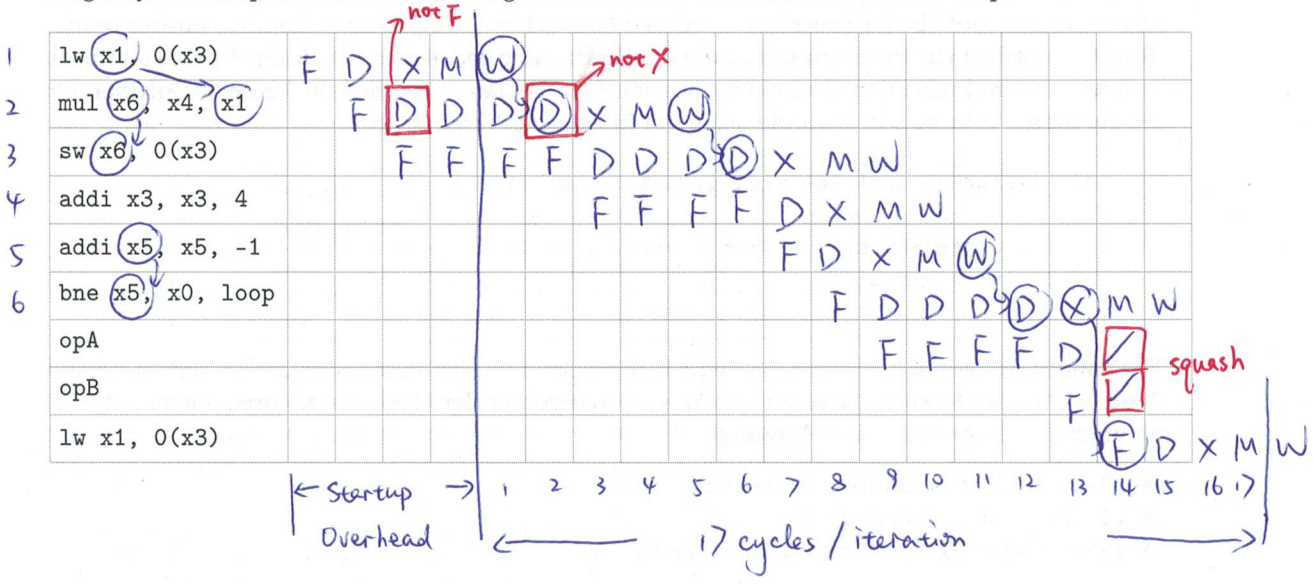
```
1 loop:
2  lw    x1, 0(x3)
3  mul   x6, x4, x1
4  sw    x6, 0(x3)
5  addi  x3, x3,  4
6  addi  x5, x5, -1
7  bne   x5, x0, loop
8  opA
9  opB
10 ...
```

Part 1.A TinyRV2 Processor with Single-Cycle Integer Multiplier

Figure 1 shows the datapath diagram of the TinyRV2 processor integrated with a single-cycle integer multiplier unit. The multiplier unit exists in the excute (X) stage of the datapath in parallel to the ALU unit as shown. Spend some time studying this datapath to understand how the single-cycle integer multiplier affects the structural, data, and control hazards in the pipeline. Example below shows the execution of a multiplication operation in this pipeline.

Dynamic Transaction		Cycle 0	1	2	3	4	5	6
1 `addi`	`x1, x3, 1`	F	D	X	M	W		
2 `mul`	`x2, x6, x4`		F	D	X	M	W	

Draw a pipeline diagram illustrating the first iteration of the loop. Note, you should include the first instruction of the second iteration as shown in the instruction sequence. Draw arrows in your pipeline diagram to indicate any data or control hazards. Ignoring the startup overhead calculate the total number of cycles required to execute the loop. Assume that this processor in a given standard CMOS technology take 1 ns for a clock cycle and that the critcal path is through the single-cycle multiplier in execute (X) stage. Calculate the execution time for the loop.



Total number of cycles = 64 × 17
(ignoring the startup overhead & squashes due to the last iteration)

64 → loop count

17 → cycles for 1 loop iteration

$$CPI = \frac{17}{6}$$ → not 8

Execution Time = 64 × 17 × 1 ns = 1088 ns

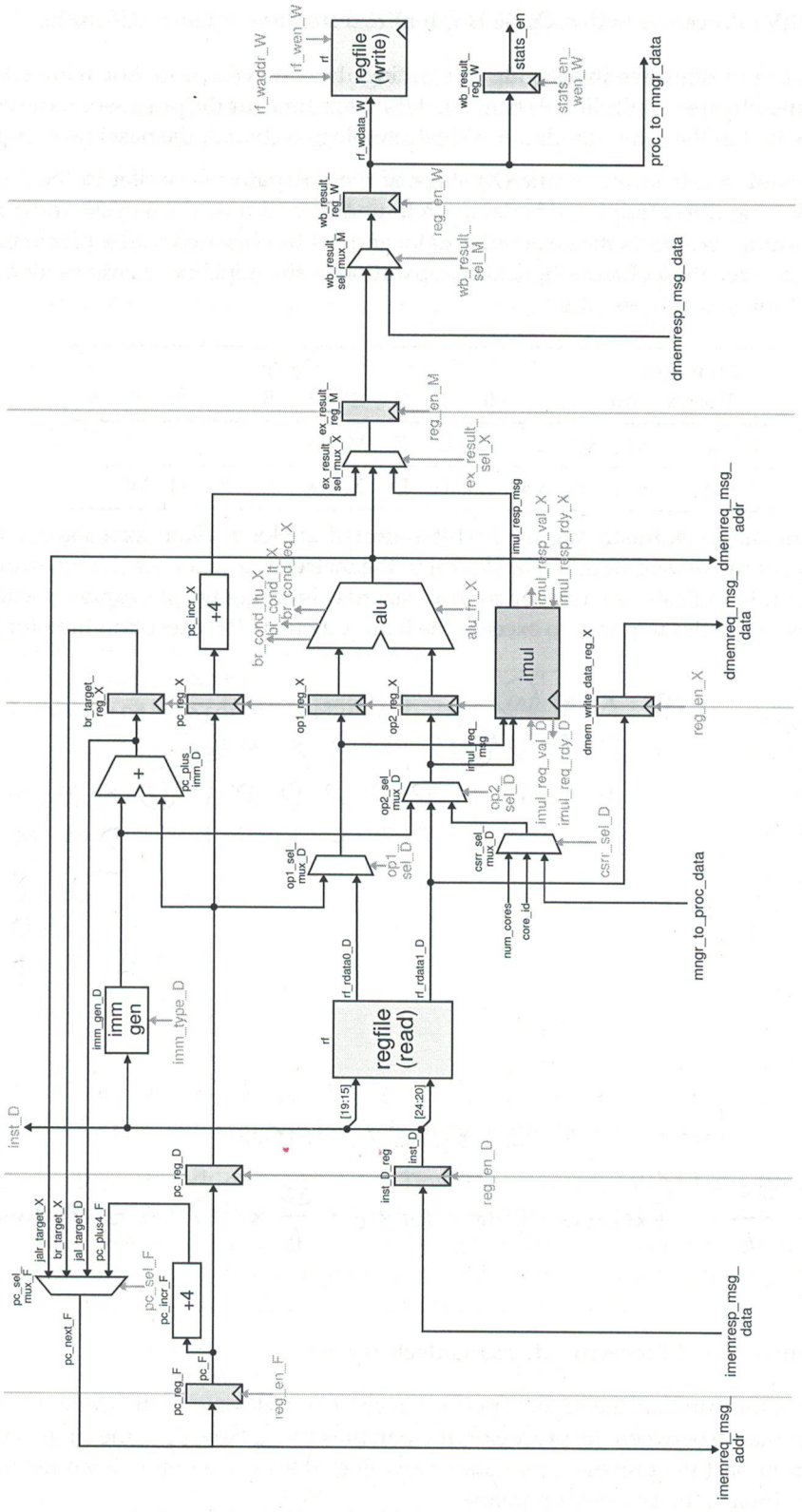


Figure 1: TinyRV2 Processor with Single-Cycle Integer Multiplier Datapath

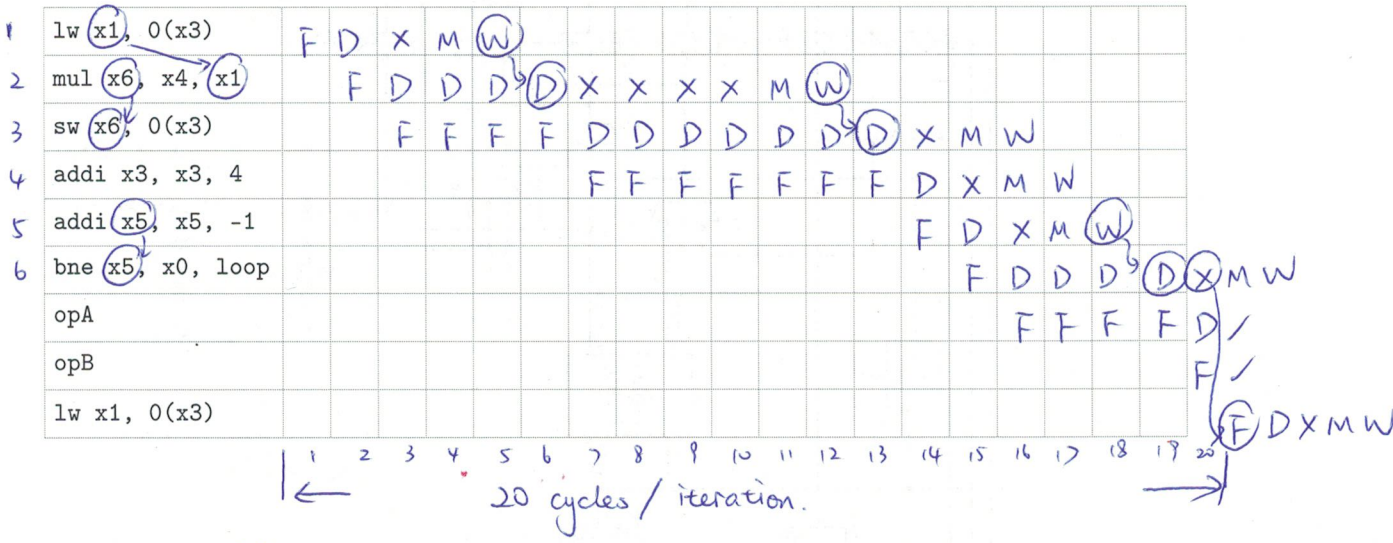
Part 1.B TinyRV2 Processor with 4-Cycle Unpipelined Iterative Integer Multiplier

For the alternative design, we investigate integrating the TinyRV2 processor with a 4-cycle unpipelined iterative Integer Multplier. Assume that the cycle time for the processor reduces to 0.75ns when implemented in the same standard CMOS technology as that of the baseline design.

The multiplier unit exists in the excute (X) stage of the datapath in parallel to the ALU unit as shown. Spend some time studying this datapath to understand how the 4-cycle unpipelined iterative integer multiplier affects the structural, data, and control hazards in the pipeline. Example below shows the execution of a multiplication operation in this pipeline. A transaction which occurs in the multiplier is indicated by X.

Dynamic Transaction	0	1	2	3	4	5	6	7	8	9
1 addi x1, x3, 1	F	D	X	M	W					
2 mul x2, x6, x4		F	D	X	X	X	X	M	W	

Draw a pipeline diagram illustrating the first iteration of the loop. Note, you should include the first instruction of the second iteration as shown in the instruction sequence. Draw arrows in your pipeline diagram to indicate any data or control hazards. Ignoring the startup overhead calculate the total number of cycles required to execute the loop. Calculate the execution time for the loop.



$CPI = \frac{20}{6}$ Execution Time $= 64 \times 6 \times \frac{20}{6} \times 0.75\,ns = 960\,ns$

Part 1.C Comparison of Processor Microarchitectures

Which microarchitecture has the highest performance? Discuss some of the trade-offs in terms of area and performance between the processor microarchitectures. How does the single-cycle integer multiplier unit impact the processor pipeline? How does the 4-cycle unpipelined iterative integer multiplier unit impact the processor pipeline?

The processor with 4-cycle unpipelined iterative integer multiplier has better performance.