ECE6005

Sep 28, 2022

Yihui Wang

Lab Assignment 1

Problem 1

a)

```
	.data
	# Prepare data
	CONTROL:	.word32 0x10000
	DATA:		.word32 0x10008
	A:		.word 2063
	B:		.word 36725
	C:		.space 16
	I:		.word 12
	J:		.word 15
	EVEN:		.asciiz " Even:"
	ODD:		.asciiz " Odd:"

.text
main:
	# Load data to register
	ld $s0, A($zero)
	ld $s1, B($zero)
	ld $s2, C($zero)
	ld $s3, I($zero)
	ld $s4, J($zero)
	daddi $s5, $zero, 2
	daddi $s6, $zero, 4
```

```
loop:
	# judge if i<15
	slt $t0, $s3, $s4
	beq $t0, $zero, done
	# Multiplication
	dmul $t1, $s0, $s5
	dmul $t2, $s1, $s6
	dadd $s2, $t1, $t2
	daddi $s2, $s2, 1
	# Store the result of C
	sd $s2, C($zero)
	ddiv $t3, $s2, $s5
	dmul $t4, $t3, $s5
	# Judge if even and jump to even function
	beq $t4, $s2, even
	# prepare to output
	daddi $t0, $zero, 4
	daddi $t1, $zero, ODD
	lwu $t5, DATA($zero)
	lwu $t6, CONTROL($zero)
	sd $t1, ($t5)
	sd $t0, ($t6)
	daddi $t0, $zero, 2
	dadd $t1, $zero, $s2
	sd $t1, ($t5)
	sd $t0, ($t6)
	daddi $s3, $s3, 1
	j loop
```

even:

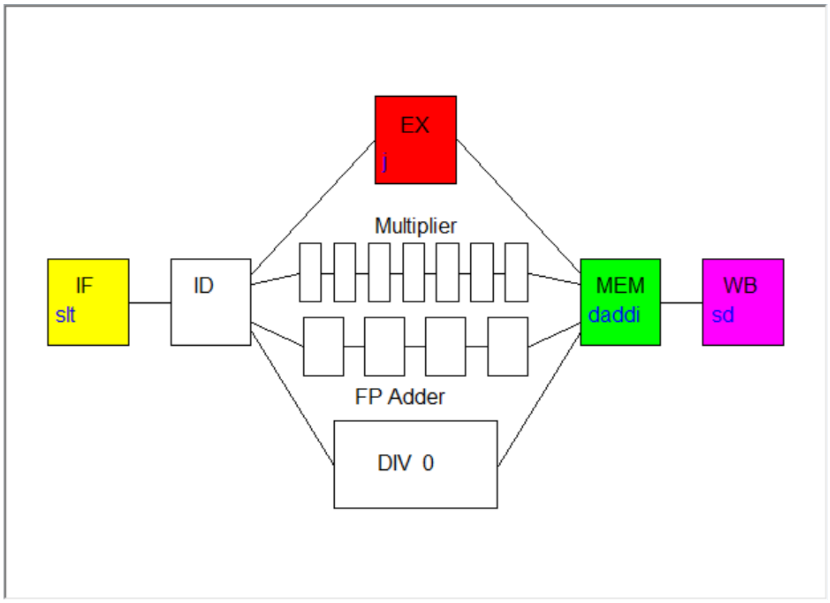
```
daddi $t0, $zero, 4
daddi $t1, $zero, EVEN
lwu $t5, DATA($zero)
lwu $t6, CONTROL($zero)
sd $t1, ($t5)
sd $t0, ($t6)
daddi $t0, $zero, 2
dadd $t1, $zero, $t3
sd $t1, ($t5)
sd $t0, ($t6)
daddi $s3, $s3, 1
j loop
```

done:

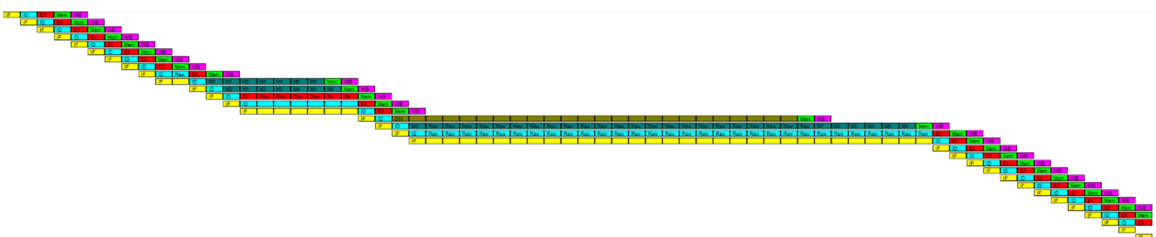
```
halt
```

b)

The pipline of one iteration is as follows:



The cycles of one iteration is as follows:



The statistics information of one iteration is as follows:

Execution

67 Cycles
26 Instructions
2.577 Cycles Per Instruction (CPI)

Stalls

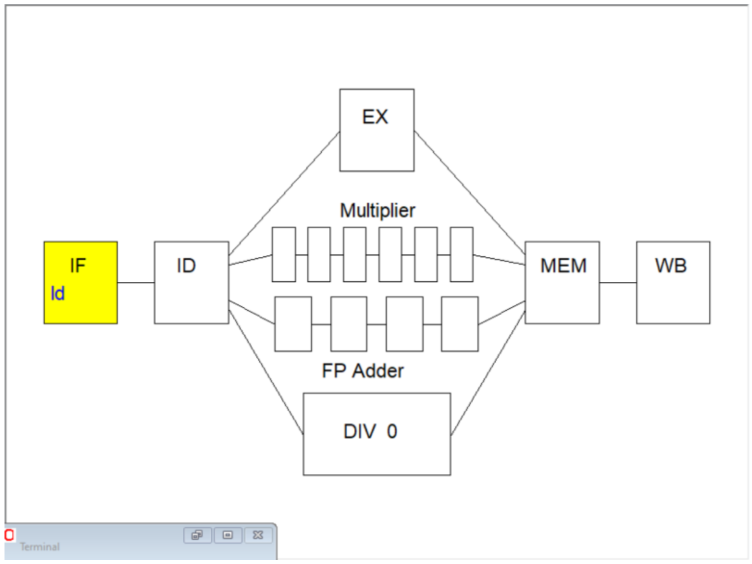
58 RAW Stalls
0 WAW Stalls
0 WAR Stalls
2 Structural Stalls
1 Branch Taken Stall
0 Branch Misprediction Stalls

Code size

168 Bytes

c)

This is the pipeline of 6 cycles of multiplication:



The statistics of architecture of 6 cycles of multiplication is as follows:

```
Execution
65 Cycles
26 Instructions
2.500 Cycles Per Instruction (CPI)

Stalls
56 RAW Stalls
0 WAW Stalls
0 WAR Stalls
2 Structural Stalls
1 Branch Taken Stall
0 Branch Misprediction Stalls

Code size
168 Bytes
```

The data access latency will directly impact the CPI by add the cycles a data access needs to the CPI, because according to the pipeline, every instruction needs to access memory for loading data and store data.

d)

We can increase the program's efficiency by decreasing the multiplication and division latency, the original statistics is:

```
Execution
196 Cycles
76 Instructions
2.579 Cycles Per Instruction (CPI)

Stalls
175 RAW Stalls
0 WAW Stalls
0 WAR Stalls
6 Structural Stalls
4 Branch Taken Stalls
0 Branch Misprediction Stalls

Code size
168 Bytes
```

After improving the architecture, the statics changes to:

```
Execution
124 Cycles
76 Instructions
1.632 Cycles Per Instruction (CPI)

Stalls
64 RAW Stalls
0 WAW Stalls
0 WAR Stalls
3 Structural Stalls
4 Branch Taken Stalls
0 Branch Misprediction Stalls

Code size
168 Bytes
```

We can see that the CPI improved significantly.

Also, we can delete the for loop since it has no contribution to the function of the code.

e)

Single precision has 32 bits while double precision has 64 bits. The statistics changes to this:

```
Execution

240 Cycles
102 Instructions
2.353 Cycles Per Instruction (CPI)

Stalls

214 RAW Stalls
0 WAW Stalls
0 WAR Stalls
9 Structural Stalls
4 Branch Taken Stalls
0 Branch Misprediction Stalls

Code size

196 Bytes
```

The instructions increased much because of the process of converting the floating to integer.

Problem 2.

a)

```
	.data
	# Prepare data
	A:	.word 15
	N:	.double 8
	R:	.double 0
	RD:	.double 0

.text
main:
	# Load data to register
	ld $s0, A($zero)
	daddi $s0, $s0, 1
	daddi $t0, $zero, 1
```

```
loop:
        daddi $s1, $s1, 1
        # judge if i<15
        beq $s1, $s0, done
        dmul $t0, $t0, $s1
        sd $t0, R($zero)

        j loop




done:
        l.d f0, N($zero)
        # convert the float to integer for output
        mtc1 $t0, f1
        cvt.d.l f1, f1
        div.d f2, f1, f0
        s.d f2, RD($zero)

        halt
```

the statistics is as follows:

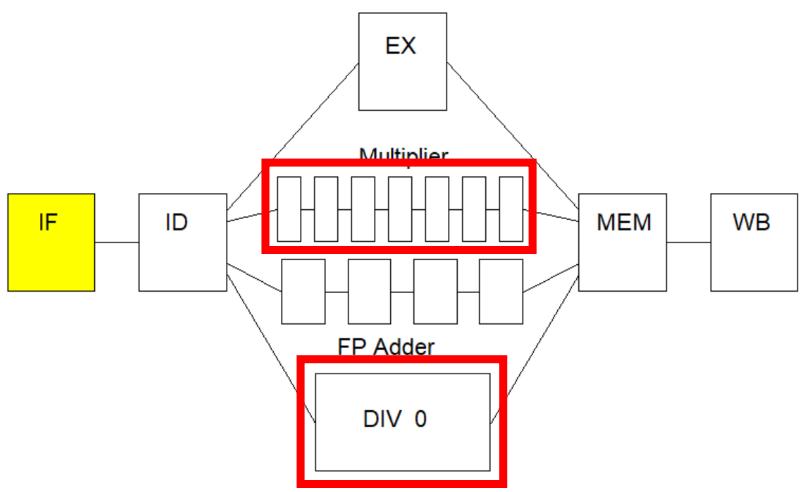
```
Execution
236 Cycles
86 Instructions
2.744 Cycles Per Instruction (CPI)

Stalls
114 RAW Stalls
0 WAW Stalls
0 WAR Stalls
16 Structural Stalls
16 Branch Taken Stalls
0 Branch Misprediction Stalls

Code size
56 Bytes
```

b)

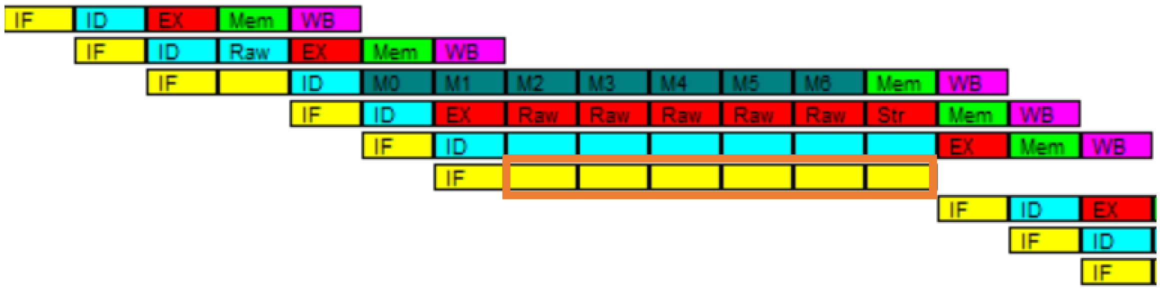
The total number of instructions should be 86, including iterations, according to the pipeline structure, the total number of cycles should be $86 + 4 = 90$, so the ideal CPI is $\frac{90}{86} = 1.047$, however, the real CPI is 2.744 according to the result of a).

c)

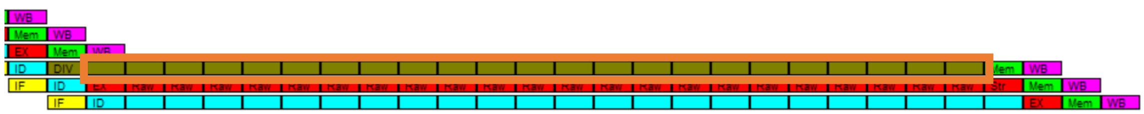
The cause of difference is the latency of multiplication and division, which are shown in the pipeline:



It also could be shown on the cycle figure that each multiplication takes 6 more cycles to process:

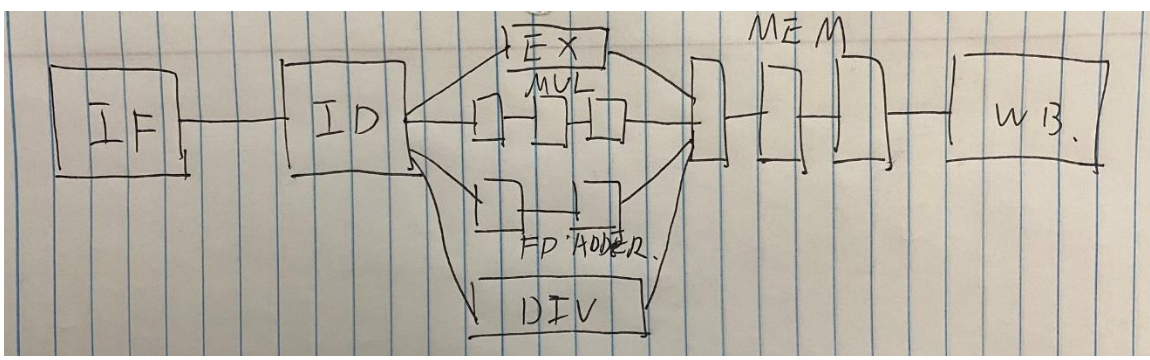


And the division takes 23 cycles more:



d)

The pipeline is as follows:



Since each instruction need to access memory, so the CPI will increase 3, in our case it will be around 5.744, because of the overlaps with multiplication and division, the final CPI will change a little bit.