Thursday, September 3, 2020 2:11 PM

Assignment 2 (100 points) Due Thursday, September 10th

1 (60 pts) Schedule the following code segment by applying Tomasulo' algorithm. Assume the floating point addition takes 2 cycles, the floating point multiplication takes 3 cycles and other operations take only 1 cycle. Figure 1 shows the initial states of the reservation station for the adder and multiplier, and the initial values of the floating point registers (FLR). Please schedule the instructions and update the states of the reservation stations and FLR for each clock cycle, until this code segment is finished.

i: R0 ← R0 * R2
j: R2 ← R4 + R8
k: R4 ← R0 * R8

cycle1: i and j dispatch

RS	Tag	Sink	Tag	Src
1	0	10.0	0	7.8
2				
3				

Adder

RS	Tag	Sink	Tag	Src
4	0	6.0	0	3.5
5				

Mult/Div

FLR	Busy	Tag	Data
0	X	4	6.0
2	X	1	3.5
4			10.0
8			7.8

cycle 2: k dispatch

RS	Tag	Sink	Tag	Src
1	0	10.0	0	7.8
2				
3				

Adder

RS	Tag	Sink	Tag	Src
4	0	6.0	0	3.5
5	4	—	0	7.8

Mult/Div

FLR	Busy	Tag	Data
0	X	4	6.0
2	X	1	3.5
4	X	5	10.0
8			7.8

cycle 3: i broadcasts:

<1, 17.8>

RS	Tag	Sink	Tag	Src
1				
2				
3				

Adder

RS	Tag	Sink	Tag	Src
4	0	6.0	0	3.5
5	4	—	0	7.8

Mult/Div

FLR	Busy	Tag	Data
0	X	4	6.0
2			17.8
4	X	5	10.0
8			7.8

i: R0 ← R0 * R2
j: R2 ← R4 + R8
k: R4 ← R0 * R8

cycle 4: j broadcasts:

<4, 21.0>

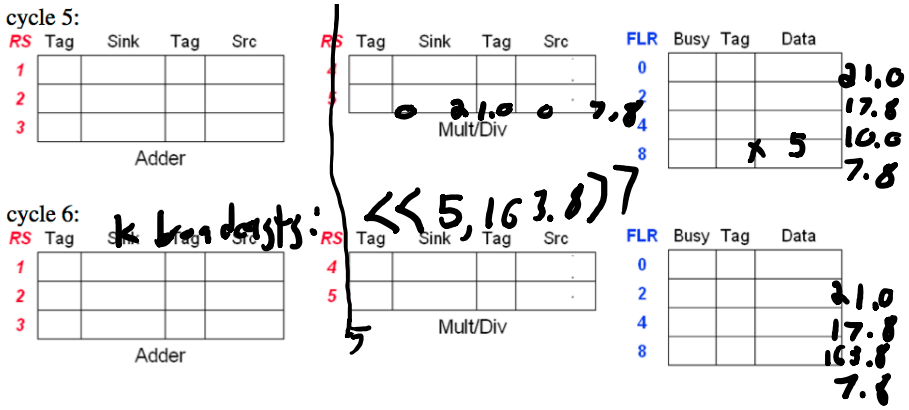
RS	Tag	Sink	Tag	Src
1				
2				
3				

Adder

RS	Tag	Sink	Tag	Src
4				
5	0	21.0	0	7.8

Mult/Div

FLR	Busy	Tag	Data
0			21.0
2			17.8
4	X	5	10.0
8			7.8



2. (40 pts) In your own words, summarize how are RAW, WAR, and WAR dependences handled by the Tomasulo's algorithm.

WAR is handled by copying the read data into the reservation station during dispatch, so any subsequent writes don't affect the operand data.

RAW is handled by marking the incomplete register data as busy, and at dispatch time the reading (latter) instruction takes the tag instead of the data. Now the former instruction can fill in the tag with the actual data when it completes.

WAW is handled in a way similar to register forwarding. The first write writes its tag to the FLR and any subsequent instruction before the later write will use that tag and eventually get the correct data from the CDB. So, the later instruction can overwrite anything in the FLR and not cause any data to be incorrect.