Computer Architectures
Exam of 22.6.2022 - part I

First name, Last name, ID...

Question #1

The Tomasulo architecture for superscalar processors with dynamic scheduling and speculation often uses a Reorder Buffer (ROB)

You are requested to

1. Explain what the ROB is and how it works
2. Describe its architecture
3. Detail when data are written in the ROB, when an entry is marked as busy and when it is released
4. List the advantages stemming from the use of the ROB.

Possible answer

1. The ROB is a module allowing in-order instruction commitment while execution is performed out-of-order.
2. The ROB is mainly composed of a FIFO buffer including several entries. Each ROB entry is composed of the following fields:
   - *Instruction type*
   - *Destination*
   - *Value*
   - *Ready*
3. An entry in the ROB is allocated to an instruction when this is successfully issued. Its content is written via the CDB when the execution finishes its execution. It is read when other instructions use the result produced by the instruction while this is not yet committed. The ROB is a FIFO (i.e., a circular buffer). When an instruction becomes the oldest in the ROB, it is committed (i.e., its result is written in the destination) and the entry is de-allocated. If the instruction to be committed is a mispredicted branch, the whole buffer is flushed. If an instruction triggers an exception, this is typically managed when the instruction is committed.
4. The ROB allows the processor to implement dynamic scheduling with speculation. It also allows precise exception management.

Computer Architectures
Exam of 22.6.2022 - part I

First name, Last name, ID...

Question #2

Let consider a MIPS64 architecture including the following functional units (for each unit the number of clock periods to complete one instruction is reported):

- Integer ALU: 1 clock period
- Data memory: 1 clock period
- FP arithmetic unit: 2 clock periods (pipelined)
- FP multiplier unit: 5 clock periods (pipelined)
- FP divider unit: 7 clock periods (unpipelined)

You should also assume that

- The branch delay slot corresponds to 1 clock cycle, and the branch delay slot is not enabled
- Data forwarding is enabled
- The EXE phase can be completed out-of-order.

You should consider the following code fragment and, filling the following tables, determine the pipeline behavior in each clock period, as well as the total number of clock periods required to execute the fragment. The values of the constants k1 and k2 are written in f10 and f11 before the beginning of the code fragment.

```
; ********************* MIPS64 ***********************
;   for (i = 0; i < 100; i++) {
;
      v4[i] = ((v1[i]*k1) + (v2[i]*k2))/v3[i];
;   }

              .data
v1:     .double "100 values"
v2:     .double "100 values"
v3:     .double "100 values"
v4:     .double "100 values"


              .text
main:  daddui r1,r0,0      I1
       daddui r2,r0,100    I2
loop:  l.d  f1,v1(r1)      I3
       mul.d  f2, f1, f10  I4
       l.d  f3,v2(r1)      I5
       mul.d  f4, f3, f11  I6
       add.d  f5,f2,f4     I7
       l.d  f6,v3(r1)      I8
       div.d  f7,f5,f6     I9
       s.d  f7,v4(r1)      I10
       daddui  r1,r1,8     I11
       daddi  r2,r2,-1     I12
       bnez  r2,loop       I13
       Halt                I14
```

	Comments	Clock cycles
	r1← pointer	5
	r2 <= 100	1
	f1 <= v1[i]	1
	f2 <= v1[i]*k1	6
	f3 <= v2[i]	0
	f4 <= v2[i]*k2	3
	f6 <= v1[i]+v2[i]	2
	f5 <= v3[i]	1
	f7 <= (v1[i]*k1+v2[i]*k2)/v3[i]	8
	v4[i] <= f7	1
	r1 <= r1 + 8	1
	r2 <= r2 - 1	1
		2
		1
Total		6+(27*100)= 2,706

Computer Architectures
Exam of 22.6.2022 - part I

First name, Last name, ID..

main: daddui r1,r0,0	F	D	E	M	W																																	5
daddui r2,r0,100		F	D	E	M	W																																1
loop: l.d f1,v1(r1)			F	D	E	M	W																															1
mul.d f2, f1, f10				F	D		E	E	E	E	E	M	W																									6
l.d f3,v2(r1)					F		D	E	M	W																												0
mul.d f4, f3, f11							F	D		E	E	E	E	E	M	W																						3
add.d f5,f2,f4								F				D			E	E	M	W																				2
l.d f6,v3(r1)												F			D	E		M	W																			1
div.d f7,f5,f6															F	D			E	E	E	E	E	E	E	M	W											8
s.d f7,v4(r1)																F			D	E							M	W										1
daddui r1,r1,8																			F	D							E	M	W									1
daddi r2,r2,-1																				F							D	E	M	W								1
bnez r2,loop																											F		D	E	M	W						2
halt																																						1