Computer Architectures
Exam of 28.1.2020 - part I – ver. A2

First name, Last name, ID.....

Question #1

Let consider the issue of predicting branch results.
You are requested to

1. Explain why this issue is important in pipelined architectures
2. Summarize the characteristics of Static and Dynamic Branch Prediction mechanisms, highlighting their main advantages and disadvantages
3. Describe the general architecture and behavior of a BHT
4. Detail the architecture and behavior of a 1-bit BHT.

Hints for answers

1. Branch prediction is important to avoid the effects of control dependencies.
   Without branch prediction, after a branch there is the risk that a wrong instruction is fetched, hence causing the need for flushing it, and thus reducing the performance.
2. Static branch prediction mechanisms are implemented by the compiler. They are based on analyzing the code and then making hypothesis on whether any branch is more likely to be taken or not. Examples of this approach include
   - Branch always taken
   - Branch always not taken
   - Prediction based on branch direction
   - Profile-based prediction.

   Dynamic branch prediction techniques are implemented by the processor at run-time resorting to Branch prediction Units (such as BHT and BTB).
   Static branch prediction techniques do not require making the processor more complex, but achieve lower performance.
3. A BHT is based on a memory composed of n words, each composed of m bits. Each word stores one prediction. Each time a branch instruction is decoded, the least log n significant bits of its address (apart from those blocked by the instruction alignment) are used to select an entry in the BHT. This mechanism may cause some collisions in accessing the BHT (different branch instructions refer to the same BHT line). The prediction is usually based on the result of the branch during previous executions. After the branch is executed, the entry it refers to is possibly updated.
4. In the case of a 1-bit BHT, m=1. This means that the prediction is simply based on the result of the branch the last time it is executed.

Computer Architectures
Exam of 28.1.2020 - part I – ver. A2

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: 4 clock periods (pipelined)
- FP divider unit: 8 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, using the table in the following page (where each column corresponds to a clock period), and determine the pipeline behavior in each clock period, as well as the total number of clock periods required to execute the fragment, reporting the result in the right column in the table below.

```
; ********************* MIPS64 ***********************
;  for (i = 0; i < 10; i++) {
;       v5[i] = v1[i]/v2[i] + v3[i]*v4[i];
;  }
```

```
            .data
V1:    .double “10 values”
V2:    .double “10 values”
V3:    .double “10 values”
V4:    .double “10 values”
V5:    .double “10 values”

            .text
main:  daddui r1,r0,0
       daddui r2,r0,10
loop:  l.d  f1,v1(r1)
       l.d  f2,v2(r1)
       l.d  f3,v3(r1)
       l.d  f4,v4(r1)
       mul.d  f5,f3,f4
       div.d  f6, f1, f2
       add.d f7, f6, f5
       s.d  f7,v4(r1)
       daddui  r1,r1,8
       daddi  r2,r2,-1
       bnez  r2,loop
       halt
```

	comments	Clock cycles
	r1← pointer	5
	r2 <= 20	1
	f1 <= v1[i]	1
	f2 <= v2[i]	1
	f3 <= v3[i]	1
	f4 <= v4[i]	1
	f5 <= v3[i]*v4[i]	5
	f6 <= v1[i] / v2[i]	5
	f7 <= v3[i]*v4[i] + v1[i] / v2[i]	2
	v5[i] <= f7	1
	r1 <= r1 + 8	1
	r2 <= r2 - 1	1
		2
		1
total		226

Computer Architectures
Exam of 28.1.2020 - part I – ver. A2

First name, Last name, ID.....

daddui r1,r0,0	F	D	E	M	W																										
daddui r2,r0,10		F	D	E	M	W																									
l.d f1,v1(r1)			F	D	E	M	W																								
l.d f2,v2(r1)				F	D	E	M	W																							
l.d f3,v3(r1)					F	D	E	M	W																						
l.d f4,v4(r1)						F	D	E	M	W																					
mul.d f5,f3,f4							F	D		E	E	E	E	M	W																
div.d f6, f1, f2								F		D	E	E	E	E	E	E	E	E	M	W											
add.d f7, f6, f5										F	D							E	E	M	W										
s.d f7,v5(r1)											F							D		E	M	W									
daddui r1,r1,8																		F		D	E	M	W								
daddi r2,r2,-1																				F	D	E	M	W							
bnez r2,loop																					F		D	E	M	W					
halt																							F	D	E	M	W					