Computer Architectures
Exam of February 22, 2021
Part 1 – possible answers

Exe #1, Point #1

- *Explain what Loop Unrolling is, stating who is in charge of applying it*

- Loop unrolling is a static technique based on reducing the number of iterations a given loop is executed, modifying its body. It is normally implemented by the compiler.

Exe #1, Point #2

- *Describe the advantages and disadvantages it introduces*

- Loop unrolling improves the performance by
  - Reducing the number of branches
  - Increasing the size of the loop body, often increasing the chances of identifying ILP in it
- Its main disadvantage lies in the increased code size.

Exe #1, Point #3

```
for (i=0;i<MAX;i++ )
{
        y[i] = x[i]+ 5;
}

for (i=0;i<MAX;i+=4 )
{
        y[i] = x[i]+ 5;
        y[i+1] = x[i+1]+ 5;
        y[i+2] = x[i+2]+ 5;
        y[i+3] = x[i+3]+ 5;
}
```

Exe #2, Point #1

Let consider a processor including a Branch Target Buffer (BTB).

Assuming that the processor uses 32 bit addresses, each instruction is 4 byte wide, and the BTB is composed of 4 entries, you are requested to

1. *Describe the architecture of the BTB*

The BTB is a table composed of 4 entries, each composed of 2 32-bit fields: Address and Target.

An additional 1 bit field may be present.

Exe #2, Point #2

2. *Describe in details the behavior of a BTB, explaining when it is accessed, and which input and output information are involved with each access*

The BTB is accessed each time an instruction is fetched.

Using the least significant log n bits of the instruction address (excluding the word alignment bits) an entry is selected. In this case n=4. The Address field of the entry is compared with the address of the fetched instruction: if they match, the Target field is uploaded in the PC.

When the instruction is completed, the BTB is possibly updated.

Exe #2, Point #3

3. *Identify the final content of the BTB if*
   - *The BTB is initially empty (i.e., full of 0s)*
   - *The following instructions are executed in sequence*

add.d f1,f2,f3	*located at the address 0x00A50050*
bnez r4,l1	*located at the address 0x00A50054; the branch is taken, and the branch target address is 0x00A60050*
mul.d f5,f6,f7	*located at the address 0x00A60050*
daddi r2,r2,-1	*located at the address 0x00A60054*
bez r2,l2	*located at the address 0x00A60058; the branch is not taken*
j l3	*located at the address 0x00A6005C; the branch target address is 0x00A60AA0*

Exe ##2, Point #3: BTB initial content

address	target
0x00000000	0x00000000
0x00000000	0x00000000
0x00000000	0x00000000
0x00000000	0x00000000

Exe #2, Point #3: BTB content after add.d execution

add.d address	
0x00A50050	0000 0000 1010 0101 0000 0000 0101 **00**00

address	target
0x00000000	0x00000000
0x00000000	0x00000000
0x00000000	0x00000000
0x00000000	0x00000000

Entry #0 is accessed

Exe #2, Point #3: BTB content after bnez execution

bnez address	
0x00A50054	0000 0000 1010 0101 0000 0000 0101 **01**00

address	target
0x00000000	0x00000000
0x00A50054	0x00A60050
0x00000000	0x00000000
0x00000000	0x00000000

Entry #1 is accessed

Exe #2, Point #3: BTB content after mul.d execution

mul.d address		
0x00A60050		0000 0000 1010 0110 0000 0000 0101 **00**00

address	target
0x00000000	0x00000000
0x00A50054	0x00A60050
0x00000000	0x00000000
0x00000000	0x00000000

Entry #0 is accessed

Exe #2, Point #3: BTB content after daddi execution

daddi address		
0x00A60054		0000 0000 1010 0110 0000 0000 0101 **01**00

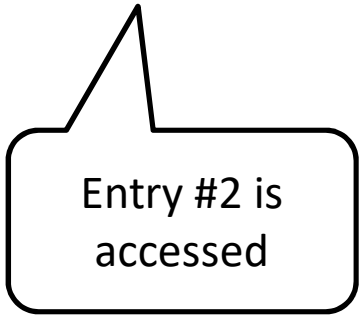
address	target
0x00000000	0x00000000
0x00A50054	0x00A60050
0x00000000	0x00000000
0x00000000	0x00000000

Entry #1 is accessed

Exe #2, Point #3: BTB content after bez execution

bez address		
0x00A60058		0000 0000 1010 0110 0000 0000 0101 **10**00

address	target
0x00000000	0x0000000
0x00A50054	0x00A60050
0x00000000	0x0000000
0x00000000	0x0000000



Exe #2, Point #3: BTB content after j execution

j address	
0x00A6005C	0000 0000 1010 0110 0000 0000 0101 **11**00

address	target
0x00000000	0x00000000
0x00A50054	0x00A60050
0x00000000	0x00000000
0x00A6005C	0x00A60AA0

Entry #3 is accessed

Exe #2, Point #3: BTB final content

address	target
0x00000000	0x00000000
0x00A50054	0x00A60050
0x00000000	0x00000000
0x00A6005C	0x00A60AA0