CMPUT 229 (Fall 2021) - Assignment #3
Instructor: José Nelson Amaral

►Solution◄

Question 1: (0 points)
Bank of Questions

Binary Representation of `jal` and Branches (V05)

For the RISC-V code given in Figure 1, notice the `bne` and `jal` instructions at the addresses `0x0040 0010` and `0x0040 0020` respectively. What are the hexadecimal representations of these instructions? To solve this problem, you need to recall that registers `t3` and `zero` are mapped to register number 28 and 0, respectively. Also, you need to recall the formats and effects of branch and jump instructions. They are shown below.

Question 2: (10 points)
Branch Instruction

31	30 25	24 20	19 15	14 12	11 8	7	6 0
imm[12]	imm[10:5]	rs2	rs1	funct3	imm[4:1]	imm[11]	opcode

```
bne rs, rt, imm <==> if(rs != rt)   PC <-- PC + {imm,1b'0}
                     else           PC <-- PC + 4
```

The opcode of `bne` instruction is `1100011` and the func3 is `001`.

Solution: When the processor is executing the `bne` instruction, the PC is at `0x0040 0010`. The target of the branch, i.e., `L2`, is 8 bytes away from this instruction. Therefore, the immediate value that needs to be added to the PC is 8. Remember that the instruction binary representation does not contain a value for the least-significant (bit zero) of the immediate

Thus, for the `bne` instruction, we have the values of each field as follows:

$$\begin{aligned}
\texttt{opcode} &= \texttt{1100011}\\
\texttt{func3} &= \texttt{001}\\
\texttt{rs1} &= \texttt{11100}\\
\texttt{rs2} &= \texttt{00000}\\
\texttt{imm[12:1]} &= \texttt{000000000100} = \texttt{0 0 000000 0100}
\end{aligned}$$

Therefore, the binary representation of the `bne` in the given RISC-V program is: `0000 0000 0000 1110 0001 0100 0110 0011`.

The hexadecimal representation is: `0x 000E 1463`.

Question 3: (10 points)
Jump and Link Instruction

31	30 21	20	19 12	11 7	6 0
imm[20]	imm[10:1]	imm[11]	imm[19:12]	rd	opcode

```
jal rd, imm <==> rd <-- PC + 4
                 PC <-- PC + {imm,1b'0}
```

The opcode of `jal` instruction is `1101111`.

Solution: When the processor is executing the `jal` instruction, the PC is at `0x0040 0020`. The target of the jump, i.e., `L1`, is -24 bytes away from this instruction. While computing the target address of a jump, the processor internally left-shifts the immediate specified in the jump instruction by one bit. Therefore, the immediate that we must specify in the `jal` instruction should be $-24/2 = -12$.

Thus, for the `jal` instruction above, we have values of each field as follows:

$$\begin{aligned}\texttt{opcode} &= \texttt{1101111}\\ \texttt{rd} &= \texttt{00000}\\ \texttt{imm[20:1]} &= \texttt{11111111111111110100}\end{aligned}$$

Therefore, the binary representation of the `jal` instruction in the above RISC-V program is: `1111 1110 1001 1111 1111 0000 0110 1111`.

The hexadecimal representation is: `0x FE9F F06F`.

```
0x0040 0000    mysteryProc:  addi  t1, zero, 32
0x0040 0004                  sll   s0, s0, t1
0x0040 0008             L1:  add   t2, a0, zero
0x0040 000C                  lbu   t3, 0(t2)
0x0040 0010                  bne   t3, zero, L2
0x0040 0014                  jal   zero, L3
0x0040 0018             L2:  addi  a0, a0, 1
0x0040 001C                  addi  s0, s0, 1
0x0040 0020                  jal   zero, L1
0x0040 0024             L3:  add   a0, zero, s0
0x0040 0028                  jalr  zero, ra, 0
```

Figure 1: Mystery code procedure