**2.24 Suppose the program counter (PC) is set to 0x2000 0000. Is it possible to use the jump (j) MIPS assembly instruction to set the PC to the address as 0x4000 0000? Is it possible to use the branch-on-equal (beq) MIPS assembly instruction to set the PC to this same address?**

**Jump Instruction (`j`) (No):**
The MIPS `j` instruction is a J-type instruction. It performs jump within a specific range. It includes a 26-bit address field that specifies the jump target. The 26-bit field in the jump instruction specifies the lower 26 bits of the target address. The upper 4 bits of the PC are concatenated with these 26 bits shifted left by 2 bits (to account for the word alignment, as MIPS instructions are aligned to word boundaries). This results in a 28-bit address. The starting PC is 0x2000 0000. The upper 4 bits of this address are `0010`. Since the `j` instruction does not alter these upper 4 bits, the only addresses we can jump to must start with `0010`. Therefore, jumping directly to 0x4000 0000 (which starts with `0100`) using a single `j` instruction is impossible.

**Branch-on-Equal Instruction (`beq`) (No):**
The `beq` instruction is an I-type instruction. It branches based on the equality of two registers. It uses a 16-bit signed immediate field that specifies the branch offset relative to the address of the next instruction. The offset is shifted left by 2 (again, for word alignment) and then added to the address immediately following the branch instruction (PC + 4). This offset allows for a branch range of ±131,072 bytes from the following instruction address, which is ±32,768 instruction words. The difference between 0x4000 0000 and 0x2000 0000 is 0x2000 0000, or 536,870,912 bytes. Hence, it is not possible with `beq`, either.

**2.40 If the current value of the PC is 0x00000000, can you use a single jump instruction to get to the PC address as shown in Exercise 2.39 (0010 0000 0000 0001 0100 1001 0010 0100)?**

**No**

**Current PC address:** 0000 0000 0000 0000 0000 0000 0000 0000
**Target Address:** 0010 0000 0000 0001 0100 1001 0010 0100

The target address from the instruction needs to have the upper 4 bits `0000` (since these are derived from the current PC and do not change with the `j` instruction).

This is not the case with the given target address.

Hence, moving the PC to the given target using a single MIPS `j` (jump) instruction is impossible.

**2.42 If the current value of the PC is 0x1FFFf000, can you use a single branch instruction to get to the PC address as shown in Exercise 2.39 (0010 0000 0000 0001 0100 1001 0010 0100)?**

**Yes**

0010 0000 0000 0001 0100 1001 0010 0100 = 0x20014924

Branch instruction uses a 16-bit immediate field that specifies a signed offset from the address of the instruction following the `beq`, which is PC + 4. The immediate field specifies the number of instruction words to jump forward or backward. The offset is multiplied by 4 (to convert from word offset to byte offset, as MIPS instructions are word-aligned).

Offset (bytes)=Target Address−(PC+4)=0x20014924−(0x1FFFf000+4)

The calculated offset in bytes is 88,352; when converted to words, it is 22,088.

The branch instructions in MIPS range from -32,768 to 32,767 in word offsets.

Hence, a single branch instruction can set the PC from `0x1FFFf000` to `0x20014924`.