Video explanation of solution is provided below the problem.

For all Beta related questions, you should make use of the Beta documentation, the Beta Instruction Summary, and the Beta Diagram.

Beta Control Signals

11/11 points (ungraded)
Following is an incomplete table listing control signals for several instructions on an unpipelined Beta. You may wish to consult Beta diagram and instruction set summary handouts.

The operations listed include two existing instructions and three proposed additions to the Beta instruction set:

```
LDX(Ra, Rb, Rc)            // Load, double indexed
    EA ← Reg[Ra] + Reg[Rb]
    Reg[Rc] ← Mem[EA]
    PC ← PC + 4

MVZC(Ra, literal, Rc)      // Move constant if zero
    If Reg[Ra] == 0 then Reg[Rc] ← SXT(literal)
    PC ← PC + 4

STR(Rc, C )               // Store relative
    EA ← PC+4+4*SEXT(C)
    Mem[EA] ← Reg[Rc]
    PC ← PC + 4
```

In the following table, **-** represents a **"don't care"** or unspecified value; Z is the value (0 or 1) output by the 32-input NOR in the unpipelined Beta diagram. Your job is to complete the table by filling in all the missing values.

Instr	ALUFN	WERF	BSEL	WDSEL	MOE	MWR	RA2SEL	PCSEL	ASEL	WASEL
JMP ✔ Answer: JMP	-	1 ✔ Answer: 1	-	0	-	0	-	2	-	0
BEQ ✔ Answer: BEQ	-	1	-	0	-	0	-	Z	-	0
LDX	A+B ✔ Answer: A+B	1	0 ✔ Answer: 0	2 ✔ Answer: 2	1	0	0	0	0	0
MVZC ✔ Answer: MVZC	B	Z	1	1	-	0	-	0	-	0
STR ✔ Answer: STR	A	0	- ✔ Answer: -	-	0	1 ✔ Answer: 1	1 ✔ Answer: 1	0	1	-

Explanation

1. In the first row of the opcode table, we see that PCSEL=2. This means that a JMP instruction is being executed. When executing a JMP instruction, the address of the instruction immediately following the JMP is stored in Rc. In order to be able to write to Rc, WERF must be 1.
2. In the second row of the table, we are given all of the control bits except for the actual instruction. Again we see that PCSEL!=0. In this case it is = Z. This means that instead of setting PC to PC+4, a new destination address is loaded into the PC depending on the value of Z. This means that this is some sort of branch instruction. Since the branch does happen when Z = 1, that means that the branch is taken when the register read equals 0, so this is a BEQ instruction.
3. The missing control bits for the LDX instruction are ALUFN, BSEL, and WDSEL. The effective address for the load is the sum of registers Ra and Rb. This means that BSEL = 0 to pass Rb through to the register file, and ALUFN = A + B to add the two registers together. The data that is loaded from this memory location is then written back into the register file by setting WDSEL = 2.
4. In the fourth row, we just need to fill in the missing operation. We see that WERF = Z which tells us that the register file is only written to if Z = 1. From this we can infer that this row corresponds to the MVZC operation.
5. Our last row, must then correspond to the STR operation. This operation writes the contents of register Rc into memory at the address that is specified by the effective address calculation. In order to write Rc to memory, RA2SEL must be 1 to pass the value of register Rc through the register file, and MWR = 1 in order to put the memory into write mode. The effective address for this instruction is PC + 4 + 4*SEXT(literal). This value comes from the additional adder and passes through the ALU by setting ASEL = 1 and ALUFN = A. This means that the B operand is ignored by the ALU so BSEL is a don't care.

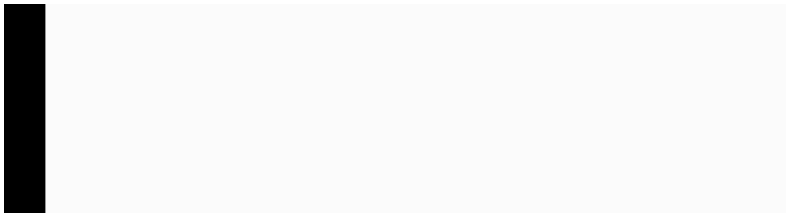
Submit

Answers are displayed within the problem

Beta Control Signals

Start of transcript. Skip to the end.

In order to better understand the role of each of the beta control signals, we will work



0:00 / 12:09 Speed 1.0x

through an example problem that provides us with a partially filled control table for 5 different instructions.

Two of these instructions are existing beta instructions that we must infer from the provided control signals.

The other three are three new instructions that we are adding to our beta by modifying the necessary control signals to produce the desired behavior of each of the operations.

The first instruction that we want to add to our beta is an LDX instruction which is a load that is double indexed.

Video
Download video file

Transcripts
Download SubRip (.srt) file
Download Text (.txt) file

Discussion

Topic: 13. Building the Beta / WE13.2

Hide Discussion

Add a Post

Show all posts | by recent activity

How much points are Lab 7 & Assignment 4	I have scored only 1.6 point up to now. Had already given up but just discovered that the Beta is due only the 1 February. So I still have time to complete it togeth...	4
Strange error detected while checking /beta/pc module	Hi All, I got a strange error (see bellow) while checking my module /beta/pc. The timestamp is outside of the test time range. I don't understand what is the issue ...	5