WE9.1

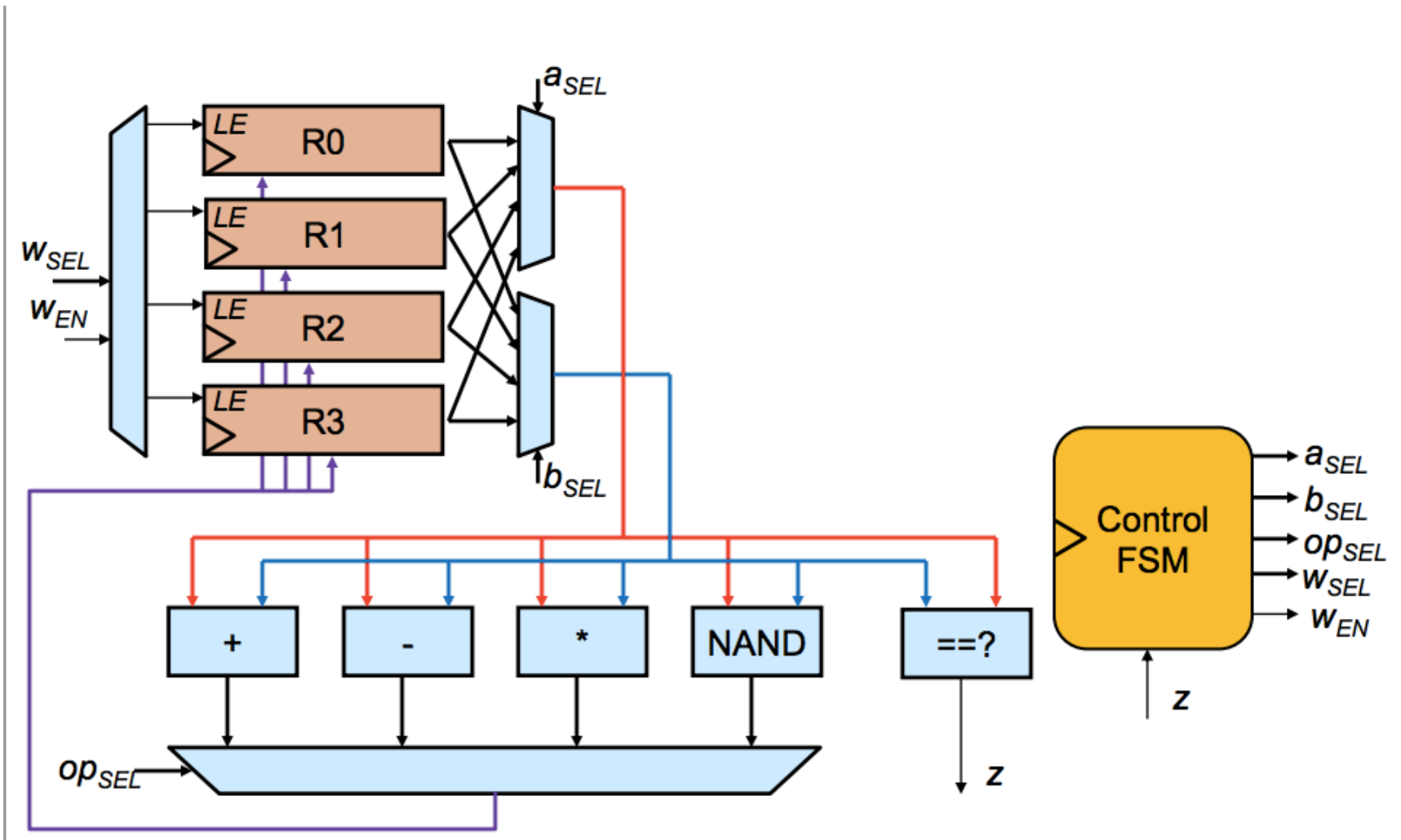
Bookmark this page

Video explanation of solution is provided below the problem.

Simple Programmable Datapath

33/33 points (ungraded)

You are given a simple programmable datapath shown below:



This datapath reads two operands A and B from a register file that contains 4 registers R0 - R3. It then performs one of four operations (ADD, SUB, MUL, or NAND) on A and B and optionally writes the result back to another register. It also has logic to determine if A is equal to B or not and sets the control signal Z accordingly.

The control signals for this datapath are described below:

Control ROM Input:

Z 1: A equals B 0: A not equal to B

Control ROM Outputs:

ASEL 2 bit selector for A input: (00 = R0, 01 = R1, 10 = R2, 11 = R3)

BSEL 2 bit selector for B input: (00 = R0, 01 = R1, 10 = R2, 11 = R3)

OPSEL 2 bit selector of operation to perform on A and B (00: ADD, 01: SUB, 10: MUL, 11: NAND)

WSEL 2 bit selector for destination register for write operations: (00 = R0, 01 = R1, 10 = R2, 11 = R3)

WEN Write enable - controls whether or not result is written back to a register or not.

You are asked to program this datapath to compute the function $3 * N - 2$, and store the result in R3. A semi-complete list of instructions and control signals is provided in the table below. Complete the missing entries to have the datapath produce the desired result.

Assume that the initial values of the registers are:

R0 = 1
R1 = 0
R2 = -1
R3 = N

A semi-complete description of what should happen in each state is shown here. You will need to determine the correct values of RX, RY, and RZ.

Calculator

```
S0: ADD(R2, RX, R2)   - produce -2 in R2
S1: ADD(RY, R0, R1)   - produce 2 in R1
S2: ADD(R0, R1, R1)   - produce 3 in R1
S3: MUL(RZ, R3, R3)   - produce 3*N in R3
S4: ADD(R3, R2, R3)   - produce 3*N - 2 on R3
S5: HALT()            - halt execution
```

Fill in the control ROM so that states S0 through S5 are executed in order to produce 3*N - 2 and store that result into R3. Use X to represent don't cares (i.e., the signal can be either a 0 or a 1).

Instr	S[2:0]	Z	S'[2:0]	ASEL	BSEL
ADD(R2, RX, R2)	000	X	001	10	10 ✔ Answer: 10
ADD(RY, R0, R1)	001 ✔ Answer: 001	X ✔ Answer: X	010 ✔ Answer: 010	00 ✔ Answer: 00	00 ✔ Answer: 00
ADD(R0, R1, R1)	010 ✔ Answer: 010	X ✔ Answer: X	011	00 ✔ Answer: 00	01 ✔ Answer: 01
MUL(RZ, R3, R3)	011	X	100 ✔ Answer: 100	01 ✔ Answer: 01	11 ✔ Answer: 11
ADD(R3, R2, R3)	100	X	101 ✔ Answer: 101	11 ✔ Answer: 11	10 ✔ Answer: 10
HALT()	101 ✔ Answer: 101	X	101 ✔ Answer: 101	XX	XX

Explanation

In order to execute $3 * N - 2$ our datapath needs to begin at state S0 and then pass through each successive state once until it reaches the HALT() state where it should remain once the function has been executed and written back to R3.

Let's examine the missing entries. In the first ADD(R2, RX, R2) we are trying to produce -2 in R2. Since we are told that R2 = -1 initially, RX must equal R2 so that this instruction will add $-1 + -1$ to produce -2 and store that result back into register R2. This means that for state S0 = 000, the missing entries are BSEL = 10 to select R2, OPSEL = ADD, WSEL = 10 to write the result back into R2, and WEN = 1 in order to enable the result to be written back into the register file. We are provided with the fact that the next state is 001 = S1.

In state 1, our goal is to produce 2 in R1. We are given the template code of ADD(RY, R0, R1). We know that R0 = 1 initially. In order to produce 2 in R1, we must add R0 to itself and store that result back to R1. This means that for the ADD(RY, R0, R1) instruction, the missing entries are as follows. The current state if 001 and the next state is 010. The control signal Z does not affect any of our computation for this problem so it will always be a don't care = X. ASEL and BSEL should both select R0 so they should be set to 00. The destination register is R1, so WSEL = 01, and finally WEN = 1 in order to permit writing the result to R1.

In state 2, we are told that an ADD(R0, R1, R1) operation is to be performed in order to add 1 to 2 to produce 3 in R1. To do this, we have current state = 010 and next state = 011. Once again Z = X. ASEL = 00 to select R0 and BSEL = 01 to select R1. WEN = 1 in order to allow the result to be written back to register R1.

In state 3, our goal is to produce 3*N in R3. Since R3 = N and at this point R1 = 3, we want to multiply R1 by R3 to produce 3*N. This result is stored back into R3. So the missing control signals are: next state (S'[2:0]) = 100, ASEL = 01 to select R1, BSEL = 11 to select R3, OPSEL = MUL to perform a multiply operation, WSEL = 11 to write our result back into R3, and finally WEN = 1 to enable the writing of the result back into the register file.

After state 3, we proceed to state 4 where we perform ADD(R3, R2, R3). To do that, the next state = 101, ASEL = 11, BSEL = 10, OPSEL = ADD, WSEL = R3, and WEN = 1. This state will write our desired result back into R3.

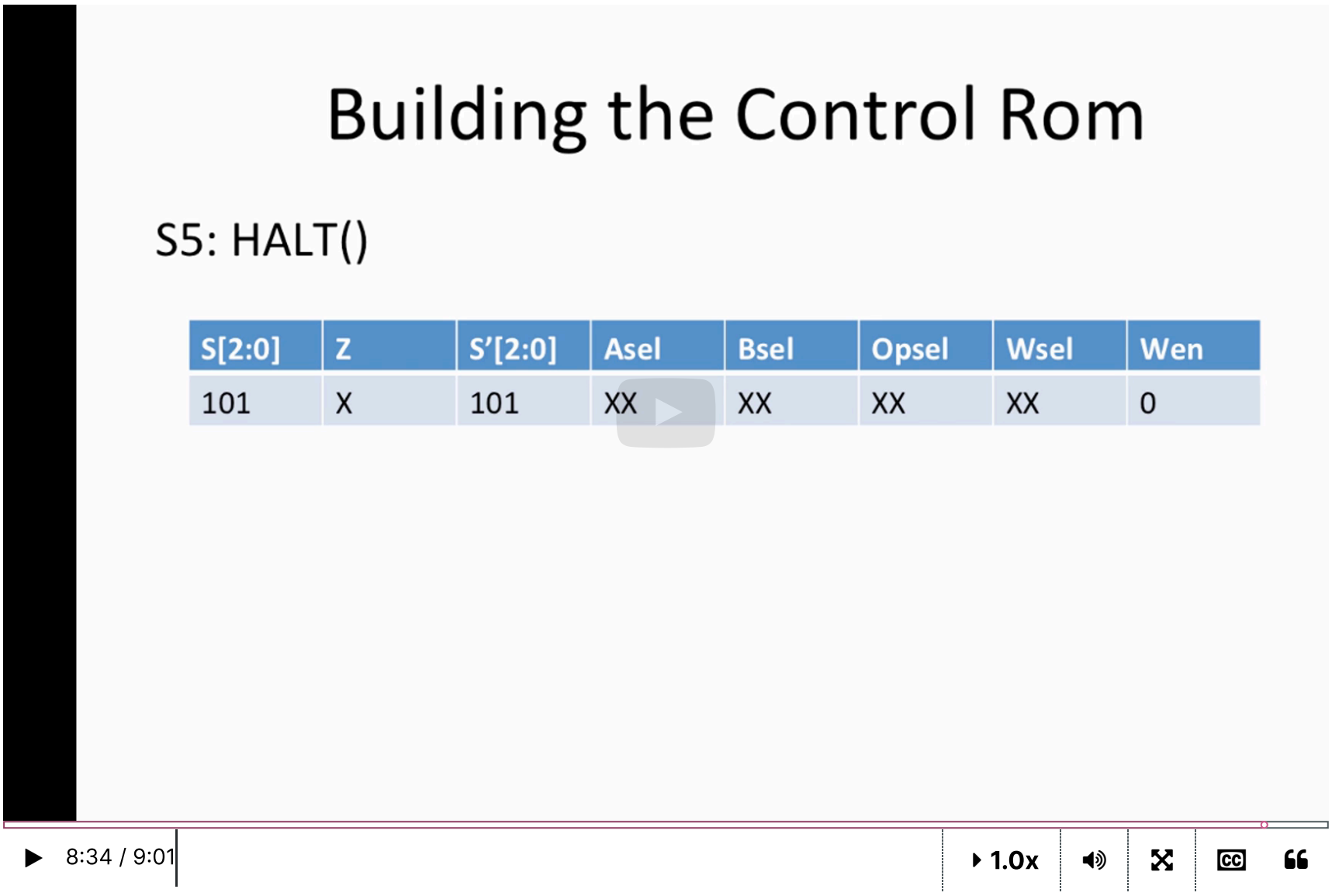
Once we have completed all the necessary tasks, we proceed to state 5 which is the HALT() state and stay there. So S[2:0] = S'[2:0] = 101 in state S5. Z continues to be X. Nowe we have a few more don't care situations. ASEL, BSEL, OPSEL, and WSEL are all don't cares. However, WEN = 0 in order to ensure that we don't accidentally write garbage back into our register file.

Submit

Answers are displayed within the problem

Calculator

Programmable Architectures



Video
Download video file

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

Discussion

Topic: 9. Designing an Instruction Set / WE9.1

Hide Discussion

Add a Post

Show all posts by recent activity

- Value of Z — 10
  I don't agree with the grader for the value of Z. Even if we don't care its value, this value exists and is perfectly determined (it can't ...
- Efficiency — 2
  Inveterate low-level mathematician/programmer here... - S0: ADD(R2, R3, R2) - produce N - 1 in R2 - S1: ADD(R2, R3, R3) - produce...
- BSEL — 2
  Information for BSEL should probably read: > BSEL 2 bit selector for **B** input: (00 = R0, 01 = R1, 10 = R2, 11 = R3) Also > Con **t...

Previous Next

edX

About
Affiliates

Calculator