Design of Digital Circuits and Systems, HW4

Algorithmic State Machines with Datapath

Deliverables (Gradescope)

1) Homework problem solutions (*i.e.*, text, diagrams, screenshots, work) in a single PDF file ending in `.pdf` (all lowercase).
    a) At the end of this document, estimate how long you spent working on the homework and rate the difficulty on the following scale: Very Hard — Hard — Moderate — Easy — Very Easy
2) (System)Verilog files *`divider.(s)v`*, *`downcount.(s)v`*, *`muxdff.(s)v`*, *`regne.(s)v`*, *`shiftlne.(s)v`*, and new test bench file *`divider_tb.sv`*.

Problems

Problem 1:

- Draw the ASMD chart for each of the following state transition descriptions:
    a) If `x = 1`, control goes from state $S1$ to state $S2$; if `x = 0`, generate a conditional operation R <= R+2 and go from $S1$ to $S2$.
    b) If `x = 1`, control goes from state $S1$ to state $S2$ and then to state $S3$; if `x = 0`, control goes from $S1$ to $S3$.
    c) Start from state $S1$; if `xy = 11`, go to state $S2$; if `xy = 01`, go to state $S3$; if `xy = 10`, go to $S1$; otherwise, go to $S3$.

Problem 2:

We want to build a digital system that counts the number of people in a room. The one door through which people enter the room has a photocell that changes a signal **x** from `1` to `0` while the light is interrupted. People leave the room from a second door with a similar photocell that changes a signal **y** from `1` to `0` while the light is interrupted. The datapath circuit consists of an **up-down counter** with a **display** that shows how many people are in the room. Only one person can pass through each door at a time and each person should only be counted once.

- Construct a block diagram and an ASMD chart for this system. Hint: You should use 4 states corresponding with each combination of the sensors' outputs x and y.

Problem 3:

You are provided a buggy Verilog implementation of a divider circuit similar to the one discussed in lecture in the files: *`divider.v`*, *`downcount.v`*, *`muxdff.v`*, *`regne.v`*, and *`shiftlne.v`*.

- Create a test bench and use it to update the provided code to work as intended (note: all Verilog code works in SystemVerilog if you would like to update to *`.sv`* files). Make sure that your code fixes are documented in code comments for each file.

- Include screenshots of any warnings, errors, or simulation results of the original code and point out the parts that helped you identify the faulty behavior that you found.

Your understanding of the divider's datapath and control path should help in figuring out the faults in the program (if any). For additional info, refer to section 7.5 in Brown & Vranesic.

Signal naming scheme:

- "E" is short for **enable**, "L" is short for **load**
- "R" is short for **register**, "C" is short for **counter**
- s is the **start** signal, z is the **zero** signal (count==0)

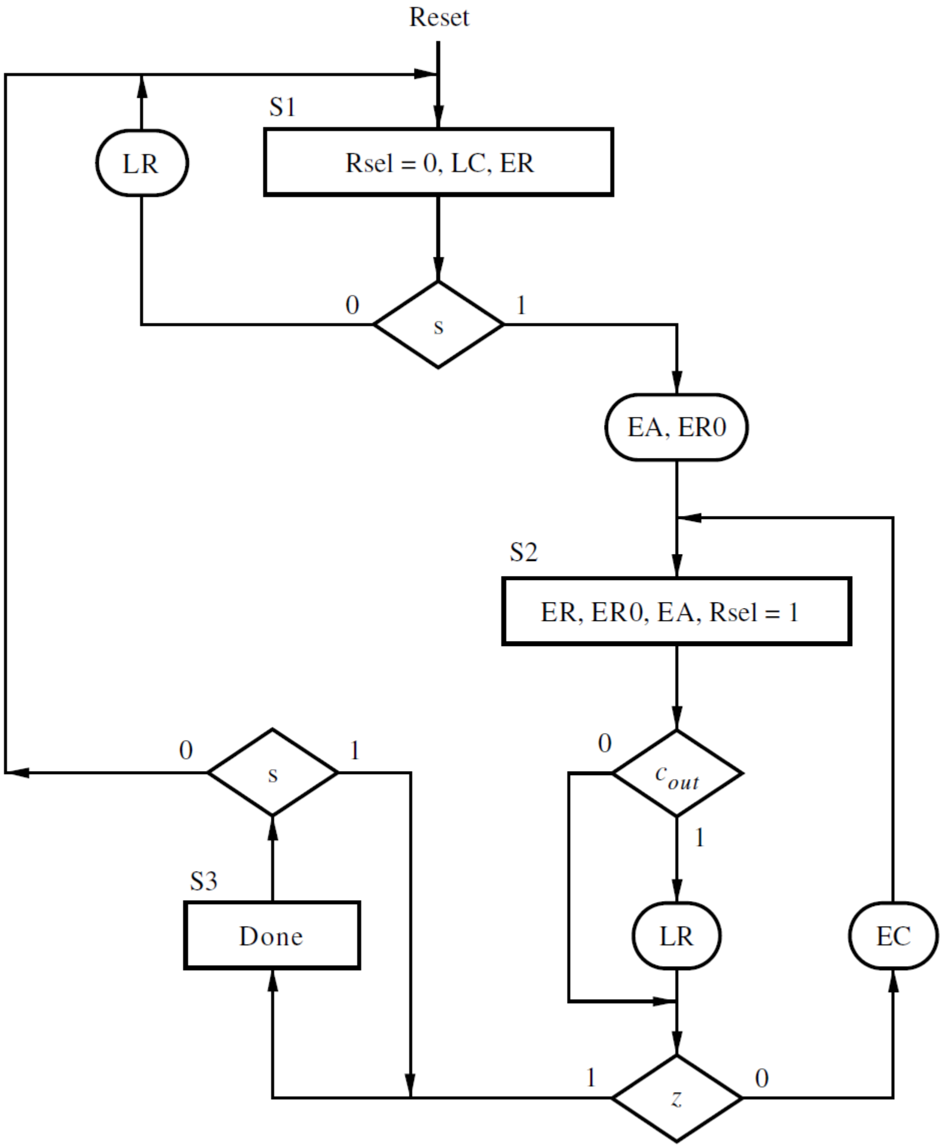

Figure 7.33 ASM chart for the enhanced divider control circuit.

Component naming scheme:

- Upper-left Left-shift register is “Register A” (A); upper-right Register is “Register B” (B)
- Lower Left-shift register is the “Remainder Register” (R, confusing, I know)

The output signals at the bottom are the quotient (Q) and remainder (R).

The down-counter (C) is not shown, but used to generate the z signal.

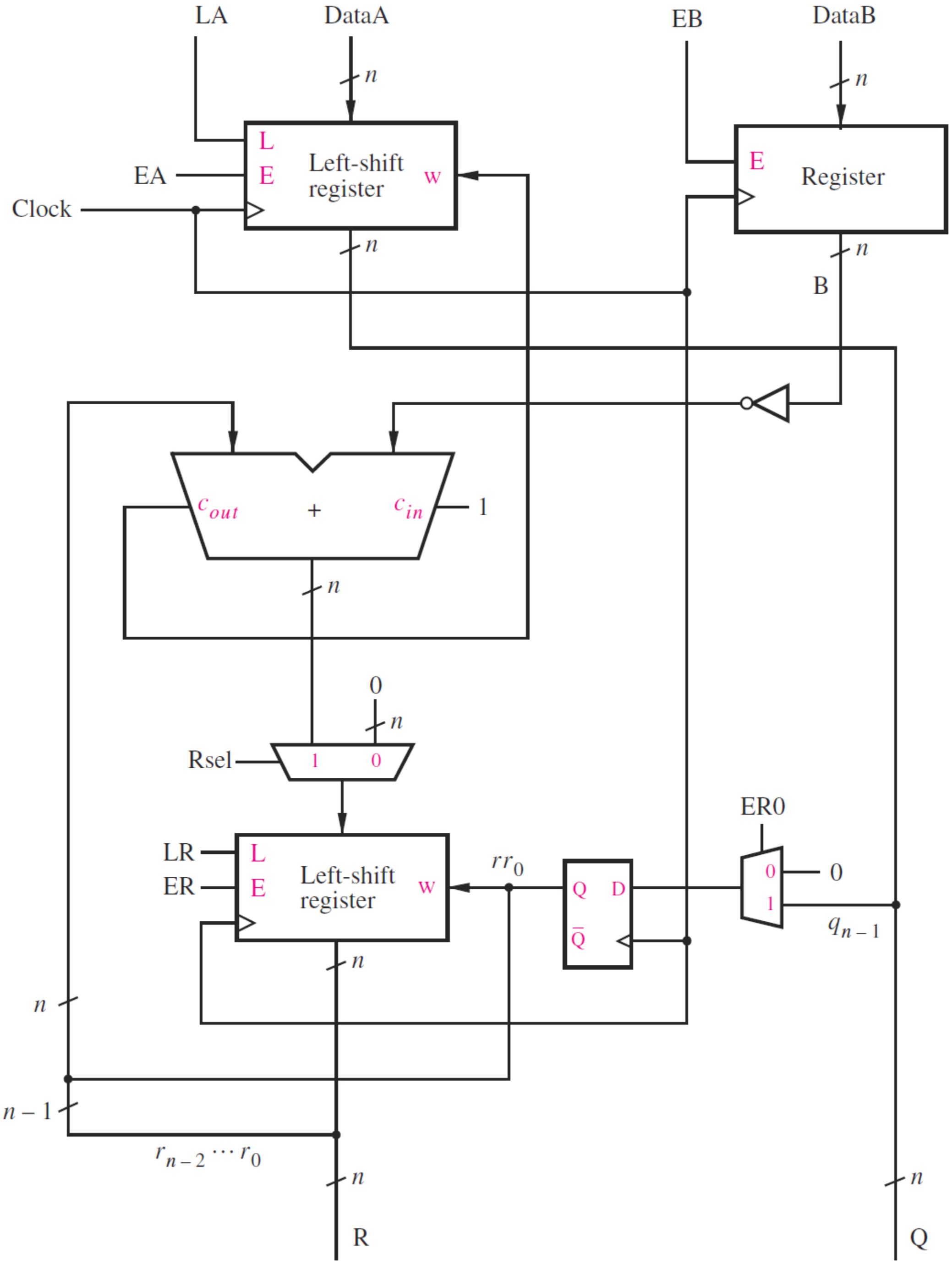

Figure 7.34 Datapath circuit for the enhanced divider.