University of Illinois at Urbana-Champaign Dept. of Electrical and Computer Engineering

ECE 120: Introduction to Computing

The Design of the Lab FSM

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved.

14. 4

What Problem Must be Solved?

In the lab, your task is

- o to build a small FSM
- to control a coin-operated vending machine.

Inputs are produced by coins.

Outputs specify

- · whether a coin should be accepted, and
- whether a product should be released.

The **design is extremely simple** so as to minimize the number of chips needed.

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved.

What Purpose Does the Lab Serve?

Help students to make the connection between lines and boxes on paper and wires and chips in a real system.

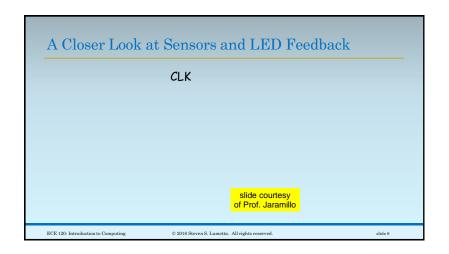
Help students to realize that the knowledge they have gained in ECE120 enables them to build real systems with sensors and actuators; in other words, to interface with the real world.

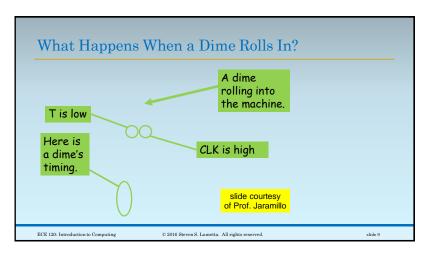
ECE 120: Introduction to Computing

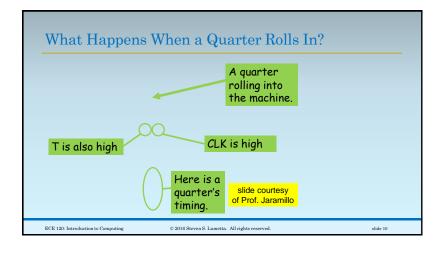
© 2016 Steven S. Lumetta. All rights reserved.

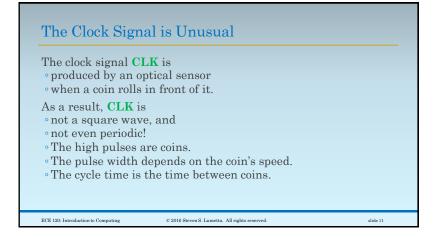
slide 3

Let's Take a Look at the Physical System These produce CLK and T. Gate is controlled by of Prof. Jaramillo ECE 120: Introduction to Computing C 2016 Steven S. Lumetta. All rights reserved.









CLK Signal is Sufficient for Our FSM's Needs

However, the **CLK** signal is **sufficient for our needs**.

You build with positive edge-triggered D flip-flops.

Because of the positioning of optical sensors, **T** is stable (0 for a dime, 1 for a quarter) when **CLK** rises.

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved.

Lab Machine as a Sequence Recognizer

A **sequence recognizer** looks for bit patterns in a serial input stream.

Previously, we developed a 01 sequence recognizer as an example of the difference between Mealy and Moore machines.

For the lab, we can

- treat the sequence of coins as a serial input (0 for dimes—\$0.10; 1 for quarters—\$0.25).
- then look for patterns that total \$0.35.

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved.

A Process for Developing a 01/10 Recognizer

If T is the serial input of coin types, we must produce product release output P = 1 whenever we see 01 or 10.

We use the following process to develop the sequence recognizer:

- 1. Create a state for each bit in each sequence.
- 2. Complete the transitions not used in the desired sequences.
- 3. Minimize by merging redundant states.

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved.

slide 14

Inputs, Output, and Notation

Inputs:

T 0 for dime, 1 for quarter

(transitions use black/red to denote T)

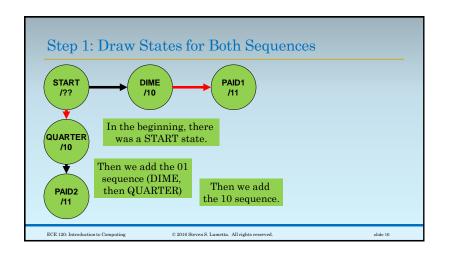
Outputs:

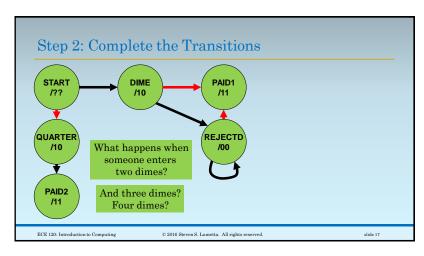
- A 1 to accept the coin just inserted (0 to reject it, returning to the user)
- P 1 to release the product

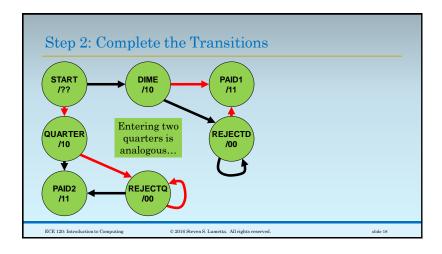
States are marked with /AP.

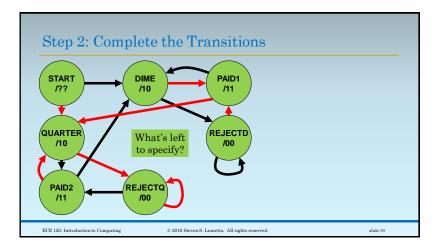
ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved.

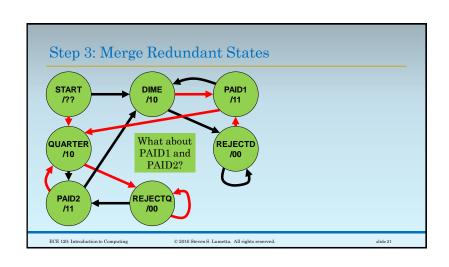


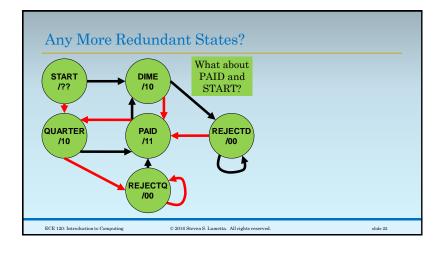


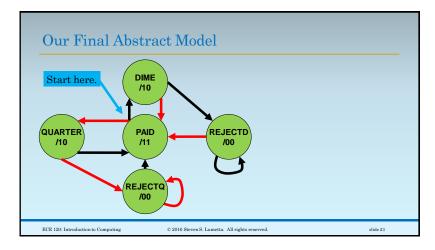




Step 3: Merge Redundant States Now we can merge redundant states. To merge states, it suffices to find two states with identical outputs and identical next states. Let's take a look.







Use Human Information to Define the Representation

We need 3 bits for 5 states.

Let's use human information to define the representation.

Think about the sequence of coins that has been inserted into the machine.

Let's call the last coin T_0 .

And the one before that T_{-1} .

And so forth.

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved

State Bit Definition

Define the state bits as follows:

 S_2 is T_0 , the last coin type.

 S_1 is 1 iff one or more quarters were inserted

• before the last coin (T₋₁, T₋₂, and so on)

• but after the last product release.

 S_0 is 1 iff one or more **dimes** were inserted

• before the last coin (T-1, T-2, and so on)

• but after the last product release.

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved

Does that Approach Work? Check the State IDs

For **DIME**, the only coin since the last payment is a single dime.

So we have... DIME

QUARTER $S_2 = 0$ (the dime)

REJECTD $S_1 = 0$ (no quarters at all)

REJECTQ

 $S_0 = 0$ (the dime is unique)

PAID

000

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved

slide 26

Calculate the State ID for QUARTER

For **QUARTER**, the only coin since the last payment is a single quarter.

So we have...

000 DIME

 $S_2 = 1$ (the quarter)

QUARTER 100

 $S_1 = 0$ (the quarter is unique)

REJECTD REJECTQ

 $S_0 = 0$ (no dimes at all)

PAID

© 2016 Steven S. Lumetta. All rights reserved. ECE 120: Introduction to Computing

Calculate the State ID for REJECTD

For **REJECTD**, we have seen two or more dimes but no quarters.

So we have... DIME 000 QUARTER 100 $S_2 = 0$ (the last dime) REJECTD 001 $S_1 = 0$ (no quarters at all) REJECTQ $S_0 = 1$ (extra dimes) **PAID**

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved

Calculate the State ID for REJECTQ

For **REJECTQ**, we have seen two or more quarters but no dimes.

So we have... DIME 000 QUARTER 100 $S_2 = 1$ (the last quarter) REJECTD 001 $S_1 = 1$ (extra quarters) REJECTQ 110 $S_0 = 0$ (no dimes at all) **PAID**

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved

What is the State ID for PAID?

For **PAID**, we could have gotten either coin last! Before the last coin, we got one or more of the other kind.

DIME 000

For a last quarter, we have... QUARTER 100

 $S_2 = 1$ (the last quarter) REJECTD 001

 $S_1 = 0$ (no extra quarters)

REJECTQ 110

 $S_0 = 1$ (at least one dime) **PAID** 101

ECE 120: Introduction to Computing

© 2016 Steven S. Lumetta. All rights reserved

What is the State ID for PAID?

But what about a last dime?

000 DIME QUARTER 100

 $S_2 = 0$ (the last dime) REJECTD 001

 $S_1 = 1$ (at least one quarter) REJECTQ 110

 $S_0 = 0$ (no extra dimes) **PAID** 101

So we have two bit patterns! **PAID** 010

ECE 120: Introduction to Computing

slide 30

© 2016 Steven S. Lumetta. All rights reserved.

That's All!

You will finish the rest of the design and implement it in the lab...

In Lab 7, you will implement and simulate the FSM in Altera Quartus.

- You already have the output logic done.

In Lab 8, you will assemble your FSM on the prototyping board.

- You already have the output logic done.

ECE 120: Introduction to Computing

 $\ensuremath{\mathbb{C}}$ 2016 Steven S. Lumetta. All rights reserved.