

Problem 4.1.3

# State Machine: Tollbooth Using PLTW S7

---

## INTRODUCTION

Now that you have seen examples of **state machines** and how they work, it is time for you to create your own. The state machine you will design is based on the tollbooth gate example.

In this design you are required to use a PLD to create a functioning prototype. Can you think of any advantages to creating this design in simulation and exporting it to a PLD rather than using the VEX Cortex Controller?

The tollbooth gate state machine has the following four inputs and four outputs.

### Inputs

<b>OS - Open Switch Push-button</b>	This input is activated by the user. When the button is pressed, it outputs a logic 1 and causes the gate to open. This input button is a push-button switch located on the PLTW S7.
<b>CS - Close Switch Push-button</b>	This input is activated by the user. When this button is pressed, it outputs a logic 1 and causes the gate to close. This input button is also a push-button switch located on the PLTW S7.
<b>OL - Open Limit Switch</b>	This input is activated by the gate mechanism when it is fully open. When this input switch is pressed, it outputs a logic 1. This limit switch is located on the test fixture.
<b>CL - Close Limit Switch</b>	This input is activated by the gate mechanism when it is fully closed. When this input switch is pressed, it outputs a logic 1. This limit switch is located on the test fixture.

### Outputs

<b>MO - Motor Open</b>	This output is a logic 1. It will cause the tollbooth gate to open.
<b>MC - Motor Close</b>	This output is a logic 1. It will cause the tollbooth gate to close.
<b>GO - Gate Open</b>	This output is connected to an LED on the PLTW S7. It is ON when the gate is fully open.
<b>GC - Gate Closed</b>	This output is connected to an LED on the PLTW S7. It is ON when the gate is fully closed.

## EQUIPMENT

- Circuit Design Software (CDS)
- Digital MiniSystem (DMS)
  - myDAQ
  - myDigital Protoboard
  - PLTW S7 FPGA Module
- SN754410 Quadruple Half-H Driver or L293 H-Bridge Driver
- Resistors: 3 100k $\Omega$ ; 1 180 $\Omega$ ; 1 330 $\Omega$
- Variable power supply or 4 AA Batteries with holder
- #22-gauge solid wire
- PLTW Digital Electronics VEX Kit
- Hex wrenches (3/32 in. and 5/64 in.)

## RESOURCES



Problem 4.1.3 Using PLTW S7 Construction Guide



Problem 4.1.3 Using PLTW S7\_Worksheet



System Behavior

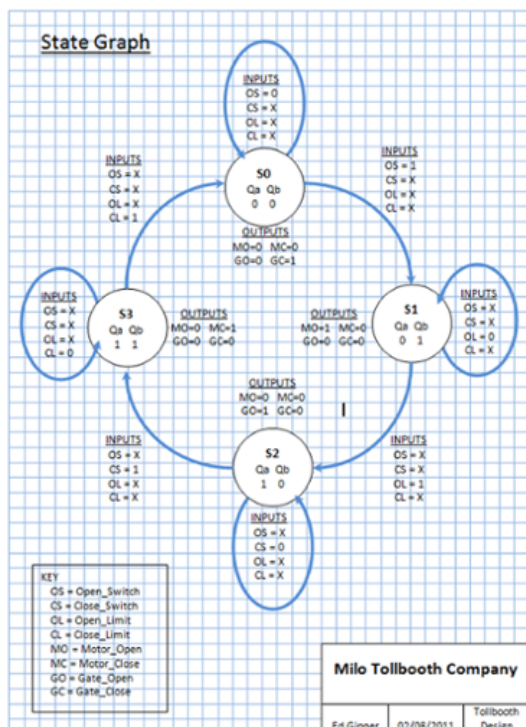


# Procedure

## Design

Using the provided state graph and [state transition table](#) as a starting point, design a state machine that will control the opening and closing of the tollbooth. This state machine will be implemented in the FPGA. Your design should clearly show how you will implement the following:

- Input Combinational Logic – What happens next based on the states of the buttons and switch inputs?
- Memory – How do you use flip-flops to create the number of states needed and to trigger the state transition when an input is made?
- Output Combinational Logic – Which motor output (Open/Close) and what LED indicator should come on to indicate the current state of the gate?
- Refer to the [System Behavior video](#) for guidance.



## State Transition Table

Present State		Inputs				Next State		Outputs				
Qa	Qb	OS	CS	OL	CL	Qa*	Qb*	MO	MC	GO	GC	
S0	0	0	0	X	X	X	S0	0	0	0	0	1
S0	0	0	1	X	X	X	S1	0	1	0	0	1
S1	0	1	X	X	0	X	S1	0	1	1	0	0
S1	0	1	X	X	1	X	S2	1	0	1	0	0
S2	1	0	X	0	X	X	S2	1	0	0	0	1
S2	1	0	X	1	X	X	S3	1	1	0	0	1
S3	1	1	X	X	X	0	S3	1	1	0	1	0
S3	1	1	X	X	X	1	S0	0	0	0	1	0

## Design Equations

$$Qa = Qa^* = \overline{Qa} \cdot Qb \cdot OL + Qa \cdot \overline{Qb} \cdot \overline{CS} + Qa \cdot \overline{Qb} \cdot CS + Qa \cdot Qb \cdot \overline{CL}$$

$$Qb = Qb^* = \overline{Qa} \cdot \overline{Qb} \cdot OS + \overline{Qa} \cdot Qb \cdot \overline{OL} + Qa \cdot \overline{Qb} \cdot CS + Qa \cdot Qb \cdot \overline{CL}$$

$$MO = \overline{Qa} \cdot Qb$$

$$MC = Qa \cdot Qb$$

$$GO = Qa \cdot \overline{Qb}$$

$$GC = \overline{Qa} \cdot \overline{Qb}$$

Milo Tollbooth Company

Ed Ginner

02/08/2011

Tollbooth Design

## State Graph Analysis

---

From the provided state graph, you can see that the tollbooth design requires four states (S0–S3). The two required **state variables** have been identified as Qa and Qb. You should maintain these labels in your design. How did you know that two state variables were required for this design?

With each clock signal, the buttons and switches are checked to determine whether the next state should be transitioned to or there is no change in the state. Can you identify the physical state of the tollbooth based on the provided state graph?

State	Qa	Qb	Physical State of the Gate (Open/Closed) or (Opening/Closing)
S0	0	0	The gate is...
S1	0	1	The gate is...
S2	1	0	The gate is...
S3	1	1	The gate is...

## State Transition Table Analysis

---

Now let’s look at how the *State Transition Table* was generated from the *State Graph*. For each state, there must be 4 sets of inputs that cause a transition. So with 4 possible states, there are actually 8 sets of inputs that must be checked on each clock signal to determine whether the state is held or a transition to the next state will take place.

In the first line of the transition table (S0), the gate is closed.





In the second line of the transition table (S0), the operator presses the Open Switch Push-button.

Without looking at the notes above, begin on LINE 2 and see if you can accurately predict the NEXT STATE and the outputs as you move down each consecutive line. Below each line in the transition table is a box where you can describe (in your own words) what is happening at the tollbooth. The first few descriptions have been given as examples to get you started. After you have worked through the table, you can go back and put in the “Don’t Care” comments.




**Note:** This content is meant to help you understand the tollbooth state machine design. It is for your use only. You should document all of your work for final evaluation in your engineering notebook.

### Tollbooth State Transition Table Analysis





---

PRESENT STATE			Operator Push-buttons		Limit Switches		NEXT STATE*			Outputs	
State	Qa	Qb	Open Switch	Close Switch	Open Limit	Close Limit	State	Qa*	Qb*	Motor Open	Motor Closed
S0	0	0	0	0			S0	0	0		





The gate is closed with no inputs from the operator.

S0	0	0	1	0			S1	0	1	0	
----	---	---	---	---	---	---	----	---	---	---	---




























The operator has pressed the push-button "OPEN SWITCH", but the gate is still closed. "MOTOR OPEN" state.

S1	0	1			0		S1	0	1	1	
----	---	---	--	--	---	--	----	---	---	---	--

The gate is opening but has not reached the "OPEN LIMIT". Reaching the "OPEN LIMIT" should be the

S1	0	1			1		S2	1	0	1	
----	---	---	---	---	---	---	----	---	---	---	---

The gate has reached the "OPEN LIMIT", but "MOTOR OPEN" is still on. "GATE OPEN" with motor off state.

S2	1	0									
S2	1	0									
S3	1	1									



S3	1	1										
----	---	---	--	--	--	--	--	--	--	--	--	--

## Design Equations

---

The input combinational logic to your flip-flops is represented by the NEXT STATE\* in your transition table. That is, the Da and Db into your flip-flops are the same as the Qa\* and the Qb\* in your transition table. Therefore, you should be able to generate two logic expressions from the INPUTS listed as a (logic 1) for Qa\* and Qb\*, respectively. See whether you can generate the unsimplified expressions without referring to the previous information.

### *Input Combinational Logic*

$$Da = Qa^* =$$

$$Db = Qb^* =$$

Capture these expressions in your engineering notebook and simplify into your own unique design.

There are four possible outputs: “Gate Closed Indicator”, “Motor Open Signal”, “Gate Open Indicator”, and “Motor Close Signal”. What are the logic expressions for these four outputs based on the transition table?

### *Output Combinational Logic*

$$MO =$$

$$MC =$$

GO =

GC =

## Simulation

---

Simulate the design in the CDS. Remember, you can simulate in PLD mode using the interactive digital constant for the switches and probes to represent the outputs.

Start the simulation with CL activated (logic 1), while OS, OL, and CS are not activated (logic 0). This will simulate the gate closed and no buttons activated. In simulation, it is fine to use a low frequency clock signal (20 Hz or lower). Would this be a good idea when you actually create the prototype? Use the simulation to verify that your design works.

## Prototyping

---

Create the tollbooth test fixture using the provided VEX structural components, two limit switches and a 2-wire motor. Images of an example test fixture are provided in the next section.

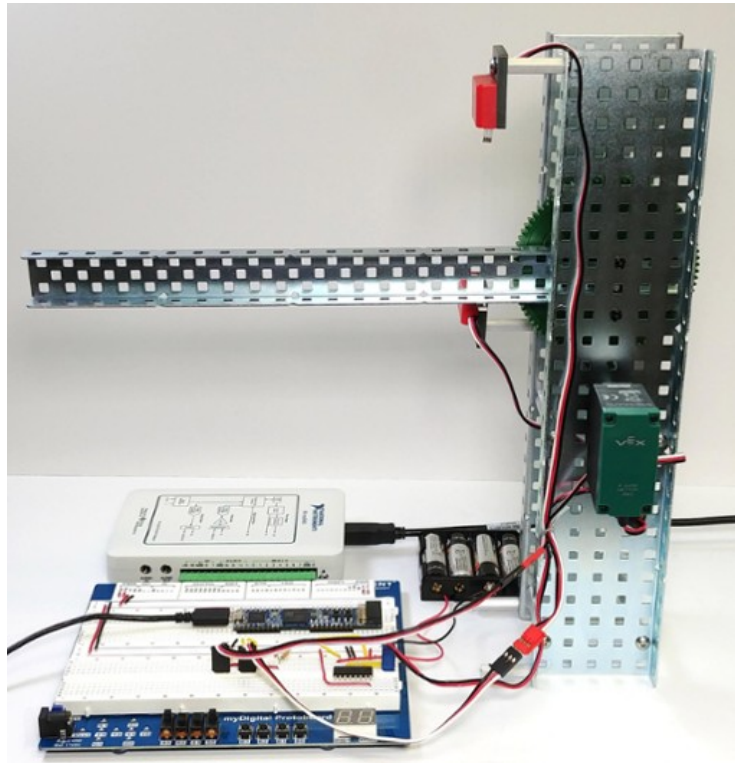
In addition to creating the mechanical tollbooth test fixture, you need to use additional components so that you can drive a DC motor using digital logic signals (SN754410 Quadruple Half-H Driver or the L298 Dual Full-Bridge Driver). VEX motors operate at voltages higher than the digital 5V used by TTL. It is also a good practice to use a different or backup voltage source for logic circuits that drive motors. Why do you think this is a good idea?

**Note:** To protect the DMS, it is important to use an EXTERNAL power supply for the 6V necessary to drive the motor. Check that your external power supply and DMS share a common ground.

## Tollbooth Test Fixture – Build

---

Use the Construction Guide in the Resources section to build the test fixture.

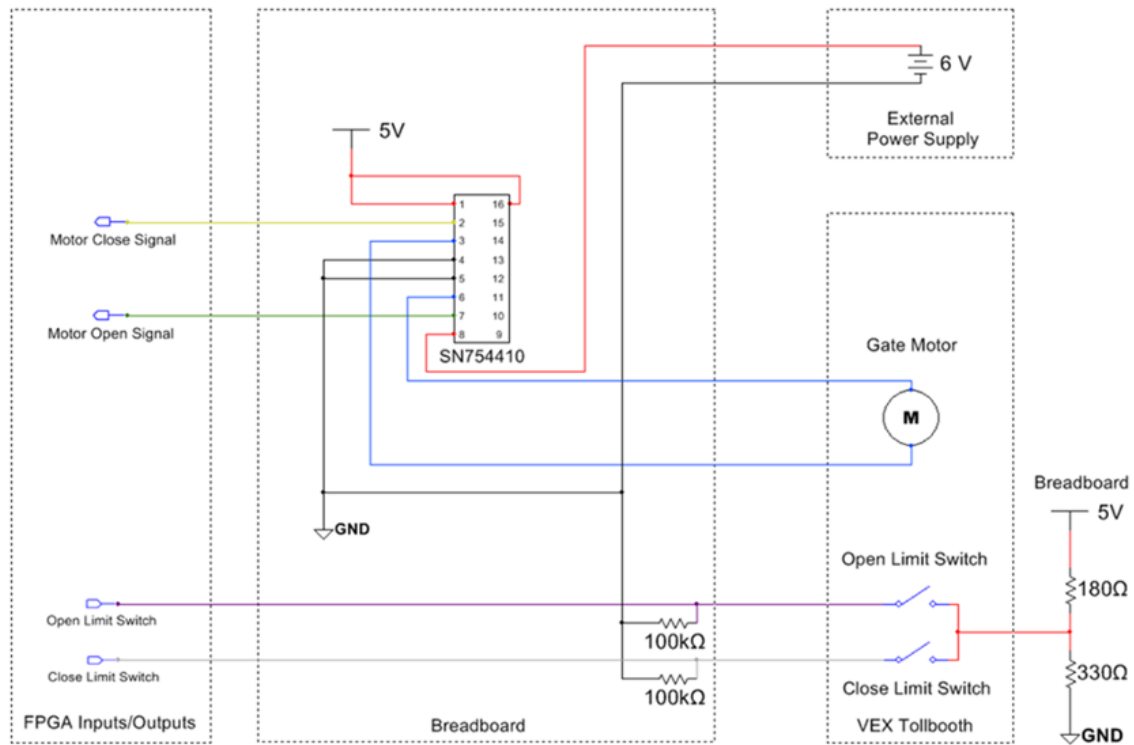


## Tollbooth Test Fixture – Wiring Diagram

---

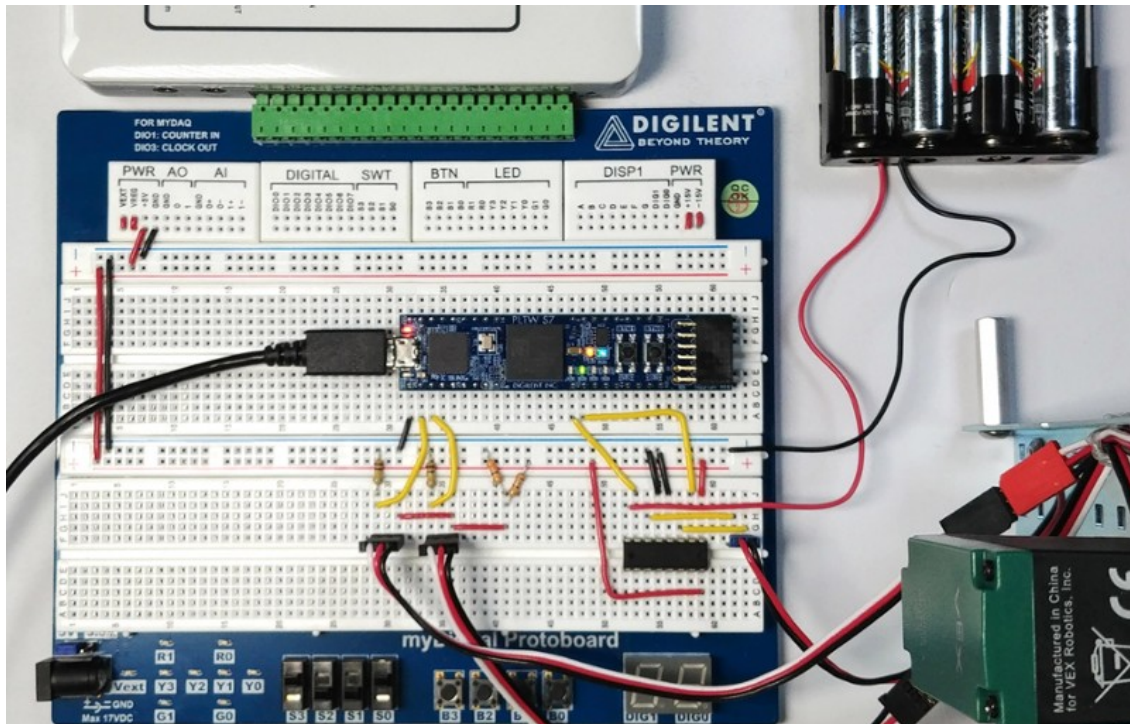
The schematic below illustrates how you can interface the tollbooth test fixture with the Digital Minisystem (DMS). The SN754410 is an H-bridge driver designed to accept standard TTL logic levels and drive DC motors. For pin assignments, refer to the SN754410 Quadruple Half-H Driver data sheet online. The open/close push-buttons and open/close LED indicators are not included in the diagrams below. You will need to determine how to include them in your PLD design and how you want to assign your inputs/outputs to the FPGA. Remember, you will need a clock signal to trigger your flip-flops.

**Note:** A L293 Dual Full-Bridge Driver can be used in place of the SN754410 Quadruple Half-H Driver. (See the DLB solution guide for block diagram and wiring diagram.)



Block Diagram

SN754410 Quadruple Half-H Driver



Physical Wiring

Use the Digital MiniSystem and the Tollbooth test fixture to build and test your state machine design. Verify that the circuit is working as designed. If the circuit is not working properly, review your design work and circuit implementation. Make necessary corrections and retest. **Document all changes in your engineering notebook.**

Suggestion: If you are using the myDAQ and Digital Writer as the clock signal, use DIO0.

---

## CONCLUSION

In your engineering notebook, write a conclusion (minimum 250 words) that describes the process you used to design, simulate, and prototype your tollbooth state machine. This conclusion must include all of your design work, preliminary and final schematics, a parts list, and a digital photograph of your final circuit. The documentation should be complete enough that another student, with that same knowledge of digital electronics, could reproduce your design without additional assistance.

---

[Proceed to next lesson](#)