The University of Toledo

EECS 3100 Embedded Systems

Lab Project #7 Revision Addendum

Existing Text	Revised Text
● Inputs - there are 3 inputs to the system as follows: 1. ***South Sensor****:* This sensor is to be activated (set to logic 1), if one or more cars are near the intersection on the South road. You will interface an on-off binary switch that will generate the activation signal for a simulated sensor. Pressing the switch will simulate sensor activation. 2. ***West Sensor****:* Same as south sensor but for westbound traffic. 3. ***Walk Button:*** Will be pushed by a pedestrian when he or she wishes to cross in any direction. A hardware switch will implement this button functionality. Closing or pressing the switch will simulate the button press effect.	● **INPUTS - 3** 1. ***South Sensor:*** This sensor or button be activated (set to logic 1), if one or more cars are near the intersection on the South road. You will simulate this on the hardware by interfacing a button and holding it down. You will push and hold the button for as long as there are cars on the South road. 2. ***West Sensor:*** This sensor or button be activated (set to logic 1), if one or more cars are near the intersection on the West road. You will simulate this on the hardware by interfacing a button and holding it down. You will push and hold the button for as long as there are cars on the West road. 3. ***Pedestrian sensor:*** Will be activated when a pedestrian is waiting to cross in any direction. You will simulate this effect by pushing a button. You will simulate this on the hardware by interfacing a button and holding it down. You will push and hold the button for as long as there are pedestrians wishing to cross. *This is not a traditional walk button that is pushed and released to request a walk signal.*
● Outputs – you will use 8 outputs from your microcontroller to control the traffic lights. 1. ***(6 LEDs) South & West, R/Y/G Lights***: You will have to interface a total of 6 LED's for the South and West's red, yellow and green lights. 2. ***(1 LED) Walk light:*** This will be the <u>green LED (PF3)</u> on the LaunchPad, and will be turned on when pedestrians are allowed to cross. To request a walk, a pedestrian must push and hold the walk button for at least 2 seconds, after which a person can release the button, and the walk request should be serviced eventually. If the user does not hold down the button for 2 seconds this	● **OUTPUTS** You will use 9 outputs from your microcomputer that control the traffic lights. 1. ***(6 LEDS) South and West, R/Y/G Lights***: You will have to interface a total of 6 total LED's for the South and West's, red, yellow and green lights. To reduce accidents, there should be a short time with all lights red, as it transitions from yellow on one road to green on the other road.

document does not specify what will happen, you may either go to the walking state(s) or not. We leave this decision up to the engineers designing the FSM. Your engineering decisions need to be documented as part of your project report. The walk sequence should be realistic, showing three separate conditions: 1. ***Walk:*** Your walk light should be on signifying the pedestrians may cross. You may decide how long you want to allow pedestrians to cross. The two traffic signals will be red in this state. 2. ***Warning***: Flash your “don't walk” LED signifying that pedestrians need to hurry up. You may decide how long you want to flash it. 3. ***Don’t Walk:*** Your “don't walk” LED should be on and constant. The two traffic signals must change allowing the normal traffic flow. 3. ***(1 LED) Don’t Walk light:*** This will be the red LED (PF1) on the LaunchPad. a. When the “don’t walk” condition flashes (and the two traffic signals are red), pedestrians should hurry up and finish crossing in any direction. b. When the “don’t walk” condition is on steady, pedestrians should not enter the intersection.	2. ***(3-color LED) Walk light:*** This will be the white LED (PF3,2,1) on the LaunchPad, and will be turned white when pedestrians are allowed to cross. The walk sequence should be realistic, showing three separate conditions: 1. ***Walk:*** Your white walk light should be on signifying the pedestrians may cross. The two road signals should be red while pedestrians are walking. 2. ***Warning***: Flash your red “don't walk LED” (PF1) at least three times signifying that pedestrians need to hurry up (e.g., white-red-off-red-off-red-off-red). You may decide how long you want to flash it. The two road signals should be red while pedestrians are hurrying up to cross the street. There should be a short time with all lights red, as it transitions from warning to allowing cars to pass. 3. ***Don’t Walk:*** Your red don't walk LED should be on and constant.
Part E - Testing on Real Hardware Debug your combined hardware/software system on the actual TM4C123 board. Then test the following scenarios, take a picture of the outputs for each. ***Walk Behavior:*** • The green LED is turned on when pedestrians are allowed to cross. • System accounts for pedestrian pressing and releasing button after 2 seconds. • System eventually processes walk request.	**Part E - Testing on Real Hardware** Debug your combined hardware/software system on the actual TM4C123 board. Then test the following scenarios, take a picture of the outputs for each. ***Walk Behavior:*** • The green LED is turned on when pedestrians are allowed to cross. • System accounts for pedestrian pressing and holding button until walk light is turned on. • System eventually processes walk request.

7. *No Accidents:* Do not allow cars to crash into each other. Do not allow pedestrians to walk in one direction while any cars are allowed to go. Engineers do not want people to get hurt. i.e., there should not be only a green or only a yellow LED on one road at the same time there is only a green or only a yellow LED on the other road.	7. **No Accidents:** Do not allow cars to crash into each other or into pedestrians. On the South road, exactly one LED (red yellow or green) should be on at all times. On the West road, exactly one LED (red yellow or green) should be on at all times. When the South road is green or yellow, the West and Walk lights should be red. When the West road is green or yellow, the South and Walk lights should be red. When the walk signal is white or or flashing red, the South and West lights should be red. Engineers do not want people to get hurt. There should be a short time of all red between yellow on one road and green on another. There should be a short time of all red before and after the walk sequence.
9. *"2 Second Walk" Button Timing:* If the pedestrian pushes the walk button for 2 or more seconds, eventually a walk light must occur. If the pedestrian pushed the walk button for less than 2 seconds, it is up to you to decide what happens.	9. **Pedestrian sensor**: If the pedestrian sensor is pushed and held down/on continuously, eventually a walk light must occur regardless if cars are present.
Current text does not entail this requirement explicitly!	*10.* **No Starvation:** When all of the inputs are held active, your FSM should cycle through allowing southbound cars to go, westbound cars to go, and pedestrians to cross the street (in any order).
Tips and Tricks ● When a car sensor is released or deactivated (set to logic 0), it means no cars are waiting to enter the intersection: i.e. when you are not pressing the button no cars are on the road. ● You should exercise common sense when assigning the length of time that the traffic light will spend in each state; so that the system changes at a speed convenient for the TA (stuff changes fast enough so the TA doesn't get bored, but not so fast that the TA can't see what is happening).	**Tips and Tricks:** ● When a car sensor is released or deactivated (set to logic 0), it means no cars are waiting to enter the intersection. i.e., when you are not pressing the button no cars are on the road. ● When a pedestrian sensor is released or deactivated (set to logic 0), it means no people are waiting to enter the intersection. i.e., when you are not pressing the button no people at the intersection. ● You should exercise common sense when assigning the length of time that the traffic light will spend in each state; so that the system changes at a speed convenient for the TA (stuff changes

	fast enough so the TA doesn't get bored, but not so fast that the TA can't see what is happening).
Handling the 2 Second Walk ● One way to handle the 2-second walk button requirement is to have a simplified state graph centered around a "center home" or "check all" state. The idea is that you have one state that you can always come back to that controls what your state graph is doing. Initially when thinking about this problem you can identify that servicing traffic on the east or west road or servicing the pedestrians can be thought of a set sequence of events that have to start and end somewhere. So just make them start and end at the "check all" state. This approach may make sense to some of you. If not, there is another approach mentioned next. ● Another way to handle the 2-second walk button requirement is to add a duplicate set of states. The first set of states means the walk button is not pressed. The second set of states means the walk operation is requested. Go from the first set to the second set whenever the walk is pushed. Go from the second back to the first, whenever the walk condition is output. The two sets of states allow you to remember that a walk has been requested; in this way the request is serviced when appropriate.	This section is removed from the assignment document! Therefore you may simply ignore this requirement in the existing text.