Computation II: embedded system design (5EIB0)

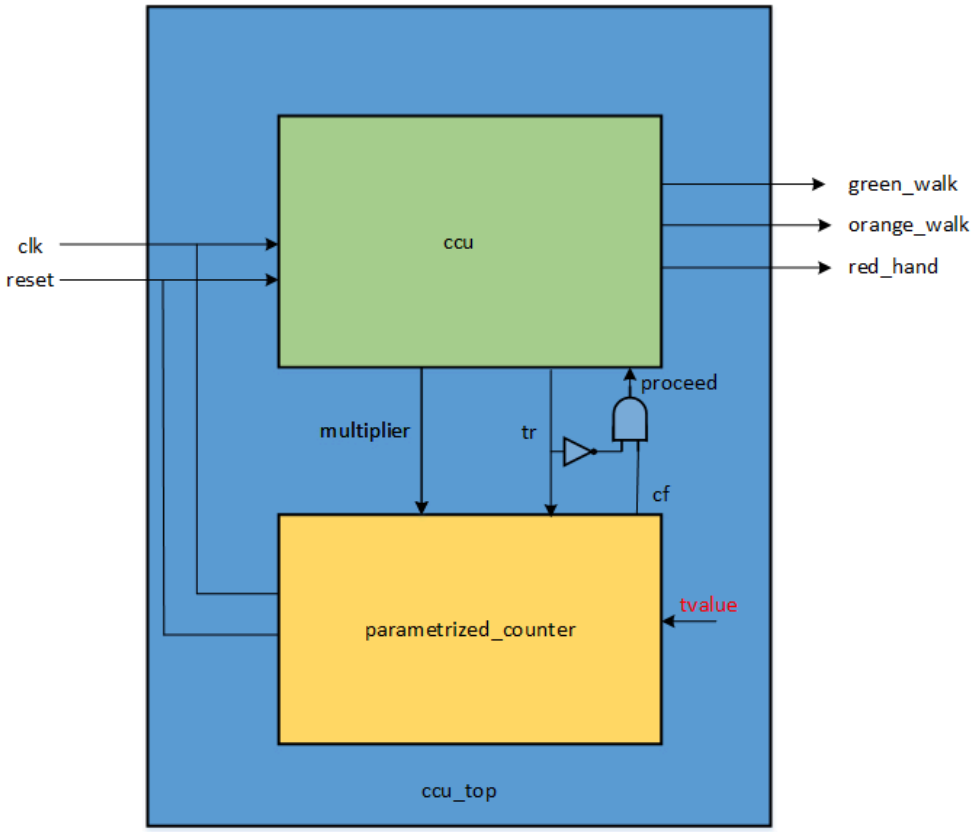
Available from: Friday, 5 April 2024, 2:00 PM
Requested files: ccu.v, ccu_top.v (Download)
Type of work: Individual work

Introduction

This assignment is about the Verilog implementation of a *crosswalk control unit* (ccu):

The behaviour of the ccu repeats as follows. First the walk signal should be green for 9 clocks cycles. Then the walk signal should be orange f 5 clock cycles. Finally the hand signal should be on for 33 clock cycles.

The top block diagram of the crosswalk control unit is shown below:



The block diagram consists of two modules: a finite state machine (FSM) module named **ccu,** and the parametrized counter with adjustable multiplier (**parametrized_counter**) from the previous assignment. A module named **ccu_top** wraps these two modules.

The details of the signals are given below:

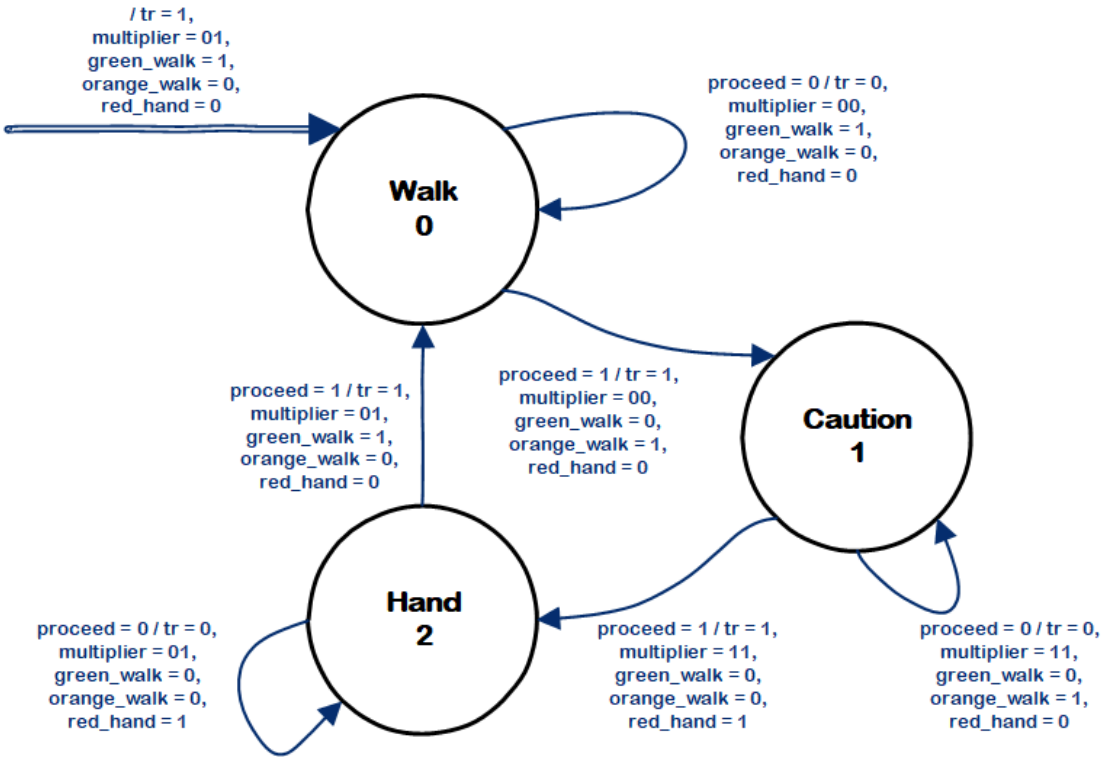
- **clk (1 bit):** Clock (positive edge)
- **reset (1 bit):** Reset (active high, synchronous)
- **walk (1 bit):** Output signal for green *walk* light (active high)
- **hand (1 bit):** Output signal for red *hand* light (active high)
- **tr (1 bit):** Signal to *parametrized counter* to start counting for *tvalue* cycles
- **cf (1 bit):** Signal from *parametrized counter* that indicates *tvalue* cycles have passed
- **proceed (1 bit)**: Signal that the crosswalk control unit should proceed to the next light signal; asserted iff tr is low and cf is high (active high)
- **multiplier (2 bit):** Signal indicating the multiplier of the parametrized counter (little-endian).
  - 00: Run for ***tvalue*** - *1* cycles
  - 01: Run for *2* × ***tvalue*** - *1* cycles
  - 10: Run for 4 × ***tvalue*** - *1* cycles
  - 11: Run for 8 × ***tvalue*** - *1* cycles

Assignment

In this assignment, you have to:

1. Set the **tvalue** parameter in the **ccu_top** module to the correct value of **4**.
2. Assign **proceed** in the **ccu_top** module using the correct combinatorial logic.
3. Instantiate the **ccu** and **parametrized_counter** modules in the **ccu_top** module. (Follow the instructions regarding naming in the provided template!)
4. Implement the FSM module **ccu**.

The Mealy state transition diagram for the **ccu** FSM module is given below:



The state names are indicated in black (Walk, Caution, Hand). Assertion of reset always causes a transition to the **Walk** state, which is indicated by the double-lined arrow into that state. **The outputs of the Mealy machine should be registered on the positive edge of the clock before being output by the ccu module.**

Debug

Click on the symbol marked below to see the waveforms produced by your design. Please note that if your code has an error that prevents it being simulated it will not produce any waveforms.

Requested files

ccu.v

```
module ccu (
    input  clk,
    input  reset,
    input  proceed,
    output green_walk,
    output orange_walk,
    output red_hand,
    output [1:0] multiplier,
    output tr
);

// Add your FSM implementation here...

endmodule

```

ccu_top.v

```
/* Top-level module for crosswalk control unit */
module ccu_top (
    input  clk,
    input  reset,
    output green_walk,
    output orange_walk,
    output red_hand
);

// Number of cycles for each signal
localparam tvalue = ...; // TODO: replace this with the correct value

/* Use these wires for the signals between ccu and parametrized counter.
Do not change the names of these signals, as the testbench will
inspect them. */
wire tr;
wire cf;
wire [1:0] multiplier;
wire proceed;

/* TODO: Assign proceed to be high iff cf is high and tr is low. */
assign proceed = ...;

/* TODO: Instantiate parametrized_counter module (provided for you).
Make sure the instance is called parametrized_counter_inst. */

/* TODO: Instantiate FSM module (which you have to implement in ccu.v).
Make sure the instance is called ccu_inst. */

endmodule

```

VPL