UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

TIMER

DSD FIRST YEAR PROJECT

Mara Mureșan

31.05.2023 | PROJECT SUPERVISOR: ING. MIHAI TIMAR

Table of Contents

1. Specifications

Design a timer with the next functionality: the device has 4 displays BCD – 7 segments. The first two displays are for the minutes, the next two show the seconds, such that the maximum value that can be shown is 99 minutes and 59 seconds.

The device has 3 buttons: M (from Minutes), S (from Seconds) and START/STOP.

Suppose that initially the device is in State Zero, if the START/STOP button is pressed, the timer starts to count in increasing order. If the START/STOP button is pressed again, the timer stops at the value reached in that moment. If the START/STOP button is pressed once again, the timer continues to count. If it reaches 99 minutes and 59 seconds, follows State Zero again. If M and S are pressed simultaneously, the timer resets (becomes State Zero).

In each state, if the M button is pressed, it will increment and show the value of the minutes. In each state, if the S button is pressed it will increment and show the value of the seconds. Once a value is set for minutes and/or seconds (by pressing the M and S buttons), when the START/STOP button is pressed, the timer starts to count in descending order from the current value "Minutes/Seconds" until State Zero and then when it reaches State Zero, an audio signal (alarm) is emitted for a period of time previously settled.

It is considered that there exists a periodic signal with the frequency of 1 Hz.

The project will be done by one student.

The project is made using **VHDL** language in **Logisim Evolution** app.

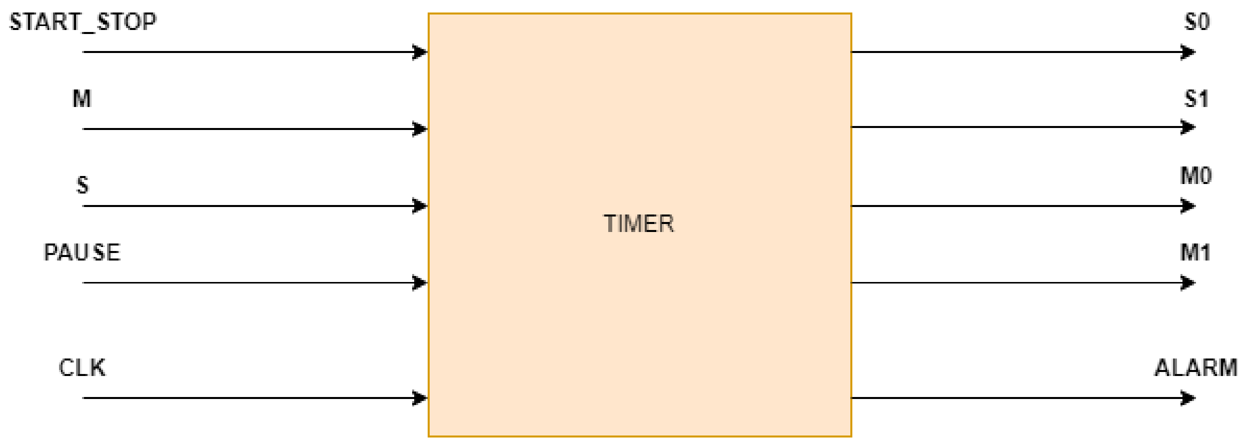
2. Design

2.1 Extra Functionalities

For easier manipulation of the device and for a clear understanding of the processes through which the timer goes, a new input (button PAUSE) was added in order to stop the timer at a certain value (this button replaces the START_STOP button in the case of putting a pause in the process of counting in increasing order).

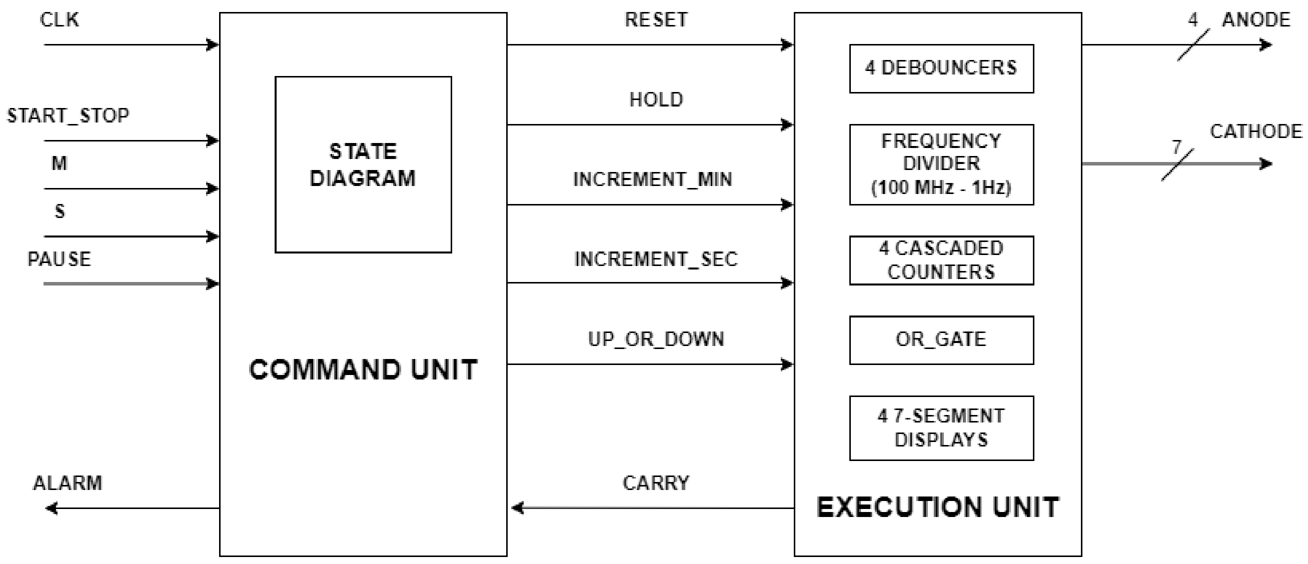
Also, for an easier implementation of the device, the audio signal (the ALARM) is replaced by a LED which stays on until the START_STOP button is pressed.

2.2 Black Box



2.3 Control and Execution Unit

2.3.1 Mapping the Inputs and the Outputs of the Black Box on the Two Components

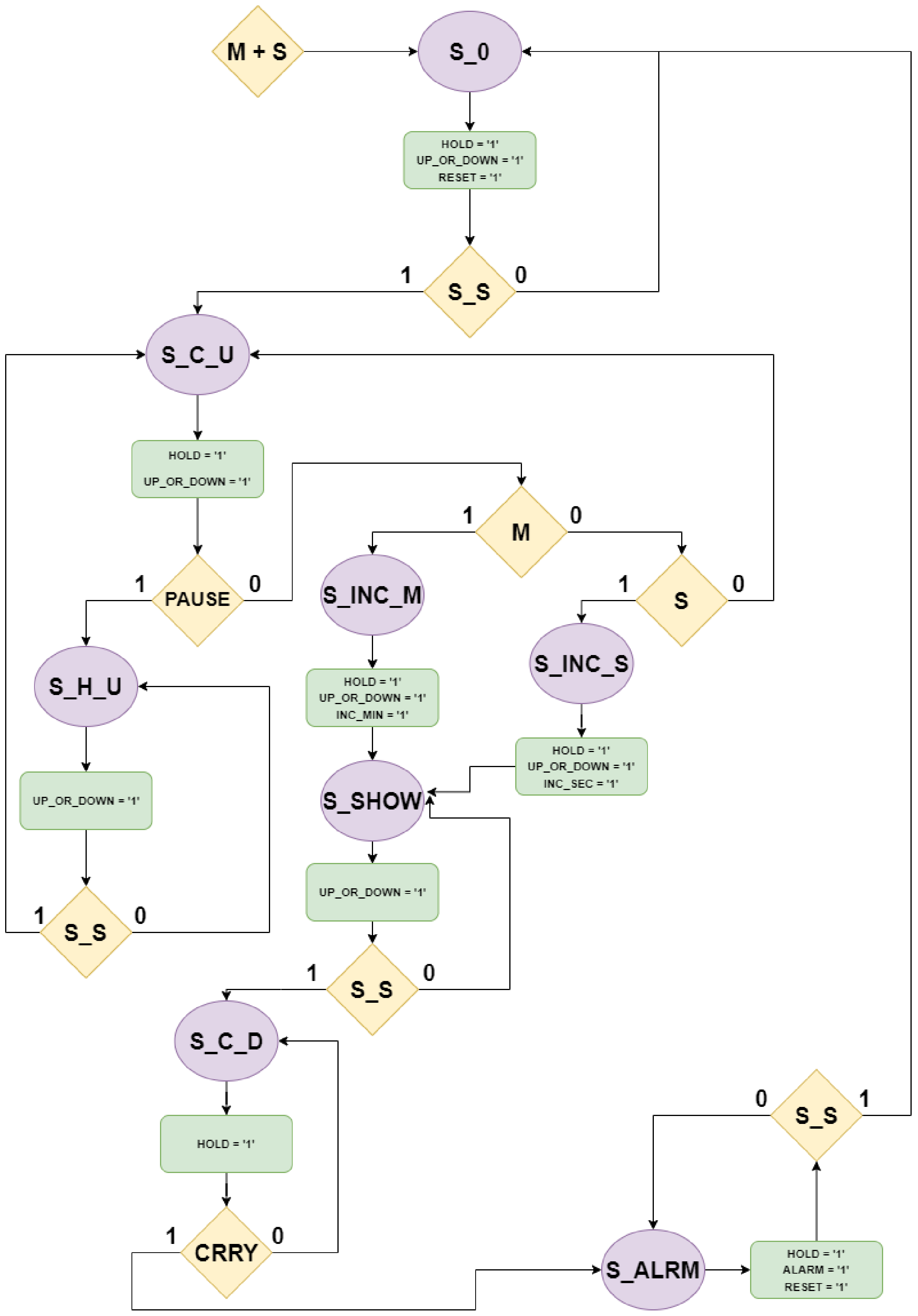


We can divide both inputs and outputs into 2 categories: **data** and **control**. This separation is essential at the beginning.

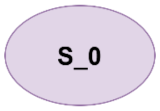
Data inputs: values for different things (CLK pulse = 1 Hz)
Control inputs: the buttons (START_STOP, M, S, PAUSE).

Data outputs: values to be displayed to the user (the time displayed using the anodes and cathodes)
Control outputs: warning or warning signals to the user, through which we can control and guide the user through the operation of the system (the ALARM = a LED).

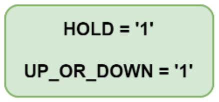
2.3.2 State Diagram for the Control Unit



The state diagram represents the control part, the decision part of any algorithm, and it can then be implemented directly in VHDL.

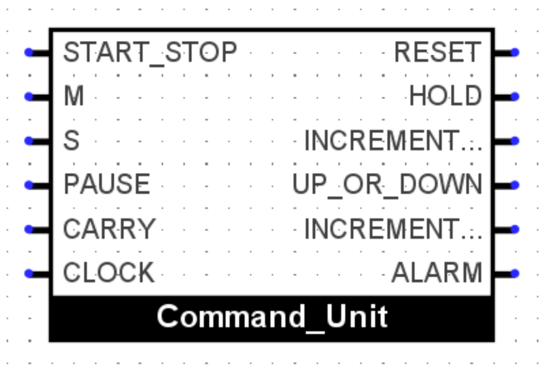
A state represents a moment of time (a period). **States** are represented by 

The **decisions** made in each state are represented by

The **outputs** generated in each state are represented by 

. Inside the rectangle are the outputs that are true at that time.

To implement the State Diagram, the method of behavioral description with 2 processed was used.



- **Inputs**:
  - the buttons: START_STOP, M, S and PAUSE
  - CARRY which will take the value of the counter's carry (CARRY will be '1' when the counters finish counting in decreasing order)
  - The CLOCK pulse from the FPGA board (with the frequency equal to 100 MHz)
- **Outputs**:
  - RESET: resets the counters with the value "0000"
  - HOLD: enables or disables the counters
  - INCREMENT_MIN: increases the value of the minutes with 1
  - UP_OR_DOWN: sets the mode of the counting (if '1', counts up; if '0', counts down)
  - INCREMENT_SEC: increases the value of the seconds with 1
  - ALARM: activates a LED when counting in decreasing order reaches the value "0000"

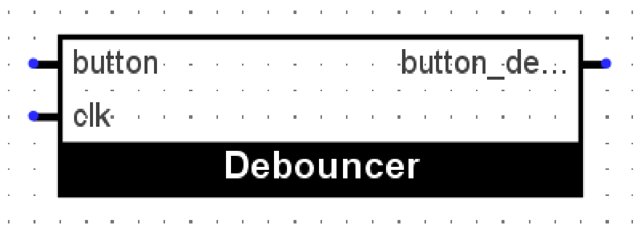
2.3.3 Resources. Breakdown of the Execution Unit

In order to further establish the links between the CU and the EU, we must first identify the resources on the basis of which we make decisions. These resources can generate signals to the control unit or can be controlled by it via ENABLE or RESET signals.

Any decision-making information must come from a resource that generates that information and passes it on to CU.

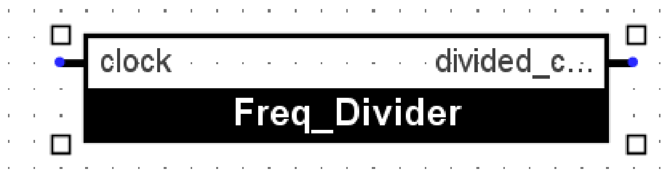
2.3.3.1 Debouncer

Verifies if the buttons were pressed and transmits the responses via **button_debounced**.



2.3.3.2 Frequency Divider

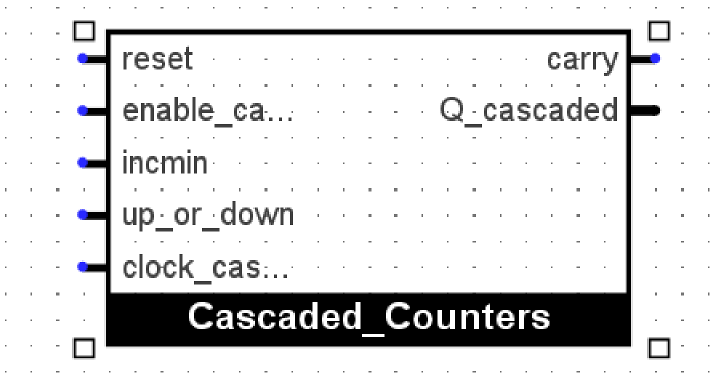
Divides the frequency of the FPGA board (100 MHz) with the proportion of 50%, resulting a clock pulse of 1 Hz (100 million smaller than the board's clock pulse). This component is used to divide the clock that generates the cascaded counters.



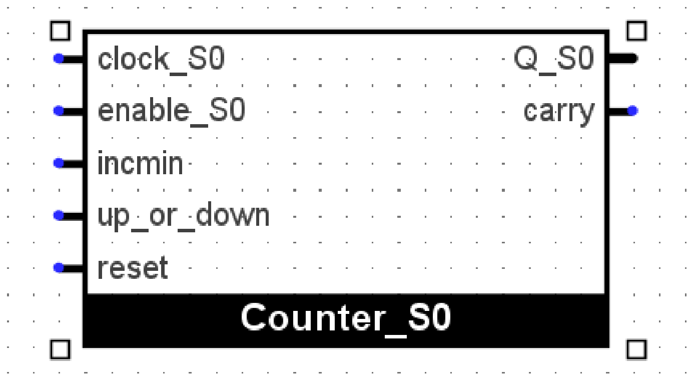
2.3.3.3 Cascaded Counters

This component was made by cascading 4 reversible 4-bit counters. The implementation in VHDL language was done using 4 **port map()** instructions. The first counter (S0) receives the main clock pulse. For the next 3 counters (S1, M0, M1), the **carry** generated by the previous counter becomes the **clock** pulse for the next counter. The last carry (generated by M1) is linked to the CARRY input from the CU in order to show when the counting in decreasing order is finished.

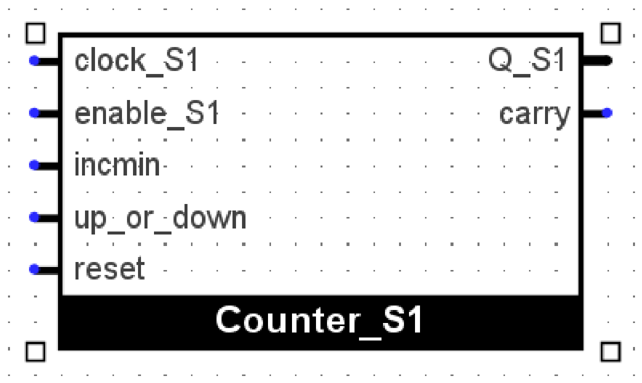
- **reset** = resets the counters value with "0000"
- **enable_cascaded** = enables ('1') or disables ('0') the counters
- **incmin** = used only for M0; increases the value of the minutes with 1, on the next clock pulse
- **up_or_down** = determines the mode of the counting ('1' = count up; '0' = count down)
- **clock cascaded** = clock pulse
- **carry** = carry out
- **Q_cascaded** = the time composed by 4 numbers (in total there are 16 bits)



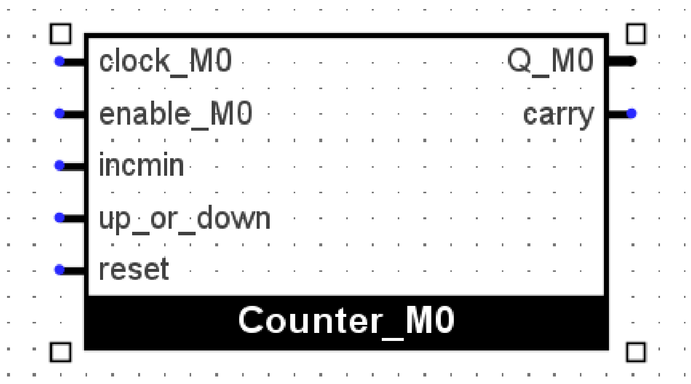
- **Counter_S0**: reversible counter modulo 10 (0-9) on 4 bits (the unit digit of the seconds)



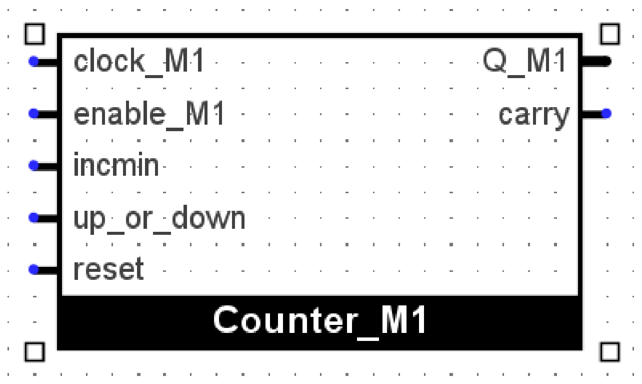
- **Counter_S1**: reversible counter modulo 6 (0-5) on 4 bits (the tens digit of the seconds)



- **Counter_M0**: reversible counter modulo 10 (0-9) on 4 bits (the unit digit of the minutes)

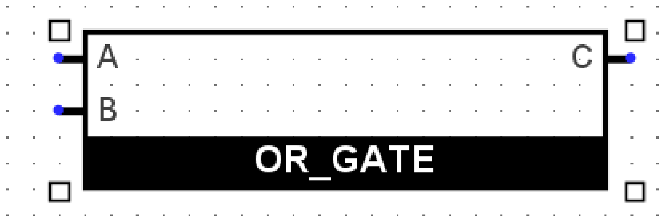


- **Counter_M1**: reversible counter modulo 10 (0-9) on 4 bits (the tens digit of the minutes)



2.3.3.4 OR gate

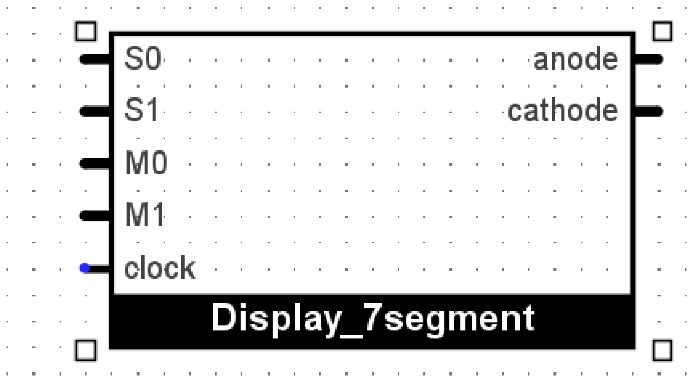
Links the divided **clock** to the **INCREMENT_SEC** output of the CU in order to increase the value of the second with 1 by adding an extra clock pulse.



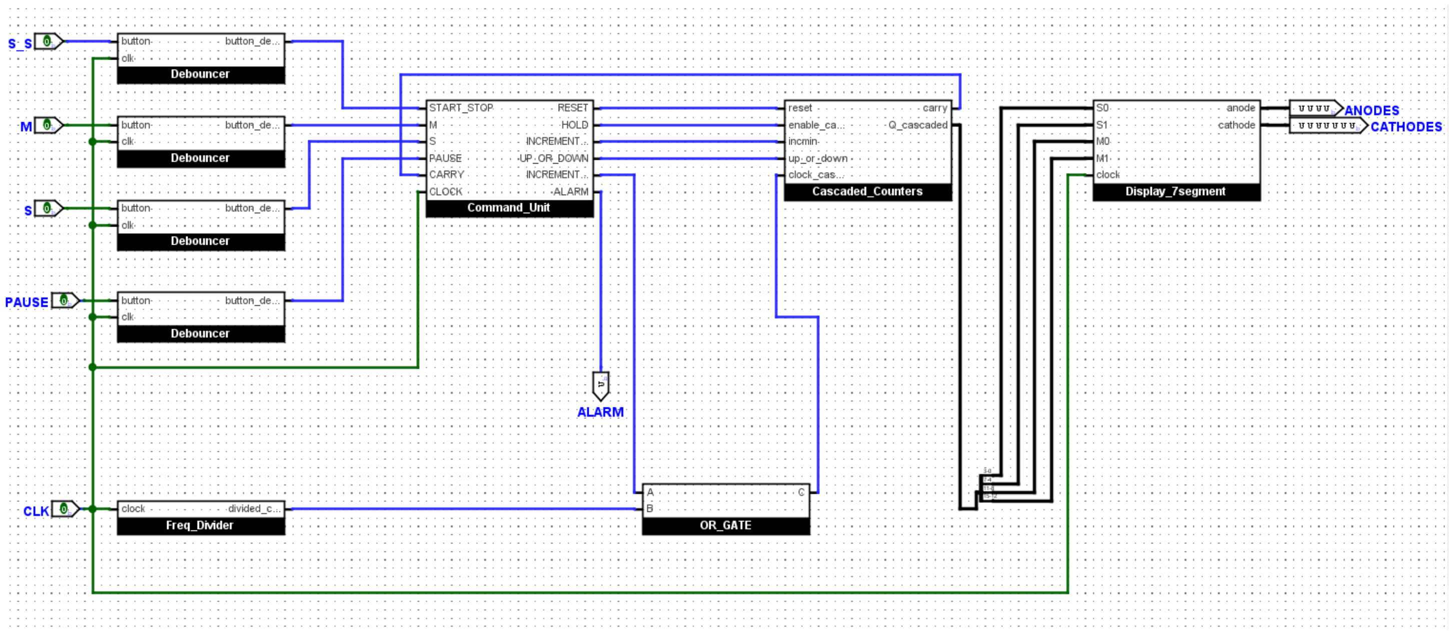
2.3.3.5 7-Segment Display

This component is used to make possible the show of the time on 4 7-Segment Displays which are on the FPGA board.

- Inputs: the numbers generated by the counters (S0, S1, M0, M1) and the undivided clock
- Outputs: the anodes and cathodes on the FPGA board

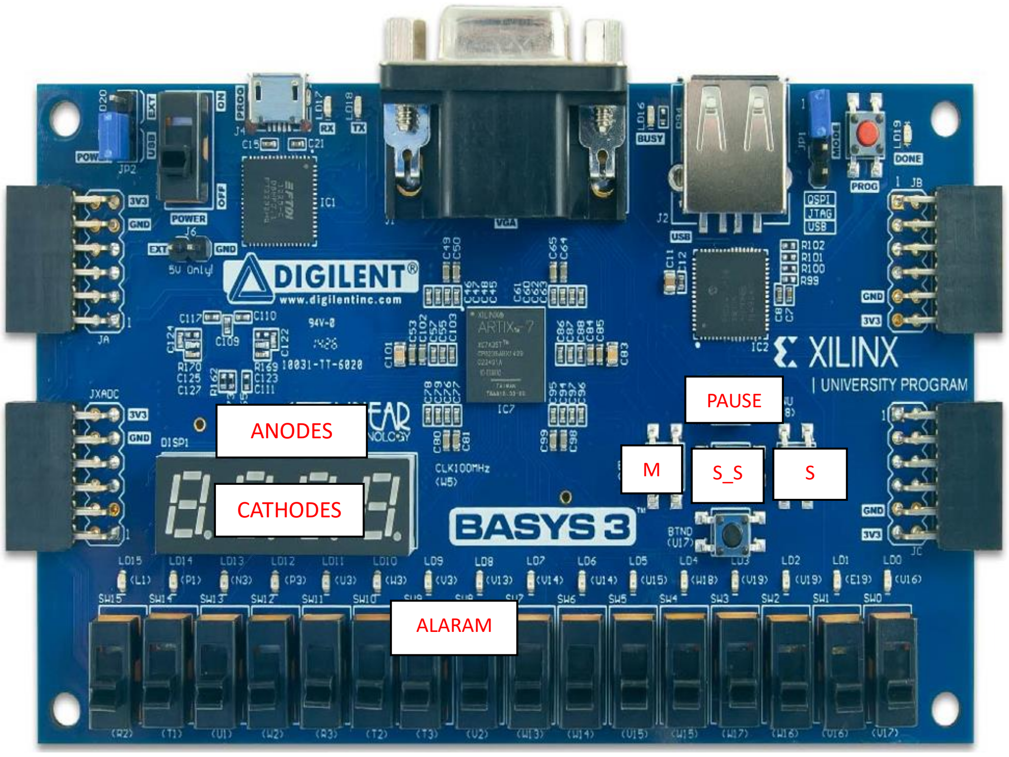


2.3.4 Detailed Diagram of the Project



3. User Manual

This timer was tested using the Basys 3 FPGA board and the user inputs (the buttons) and the outputs are shown in the picture below:



4. Technical Justifications for the Design

I chose to cascade the 4 counters using VHDL code rather than linking them with wires in the Logisim Evolution app because it is a more compact approach and it is easier to have the whole code in only one component.

I also added the PAUSE button in order to solve the problem of an infinite loop in the State Diagram that I observed during implementation.

5. Future Developments

In the future I could develop the project by adding additional features such as a new button which could change the way in which the counting is made (counting from 2 to 2 or from 5 to 5) or adding a new LED signal which will light on when the timer reaches an even number.

6. References

- The Digital System Design Lab Works and Courses