|  |
| --- |
|  |
| DSD Project: Apartment thermostat |
|  |

|  |
| --- |
| Student: George Cătălin Robotin  Coordinator: Diana Irena Pop |

Apartment thermostat

Description:

My project consists in designing a thermostat, for apartment use. It should be implemented on an FPGA board, using VHDL. As for its functionality, it should allow the user to:

* See the time (hours and minutes), as well as the current temperature
* Set the global minimum and maximum values for the temperature
* Set minimum and maximum values for any specific hour of the day
* Have an extra simulation mode, where the temperature should increase or decrease by 1 degree every 3 seconds
* Reset all values, using a button
* Additionally, it signals each confirmation through an led

The temperature sensor is considered to already exist.

Contents

[Chapter 1. Design 2](#_Toc104235466)

[1.1 The Black Box 2](#_Toc104235467)

[1.2 Control and execution unit 3](#_Toc104235468)

[1.2.1. Mapping the inputs and outputs to the Control and Execution Units 4](#_Toc104235469)

[1.2.2. Establishing the resources 4](#_Toc104235470)

[Resources: 5](#_Toc104235471)

[1.2.3. Block diagram for first breakdown 8](#_Toc104235472)

[1.3 The State Diagram for the Control Unit 9](#_Toc104235473)

[1.4 Detailed diagram 10](#_Toc104235474)

[Chapter2. Justification for chosen solution 11](#_Toc104235475)

[Chapter 3. User Manual 12](#_Toc104235476)

[Chapter 4. Future developments 14](#_Toc104235477)

# Chapter 1. Design

# The Black Box

A screenshot of a computer

Description automatically generated with low confidence

The Black Box shows an overview of all the inputs and outputs, visible for the end user, which correspond to the operations that can be executed using the thermostat. The “dir” input is used for the simulation mode, “clk” as we have a synchronous device, whilst “set\_min\_max” and “set\_specific” are used for the global and detailed setting of the temperature, respectively. The outputs are 8 seven segment displays, each with seven cathodes. The leftmost 6 can not be changed during normal execution, and the last 2 are used for visualization during data input. “Led\_heat” lights up when the setting of a value is confirmed, and “led\_rst” turns on when the reset button is pressed.

# Control and execution unit

A screenshot of a computer

Description automatically generated with low confidence

Execution unit

Control unit

The design is split into 2 main components, namely the Execution and Control Units. The Control Unit takes care of all the logic that needs to be taken into account for the system to work properly, sending signals to the Execution Unit, which contains all the components needed for each command to run properly.

# 1.2.1. Mapping the inputs and outputs to the Control and Execution Units

A picture containing diagram

Description automatically generated

E.U.

C.U.

All inputs and outputs are driven by the Control Unit, which dictates the behaviour of the system based on signals which it gives to and receives from the Execution Unit.

# 1.2.2. Establishing the resources

The resources are components from the Execution Unit which ensure that the system is working, exchanging signals with the Control Unit. For my project, they differ in complexity, from usual multiplexers to the simulation component, which deals with all the signals needed to perform the settings.

# Resources:

1. Seven segment display

Graphical user interface

Description automatically generated with medium confidenceThe component takes as input 8 4-bit signals, corresponding to each of the 8 displays on the board. Each display (anode) consists of 7 segments (cathodes), used to show the necessary information. The logic to do it requires other components, which are part of this one, namely:

* 1. A clock divider

A screenshot of a computer

Description automatically generated with low confidenceIt is used to make the displayed information readable for the human eye, splitting the 100MHz of the board’s clock in such a way that every display is visible.

* 1. 8:1 Multiplexers

A screenshot of a computer

Description automatically generated with low confidenceIt is used to select the display on which each digit goes. It has a width of 8 bits, each of which represents one of the displays.

There is also another 4 bit wide 8:1 multiplexer, which decides the input number, or the number to be converted, for the next

resource, the decoder.

* 1. A picture containing diagram

     Description automatically generatedDecoder

The decoder takes the 4 bit digit and activates the corresponding anodes of the seven segment display, in order to show the correct symbol,

1. Random number generator

A screenshot of a computer

Description automatically generated with medium confidenceThe generator takes a 6 bit input, using the most significant one like a serial load and performs logic computations in order to generate a number in the range between 0 and 39, which acts as the temperature we would have from the sensor.

1. A screenshot of a computer

   Description automatically generated with medium confidenceA 24 hour digital clock

It is a more complex component, which includes a 1 second clock divider (shortened to verify functionality). It also combines 2 modulo 60 and 1 modulo 24 counter, for seconds, minutes and hours respectively. The current hour can only be input from the program.

1. The user input component

A screenshot of a computer

Description automatically generated with medium confidenceThis is the most complex resource of the project, dealing with both the simulation mode, controlled by the “dir” input, as well as the minimum and maximum temperature settings, through “set\_min\_max” and also the specific time setting, “set\_specific”. It outputs on the 2 remaining displays, which the user can control. Most of its logic is described behaviourally, but it still has another 2 distinct subcomponents:

* 1. The debouncer

A screenshot of a computer

Description automatically generated with low confidenceIt is used to delay the signal of a button, namely the reset button, in order not to produce misinputs.

* 1. A picture containing diagram

     Description automatically generatedThe 3 second clock divider

Frequency divider with the factor 300 million, in order to have a period of 3 seconds.

# 1.2.3. Block diagram for first breakdown

Diagram

Description automatically generated

Diagram

Description automatically generated

# 1.3 The State Diagram for the Control Unit

# 1.4 Detailed diagram

A screenshot of a computer

Description automatically generated with low confidence

Each of the 4 components are made up of more smaller resources, as I showed in the Resources chapter. For example, the ssd is made up of the frequency divider, the 2 multiplexers and the decoder.

A screenshot of a computer

Description automatically generated with low confidence

# Chapter2. Justification for chosen solution

I’ve written my code in such a way that it is as modular as possible, so that future developments are made easily, and its functionality is easy to understand by looking at the top module. Also, my implementation takes into account the ease of access for end users, as all counters are automatic and you only need to press a confirmation switch. As far as the small components themselves, they are described mostly behaviourally, making the code easier to follow.

# Chapter 3. User Manual

A picture containing text, electronics, circuit

Description automatically generated

1. 2. 3. 4. 5. 6. 7. 8. 9. 10. 11. 12.

Display

Led Reset

Led Store

Button Reset

Switch ON/OFF

Switches:

1. Enable simulation mode
2. Sim up
3. Sim down
4. Enable global minimum and maximum set
5. Max set
6. Min set
7. Confirm selection (for global values)
8. Enable specific setting for temperature
9. Hour select
10. Minimum set for selected hour
11. Maximum set for selected hour
12. Confirm selection (for specific hours)

Utilization:

* Power switch must be turned to “on”
* First 4 displays on the left show the time, with hours and minutes
* The next 2 show the current temperature
* When no other setting is done, the last 2 will show 0’s

Simulation mode:

* Flick on switch 1 to enable the simulation mode. It will display the temperature on the rightmost 2 displays. To simulate the temperature going down, flick on switch 2 (while keeping the first one on). To sim up, flick on switch 3. It will show the temperature changing by 1 degree every 3 seconds. If the operation is finished, turn all the used switches back to their original “off” position

Global setting mode:

* Turn on switch 4 to enable the setting. For the minimum temperature, flick on switch 5; for the maximum, use switch 6. In both cases, the temperature to be set will start at the value of 18 degrees and automatically go up or down slowly, signaling the temperature to be set. Once the value is satisfactory, flick on switch 7, which confirms the current setting. Upon pressing it, the Store Led will turn on, signaling the fact that the value has been memorized. Again, when the operation is finished, turn all the switches back to their original position.

Specific hour setting mode:

* Turn on switch 8 to start the counter for the hour. It loops in the interval 0 to 23, corresponding to each hour. When you reached the desired time, flick on switch 9. This will set the display back to 0. To set the minimum for that hour, turn the hour select switch back to off and turn on switch 10. For maximum, use switch 11. When the desired temperature is reached, turn on switch 12 to confirm the selection. Once again, the confirmation led will turn on. Upon finishing the operation, turn all switches back to their original position.

Reset button:

* It can be pressed at any time. When it is activated, all the memorized values are reset. The reset led will light up, and the message “S0” will be shown on the 2 displays, meaning all the saved values are set to 0.

Important!

Only 1 mode should be used at one specific moment (either sim, global set or specific set). The reset button deletes the values saved in all modes.

# Chapter 4. Future developments

In the future, I would like to split the simulation component into 3 smaller ones, corresponding to each functioning mode. Besides that, I would like to have a separate component for memorizing each value, not needing to use an array of signals, so that they could be read and written in a more centralized way.

# Bibliography:

<https://github.com/ninadwaingankar/Digital-Time-clock/blob/master/watch.vhd>

(used as template for the digital clock component)