Thermostat

Table of Contents

1. [Specifications 1](#_Toc136426129)
2. [Design 1](#_Toc136426130)
   1. [Black box 1](#_Toc136426131)
   2. [Control and Execution Unit 1](#_Toc136426133)
   3. [Mapping the inputs and outputs of the black box 2](#_Toc136426134)
   4. [Resources 2](#_Toc136426135)
3. [State diagram 7](#_Toc136426136)
4. [Detailed diagram of the project 8](#_Toc136426137)
5. [User manual 8](#_Toc136426138)
6. [Future Developments 9](#_Toc136426139)

# **Specifications**

Design a thermostat for an apartment heating system. There is a clock to display the time (hour, minutes) and a temperature display. A minimum and maximum value for the apartment temperature can be programmed. These values can be associated for each hour (out of the 24 hours of the day). Depending on the programming, the thermostat must command the heating to turn on or off. Additionally, the thermostat will receive data from the simulation unit, which will simulate heating by 1 ° for every 3 seconds with the heating element on and the same for cooling.

# **Design**

# Black box

# O imagine care conține text, captură de ecran, proiectare Descriere generată automat

# Control and Execution Unit

The system's black box must be further broken down in order to find implementable components. We will do a **top-down** breakdown of the problem until we get to known circuits, and then we will implement **bottom-up**.

The first breakdown of any system is one in which we will differentiate between the **control logic** in the system and the **system resources**. The control logic is represented by the Control Unit (CU) and the resources are represented by the Execution Unit (EU). Any algorithm can be broken down in this way.

O imagine care conține captură de ecran, text, diagramă, Dreptunghi

Descriere generată automat

# Mapping the inputs and 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 (ticket cost, distance traveled, pin card number, number of tickets, etc.).
* **Control inputs**: confirmation button, program selection button, cancel button, etc.
* **Data outputs**: values to be displayed to the user (time remaining in a program, rest due, calculated cost of a ticket, etc.)
* **Control outputs**: warning or warning signals to the user, through which we can control and guide the user through the operation of the system (LEDs, audible signal).

# Resources

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** and can be **controlled by the CU** via Enable or Reset signals.

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

Resources can be **simple circuits**, which can be implemented directly (counter, register, etc.) or **complex resources** (remainder algorithm, multiplication algorithm, etc.). These complex resources may appear in the first breakdown with black boxes to which we must establish inputs and outputs, but later they must be further broken down (usually also in CU and EU) until we reach known circuits.

**RESOURCES:**

1. **Clock Divider**

We use it to transmit the counter a clock signal of one second.

Nexys4 clock frequency: 100Mhz= Hz

Remark: 1 sec = 1Hz / =(aprox.) 1Hz

O imagine care conține text, captură de ecran, software, număr

Descriere generată automat

1. **Digital clock (00-23)**

It has the 1s clock input, a reset that brings the clock to 00:00 and 4 outputs on 4 bits, for the tens digit and the units digit for the hour and for the minutes.

O imagine care conține text, diagramă, captură de ecran, Font

Descriere generată automat

For this clock we will use 2 types of counters:

1. Counter (0-9), which can be implemented with JK Flip-Flops

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Qn |  |  |  | QN+1 |  |  |  |  |  |  |  |  |  |  |  |
| Q3 | Q2 | Q1 | Q0 | Q3 | Q2 | Q1 | Q0 | J3 | K3 | J2 | K2 | J1 | K1 | J0 | K0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | X | 0 | X | 0 | X | 1 | X |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | X | 0 | X | 1 | X | X | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | X | 0 | X | X | 0 | 1 | X |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | X | 1 | X | X | 1 | X | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | X | X | 0 | 0 | X | 1 | X |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | X | X | 0 | 1 | X | X | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | X | X | 0 | X | 0 | 1 | X |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | X | X | 1 | X | 1 | X | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | X | 0 | X | 0 | 0 | X | 1 | X |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | X | 1 | X | 0 | 0 | X | X | 1 |

O imagine care conține text, diagramă, hartă, captură de ecran

Descriere generată automat

1. Counter (0-5)  
   O imagine care conține text, captură de ecran, afișaj, diagramă

   Descriere generată automat

O imagine care conține text, captură de ecran, diagramă, Plan

Descriere generată automatThe digital clock made of the two counters:

1. **“TransfOra\_AdrRam”**: takes as inputs the outputs of the digital clock for the hour and transforms into a 5-bit value which will then be transmitted to the RAM as an address.

O imagine care conține text, diagramă, captură de ecran, Font

Descriere generată automat

Pseudocode:

ORA\_unit // 4-bit signal for the unit’s digit

ORA\_zeci //4-bit signal for the tens digit

ORA\_5BIT //signal that contains the hour on 5 bits

If ORA\_zeci = “0000” then

If ORA\_unit = “0000” then

ORA\_5BIT <= “00000”

if ORA\_unit = “0001” then

ORA\_5BIT <= “00001”

………………………………….

If ORA\_zeci = “0001” then

………………………………….

1. **RAM MEMORY**

It has 32 rows from which we only use 24 (one for each hour), each row has 10 bits. First 5 bits hold the minimum temperature, and the next 5 bits hold the maximum temperature.

O imagine care conține text, diagramă, captură de ecran, Font

Descriere generată automat

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Hour/Address 00 | tmin | tmin | …… |  |  | tmax | tmax | ……. |  |  |
| Hour /Address 01 |  |  |  |  |  |  |  |  |  |  |
| Hour /Address 02 |  |  |  |  |  |  |  |  |  |  |
| Hour /Address 03 |  |  |  |  |  |  |  |  |  |  |

………………………………………………………………………………………………

1. **Simulation Unit**

It has as inputs the clock signal, reset and the output from the thermostat which tells the simulation unit to increase or decrease the temperature. It gets an initial value for the temperature of 5 degrees. It also contains a 3 second counter for when the temperature should change. The outputs are the temperature after being heated or cooled and the led that announces if the heating is on or off.

1. **Thermostat**

O imagine care conține text, diagramă, Plan, captură de ecran

Descriere generată automatIt has as input the output from the RAM which contains the min and max values for the temperature. Its purpose is to compare the current value of the temperature, which it gets as an input from the simulation unit, with the min and max value set for the temperature. It sends the simulation unit the signal that makes it increase or decrease the temperature.

1. **Seven Segment Display**

This component decodes the 4-bit signals that it gets from the inputs, transforming them into a 7-bit signal that can be shown on the displays on the FPGA. It also commands the successive lighting of the anodes in order to simulate the continuous lighting of all anodes.

O imagine care conține Color, Dreptunghi, captură de ecran, Paralel

Descriere generată automat

# **State Diagram**

The state diagram is **not a flow chart**, but represents the control part, the decision part of any algorithm, and it can then be implemented directly in VHDL if done correctly.

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

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

O imagine care conține captură de ecran, cerc, Grafică, proiectare

Descriere generată automatThe **outputs** generated in each state are represented by . Inside the rectangle are the outputs that are true at that time.

# **DeO imagine care conține text, diagramă, Plan, schematic Descriere generată automattailed diagram of the project**

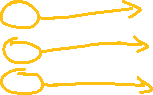
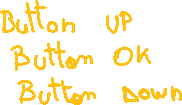
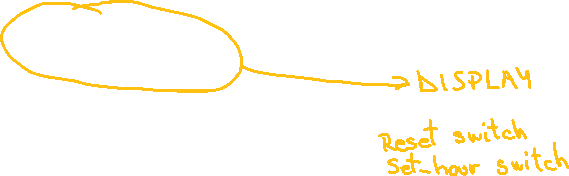
# **User manual**

After plugging in the FPGA board, the user must switch the reset to 0 and then to 1 for the thermostat to start functioning.

If the “set\_hour” switch is set to 0, then, on the first four digits of the display will appear the current hour and minutes that start from 00:00 after the reset. On the next 2 digits of the display will appear the current temperature that starts from 5 after the reset and oscillates between the minimum and maximum temperature.

If the user sets the “set\_hour” switch to 1, he will be able to change the minimum and maximum values for each hour. At first, he can increase the hour (shown on the first two digits of the display) using the up button or decrease it using the down button. After he gets to the hour that he wants to modify the values for, he presses the ok button and now he is able to change the minimum value for the temperature (shown on the digits 3 and 4 on the display) in the same way he changed the hour. After pressing ok again, he can change the maximum value for the temperature (shown on the digits 5 and 6 on the display). After switching the “set\_hour” back to 0 we will observe that the current temperature oscillates between the new values.

# **O imagine care conține electronice, Componentă electronică, Componenta circuitului, Inginerie electronică Descriere generată automatFuture Developments** In the future, there are several potential developments for a thermostat implemented on an FPGA (Field-Programmable Gate Array) board. Here are a few possibilities:



1. Advanced Control Algorithms: FPGA-based thermostats can leverage the power and flexibility of FPGAs to implement more advanced control algorithms. These algorithms can incorporate machine learning techniques to adaptively learn and optimize temperature control based on user preferences, environmental conditions, and energy efficiency.
2. Integration with Smart Home Systems: Future thermostats implemented on FPGAs can be designed to seamlessly integrate with smart home systems. This integration allows users to control their thermostats remotely through smartphone apps or voice assistants, receive real-time energy consumption data, and synchronize temperature settings with other smart devices in their homes.
3. Environmental Sensing and Adaptive Control: Future thermostats can integrate additional sensors, such as humidity, occupancy, and ambient light sensors, to provide a better understanding of the environment. This information can be used to dynamically adjust temperature settings and optimize comfort levels based on real-time conditions.
4. Customizable User Interfaces: FPGAs allow for flexible user interface designs. Thermostats can incorporate customizable graphical interfaces on touchscreens for intuitive and immersive user experiences. Users can personalize their thermostat interfaces.
5. Enhanced Security: As smart home devices become more prevalent, security is of critical importance. FPGA-based thermostats can implement robust security measures, such as hardware-based encryption, secure boot, and authentication protocols, to ensure data privacy and protect against unauthorized access or tampering.

These are just a few potential developments for a thermostat implemented on an FPGA board. The flexibility and programmability of FPGAs open up a wide range of possibilities for innovation in the field of thermostat technology.