

**THERMOSTAT**

**Faculty:** Facultatea de Automatica si Calculatoare, CTI engleza, an I, grupa 30414

**Student:** Radu Cosmina

**Laboratory Professor:** Cret Octavian

**Content:**

1.Specification…………………………………………3

1.1Requirement………………………………………3

1.2Analisys of the requirement……………… .……...3

2.Black box……………………………………………..4

2.1State diagram……………………………….……..5

2.2Component blocks………………………….……..6

3.Main components description…………...…………..7

3.1.Program unit ……………………….….…………7

3.2.Clock unit ……………………………………….10

3.3.Functioning unit………………………………... 13

3.4.Display unit …………………………………….16

3.5Command unit ………………………………….17

3.5. Frequency divide ………………………………18

3.6. Debouncer ……………………………………...18

4.The meaning of the notations and the interface with

the outside ……………………………………………....19

4.1 Inputs:……………………………………………. 19

4.2.Outputs: …………………………………………..21

5.Justification of the chosen solution………………….. 22

6.Instructions for use and maintenance ………………...22

7. Possibilities for further development ………………..23

**1.Specification**

**1.1. Requirement**

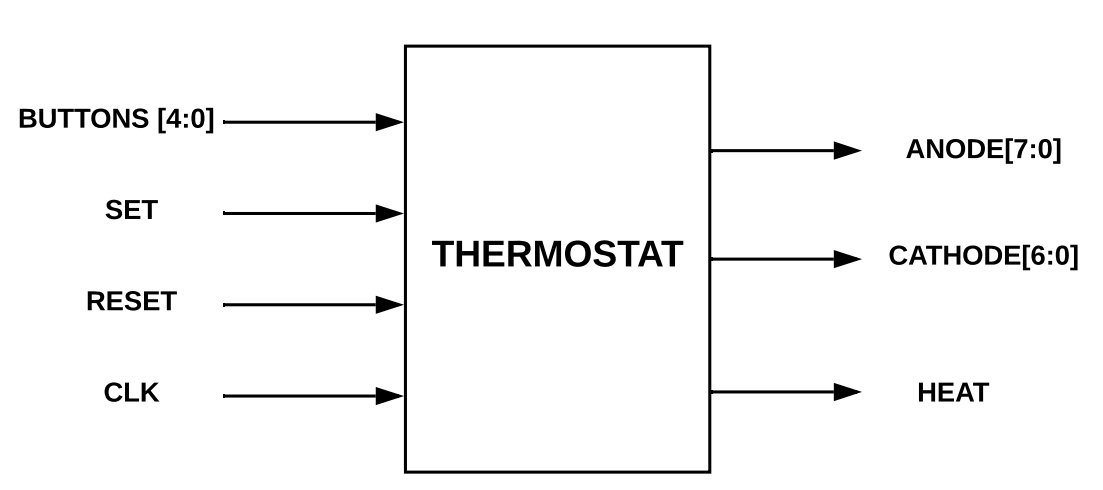
Sa se proiecteze un **termostat pentru o centrala termica de apartament**. Termostatul este prevazut cu un termisor pentru masurarea temperaturii. (Exista un ceas pentru afisarea timpului) si un afisaj pentru temperatura. Se pot programa o valoare minima si una maxima de temperatura a partamentului. Aceste valori pot fi apoi asociate pentru fiecare ora ( din cele 24 de ore ale zilei ). In functie de programare, termostatul trebuie sa comande pornirea sau oprirea incalzirii. Aditional termisorul va primi date de la o unitate de simulare, care va simula incalzirea cu 1 grad pentru fiecare 3 secunde cu elementul de incalzire pornit si similar pentru racire.

* 1. **Analysis of the requirement**

To start, we divide the machine into 2 modules:

1. The Programming module: here, the user can make all the necessary changes for the thermostat’s functioning. Even if the user doesn’t make any changes in the settings, the thermostat will still work by the default settings. The user can make any changes concerning the interval of temperatures by switching to the “setting mode”;
2. The Functioning module: here, the thermostat will work by the settings done by the user or the default ones but without the user’s direct implication. It’s functioning will be visible when the “setting mode” is off, the previous settings working on the current time shown by the clock;

**2.Black Box**



**Inputs:**

Buttons: the 5 buttons on the FPGA board, each having its own utilization

Set: the “setting mode” switch

Reset: the resetting switch

Clk: the clock from the FPGA board

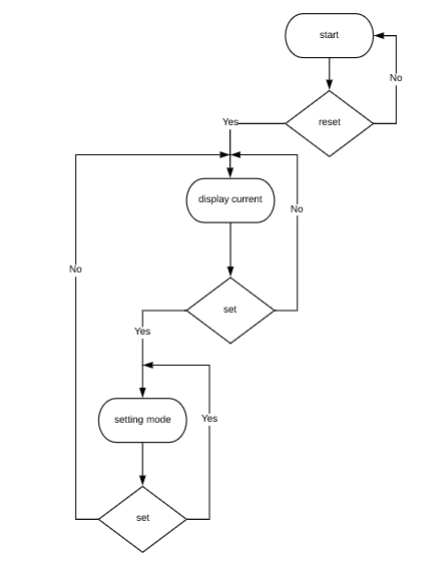
**Outputs:**

Anode[7:0]: the display

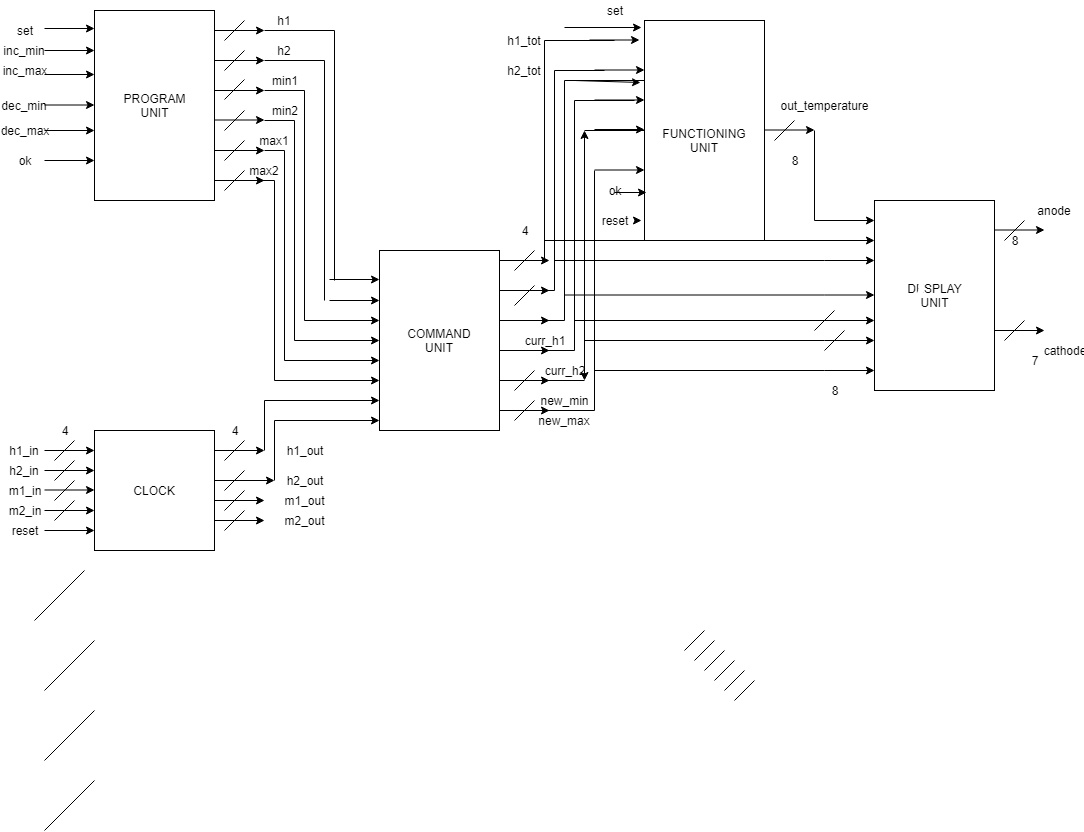
Cathode[6:0]: BCD 7 segments

Heat: the heat signal

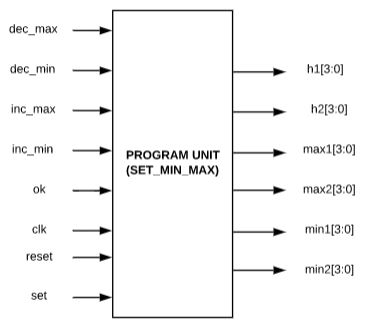
**2.1 State diagram**

****

**2.2 Component blocks**

****

1. **Main components description**

** 3.1.Program unit:** using the buttons (dec\_max, dec\_min, inc\_max, inc\_min, ok) and the switches (set, reset), the user can set the range of minimum and maximum temperature for each hour. To set a specific range of temperatures for a specific hour, the user must set the “set” switch to 1. By pressing the “ok” button , the minimum and maximum temperatures will be stored in a RAM (one for the minimum, one for the maximum) memory for the selected hour, so when it becomes the current hour, the thermostat will work within the set range. Also , besides storing the values in the memory, by pressing the “ok” button , the shown hour will increment so the user can set a range of temperatures for every hour in the day. This way, the values that are introduced will be stored in the memory at the address that the hour indicates.

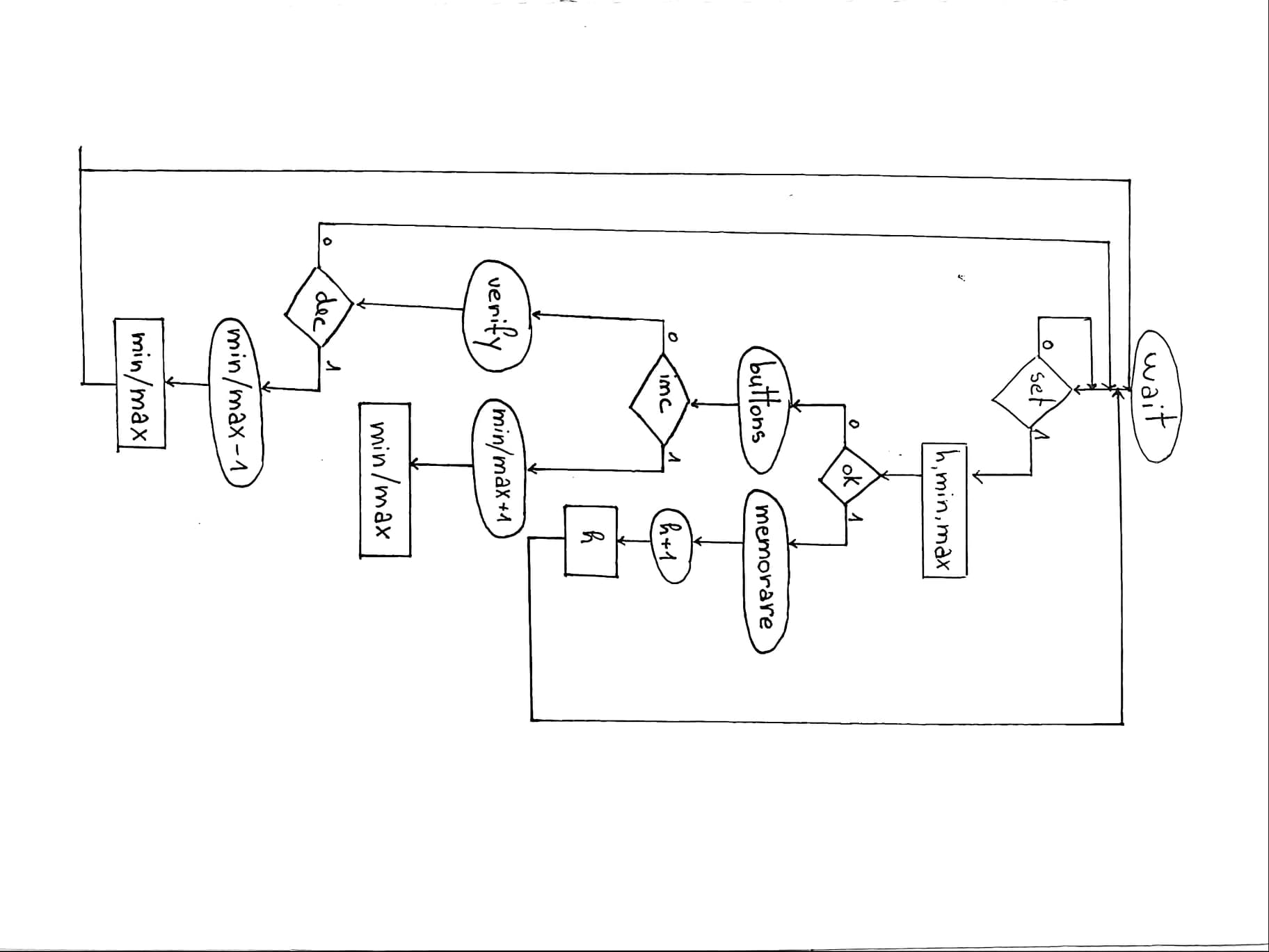
Capture.PNG

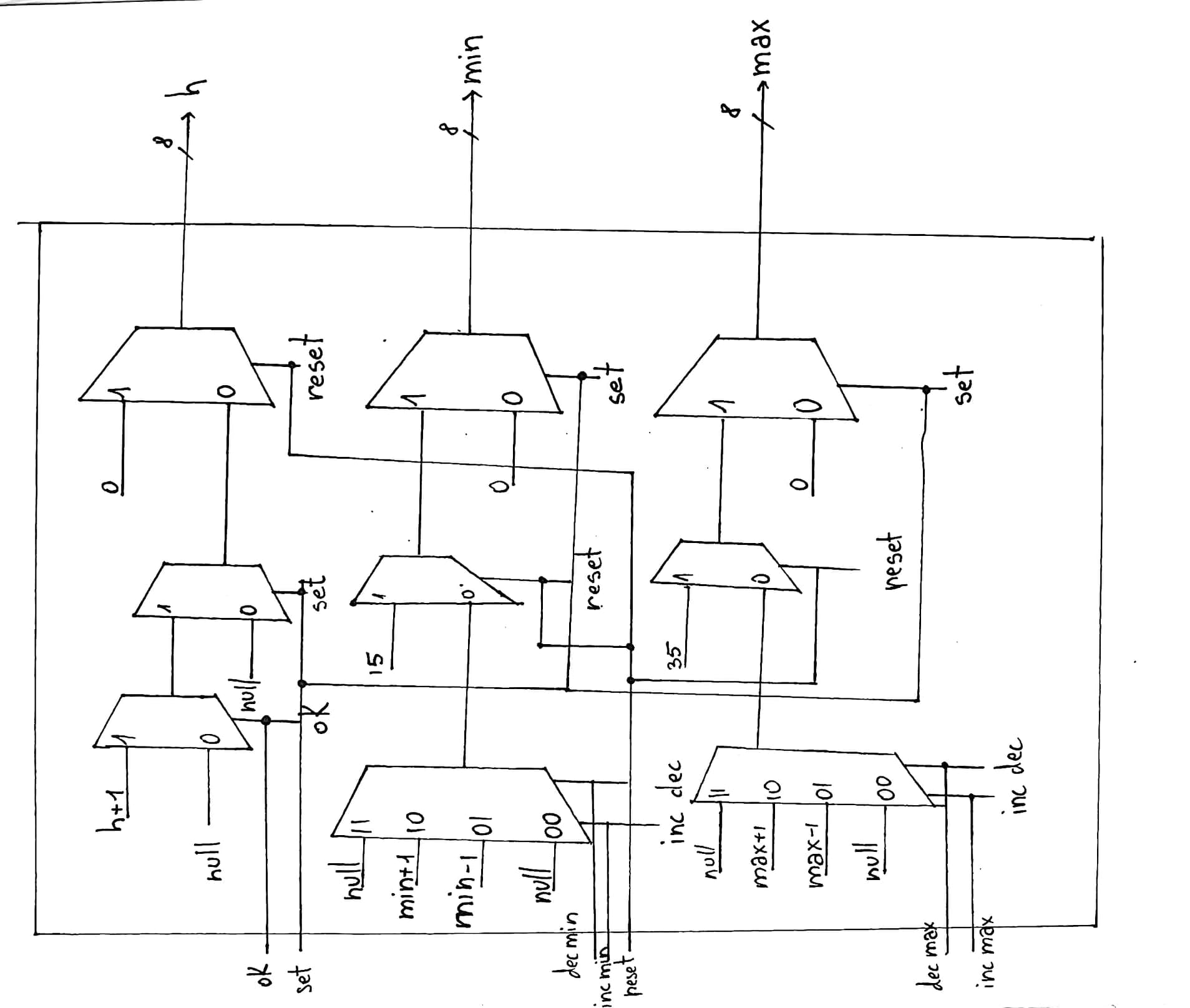
the hour for which the minimum the maximum

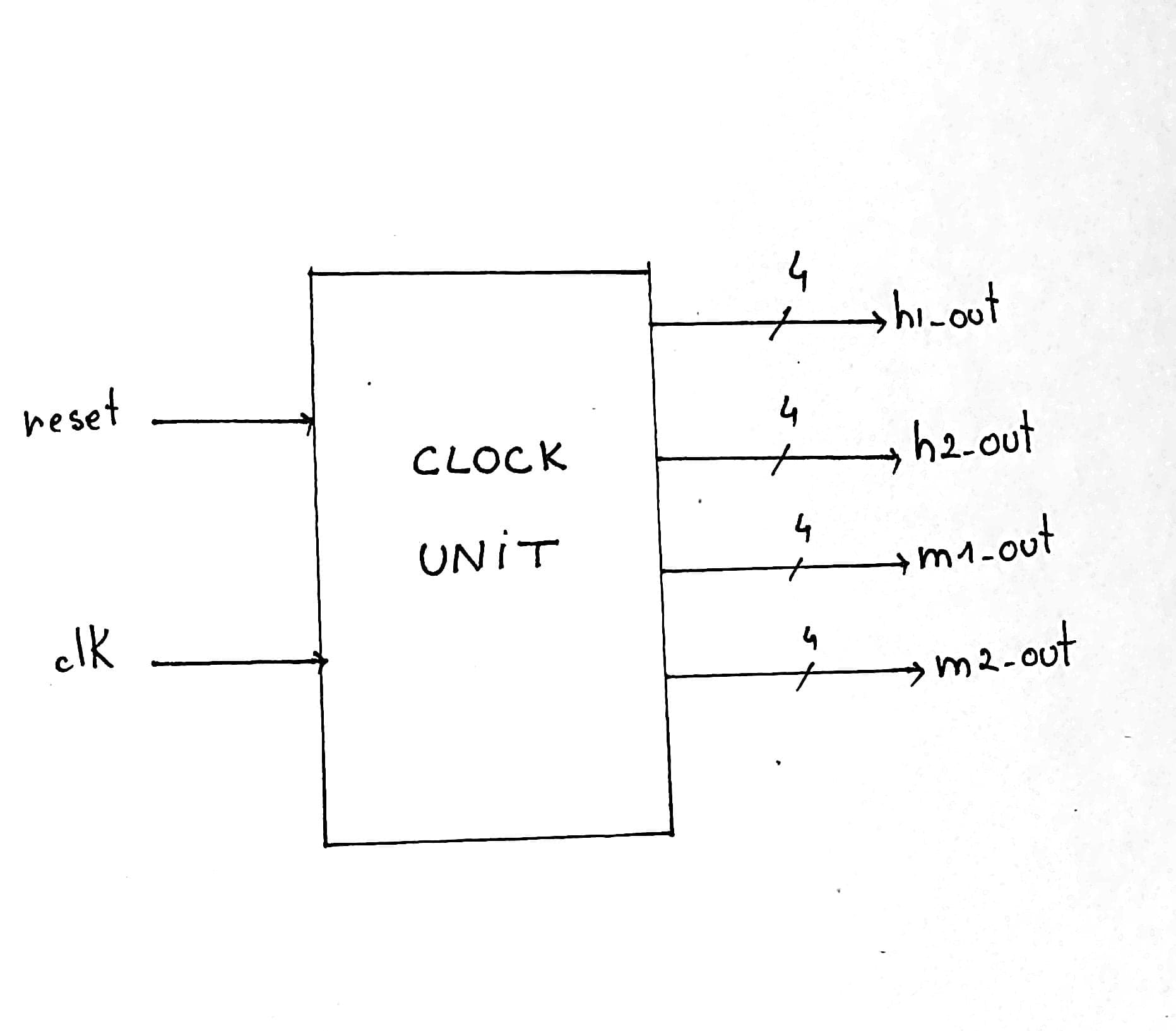
we set the minimum set temperature set temperature

and maximum

temperature





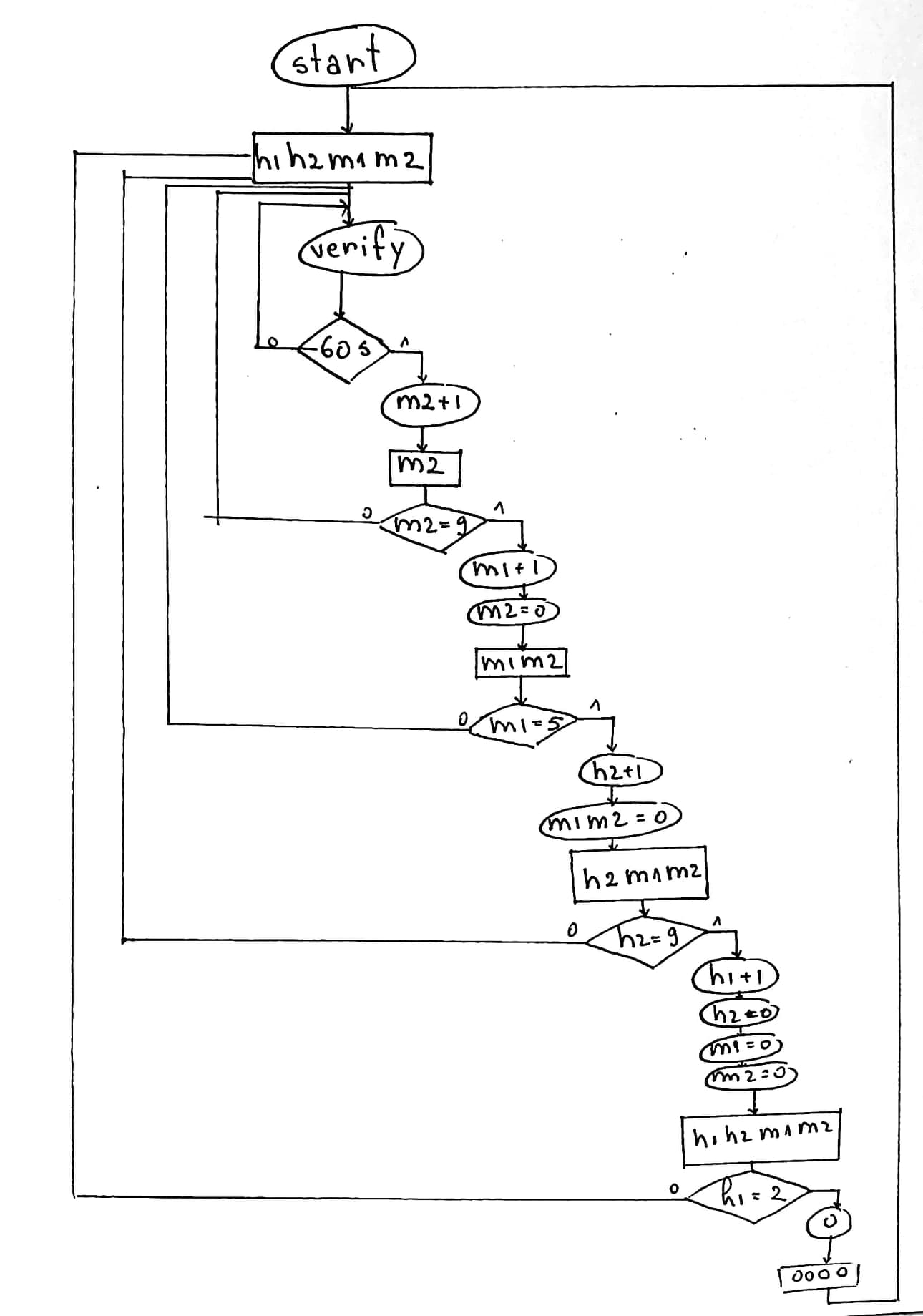
** 3.2.Clock unit:** when the “set” switch is off, in the display will be shown the current time (ex. 15:47) and the current temperature. The current time is automatically set from the code and it will work properly. When the “reset” signal becomes 1, the time will come back to its initial state. The “clk” signal comes from the clock of the FPGA board.

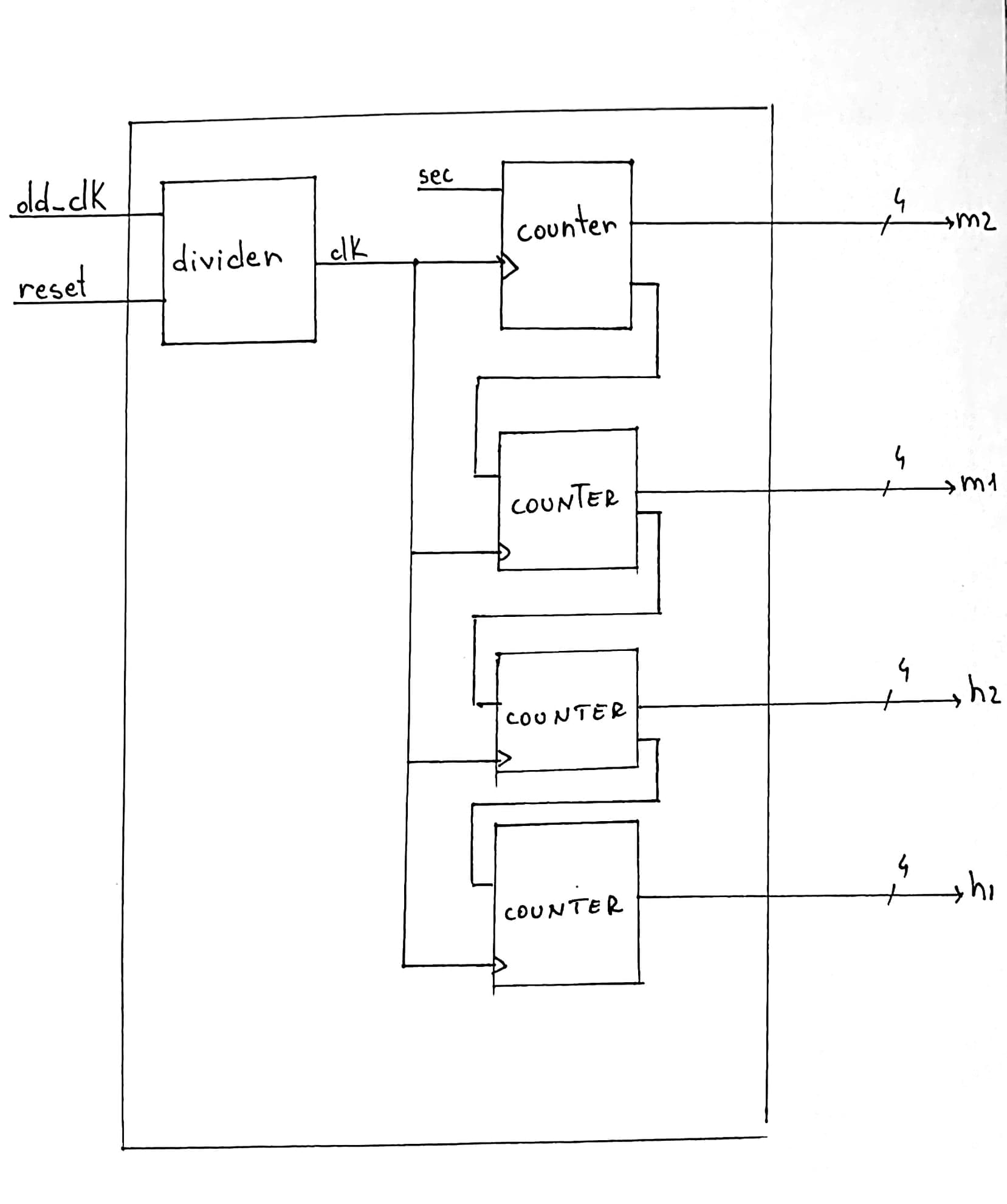
This unit is done by cascading 4 integer counters (h1, h2, m1, m2). They work for the units and tens of the hour and minutes. Two of them go from 0 to 9 (h2, m2) , one of them counts from 0 to 5( m2 ) an the other one from 0 to 2 ( h1 ).

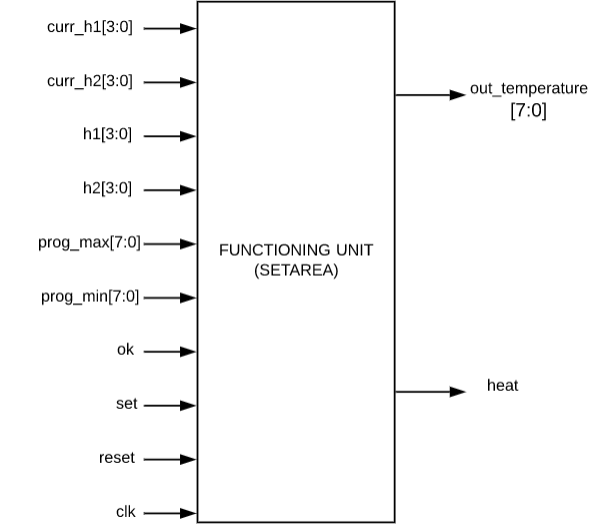
Capture.PNG

the current time the current

temperature



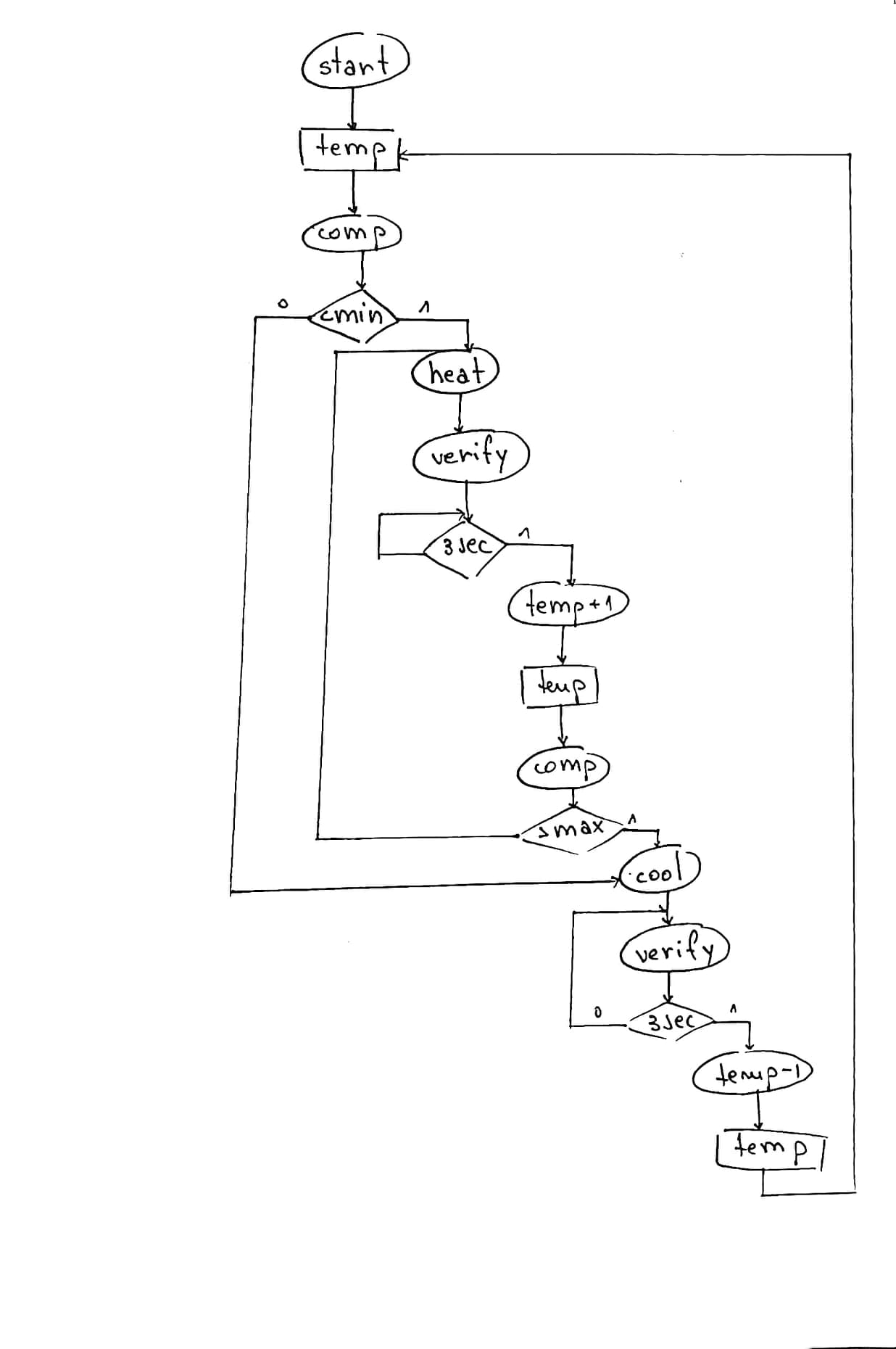


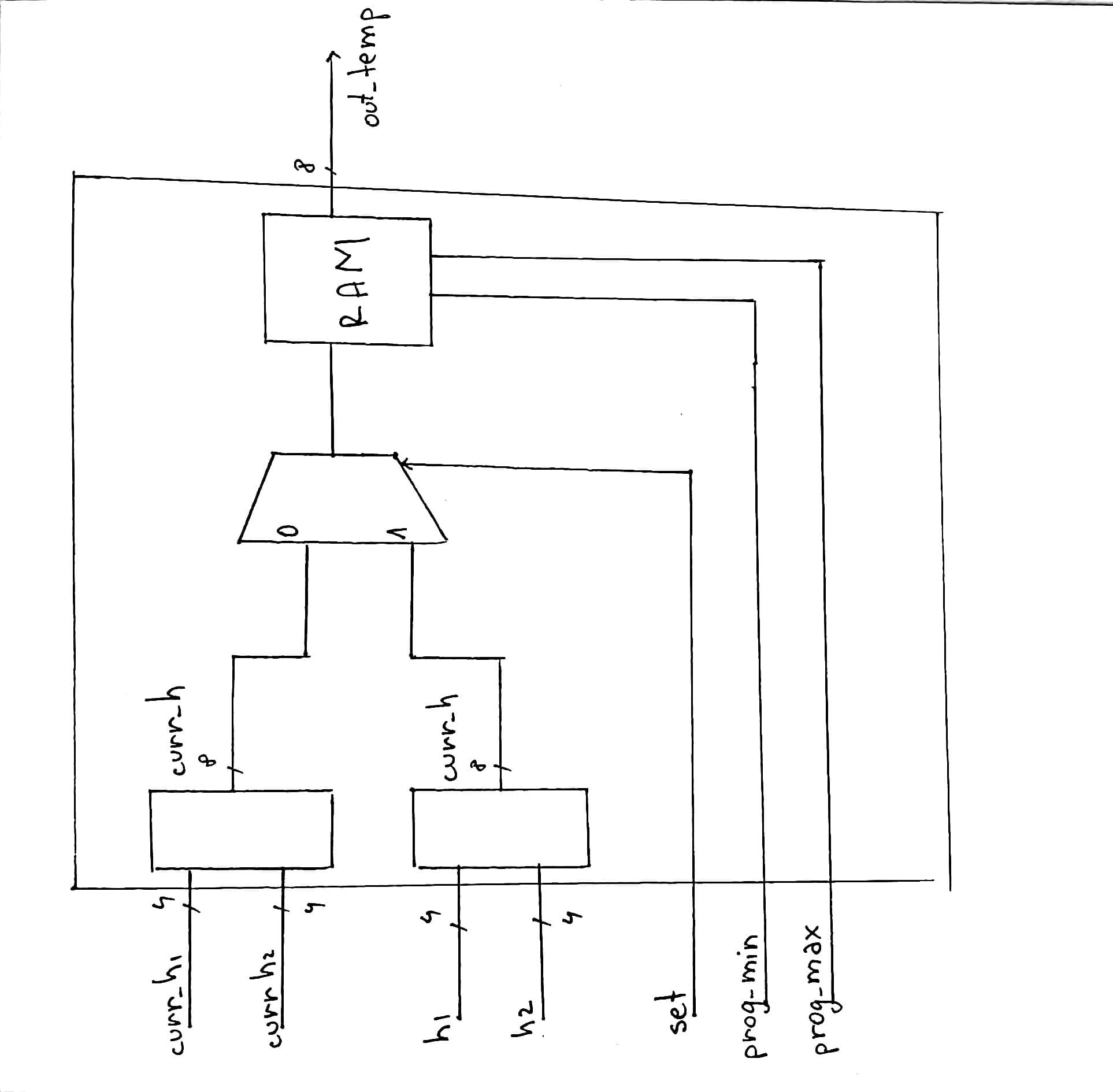
****

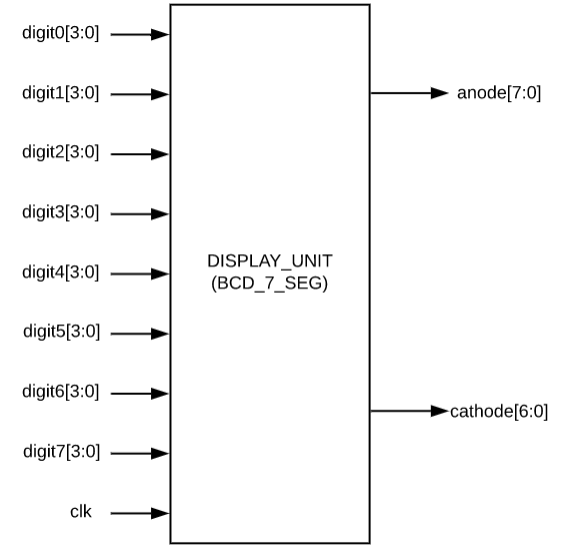
**3.3.Functioning unit:** this unit is where the set temperatures are stored: by pressing the “ok” button while the “set” switch is set on 1, the temperatures that the users chose ( prog\_min[7:0], prog\_max[7:0] ) will be stored in the memory, having as address the hour ( h1[3:0], h2[3:0] ) for which we set the temperatures. This also contains the simulationunit: while “set” switch is set on 0, the temperature showed on display ( out\_temperature[7:0] ) will increase or decrease, depending on the set temperatures

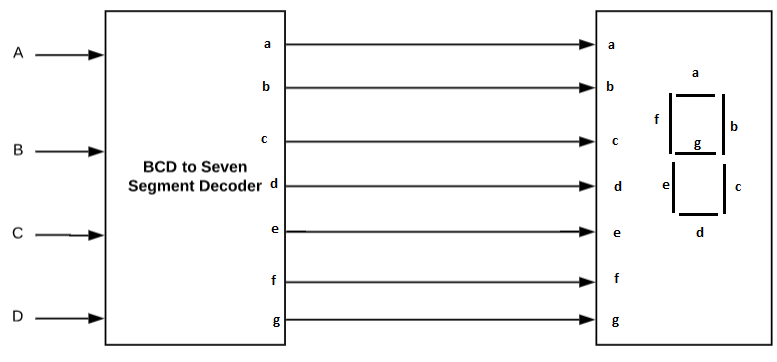
for the respective hours, stored in the memory. This command will be given by the signal “heat” which is also connected to a led : this signal will be 1 and the led on when the temperatures need to increase, and the signal will be 0 and the led off when the temperature will need to decrease. Every 3 seconds, the temperature increases or decreases, depending on the settings done for the programmed temperatures and on the current time ( curr\_h1[7:0] , curr\_h2[7:0] ) .

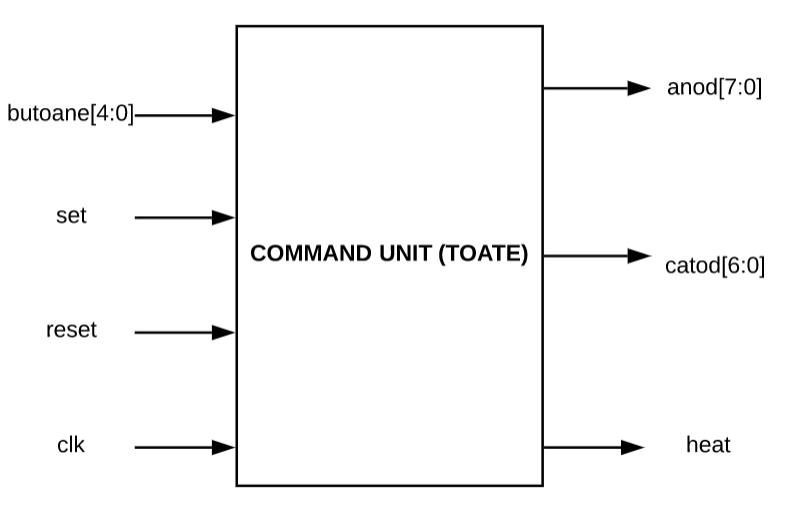
If the “reset” switch is set on 0, then the set temperatures will become the default ones ( 15 – 35 ) and what was before written in the memory will be erased.





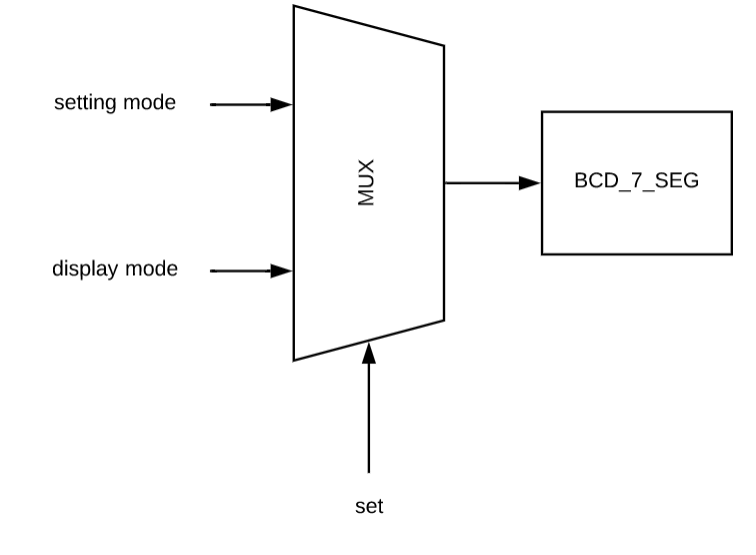
**3.4.Display unit**: in BCD encoding scheme , each of the decimal numbers (0-9) is represented by its equivalent binary pattern (on 4 bits). In Seven Segment display is an electronic device which consists of 7 leds arranged in some definite pattern (cathode or anode type) which is used to display hexadecimal numbers. In the cathode type of display, all cathodes of the seven leds are connected together to the ground or -Vcc and led displays digits when some high signal is supplied to the individual anodes. In the anode type of display, all the anodes of the seven leds are connected to battery of +Vcc and led displays digits when some low signal is supplied to the individual cathodes. But, seven segment display does not work by directly supplying voltage to different segments of LEDs. First, our decimal number is changed to its BCD equivalent signal then BCD to seven segment decoder converts that signals to the form which is fed to seven segment display. This BCD to seven segment decoder has four input lines (A, B, C and D) and 7 output lines (a, b, c, d, e, f and g), this output is given to seven segment LED display which displays the decimal number depending upon inputs.

****

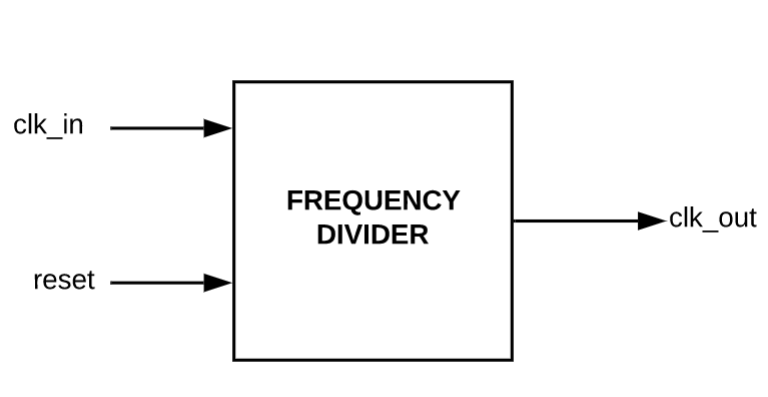
****

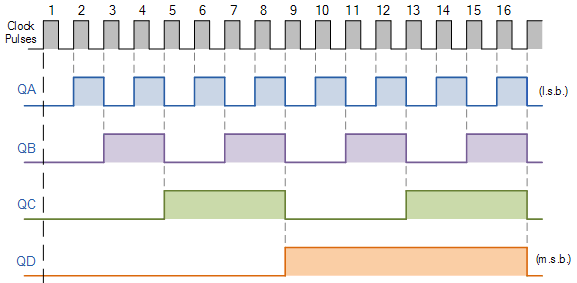
**3.5Command unit :** this unit receives the signals from the other units (execution units) : the minimum and maximum temperatures set for a specific hour with the help of the 5 buttons: inc\_min, inc\_max, dec\_min, dec\_max and ok) , as well as the current time with the help of the two RAM memories, one for the minimum temperature and

one for the maximum temperature, such that it can give commands concerning the settings done by the user with respect to the current time. It also commands the “heat” signal to be on whenever the temperature needs to increase or off when the temperature needs to decrease.



The command unit also has a multiplexor which, with the help of the “set” switch, decides what sould be displayed: when the switch is on “0”, on the display will pe shown the current time and the current temperature, and when the set switch is on “1’, on the display will be shown the setting mode, where the user can make the changes for each hour.

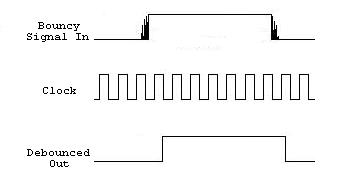
**3.5. Frequency divider:** it is a circuit that takes an input signal of a frequency and generates an output signal of the frequency. Nexys 4 has a frequency of 100MHz and the FPGA board’s clock is divided to 1 Hz ( 1 second ). This was done such that the clock can work in real time. Otherwise, the clock would have worked much faster.

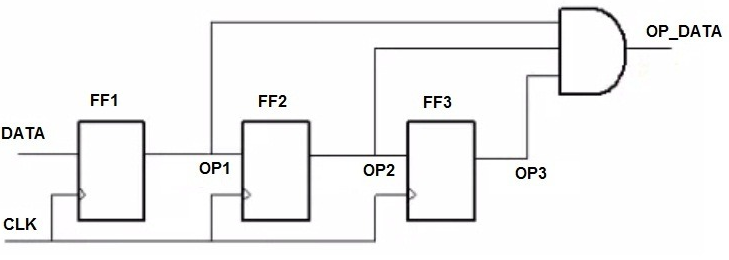


(this is an example of how a clock divider works)

**3.6. Debouncer:** thiscomponent takes an input signal from a bouncing contact and generates a clean output for digital circuits. The component will not pass the signal to the output until the predetermined period of time when the switch bouncing settles down. In this way, the circuit will respond to only one pulse generation performed by the pressing or releasing of the switch and not several state transitions caused by contact bouncing.

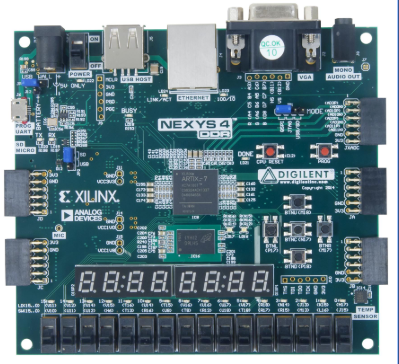
(the debouncing phenomena) (debouncer done with D flip-flops)





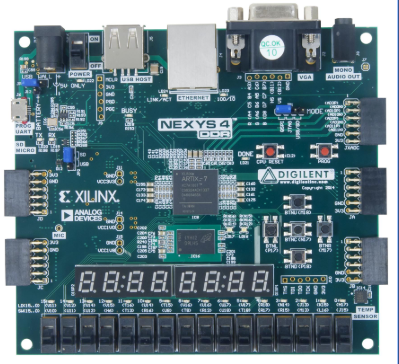
**4.The meaning of the notations and the interface with the outside**

**4.1 Inputs:**



“**set**”-switch to set minimum and maximum temperatures or to display current time and temperature

“**reset**”-switch to reset time and previous settings



“**ok**”-button to store the set temperatures and change the hour

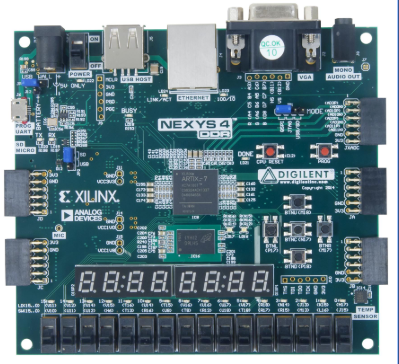
“**inc\_max**”-button to increment the maximum temperature

“**inc\_min**”-button to increment the minimum temperature

“**dec\_min**”-button to decrement the minimum temperature

“**dec\_max**”-button to decrement the maximum temperature

**4.2.Outputs:**

****

**Seven segment display**- shows current time and temperature when the “set” switch is on “0” and the hour, minimum and maximum temperature when the “set” switch is on”1”

“**heat**” –led that is on when the temperature increases and off when the temperature decreases

**5.Justification of the chosen solution**

I chose to implement the thermostat this way because it’s easy to understand by the user: the settings can be done from the switches and buttons , without the user to make changes in the code or to work in binary numerical system, the instructions being easy to follow.

Also, this implementation shows a certain degree of generality thus it be further developed.

**6.Instructions for use and maintenance**

To first start the thermostat, the power switch of the FPGA board must be on . For the hours and temperatures to initialize, the “reset” switch on “1”. After the values have been initialized, the “reset” is switched on “0”. By now, on the display should appear the current time (hh:mm format) and the current temperature, which increases/decreases every 3 seconds in the interval 15-35 degrees which is the default setting.

To set a range of temperatures for each hour of the day, switch the “set” switch on “1” while the “reset” switch is still on “0” ( in case the user doesn’t want to reset al previous settings, the “reset” switch should stay on “0”! ). By now, on the display should be shown the hour (starting from 00), the minimum temperature (15-default) and the maximum temperature(35-default). If the user wants to change the range of temperatures, by using the increment/ decrement buttons for the minimum / maximum temperatures, it is possible to increase/ decrease the current shown temperatures until the user gets to the desired settings (note that the temperatures can’t pass the limits of 0-40 degrees. In case the user keeps incrementing/ decrementing the temperatures beyond this range, the values won’t go further). In case the user doesn’t set any specific temperatures for the hours, the thermostat will work by the default settings. After the setting values of temperatures for a specific hour, the “ok” button should be pressed such that the new values are stored and settings for the next hour are possible.

If all the changes are made, turn the “set” switch on “0” such that the thermostat can shouw the current time and temperature which will be kept in the established range.

To reset the thermostat ( to come back to the default settings ), the “reset” must be switched on “1”.

For the thermostat to work properly for a longer period of time, the user should avoid spilling any kind of substances on the FPGA board, avoid dropping or applying force on the board, should keep it away from any radiant source which can cause damage to the hardware.

Also, the board should be kept connected to the power supply for the thermostat to work allthe time.

When using the buttons and the switches, not mush pressure should be applied, the board is programmed in such a way the that signals are easily transmitted to the command unit.

**7. Possibilities for further development**

The thermostat works properly for the initial requirements. Even so, some characteristics could be further developed: the initial hour could be set from the board, not inside the code, a temperature sensor could be connected to the board such that it can read the actual temperature in the room, not a random number set from the code.

Also, for a more advanced possibilities, the thermostat could also show the date such that a range of temperatures could be set for every day in the year, or just for seasons / periods.