TECHNICAL UNIVERSITY OF CLUJ-NAPOCA ROMANIA

# PROGRAMMABLE OFFICE DIARY

# (B6)

DISCIPLINE: DIGITAL SYSTEM DESIGN

Student: URDEA MARA-CRISTINA
Group: 30412
Laboratory guide: Pop Diana-Irena

# Table of contents

# 1. Task

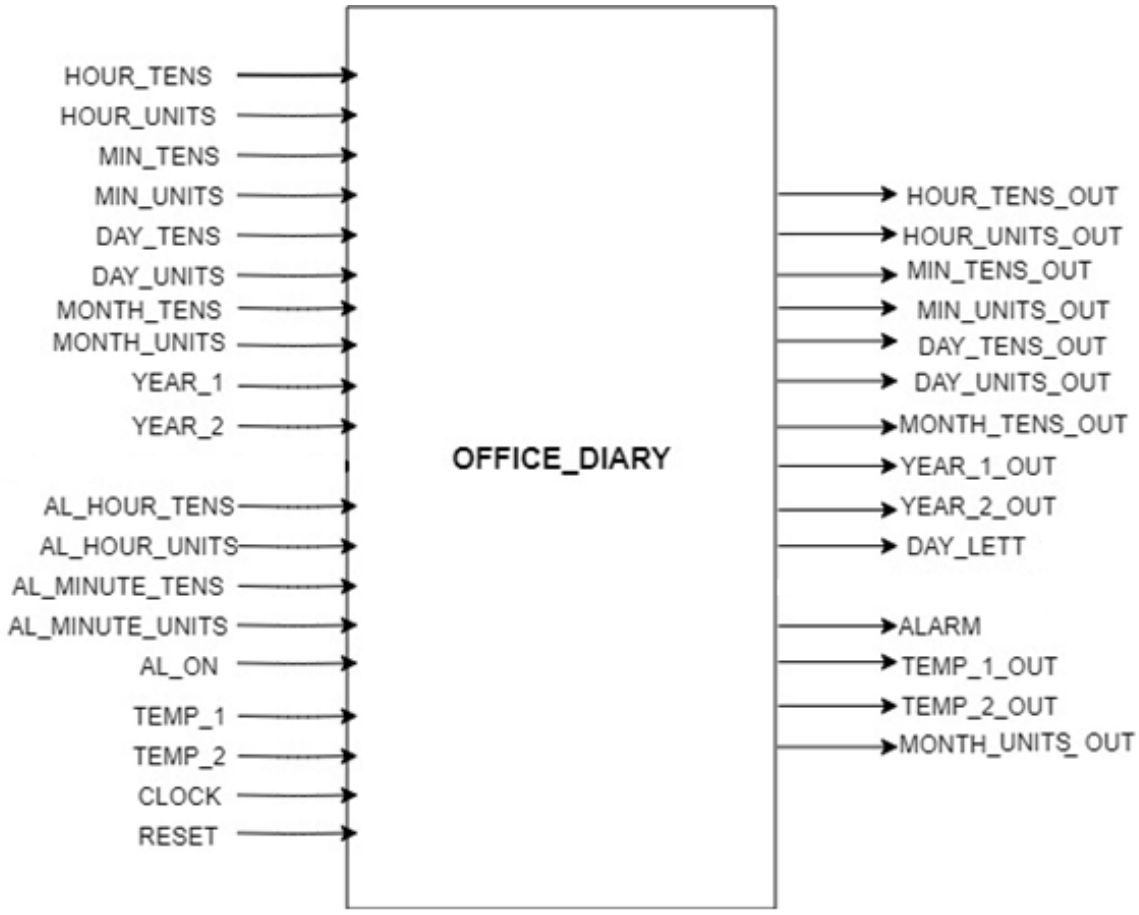
Design a programmable office diary that displays:

- year, month, day: with numbers;
- day: with letters;
- hour and minute: with numbers;
- ambient temperature in degrees Celsius.

The agenda will also be provided with an audible alarm associated with the hour and minute.

# Chapter 1: Design

## 1.1 Black box



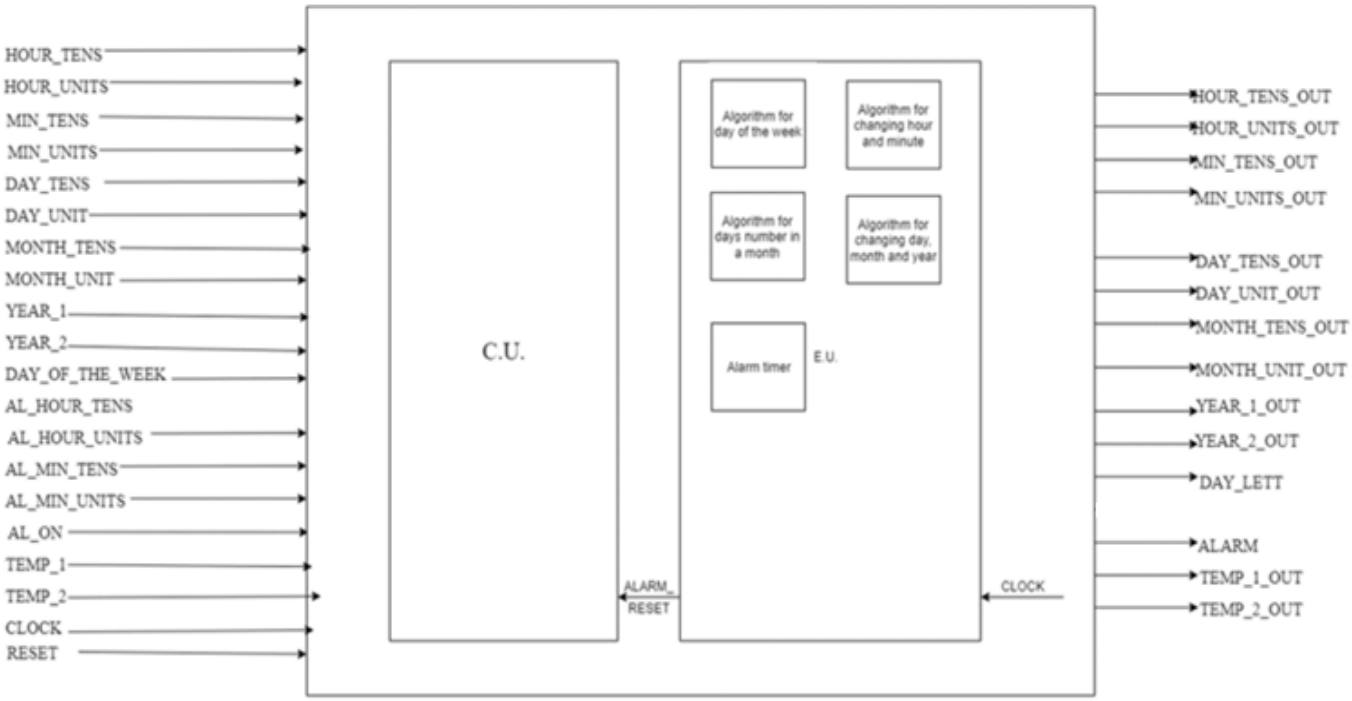
Inputs:

1. HOUR_TENS (4 bits): sets the tens part for hour
2. HOUR_UNITS (4 bits): sets the units part for hour
3. MIN_TENS (4 bits): sets the tens part for minute
4. MIN_UNITS (4 bits): sets the units part for minute
5. DAY_TENS (2 bits): sets the tens part for day
6. DAY_UNIT (4 bits): sets the units part for day
7. MONTH_TENS (4 bits): sets the tens part for month
8. MONTH_UNIT (4 bits): sets the units part for month
9. YEAR_1 (4 bits): sets the penultimate digit for year
10. YEAR_2 (4 bits): sets the last digit for year
11. AL_HOUR_TENS (4 bits): sets the tens part for alarm's hour
12. AL_HOUR_UNITS (4 bits): sets the units part for alarm's hour
13. AL_MIN_TENS (4 bits): sets the tens part for alarm's minute
14. AL_MIN_UNITS (4 bits): sets the units part for alarm's minute
15. AL_ON (1 bit): sets if alarm is on
16. TEMP_1 (4 bits): sets the tens part for temperature
17. TEMP_2 (4 bits): sets the units part for temperature
18. CLOCK
19. RESET (1 bit): when 0, saves data, when 1, changes are made

Outputs:

1. HOUR_TENS_OUT (4 bits)
2. HOUR_UNITS_OUT (4 bits)
3. MIN_TENS_OUT (4 bits)
4. MIN_UNITS_OUT (4 bits)
5. DAY_TENS_OUT (4 bits)
6. DAY_UNIT_OUT (4 bits)
7. MONTH_TENS_OUT (4 bits)
8. MONTH_UNIT_OUT (4 bits)
9. YEAR_1_OUT (4 bits)
10. YEAR_2_OUT (4 bits)
11. DAY_LETT (4 bits)
12. ALARM (4 bits)
13. TEMP_1_OUT (4 bits)
14. TEMP_2_OUT (4 bits)

## 1.2. Control and execution unit

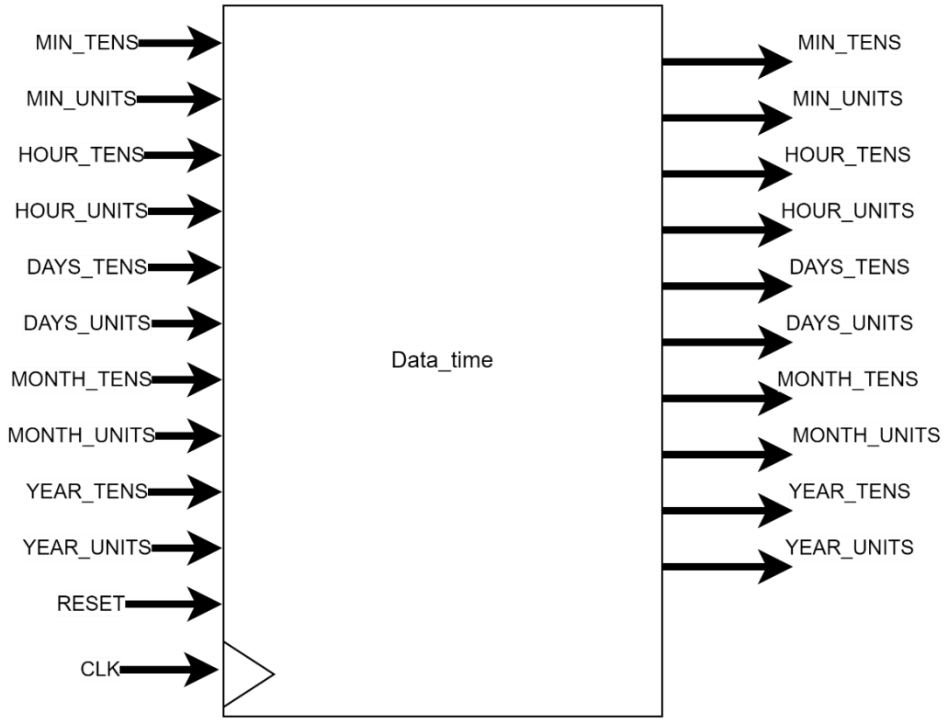



Signals:

ALARM_RESET: resets the alarm

## 1.3. Resources

1. Date and Time Unit




Here I used an algorithm, The Key Value Algorithm, to calculate the day of the week using codes for different months and years.

Suppose the year is not in the above table. In this case all we have to do is add or subtract 400 until we have a year (century) that is in the table. Then get the code for the year from the above table and add the value to the previous step (our running total).

7. Add the last two digits of the year to the value we obtained in the previous step.
8. Divide this value by 7 and take the remainder. Get the day from the following table based on the value of the remainder.

| Sun | Mon | Tue | Wed | Thurs | Fri | Sat |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 0 |

**Steps:**

1. Take the last 2 digits of the year.
2. Divide it by 4 and discard any remainder.
3. Add the day of the month to the value obtained in step 2.
4. Add the month's key value, from the following table to the value obtained in step 3.

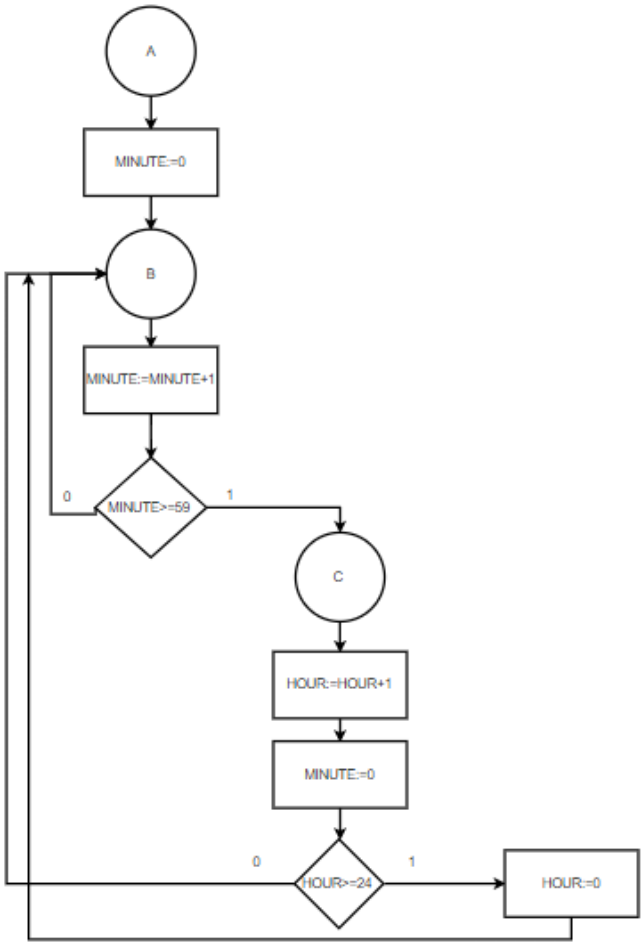
| Jan | Feb | Mar | Apr | May | June | July | Aug | Sept | Oct | Nov | Dec |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 4 | 4 | 0 | 2 | 5 | 0 | 3 | 6 | 1 | 4 | 6 |

5. If the date is in January or February of a leap year, subtract 1 from step 4.
6. Add the year (century) code from the following table.

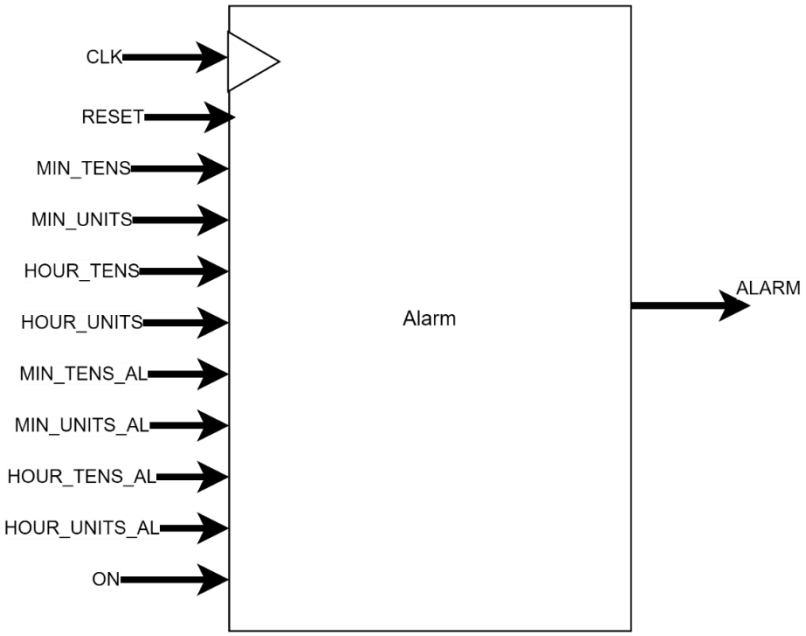
| 1700s | 1800s | 1900s | 2000s |
|---|---|---|---|
| 4 | 2 | 0 | 6 |

The algorithm for changing hours, minutes, day, month and year is similar to the one in C language.
Code example:

```
IF rising_edge(clk) then
    --change minute units
    m0<= m0 + "0001";
    --change minute tens
    if m0= "1001" then
        m0<= "0000";
        m1<= m1 + "0001";
    end if;
```



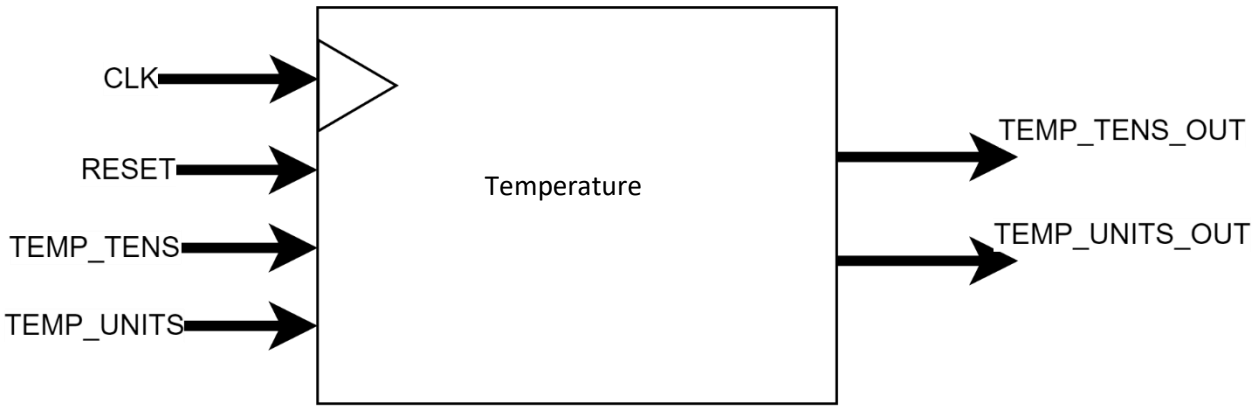
2. Alarm




The piece of code that checks the alarm condition:

```
-- verify the condition to "ring" the alarm
IF (alarm_h1=h1) and (alarm_h0=h0) and  (alarm_m1 = m1) and (alarm_m0 = m0) then
    alarma_o<= '1';
else
    alarma_o <= '0';
end if;
```

3. Temperature



## Chapter 2: Instructions for use

• Enter the number of tens and the number of units of the hour
• Enter the number of tens and the number of units of the minute
• Enter the number of tens and the number of units of the day
• Enter the number of tens and the number of units of the month
• Enter the penultimate digit and the last digit of the year

The device has the option to set an alarm. To set the alarm, proceed as follows:
• Enter the number of tens and the number of units of the alarm time
• Enter the number of tens and the number of units of the minute from the alarm
• AL_ON is set to 1, marking that the alarm is set

## Chapter 3: Justification for chosen solution

I have chosen this project because it seemed interesting and because it has everyday applications. When I have started I realized that it is not that easy as I thought. I tried to use my algorithmic thinking and make everything to look well developed.

## Chapter 4: Possibilities for further development

Each program / project is designed to work as well as possible, but there is always room for change and improvement. In the case of my project, here are some things that could be added to improve the project:

- Adding a button to be able to set an alarm for several days, so that the user does not have to enter it every time.

## Bibliography:


- DSD lectures
- https://www.diagrameditor.com/
- https://www.geeksforgeeks.org/digital-electronics-logic-design-tutorials/?ref=lbp
- https://vhdlguru.blogspot.com/
- https://www.aldec.com/en/support/resources/documentation/articles/1054
- https://www.fpga4student.com/2016/11/vhdl-code-for-digital-clock-on-fpga.html
- https://beginnersbook.com/2013/04/calculating-day-given-date/