|  |  |
| --- | --- |
|  | Digital Clock & Weather Station |
|  |  |
|  | Mamadu Balde U1572298  BSc(Hons) Software Engineering  4/25/17 |

Topics

[Introduction 2](#_Toc481159311)

[Design Specification 2](#_Toc481159312)

[Design Aim 2](#_Toc481159313)

[Hardware Specification 3](#_Toc481159314)

[Operational Specification 3](#_Toc481159315)

[Running Mode 3](#_Toc481159316)

[Configuration Mode 4](#_Toc481159317)

[Alarm Mode 5](#_Toc481159318)

[Further Requirements 5](#_Toc481159319)

[Use of Interrupts 6](#_Toc481159320)

[Design and Implementations 6](#_Toc481159321)

[Design Flow Chart 7](#_Toc481159322)

[Reflection 7](#_Toc481159323)

[References 8](#_Toc481159324)

# Introduction

For this second term, I am required to complete an embedded systems coding project to a design specification. This assignment comprises four elements:

1. **Logbook**: It is a set of exercises provided in the lab handbook and these exercises are to be completed and evidenced in an electronic logbook
2. **Developed project code**: A challenge to create an embedded systems solution which satisfies the stated specs
3. **Short report**: This report will contain the summary of the design and submitted project
4. **Demonstration**: A demonstration of the design and some technical questions related to the project

## Design Specification

As we know, alarm clocks are an indispensable item for the modern living, helping us to manage our activities and attendance for a specific matter.

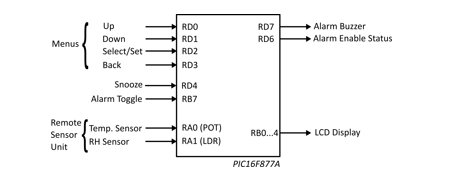
For this project, I have been challenged to use the Matrix Multimedia Pic development platform to create an embedded system which will have a clock with specific functions such as weather monitoring and this Matrix has a remote sensor unit that is located outside which will be responsible for measuring the humidity and temperature.

## Design Aim

The aim is to develop this project is C and simulate the embedded system that will contain an alarm clock weather and tested using the Matrix above mentioned. Using LCD to display menu options, and other information such as current time, alarm time, relative humidity and outside temperature. Using pushbutton switches to simulate inputs.

Using LEDs to simulate the alarm output and alarm enable.

## Hardware Specification

For this project, I have used the PIC16F877A microcontroller to simulate the development of the proposed project and the inputs and outputs are summarised as follow:

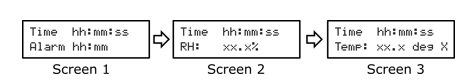
From the above diagram, we can see that the microcontroller push buttons are defined to do a specific task. For instance, the menus are set to be RD0 to RD3, where each one of them has a specific task as you can see from the above diagram.

## Operational Specification

As the assignment specified, this system will operate in three modes.

### Running Mode

In this mode, the device has the freedom to run as soon as the “. hex” file is uploaded to the PIC16F877A and displays the weather, humidity, time and alarm information.

For this mode, the screen cycles repeatedly, with five seconds’ interval between each screen and the current alarm time, temperature and humidity information is displayed to the user interface while the time is visible in all three screen

The above image describes how the information is represented on the LCD screen. Let’s look at the time and alarm, the time has hh: mm: ss: ss format while the alarm has only the first four digits’ formats hh:mm.

The temperature and the humidity are displayed a precision of one decimal place and the unit displayed for the temperature is depending on the mode the temperature is set to Fahrenheit or centigrade.

#### Display of Time (all screens)

Using Timer1, I could generate the interval required to update the current time on the display at one second interval through all screens.

#### Display of Temperature (screen 2)

The analogue input connected to RA1 is used to simulate the output of the temperature. Using RA0, allow the voltage to be set between 0 to 5 volts through the potentiometer on the E-block sensor board.

The temperature unit has value depends on the temperature set. If the temperature is set to Centigrade, the running mode will display a range of values from -40 to +50C. When the temperature happens to beset to Fahrenheit, the conversion is applied before displaying the data as follow:

 or Temperature (deg F) = ((Temperature (deg C) \* 9) / 5) + 32.

The temperature for Fahrenheit will vary from -40 to 122\*F.

#### Display of humidity (screen 3)

The humidity module has the same simulation as the temperature, it requires a voltage that varies between 0 to 5 volts (LDR) on the E-block sensor board.

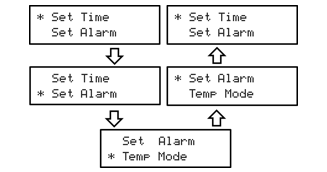
The value displayed during the running mode will vary between approximately 10% to 90% RH.

### Configuration Mode

The configuration mode is the part where the hardware specification above mentioned takes places.

Using the above description and configuration would give the user the ability to access the main menu and initialise the configuration of a specific menu item, for example, using the RD2, would give us the three main configuration options which are the “set time”, “set alarm” and “set temperature”.

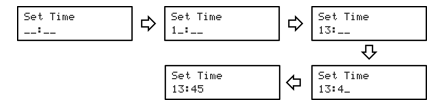
There is a sub set inside the “set temperature” option which contains the temperature format: Centigrade/Fahrenheit.

Using the RD0 which moves the cursor up and RD1 to move the cursor down and RD2 to select pushbutton switches the configuration should have the following the sequence:

The configuration can be interrupted by pressing the back button which is RD3 pushbutton where the system will revert the running mode.

#### Setting the Time

By selecting the Set time menu and using the combination of RD0 to increment the current digit value, RD1 to decrement the value and RD2 to move the LCD cursor to the next digit value we can set the time as we require and it has a sequence for the value entry.

The back button, in this case the RD3 can be pressed at any time and that would allow to rectify any errors and once the time its set, the system should return to the main screen.

#### Setting the Alarm

Setting the alarm is the same as setting the time

### Alarm Mode

The alarm is only entered when the set alarm values matches the current time and the alarm is enabled.

By pressing the RB7(Alarm Toggle) pushbutton, at any time, the alarm may be enabled or disabled.

Let suppose the RB7 is pressed, in case that happens, the LED connected to RD6 which is enable as being the Alarm Enable Status must be ON.

When the alarm is activated, the LED connected to the Alarm buzzer which is RD7 must toggled with a sequence of repetitions for 20 times:

* RD7 ON for 0.25 s
* RD7 OFF for 0.5 s

During the alarm buzzing, the normal sequence must operate as usual, the current time, the temperature and the humidity should continue to run normally.

## Further Requirements

The user of default settings for the initial run time, alarm time, temperature and alarm enable status and use of interrupts are required.

* The following default setting are expected upon system start up**.**

|  |  |
| --- | --- |
| **Item** | **Default value at start-up** |
| Time | 00:00:00 |
| Alarm Time | 08:00 |
| Temperature Mode | Centigrade |
| Alarm Enable Status | Off |

### Use of Interrupts

#### Incrementing the current time

As we know, the reason why we use time is to keep on track our important activities and for that we need it to be accurate.

For this project, the use if TMR1 and its associated overflow interrupt (TMR1E/TMR1F) was crucial and it is the timer module that makes the clock tick the current time with the right accuracy.

#### Alarm Enable Toggle

If we require to enable or disable the alarm at any point during the operation, RB7 would be the right pushbutton and for that, the PORTB interrupt (RBIE/RBIF) which generates an interrupt must the implemented.

#### Alarm mode sequence

The use of TMR0 is required to set the intervals for the alarm buzzer (RD7). The TMR0 0verflow interrupt (TMR0IE/TMR0IF) are carried out for toggling the alarm buzzer.

## Design and Implementations

The following table has the main functions I have used for this project and what each one of them is expected to do.

|  |  |
| --- | --- |
| Function Name | Function Description |
| void config(void); | function for the configuration/ settings |
| void interrupt MyISR(void); | function for the interrupt |
| void ADC\_initialise(void); | function to initialise the ADC |
| unsigned int ADC\_read(void); | function to initialise the ADC reader |
| void Timer1Delay(unsigned char delay); | TMR1 function for the right time interval counter |
| unsigned int screenConfigurationMode(void); | function to select the available options in the screen, array of strings |
| long setTimeDisplayer(void); | function to set the time to be displayed |
| long setAlarm(void); | function to set the alarm |
| char setTemperatureUnit(void); | function to set the temperature unit |
| void updateTimeDisplayed(long elapsed\_time); | function to update the time displayed |
| void updateSetAlarm(long elapsed\_time); | function to update the alarm set |
| void temperatureReader(void); | function for the temperature reader |
| void humidityReader(void); | function for the humidity reader |

## Design Flow Chart

## Reflection

I am quite thankful for the lab exercises I have been given and as the lecturer always mentioned that the labs would be essential for the project. I did not complete all the given labs for this term but I managed to complete from lab 6 to lab 9 and that helped me to do this project.

I was having difficulties on how to start the project and thanks to the lecturer who provided an example on how to initialise the most important parts of the project, I came up with my own ideas and implement the instructions I received from the code provided. I followed some examples from the code and my own research and to came up with my own ideas on how to structure my code and make it easier to access and modify in case of error and make it easier to understand.

Overall, it was a good challenge and I had the opportunity to build my own embedded system which I can mention in future job interviews.

# References

(2017). Retrieved from rapidtables: http://www.rapidtables.com/convert/temperature/celsius-to-fahrenheit.htm

Brothers, H. (2007-2017). Retrieved from calculateme: http://www.calculateme.com/Time/Hours/ToSeconds.htm

Huddersfield, U. o. (2017, 4 16). *Unilearn*. Retrieved from Unilearn: https://unilearn.hud.ac.uk/

*MicrocontrollerBoard.com*. (2008-2017). Retrieved from http://www.microcontrollerboard.com/pic-timer0-tutorial.html

(2008). *Timers: Timer0 Tutorial.* Microchip Technology Inc.