Project Report: IoT Application

Muhammad Faisal

May 2020

1 Introduction

The Internet of Things (IoT) is an emerging field that researches the communication of many devices including even the small sensors over a wide network such as the internet. In this project, I am building a simple application that allows a user to control a microcontroller through a simple user interface. The user can communicate to control an RTC module in the system. additionally, a user of this application would be able to also modify the current status of the STM32 board LEDs through the same user interface.

Github: https://github.com/mickey-me/iot-app

2 Requirements

The project have handled many important embedded system requirements such as the following:

1. Reliable communication between the different components in the system (MU - WIFI - RTC).
2. Failed requests do not affect the system's future operation.
3. The system implements the needed real time system requirements such as that requests are handled in a timely manner.

3 Features

1. Managing time/date retrieval from an RTC module.
2. Managing the microprocessor module LEDs.
3. Managing the RTC alarms.
4. Make use of a WIFI module as an access point and use it to provide access to the system through a simple user interface.

4 Used Technologies

The project has been implemented using different technologies and components such as the following ones:

4.1 used components

There have been three main electronics components in the system and all of them are cheap and low power consuming components.

4.1.1 ESP8266 module

This is a WI-FI micropship that supports TCP/IP. Mainly, it has the following features: high durability, compactness, power-saving architecture, and 32-bit tensilica processor. This module can be powered by 3.3v and therefore it is powered through the microcontroler's 3.3v output.The Wifi module is used to provide a web interface for the system. First, it outputs a user friendly interface that users can view using their normal internet browsers. The requests received from the users are then forwarded through the Wifi module to the microporcessor which processes the request and then send back its reply to the wifi module to be viewed by the user.The module code is written through the Arduino library and tools and that provides large capabilities and easiness to the development process.

1. **Arduino** is an open source hardware project. They design hardware and various designs for microprocessors and then open source these designs. They also provide an easy to use library to build an embedded system easily. I am using the Arduino library and tools to control the Wifi module and also to do the serial debugging.

4.1.2 STM32L432KC Microprocessor

This is an advanced low-cost microprocessor that has many features for the project. The microprocessor manages the communication between the different components in the system. Ideally the microprocessor is kept in an ideal state till it is invoked by the Wifi module through the microprocessor's first UART. The received buffer from the Wifi module is processed for commands to be executed in the microprocessor. The Microprocessor is also the unit responsible for controlling the RTC module through one of its I2C ports.

1. **STMCubeMX** is a useful program that provides an easy to use GUI to create a project starter with initial configuration. The program is useful to point out which connections to use for each feature such as for using UARTs or I2Cs.

2. **Arm Keil** is a useful development program that facilitates editing the starter program created by STMCubeMX. Moreover, it provides debugging features for the project in addition to board flashing.

3. **FREERTOS** is a very useful library that provides great features for real time systems. It provides features such as tasks and queues. Queues are useful for dividing the project into sub tasks where each task has its requirements such as period and priority.

4.1.3 DS3231

This is a real time clock module which consumes low power and supports alarms and powering from a small battery. The module in this project is powered and controlled from the microcontroller. The microcontroller is controlling this module through I2C communication. At the begning the RTC time is set and then alarms or enquiries to the RTC is done upon requests to the microcontroller. The RTC is also connected to a buzzer that issues sound once an alarm time is reached.

5 Design and Implementation

A simple diagram for communications between the different components in the system can be found below. The system consists of two main non-exiting programs; one on the microcontroller and the other is on the Wifi module.

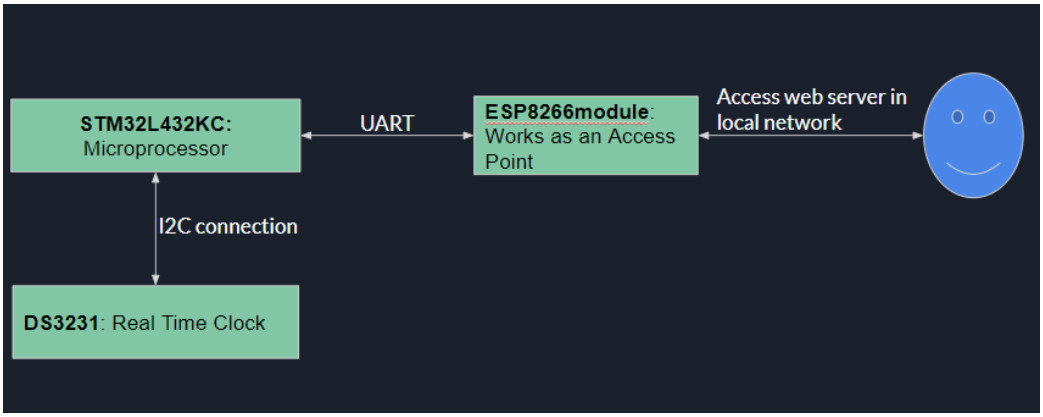

Figure 1: Communication between different system components

Each has been designed carefully in order not to get stuck or have one of their peripherals hangs them.

Figure 2 shows a boat.

5.1 Microcontroller

First, for the microconrtoller, it has been divided into 4 tasks and they are designed as follows.

1. Two tasks are responsible for managing the first UART which is connected to the Wifi module. One task is responsible for receiving the buffer and adding to the received buffer queue. The other task is responsible for transmitting the output buffer stored in the out buffer queue.

2. The third task is responsible for handling the received buffer by parsing it, extracting its different paramters. Once the parsing is done, the task starts executing the needed steps to process the received requests. If the request is a simple such as the reading a pin and changing the pin's state, it is handled within the function task itself. If the request is more complex and requires much time such as a request regarding the RTC (requires time for sending and receiving i2c data), this third task would then unblock the fourth task to handle the received request.

3. The fourth task handles requests related to the RTC module and it

handles exactly 4 different types of requests. The first one is RTC initialization and is done at the initialization of the system. The second one is for setting a specific time for the RTC. The third is for setting an alarm and the fourth is time querying. These requests are time consuming since they need to wait for the RTC module to reply to them. Moreover, when the task finishes the request, it suspends itself waiting for invocation from the third task.

5.2 Wifi Module

As per the Wifi module, at first it initializes itself setting a web server that waits for anyone to contact. The web server then communicates with users by providing them an html source code that can be viewed through any internet browser normally. Once the user requests anything through the web interface the wifi module would do the necessary communication with the microcontroller, leaving for it the needed amount of time to process the request and then received the reply and forward it to the user through the interface. The Wifi module communicates with the microcontroller using UART.

5.3 Communication

In order for the microcontroller and Wifi module to communicate together, a set of commands and replies have been developed. Having a distinct form for each request or reply helps in parallelizing the different tasks in the system and therefore reduces the busy wait. For example, two different requests may be sent to the microcprocessor at the same time. The microprocessor would then tries to process them in parallel if that is possible and then sends the replies back out of order without problems since each request has a distinct reply form.

These are examples of the different requests and replies used in our system:

1. **;ATTIME?;** This is a request to get the current time from the microcontroller.

2. **;ATLED?;** This is a request to get the current state of the microconroller led.

3. **;ATLED=0;** This request tells the microcontroller to set the Led One Off.

4. **;ATLED=1;** This request tells the microcontroller to set the Led One On.

5. **;ATTIME=1xx:xx:xx;** This request asks the microcontroller to change the current time in the RTC.

6. **;ATTIME=1xx:xx:xx;** This request asks the microcontroller to set an alarm in the RTC.

7. **xx:xx:xx** This reply tells the Wifi module the current timein the RTC.

8. **On** This reply tells the Wifi module that the led is currently On.

9. **Off** This reply tells the Wifi module that the led is currently Off.

All the requests have been put between the termination sign ";" which is used to make sure the each command do not get mixed with rubbish buffer from previous failed requests for example.

5.4 User interface

The user interface as shown below consists of 4 main parts. The first section is about controlling the microcontroller Led. It shows the current state of the led in addition to a button that can change the current state of the led. The second section is for showing the time and has a button that refreshes the time when clicked. Additionally, There are two more sections, one for setting the RTC time and the other to set an alarm in the RTC. The time shown in this page uses a 24h system not a 12h system.

6 Project Development process

The development of the project used an agile based approach and is divided as follows:

1. In the beginning, a simple led blinking project was created and tested.

Simple iot-app

Current Led3 State On

TURN OFF

Current Time: 00:12:04

Refresh Time

Time:

Set time

Alarm:

Set Alarm

2. The FREERTOS was then used to operate this simple project of blinking the led. The led blinking was done through the default task
3. Tasks and queues for the UART communication were added.
4. Support for led management.
5. Simple Support for RTC management.
6. Support for richer requests and answers (setting time and alarm).
7. More reliable communications with zero dropout requests.

7 Acknowledgments

This project is done as a part of the Embedded Systems course project for Spring 2020 in the American University in Cairo under the supervision of Dr. Mohamed Shalan.