­­­

Communication system for non

IoT ready devices

Based on nRF24 radio modules

Ivica Matic  
June 2019

# Table of Contents

[Table of Contents 2](#_Toc10401071)

[Introduction 3](#_Toc10401072)

[nRF24 Wireless module 7](#_Toc10401073)

[Custom communication system 9](#_Toc10401074)

[References 10](#_Toc10401075)

# Introduction

Today we are living in a world that is filled with a information. During our everyday activities such as a taking a bus or driving home to work we are using a Network of devices that we are unaware of. From traffic lights to bus stop schedule displays we are using some type of network capabilities. Just looking at a Information that is presented to us on a form of display, traffic lights or our phone requires a minimal effort and we can plan out next activities such as when to get a bus, decide on a quickest route home, etc. But we are unaware of how that information got there.

All that information is captured by some sort of sensor device. Devices such as a temperature sensors, traffic flow sensors, pressure sensors exist out there for many years and provide us with crucial information to plan out our daily activities. With IoT[[1]](#footnote-1) devices growing popular we can obtain that data very easily, cheap and effective.

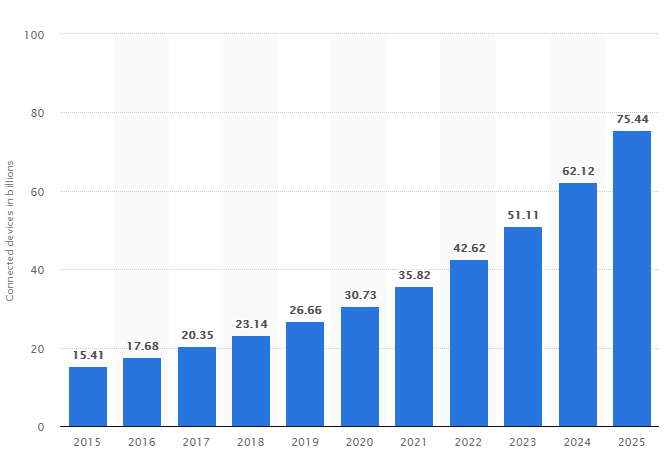


Figure 1.   
Number of IOT connected devices worldwide (in Billions)

But there is a certain problem of those IoT devices. Every IoT device will work well in the area that has good any decent Internet coverage, being it Wirelessly or Wired. Out there on the market there is a huge number of devices supporting WIFI as a mean of their primary communication protocol. Few of those devices are Espressif Systems [[2]](#footnote-2)ESP8266 and ESP32.

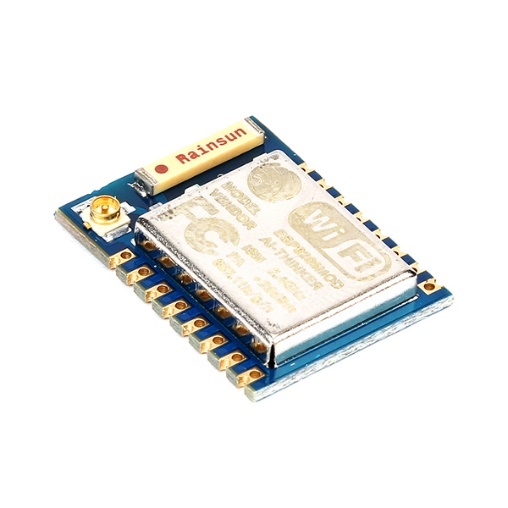


Figure 2.   
ESP8266 module (left side) and ES32 module (right side)

These 2 devices offer easy access to the world of IoT by enabling uses to program them directly to achieve specific task. As they offer on board WIFI connectivity it is relatively easy to connect external sensor to obtain data and push it to the cloud or any other device.

There are two specific problems with this solution and other similar solutions on the market.

First problem is that those devices require WIFI network in order to exchange or publish data. This problem is not that big for our home-based devices such as a Amazon Alexa, in our household WIFI signal is pretty strong and consistent, which allows our devices to function correctly. But when you want to connect devices on a large geographical area such as a city street or whole city WIFI is not viable solution because of its limitation in range and transmitting power.

That problem can usually be omitted by using alternative technology to WIFI, technology like GSM [[3]](#footnote-3)or LoRa Wan[[4]](#footnote-4). But there are also some key problems in these two solutions. GSM data transfer can, depending on provider, can be very expensive, and also, as all our mobile phones use GSM technology for data transferring and making a call, GSM frequencies can be very crowded, resulting in harder and non-reliable communication channel. Also, if we want to cover large geographic area we might stumble upon no signal spots, but that is highly unlikely.

Other technology that is alternative to GSM and WIFI is LoRa Wan. LoRa Wan is a low powered long-range wireless network specifically created for connecting various IoT devices together. LoRa can be used to connect devices across large area because of it’s high efficiency, low error rate protocols. LoRa Wan is viable solution to replace WIFI and GSM technology in connecting IoT devices but it’s high initial cost and lack of public gateways in some area might throw back users from using it.

LoRa, WIFI and GSM also use frequencies in public ISM [[5]](#footnote-5) frequency spectre, enabling them to be free of charge for every user.

Second big problem of all technology and devices listed above is a fact that you need to have WIFI, GSM or LoRa Wan certified devices in order to use it for IoT specific task. Given the cheap price of those devices that is actually not even a big deal on todays market as the price of these devices is really low (ESP devices starting from avg. 2$). But for average maker that does not want to implement and use all capabilities it might isn’t a viable solution.

For example, we have boards like Arduino Uno which does not have any networking capabilities on it’s own. If user wants to obtain data from it remotely there is a few following solutions.

1. **Use Ethernet Shield on top of Arduino UNO**

This solution could be good for connecting Arduino permanently to Internet on one fixed location like in house or in your apartment, but the fact that Ethernet shield requires physical connection (Network cable) disables it to be used outside of wired network range and ads to the size of the Arduino board itself, it might not be viable solution if user just want to upload small amount of data online.

1. **Use WIFI Shield on top of Arduino UNO**

If we plan to connect our Arduino to Wireless WIFI network, we can use WIFI shield on our Arduino to give it network capabilities similar to those of Ethernet Shield. However, price of WIFI Shield can easily be more than price of WIFI enabled solution like ESP8266 which makes it one more unviable solution if user wants to upload small amount of data online.

1. **Use LoRa Wan and Arduino UNO**

Using LoRa wan and Arduino UNO is good solution if we want to connect our device outside of our home or reach of our Internet network. But LoRa Wan module by far overprices Arduino board itself and is not viable solution for that simple task. This also implies to GSM modules.

1. **Use nRF24 Wireless Module and Arduino UNO**

Priced similarly to Arduino board and offering relatively easy mode of communication for non IoT ready devices (like UNO itself) it provides cheap way of connecting devices together. However, as nRF24 is intended for two-way communication between boards it is currently not easy to implement it for connecting multiple devices and providing Internet capabilities. This paper will be based on this module, providing a way to use non IoT ready boards, and this Wireless module to create wireless network that mimics all features of real Internet network, enabling large area connectivity to those devices.

|  |  |  |  |
| --- | --- | --- | --- |
| **Technology** | **Range** | **Complexity** | **Price** |
| **Wired Networking** | Very low | Low | Low |
| **WIFI Networking** | Low to Medium | Low | Medium |
| **GSM Networking** | Very high | Medium to High | Medium |
| **LoRa Wan** | Very high | Medium | High |
| **nRF24 Networking** | High | High | Low |

Figure 3.

Range. Complexity and Price table for mentioned connection technologies.

Fact that makes nRF24 Networking so complex it that it requires a secondary device that is connected to the Internet so other node devices can transmit to it and by that put their information online. Thing that makes this solution viable for connecting node devices is a fact that nRF24 modules are relatively inexpensive compared to other networking equipment. Problem with them is that they are not Internet devices on their own and they can’t be used for IoT themselves without some sort of gateway like device as mentioned above. During the writing of this paper I have developed custom solution to overcome that problem and I will cover it with this paper.

# nRF24 Wireless module

nRF24 Wireless module is a communication module manufactured by Nordic Semiconductors, company specialised in making low power SoCs[[6]](#footnote-6). These boards use same 2.4GHz frequencies as WIFI does, but not using WIFI protocol, enabling them low power, high reliability communication between two points.

There are 2 main nRF24 devices on market in time of writing this paper. One of them is ordinary nRF24 module with build in antenna and 4dBm of output power. This board is more suitable for short communication distances. Other device is nRF24 module with low noise amplifier providing 10 dB of receive amplification and power amplifier output, providing up to 20dBm of output power and external antenna connector for even more increased range. .

Other electrical specifications are irrelevant to this paper, but they can be found on manufacturers website.



Figure 4.  
nRF24 Modules mentioned above

Communication is governed by channels and pipes. There are 125 channels that user can use to communicate with these devices. Channel width is less than 1MHz and according to a channel selected, communication is executed within the broad range of the 2.4GHz band (2.400GHz to 2.525GHz). Be sure to check your local regulations before setting your channel. To additionally split channels users can assign custom pipe number to transmission (uint64\_t hex number) to allow multiple devices to use same frequency (and by that the channel).

In order to establish communication between devices they must be tuned in on same frequency and same pipeline. Communication is accomplished by sending a payload to the other device along with other data like address, packet control and CRC in to prevent transfer errors. Supported transfer speed on these modules are: 250kBPS, 1MBPS and 2MBPS. To make communication more reliable and extend the range use as lowest transfer speed as possible.

Both devices offer error checking and error correcting algorithms on board, so we do not need to worry about checking validity of received message

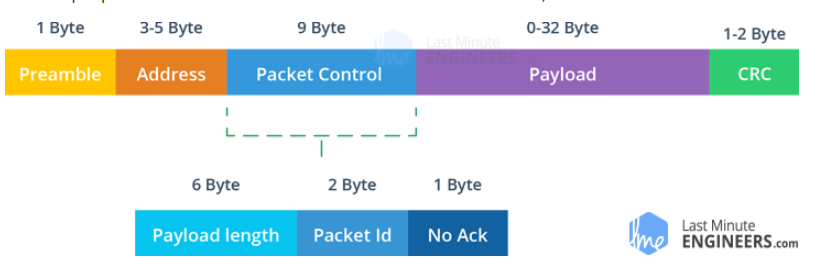


Figure 5.  
nRF24 Packet Structure  
(Last Minute Engineers, 2019)

Communication between module and micro-controller board (i.e. Arduino Mega ) is accomplished by bi-directional ISP [[7]](#footnote-7)bus. Because nRF24 is popular in world of hobby electronics there are few libraries written that we can use in order to get these modules up and running in short time.

To get the best results possible it is advised to use upper channels offered by the module because lower channels are shared with WIFI connections, and with many WIFI access points out there we are more likely to get poor results out of these modules because of background interference. It is worth mentioning that other technologies like wireless computer peripherals and Bluetooth also use lower part of the channels offered on nRF24 modules.

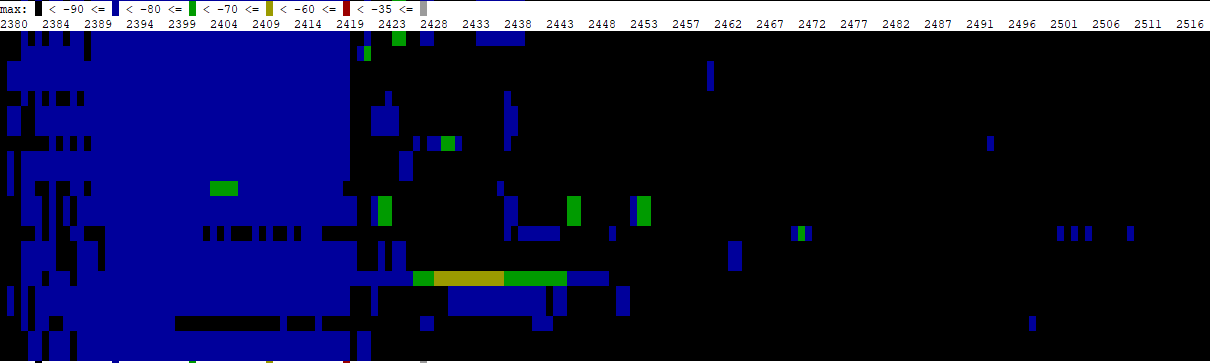


Figure 6.  
Spectral scan of 2.4GHz band using Mikrotik [[8]](#footnote-8) hAP Access Point

# Custom communication system

Now that we know how nRF24 modules communicate we can proceed on the topic. In order to establish communication between two points we need at least two nRF24 devices. As nRF24s do not have any processing power on board we need to combine them with some sort of microcontroller board capable on interfacing with them.

Because of that I have decided on implementing this solution with implementing a Gateway device. Gateway device has nRF24 connected enabling it to receive data from other nRF24 modules on same channel and pipeline. Gateway is also connected to the Internet to upload data gathered by nRF24 to the back-end application for there to be processed further.

Back-end web application and nRF24 enabled gateways form a network that can be used to transfer data wirelessly to the cloud. That data also can be stored in web application or sent out to WWW using some other protocol like MQTT or IFFT.

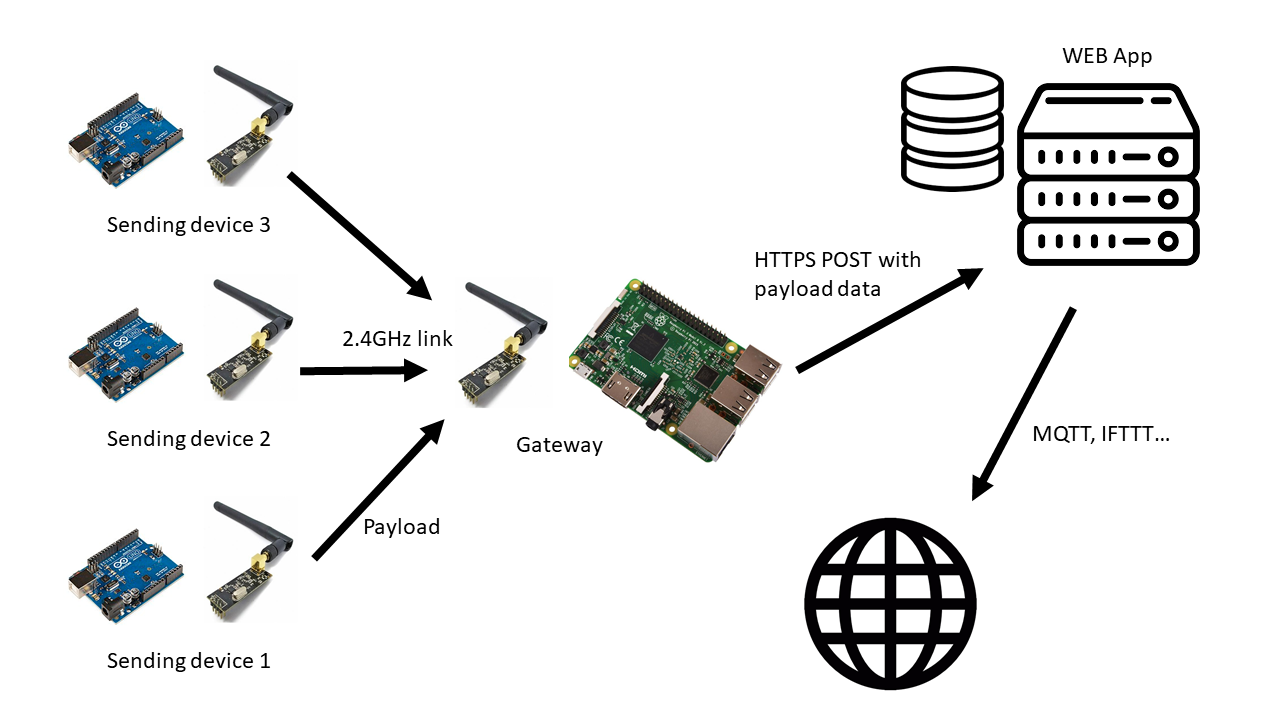


Figure  
Communication flow diagram

Whole network is designed to enable end users ease of connecting their own devices on network by creating an account and obtaining unique identifier for every information channel that they want to establish. (effectively one client device can send data to many information’s channels if user opts to do it that way. For every information channel there is unique id generated.). That piece of information can be anything that user wants, like outside temperature, battery voltage, atmospheric pressure…

As maximum payload that we can send with nRF24 is 32 bytes we need to split that into three parts necessary for this type of solution to work. For this solution first 8 bytes of payload are unique identifier for every data channel. That unique identifier is generated by web application and should be kept secret, as anyone knowing it can transmit data to that channel. Second 2 bytes indicate type of information that is sent in order to enable backend to process data accordingly. User does not need to fill in these 2 bytes as they are filled in accordingly to the information type entered. That leaves 22 bytes for information that user wants to send.

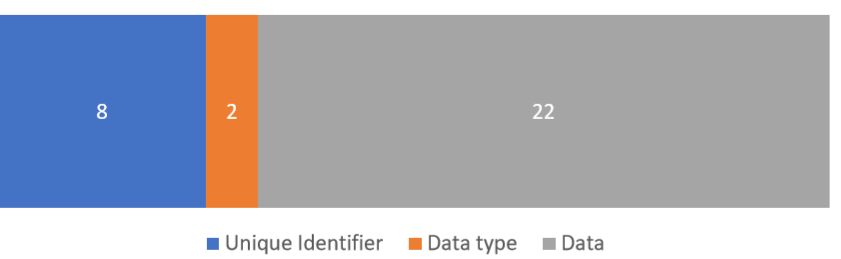


Figure   
Payload size and layout for communicating between nRF24 devices

Current plan is only to support one-way communication from Sending devices to Gateway, thus from Gateway to web application. This is because nRF24 module on gateway can communicate over 6 pipes at given time, which is not a problem from receiving data because all modules transmit on same frequency and pipeline (data is processed and routed later in web application based on unique id part of payload).

This program could be resulted by altering pipeline on the gateway according to device that user wants to receive data (pipeline number would be obtained from web application and sent back to gateway with POST request along actual data that needs to be sent)

# References

* Statista. (2019). IoT: number of connected devices worldwide 2012-2025 | Statista. [online] Available at: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/ [Accessed 31 May 2019].
* SearchMobileComputing. (2019). What is GSM (Global System for Mobile communication)? - Definition from WhatIs.com. [online] Available at: https://searchmobilecomputing.techtarget.com/definition/GSM [Accessed 31 May 2019].
* Last Minute Engineers. (2019). In-Depth: How nRF24L01 Wireless Module Works & Interface with Arduino. [online] Available at: https://lastminuteengineers.com/nrf24l01-arduino-wireless-communication/ [Accessed 2 Jun. 2019].

1. Internet Of Things [↑](#footnote-ref-1)
2. Chinese based company producing WIFI and Bluetooth IoT solutions, [www.espressif.com](http://www.espressif.com) [↑](#footnote-ref-2)
3. Digital mobile network that is widely used by mobile phone users in Europe and other parts of the world [↑](#footnote-ref-3)
4. Wide Area (LPWA) networking protocol designed to wirelessly connect battery operated ‘things’ to the internet in regional, national or global networks [↑](#footnote-ref-4)
5. Industrial, Scientific and Medical [↑](#footnote-ref-5)
6. System on chip [↑](#footnote-ref-6)
7. Serial Peripheral Interface [↑](#footnote-ref-7)
8. a Latvian network equipment manufacturer [↑](#footnote-ref-8)