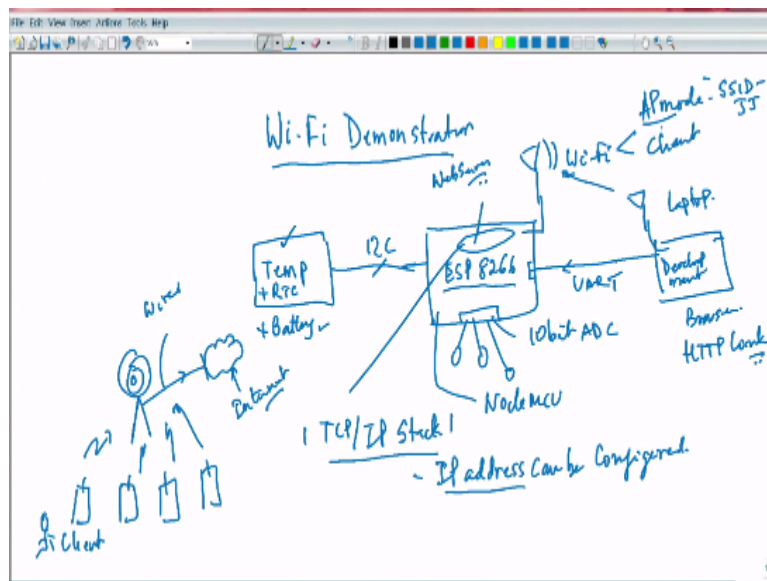


Design for Internet of Things
Prof. T V Prabhakar
Department of Electronic Systems Engineering
Indian Institute of Science-Bengaluru

Lecture - 46
Wi-Fi

Folks, let us do a demonstration of Wi Fi technology. I will show you an overview of what is out here. And then you can we can also look at the source code. So here is what the demonstration is. We have a module called 8266.

(Refer Slide Time: 00:42)



This is a very powerful I would say an SOC. It is actually you can look for node MCU. It actually integrates ESP8266 Wi-Fi module. You can see the Wi-Fi is here, right. It also has a 10 bit ADC. It has a UART port, which essentially you can use the UART port for your development purpose to dump the code onto the ESPs flash, you can use this UART module here.

Also in this demonstration is a RTC module which I will show you along with temperature which actually acts as a temperature sensor as well and that is connected over I2C bus. And since you are talking about an RTC, obviously it means that even if you power off you should be able to maintain the date and time and so on. And therefore there is a battery associated with it.

Now the 8266 module integration into node MCU is so powerful that it has a complete TCP/IP stack running on that system. Essentially, when you have TCP/IP stack, you can configure the IP address quite easily on it. And it will provide you a lot of services, okay. The Wi-Fi for instance, you can configure the Wi-Fi either in the access point mode, essentially you can associate it with a SSID.

In our case, we have given the name JJ. So you will see that JJ has one SSID that is given to it. Or essentially, if you put it in AP mode, several clients can connect to it. So I will show you a picture equivalently. This is let us say an AP access point that you may have seen in homes, in your homes or in any public places. There are many clients which can connect simultaneously.

And then there is a backhaul connection. This is a wired connection, usually, okay. And this might go to an internet gateway. Okay, this might usually go to an internet gateway. So this is internet, okay. So this clearly indicates, and these are all the clients. Let us say you are in a coffee shop, and there are several of you holding your mobile phones. And each one of you are browsing the internet.

So what is actually happening, you are connecting to the nearest wireless access point, maybe the name of the coffee shop will be what you see. And maybe there is password, maybe there is no password depending on how and how much security is required.

And then you start browsing on that right, which means each one of these clients are connected over to the internet as simple as acquiring an IP address from this AP connecting to it and then sending out data packets. So essentially all of that is possible easily with this kind of embedded module called node MCU.

Now if you talk about Wi-Fi, existence and AP existence, you also have to note that in the AP mode, the AP can actually also invoke applications which are running on 8266. Once such application could be the web server application, okay. You can also

run a web server application. In other words, the full web server application running on 8266 means you can kick up a browser here.

You can keep kick up a browser here, connect over Wi-Fi and access data of ESP8266 over a HTTP link, okay. You can acquire the data over a HTTP link. So that is the nice thing about it and what data are you acquiring? You could acquire the temperature data easily through this browser link I suppose. So that is definitely a possibility.

Now if you have to know from what we discussed in this module on Wi-Fi, the hardware, receiver sensitivity and all that you may have to know all that because if you are looking at design, you must understand those specifications very well.

(Refer Slide Time: 04:37)



Therefore let us look at the datasheet of ESP8266 which is part of the node MCU that you can buy, okay. And that is indeed this is the heart of the system here. So let us jump in and go into the details of ESP8266.

(Refer Slide Time: 04:53)

with external sensors and other devices through the GPIOs. Software Development Kit (SDK) provides sample codes for various applications.

Espressif Systems' Smart Connectivity Platform (ESP-CP) enables sophisticated features including:

- Fast switch between sleep and wakeup mode for energy-efficient purpose.
- Adaptive radio tuning for low-power operation.
- Advance signal processing.
- Spur cancellation and RF co-existence mechanisms for common cellular, Bluetooth, DDR, LVDS, LCD interference mitigation.

1.1. Wi-Fi Key Features

- 802.11 b/g/n support
- 802.11 n support (2.4 GHz), up to 72.2 Mbps
- Deinterleaving
- 2 x virtual Wi-Fi interface
- Automatic beacon monitoring (hardware TSP)
- Support Infrastructure BSS Station mode/SoftAP mode/Promiscuous mode

Espressif 1/25 2021.10
[Submit Documentation Feedback](#)

You can see that this 8266 can support 802.11 b/g/n. So all this support is very much available to it. And as we also discussed that it can do soft AP promiscuous mode and all that. What is interesting is the receiver sensitivity, okay?

(Refer Slide Time: 05:09)

1.2. Specifications

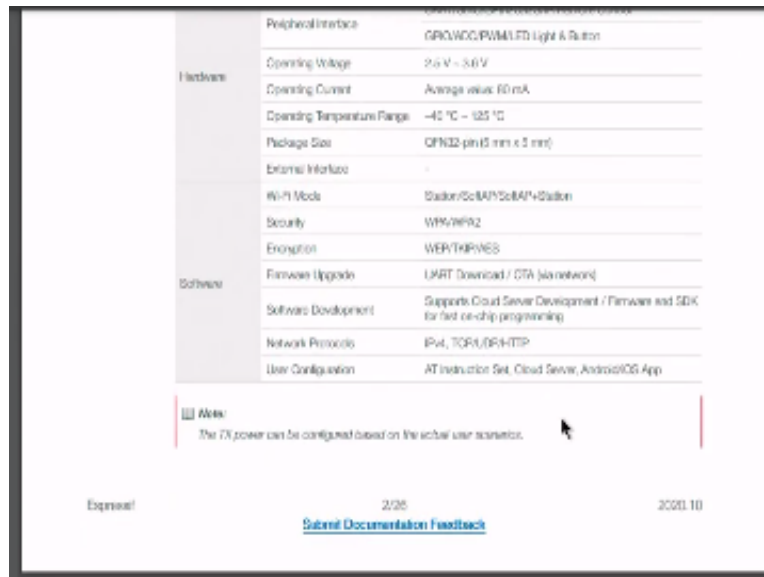
Table 1-1. Specifications

Categories	Items	Parameters	
Wi-Fi	Certification	Wi-Fi Alliance	
	Protocol	802.11 b/g/n (IEEE)	
	Frequency Range	2.4 GHz ~ 2.5 GHz (2400 MHz ~ 2483.5 MHz)	
	Tx Power	802.11 b	+20 dBm
		802.11 g	+17 dBm
		802.11 n	+14 dBm
Rx Sensitivity	802.11 b	-91 dBm (11 Mbps)	
	802.11 g	-75 dBm (54 Mbps)	
	802.11 n	-72 dBm (MCS7)	
Antenna	PCB Inset, External, IPEX Connector, Ceramic Chip		
Hardware	GPU	TeslaCo L109 32-bit processor	
	Peripheral Interface	I2C/SPI/UART/RS485/IR Remote Control	
	Operating Voltage	GPIO/WDT/PWM/LCD Light & Button	
		2.5 V ~ 3.6 V	
		Average value: 60 mA	
	Operating Temperature Range	-40 °C ~ 125 °C	
Package Size	QFN32-pin (3 mm x 3 mm)		
External Interface	-		

Look again very carefully. You can get -91 dBm if your data rate is 11 Mbps. But it is down to -75 dBm if it is 54 Mbps, okay. And it is down to -72. This is 3 dB difference. When if you choose another modulation scheme called the MCS7 modulation scheme. What that MCS7 indicates is another story. It is just very high data rate modulation scheme. But at that high data rate you have to get a compromise is it not?

The compromise is you lose 3 dB in the receiver sensitivity loss. That is essentially half the sensitivity, the sensitivity is down by half between MCS7 and the 54 Mbps links. So it takes a beating on that but that was obvious from whatever we discussed last time that Rx sensitivity is a parameter of interest to us and you have to looking at the data sheet to understand in detail about this particular parameter. And operating voltage is given. Then antenna is mentioned, okay.

(Refer Slide Time: 06:22)



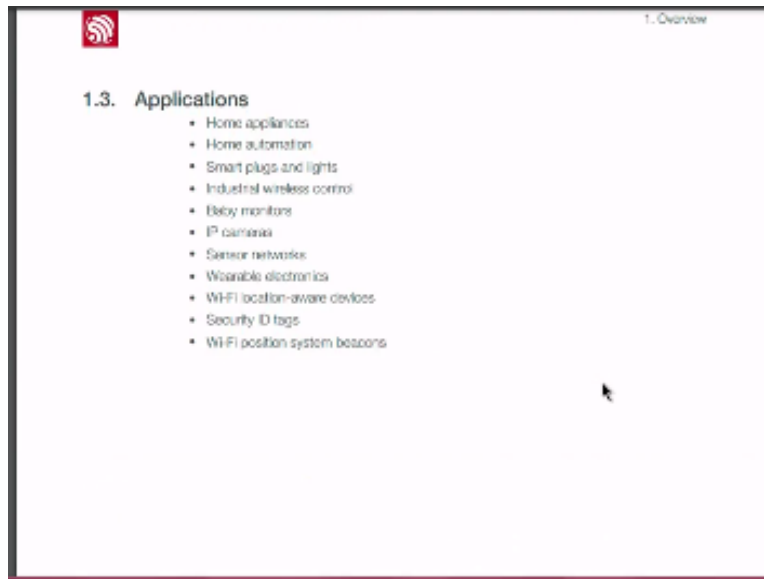
	Peripheral Interface	GPIO/ADC/PWM/LED Light & Button
Hardware	Operating Voltage	2.6 V ~ 3.6 V
	Operating Current	Average value: 60 mA
	Operating Temperature Range	-40 °C ~ 125 °C
	Package Size	QFN32 pin (5 mm x 5 mm)
	External Interface	-
Software	Wi-Fi Mode	Station/SoftAP/SoftAP+Station
	Security	WPA/WPA2
	Encryption	WEP/TKIP/AES
	Firmware Upgrade	UART Download / OTA (via network)
	Software Development	Supports Cloud Server Development / Firmware and SDK for fast on-chip programming
	Network Protocol	IPv4, TCP, UDP, HTTP
User Configuration	AT Instruction Set, Cloud Server, Android/iOS App	

Alert:
The TX power can be configured based on the actual user scenario.

Espressif 2/26 2020.10
[Subnet Documentation Feedback](#)

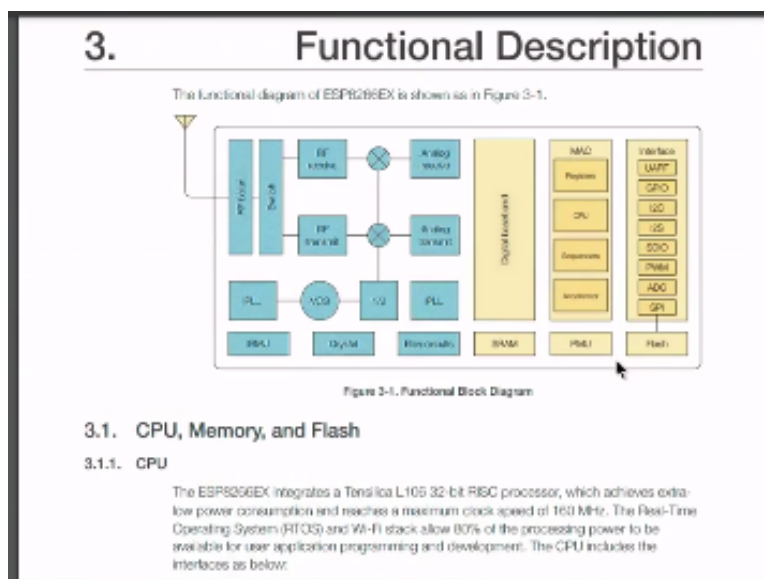
So here are the features which I was discussing. Station, you can configure it as SoftAP, SoftAP +Station also. So all these possibilities exist. Of course, security is very much part of it and it is bundled into it. Protocol support IPv4, TCP, UDP and HTTP. We spoke about browsers kicking up and asking data from temperature sensors via the HTTP server. And that is something that is easily doable with ESP 8266.

(Refer Slide Time: 06:50)



What are the typical applications? Well, there is a huge number of them. And in our lab, we have used ESP 8266 for energy monitoring applications. So that is particularly single phase energy monitoring applications. Here is a footprint of how the chip looks. And here are the pin definitions of the chip.

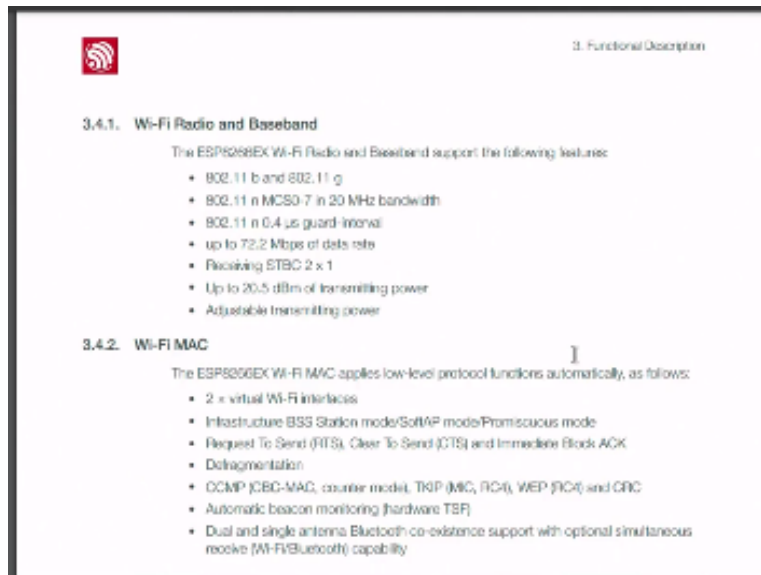
(Refer Slide Time: 07:11)



And here is a picture of the functional diagram, block diagram of ESP8266. What you see on the extreme left is all the RF part. Middle is the baseband part. And the extreme right is the MAC, PHY and all the registers which are required for configuring the, you know not just the CPU, but also the baseband and other RF related parameters. Of course, it has a bunch of interface pins. UART, GPIO, I2C and so on.

So all these are very much doable. ADC is also there. And all these are very much available for interfacing different type of sensors. ADC, yes of course, there is a 10 bit I mentioned. SPI is there and I2C is also there. If you read this document, you will clearly understand that all the parameters are easily listed there. And is useful for you to read them as thoroughly as possible.

(Refer Slide Time: 08:06)



The slide is titled "3. Functional Description" and features a red Wi-Fi icon in the top left corner. It is divided into two sections: 3.4.1. Wi-Fi Radio and Baseband and 3.4.2. Wi-Fi MAC. Section 3.4.1 lists features of the ESP8266EX Wi-Fi Radio and Baseband, including 802.11 b and g, MCS-7 in 20 MHz bandwidth, a 0.4 us guard interval, up to 72.2 Mbps data rate, 2x1 STBC, and up to 20.5 dBm power. Section 3.4.2 lists features of the ESP8266EX Wi-Fi MAC, including 2 virtual interfaces, various modes (Infrastructure BSS Station, SoftAP, Promiscuous), RTS/CTS and ACK, fragmentation, CCMP/TKIP/WEP/GCRC, beacon monitoring, and Bluetooth co-existence.

3. Functional Description

3.4.1. Wi-Fi Radio and Baseband

The ESP8266EX Wi-Fi Radio and Baseband support the following features:

- 802.11 b and 802.11 g
- 802.11 n MCS0-7 in 20 MHz bandwidth
- 802.11 n 0.4 us guard interval
- up to 72.2 Mbps of data rate
- Receiving STBC 2 x 1
- Up to 20.5 dBm of transmitting power
- Adjustable transmitting power

3.4.2. Wi-Fi MAC

The ESP8266EX Wi-Fi MAC applies low-level protocol functions automatically, as follows:

- 2 x virtual Wi-Fi interfaces
- Infrastructure BSS Station mode/SoftAP mode/Promiscuous mode
- Request To Send (RTS), Clear To Send (CTS) and Immediate Block ACK
- Defragmentation
- CCMP (CBC-MAC, counter mode), TKIP (MIC, RC4), WEP (RC4) and CRC
- Automatic beacon monitoring (hardware TSF)
- Dual and single antenna Bluetooth co-existence support with optional simultaneous receive (Wi-Fi/Bluetooth) capability

So there are details about the Wi-Fi, radio, and baseband. There are details about the MAC. You did see the MCS7 out there last time, but now you see them in detail that it is in the 20 megahertz bandwidth. So essentially giving you much higher data rates okay as compared to the normal schemes which are indicated. So this is a special data rate MCS modulation coding scheme which ESP supports for high data rate applications.

(Refer Slide Time: 08:39)

- Automatic beacon monitoring (hardware TSM)
- Dual and single antenna Bluetooth co-existence support with optional simultaneous receive (Wi-Fi/Bluetooth) capability

3.5. Power Management

ESP8266EX is designed with advanced power management technologies and intended for mobile devices, wearable electronics and the Internet of Things applications.

The low-power architecture operates in the following modes:

- Active mode: The chip radio is powered on. The chip can receive, transmit, or listen.
- Modem-sleep mode: The CPU is operational. The Wi-Fi and radio are disabled.
- Light-sleep mode: The CPU and all peripherals are paused. Any wake-up events (MAC, host, RTC timer, or external interrupts) will wake up the chip.
- Deep-sleep mode: Only the RTC is operational and all other part of the chip are powered off.

Espressif 11/26 2020.10
[Submit Documentation Feedback](#)

There is a huge discussion on the power management which is the heart of this course. So therefore I would strongly encourage you to read up on power management section of this article. You have active mode, modem-sleep mode, light-sleep mode, deep-sleep mode and so on. So all these modes are indicated.

(Refer Slide Time: 08:56)

3. Functional Description

Table 3-4. Power Consumption by Power Modes

Power Mode	Description	Power Consumption
Active (RF working)	Wi-Fi TX packet Wi-Fi RX packet	Please refer to Table 5-2.
Modem-sleep ^①	CPU is working	15 mA
Light-sleep ^②	-	0.8 mA
Deep-sleep ^③	Only RTC is working	50 uA
Shut-down	-	0.5 uA

Notes:

- ① **Modem-sleep** mode is used in the applications that require the CPU to be working, as in PWM or I2C applications. According to IEEE 802.11 standards (802.11-2016), it shuts down the Wi-Fi Modem circuit while maintaining a Wi-Fi connection with no data transmission to optimize power consumption. E.g., in DTIM3, maintaining a sleep of 300 ms with a wakeup of 2 ms cycle to receive AP's Beacon packages at interval requires about 15 mA current.
- ② During **Light-sleep** mode, the CPU may be suspended in applications like Wi-Fi switch. Without data transmission, the Wi-Fi Modem circuit can be turned off and CPU suspended to save power consumption according to the IEEE 802.11 standards (802.11-2016). E.g., in DTIM3, maintaining a sleep of 300 ms with a wakeup of 3 ms to receive AP's Beacon packages at interval requires about 0.5 mA current.
- ③ During **Deep-sleep** mode, Wi-Fi is turned off. For applications with long time intervals between sleep

The corresponding current consumption is also indicated. So look them up in great detail, understand that in great detail so that you really know that several levers can be pulled so that battery life of this NodeMCU which integrates ESP8266 really survives for the 10 plus years kind of lifetime we are looking at. Well at least the 10 plus years is indeed our goal. Whether it is doable is another story.

But it is always good to think about something which is long in what you want to achieve. Alright, these are pin definitions for SDIOs and so on. I2C is also mentioned. I gave you this idea already PWM. ADC is here.

(Refer Slide Time: 09:40)

4.9. ADC (Analog-to-Digital Converter)

ESP102/2/3/EX is embedded with a 10-bit precision SAR ADC. TOUT (Pin6) is defined as below:

Table 4-9. Pin Definition of ADC

Pin Name	Pin Num	Function Name
TOUT	6	ADC Interface

The following two measurements can be implemented using ADC (Pin6). However, they cannot be implemented at the same time.

- Measure the power supply voltage of VDDGPS (Pin3 and Pin4).

Hardware Design	TOUT must be floating.
RF Initialization Parameter	The 10th byte of esp_unity_data_default (bit 0 - 127) before vldfs_load must be set to 0x0F.
RF Calibration Process	Optimize the RF circuit conditions based on the testing results of VDDGPS (Pin3 and Pin4).
User Programming	Use system_get_vddfs instead of system_adc_read.

- Measure the input voltage of TOUT (Pin6).

Espressif 17/26 2020.10
[Submit Documentation Feedback](#)

So you can see the read the ADC in great detail here. It does talk to you about the 10-bit precision SAR ADC and some definitions are out there, okay. Then the electrical specifications are also given along with this.

(Refer Slide Time: 09:57)

D	The de-assert of EXT_RSTB	0	2	ms
E	EXT_RSTB goes high after VDD20	0.1	-	ms
F	The de-assert of CHIP_EN	0	2	ms
G	CHIP_EN goes high after EXT_RSTB	0.1	-	ms

5.2. RF Power Consumption

Unless otherwise specified, the power consumption measurements are taken with a 3.0 V supply at 25 °C of ambient temperature. All transmitters' measurements are based on a 50% duty cycle.

Table 5-3. Power Consumption

Parameters	Min	Typical	Max	Unit
TX22: 11 b, CQPSK 11 Mbps, Pout = +17 dBm	-	170	-	mA
TX22: 11 b, DQPSK 540 kbps, Pout = +15 dBm	-	140	-	mA
TX22: 11 b, MCS7, Pout = +15 dBm	-	120	-	mA
Rx22: 11 b, 1024 bytes packet length, -80 dBm	-	60	-	mA
Rx22: 11 b, 1024 bytes packet length, -70 dBm	-	60	-	mA
Rx22: 11 b, 1024 bytes packet length, -65 dBm	-	60	-	mA

And power consumption, RF power basically. Consumption for different data rates and all that is mentioned. You can see that it is not in the order of even a few, 1 or 2 milliamperes. It is in 50, 100 milliamperes or 120 milliamperes kind of range, which is

quite an amount of power compared to other Bluetooth and ZigBee kind of or 802.15.4 kind of radios. So that is a point to be noted there. And then there are characteristics of the Wi-Fi radio as well.

(Refer Slide Time: 10:28)

5.3. Wi-Fi Radio Characteristics

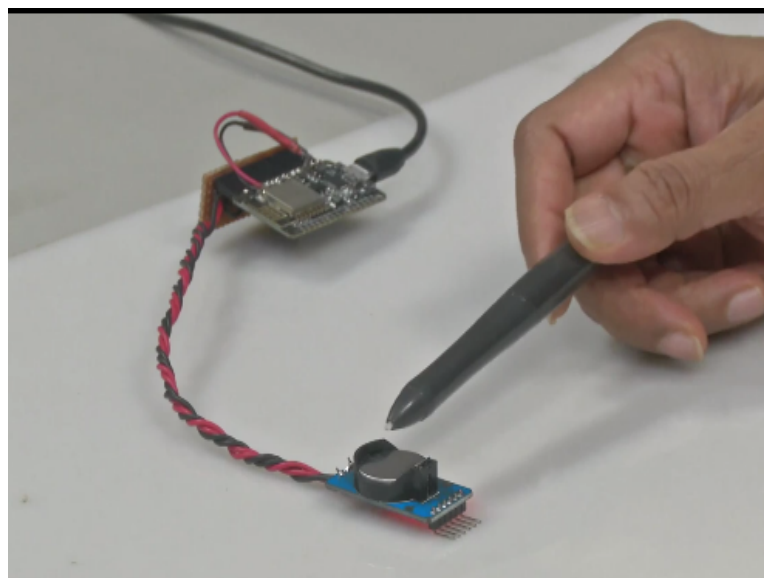
The following data are from tests conducted at room temperature, with a 3.3 V power supply

Table 6-3. Wi-Fi Radio Characteristics

Parameters	Min	Typical	Max	Unit
Input frequency	2412	-	2484	MHz
Output impedance	-	$50 + j0$	-	Ω
Output power of PA for 72.2 Mbps	19.5	18.5	17.5	dBm
Output power of PA for 11b mode	19.5	20.5	21.5	dBm
Sensitivity				
DSSS, 1 Mbps	-	-90	-	dBm
CSMA, 11 Mbps	-	-91	-	dBm
6 Mbps (1/2 BPSK)	-	-93	-	dBm
54 Mbps (2/4 64-QAM)	-	-78	-	dBm
HT20, MCS7 (3/4 64-QAM, 72.2 Mbps)	-	-72	-	dBm
Adjacent Channel Rejection				
OFDM, 6 Mbps	-	37	-	dB
OFDM, 54 Mbps	-	21	-	dB
HT20, MCS9	-	37	-	dB
HT20, MCS7	-	20	-	dB

The impedances are mentioned here so that if you have to match with the external antenna, this might be of some help. Output of power amplifier, is mentioned. The modulations, the sensitivity is mentioned, and so on. So that is about what this ESP8266 is. And do spend time reading about it so that you get some familiarity before you actually start using it. Now let us switch to the demo. What I show you here in the demonstration is essentially two parts here.

(Refer Slide Time: 10:58)

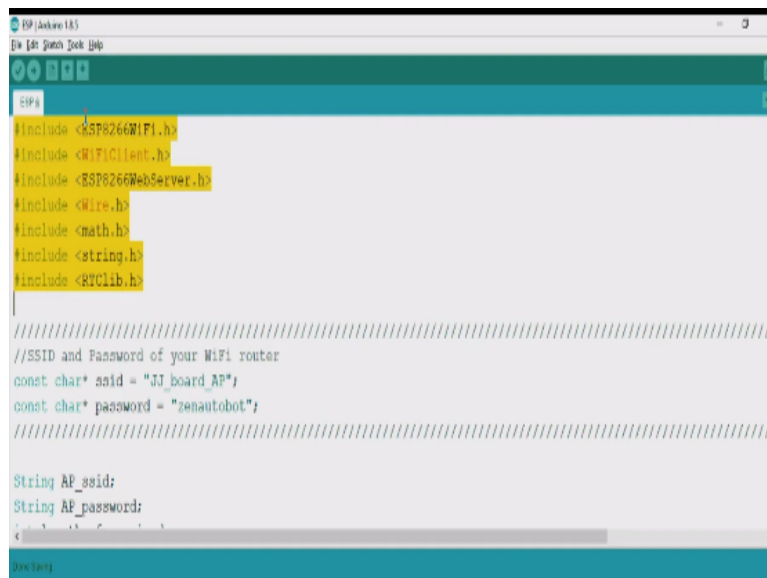


So if you look back at what I had mentioned to you, there is a temperature sensor and an RTC module, right. And now that equivalently is shown here, in this picture here. Okay, this is that module, alright? Now you have the NodeMCU, which is shown here on this screen here. This is the NodeMCU and equivalently the NodeMCU is shown here, you can see in this picture, okay.

You can see that this indeed is the NodeMCU here, which has the 8266 ESP system, which is actually the CPU is from Tensilica. And all the features are all out there; Wi-Fi, then flash, then all the configurations that you need to do for UART, for dumping the code, and so on, everything is out there. So this is that module.

What you see here, this cable is indeed the UART cable, which is quite similar to what I have shown you here in this picture. This is the UART cable, which essentially be used for configuration. And also for let us say you want to display something that, because ESP8266 is really headless in that sense. So if you want to have a screen, you may want to also use the UART for providing the screen capability for it. Now let us switch to the software part of the code.

(Refer Slide Time: 12:13)



```
ESP8266
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <Wire.h>
#include <math.h>
#include <string.h>
#include <RTClib.h>

//SSID and Password of your WiFi router
const char* ssid = "JI_board_AP";
const char* password = "zenautobot";

String AP_ssid;
String AP_password;
```

What essentially we will show you is the different libraries, which are there. And you can see that the list of libraries are shown out there. Let him just highlight it, and then you will see. So this is demonstration from Pratyush. Pratyush Shukla is from the lab.

So he is showing you a demonstration of this. So you can see that these are the main libraries, which are part of this code.

And in fact, this code will be available to you. He is kind enough to release this code, so you can use it. If you have a NodeMCU with you, you can actually use this as a starting point, okay. Now you see that there is SSID and password of your Wi-Fi router. You could mention it there. You can see that the SSID chosen is JJ board underscore AP. Okay, so here is something that you can imagine, let me draw this picture.

Here is an AP, right? Sometimes you might have seen your college SSID appearing there. Quite like that this is the AP part of the ESP8266 and its SSID. So if you come close to this ESP8266 module, you will see on your mobile phone, you will see this AP showing up that it can allow you to connect okay, so that is the nice thing about it. All right, then you have other parts, essentially going to the I2C protocol part, which is shown here.

(Refer Slide Time: 13:41)



```
void Time_Stamp()
{
  Wire.beginTransmission(0x68);           // Start I2C protocol with 0x68 address
  Wire.write(0);                          // Send register address
  Wire.endTransmission(false);            // I2C restart
  Wire.requestFrom(0x68, 7);              // Request 7 bytes from 0x68 and release I2C bus at end of read
  second = Wire.read();                    // Read seconds from register 0
  minute = Wire.read();                    // Read minutes from register 1
  hour = Wire.read();                      // Read hour from register 2
  day = Wire.read();                       // Read day from register 3
  date = Wire.read();                      // Read date from register 4
  month = Wire.read();                     // Read month from register 5
  year = Wire.read();                      // Read year from register 6
}
```

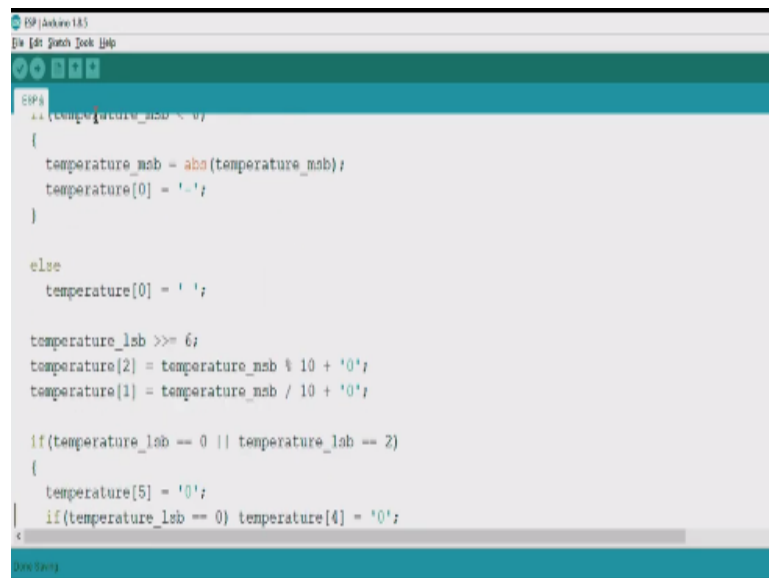
The I2C is required because you are actually acquiring the temperature, right? So what is the demo here that we have? The demo is you acquire the temperature over I2C bus. And then you should be able to, you know broadcast that or transmit that

data using Wi-Fi onto a mobile phone. So that will tell you that this module can easily communicate to mobile phone.

When it can communicate to mobile phone it can connect to any Wi-Fi compatible device to transfer its data. So the demo is what? Demo is acquire temperature data, pass it over I2C bus, give it to NodeMCU/8266 module. And that module in turn, should be able to send its data out to nearby mobile phone, okay. And that you should be able to read that data on the mobile phone.

That is essentially the demonstration here. Okay, so you can see that several parts of the code are shown here.

(Refer Slide Time: 14:44)



```
ESP8266
File Edit Search Tools Help
ESP8266
// (temperature_msb <= 0)
{
  temperature_msb = abs(temperature_msb);
  temperature[0] = '-';
}

else
  temperature[0] = '+';

temperature_lsb >>= 6;
temperature[2] = temperature_msb % 10 + '0';
temperature[1] = temperature_msb / 10 + '0';

if(temperature_lsb == 0 || temperature_lsb == 2)
{
  temperature[5] = '0';
  if(temperature_lsb == 0) temperature[4] = '0';
}
```

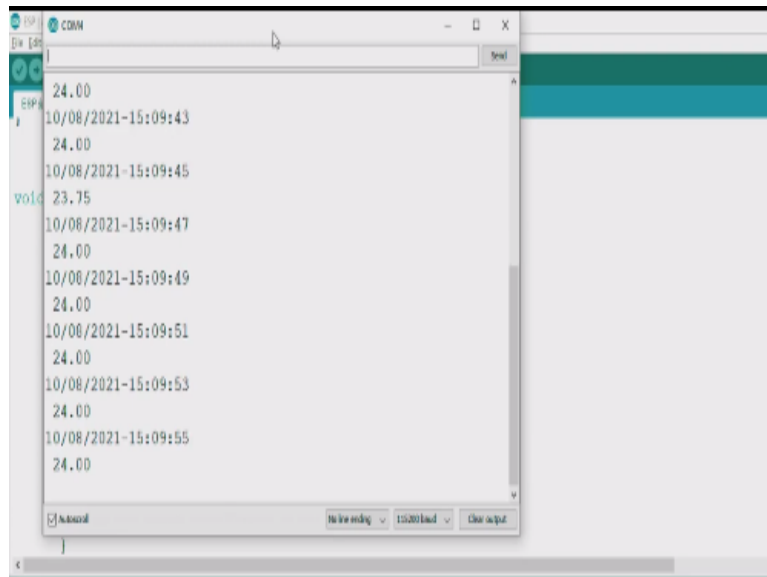
So there you are, the acquisition of temperature is happening there. He is acquiring the data. And then he is perhaps going down and he takes the data, puts it into I2C bus and then transmits it and there is a static IP that he has assigned to this NodeMCU. It is a IPv4 address, I suppose. So that he knows that it is the, that is the IP of this machine, of the machine that he is using.

And yeah, so there you are, Wi Fi and so on. And then if connection successful show IP address in serial monitor. So you can check a few things, simple things like whether the IP address is configured correctly, whether it has given you something

that is not clashing with other IP addresses in the same Wi-Fi network, and so on. All these tests are done. Then you update the HTML page that is hosted on the static IP, essentially.

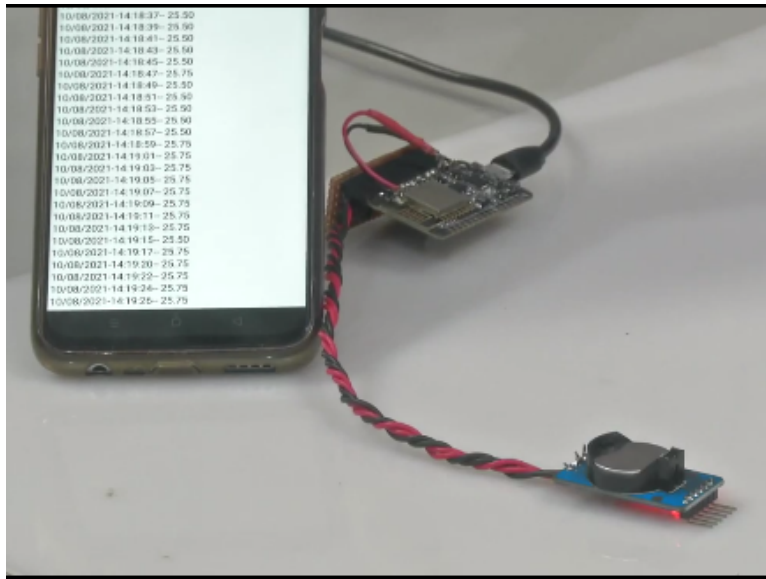
So you will actually see whether the IP address is correct and whether it is configured properly without any clashes and so on. Yeah, then you moved on.

(Refer Slide Time: 15:51)



Now there you see, you see that it is now basically displaying the date. It is displaying the time, and it is also displaying the temperature value. And this is now being shown on the laptop, by the way, which means you are actually displaying it, we are displaying it via the UART. Let us see whether you will be able to get these values also on the mobile phone. So let me show you that.

(Refer Slide Time: 16:18)

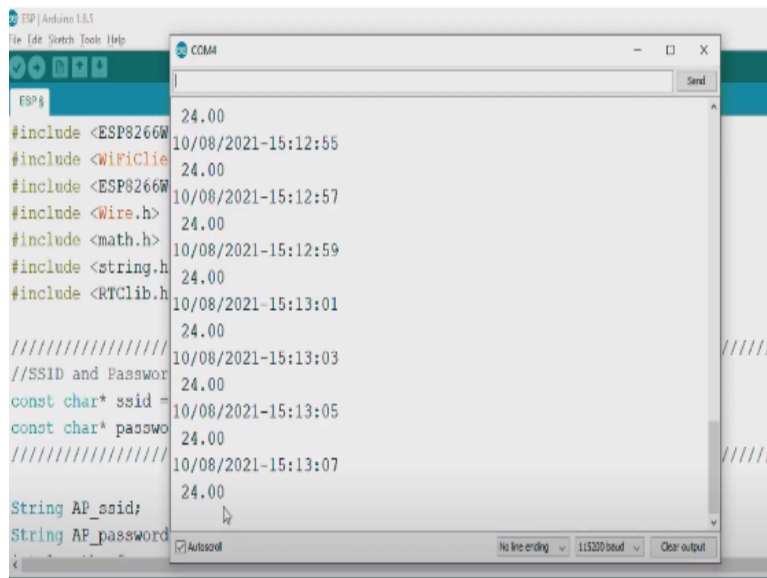


You see now it is showing you the date, time. And also it is giving you the values. So what is the takeaway from this little small demonstration? That very easily, you can interface simple systems. And you can simple sensors, acquire data and transmit that data over Wi-Fi to your mobile phone.

See folks, while it looks very simple as a demonstration, do not forget that if you have a home Wi-Fi router, this module can directly upload the sensor values directly to the cloud, okay. And since it has TCP/IP stack, you can also load other IoT protocols like MQTT or COAP and get data uploaded directly. After all, MQTT requires TCP, right? And this is running a full TCP/IP stack.

Therefore, it is a very powerful way of you know acquiring data and transmitting data. And Wi-Fi is so common everywhere, that with low power capabilities, which I showed you in the datasheet and with the simple configuration that you can do with the system, you should be able to easily get going with respect to at least a simple sensor like temperature sensor to be acquired successfully.

(Refer Slide Time: 17:40)



```
ESP: 24.00
#include <ESP8266W 10/08/2021-15:12:55
#include <WiFiClie 24.00
#include <ESP8266W 10/08/2021-15:12:57
#include <Wire.h> 24.00
#include <math.h> 10/08/2021-15:12:59
#include <string.h 24.00
#include <RTClib.h 10/08/2021-15:13:01
24.00
//////////////////// 10/08/2021-15:13:03
//SSID and Passwor 24.00
const char* ssid = 10/08/2021-15:13:05
const char* passwo 24.00
//////////////////// 10/08/2021-15:13:07
24.00
String AP_ssid;
String AP_password
```

So now let us see how this whole demonstrable system actually works. Because we have to show you something that is going live. At the moment, this module that you see here, is actually reading 24 degrees. So what we will do is we will start heating this module, this temperature, bring some hot air very close to the module. You can see that it has jumped to 32.5 degrees already, right?

(Refer Slide Time: 18:05)



This is a hot air gun. And what we did was we started heating the module using this hot air gun. This one, this is the hot air gun, okay. So we can see this demo again. We will use this hot air gun, it is already pretty hot here. Go close to the IC, where the temperature sensor is there. And that already will directly display the values to you. That is about what we wanted to show you. Thank you very much.

