

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

**Lecture - 41**  
**Bluetooth Low Energy (BLE) – 01**

Alright folks, we discussed about receiver sensitivity of general you know receiver chain and a transmit chain. I suppose now that is sufficient for you in this course to decide how to choose a radio for your application, particularly when you are looking at data sheets of SoC manufacturers. One such SoC manufacturer which again I would like to point you to is NRF52840.

**(Refer Slide Time: 01:01)**

**Feature list**

**Features:**

- Bluetooth<sup>®</sup> 5, IEEE 802.15.4-2006, 2.4 GHz transceiver
- -95 dBm sensitivity in 1 Mbps Bluetooth<sup>®</sup> low energy mode
- -103 dBm sensitivity in 125 kbps Bluetooth<sup>®</sup> low energy mode (long range)
- -20 to +8 dBm TX power, configurable in 4 dB steps
- On-air compatible with NRF52, NRF51, NRF24L, and NRF24AP Series
- Supported data rates:
  - Bluetooth<sup>®</sup> 5: 2 Mbps, 1 Mbps, 500 kbps, and 125 kbps
  - IEEE 802.15.4-2006: 250 kbps
  - Proprietary 2.4 GHz: 2 Mbps, 1 Mbps
- Single-ended antenna output (on-chip balun)
- 128-bit AES/ECB/CCM/MAC co-processor (on-the-fly packet encryption)
- 4.8 mA peak current in TX (0 dBm)
- 4.6 mA peak current in RX
- RSSI (1 dB resolution)
- ARM<sup>®</sup> Cortex<sup>®</sup>-M4 32-bit processor with FPU, 64 MHz
- Flexible power management
- 1.7 V to 5.5 V supply voltage range
- On-chip DC/DC and LDO regulators with automated low current modes
- 1.8 V to 3.3 V regulated supply for external components
- Automated peripheral power management
- Fast wake-up using 64 MHz internal oscillator
- 0.4 µA at 1 V in System OFF mode, no RAM retention
- 1.5 µA at 3 V in System ON mode, no RAM retention, wake on RTC
- 1 MB flash and 256 kB RAM
- Advanced on-chip interfaces
  - USB 2.0 full speed (12 Mbps) controller
  - QSPI 32 MHz interface
  - High-speed 32 MHz SPI

This is Nordic radio, radio SoC. Now let us just look at the radio part of the SoC. What does it say? It supports Bluetooth 5 that clearly indicates there must have been a Bluetooth 4. There must have been an earlier version of Bluetooth as well, right? And it also says that look, I am not stopping at just supporting Bluetooth 5, I can also support IEEE 802.15.4 2006 version.

That means, there must have been a version which is 15.4 2000 something else, greater than that. There is indeed a version which is 2011. There is a 2012 version and so on. So which clearly indicates if you come to the area of radios, you have to be

very clear about which version you are looking at, okay. So you can see that that is true for both types of technologies which it supports.

Now what is the beauty of this? Broadly, this is Bluetooth and this is ZigBee. IEEE802.15.4 - 2006 is the MAC and PHY medium access control and physical layer, right. MAC and PHY of the OSI stack which is exploited by a consortium called ZigBee. ZigBee uses this particular IEEE 802.15.4 – 2006 MAC and PHY layer. That is what it means. You can also develop native applications on top of this MAC and PHY.

But ZigBee is one such you know standard which is a consortium of people, of consortium of companies which wanted to use this and they have defined certain things as part of their upper layer stack. Now beauty is this same SoC can be configured for either Bluetooth or for ZigBee in a way. Why? Because remember I mentioned to you about the generic RX chain and the generic receive chain and the receiver sensitivity which I was talking to you about.

Now you have to know a little more about this particular radio. So look at what he says. -95 dBm is the sensitivity in 1 Mbps. It is greater than that. It goes down to 103 dBm sensitivity if you reduce the data rate, right? So 125 kbps gives you 103 dBm. What is the transmit power of this radio? -20 to 8 dBm is the transmit power in steps of 4 dB.

Now clearly, this Bluetooth 5 supports, what all data rate it supports is mentioned inside this SoC. Gives you 2 Mbps, 1 Mbps 500 and 125 kbps. And clearly sensitivity is changing based on the data rate. Now you can evaluate. You have now seen Nordic. Go and look at some TI SoC or look at NXP SoC or look up STM SoC and find out how these compare and what your application is, what kind of range you want to support.

Everything will depend on what is available from the radio, right? And in this complete course I never really pushed you very hard towards security. But folks, it is

actually integrated, okay. And you need to understand the security aspect all by yourself independent of what I teach in this course because everything that I teach here assumes that there is security and indeed there is.

You can see it supports 128 AES elliptic key, ECB security, CCM and so on, right? So all of that is possible well within the security module or security specification of the system. So that is possible also. So now you have the energy aspects which are indicated here. We will not get into that because we did that already in the other modules there, okay.

**(Refer Slide Time: 05:10)**

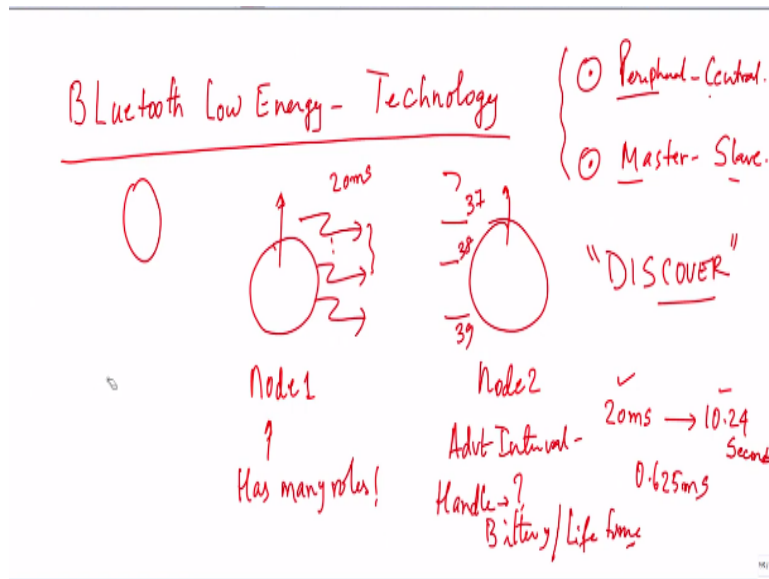
## Feature list

**Features:**

- Bluetooth<sup>®</sup> 5, IEEE 802.15.4-2006, 2.4 GHz transceiver
- -95 dBm sensitivity in 1 Mbps Bluetooth<sup>®</sup> low energy mode
- -103 dBm sensitivity in 125 kbps Bluetooth<sup>®</sup> low energy mode (long range)
- -20 to +8 dBm TX power; configurable in 4 dB steps
- On-air compatible with nRF52, nRF51, nRF24L, and nRF24AP Series
- Supported data rates:
  - Bluetooth<sup>®</sup> 5: 3 Mbps, 1 Mbps, 500 kbps, and 125 kbps
  - IEEE 802.15.4-2006: 250 kbps
  - Proprietary 2.4 GHz: 2 Mbps, 1 Mbps
- Single-ended antenna output (on-chip balun)
- 128-bit AES/ECB/CCM/AAR co-processor (on-the-fly packet encryption)
- 4.8 mA peak current in TX (0 dBm)
- Flexible power management
  - 1.7 V to 5.5 V supply voltage range
  - On-chip DC/DC and LDO regulators with automated low current modes
  - 1.8 V to 3.3 V regulated supply for external components
  - Automated peripheral power management
  - Fast wake-up using 64 MHz internal oscillator
  - 0.4 µA at 3 V in System OFF mode, no RAM retention
  - 1.5 µA at 3 V in System ON mode, no RAM retention, wake on RTC
  - 1 MB flash and 256 kB RAM
  - Advanced on-chip interfaces
  - USB 2.0 Full-speed (12 Mbps) controller

So connecting back these are the key important parameters that you may want to look up. Before we delve into anything related to Bluetooth, you must understand a few things about the Bluetooth protocol itself, okay.

**(Refer Slide Time: 05:22)**



So let us go and spend a little time about Bluetooth Low Energy as a technology. Now think of two nodes, okay. This is node 1 and this is node 2, okay. And now how these two nodes are communicating indeed depends on several you know terminologies which are part of this Bluetooth Low Energy. You should know those terminologies very well. Of prime importance to you will be these two terminologies.

What is a peripheral, and what is a central? Who is a master and who is a slave? If you know these things well, you can read the document, which are associated with Bluetooth, in a much more detailed, in a detailed way, okay. So I will get into some detailing here. And for that, I will take the support and assistance of a few files, which you can also download very easily, okay.

**(Refer Slide Time: 06:29)**

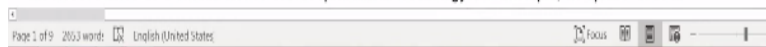
---

## Introduction to BLE

we'll be going over the definition of BLE, a little history, and how it compares to the original Bluetooth: Bluetooth Classic (*also known as BR/EDR*).

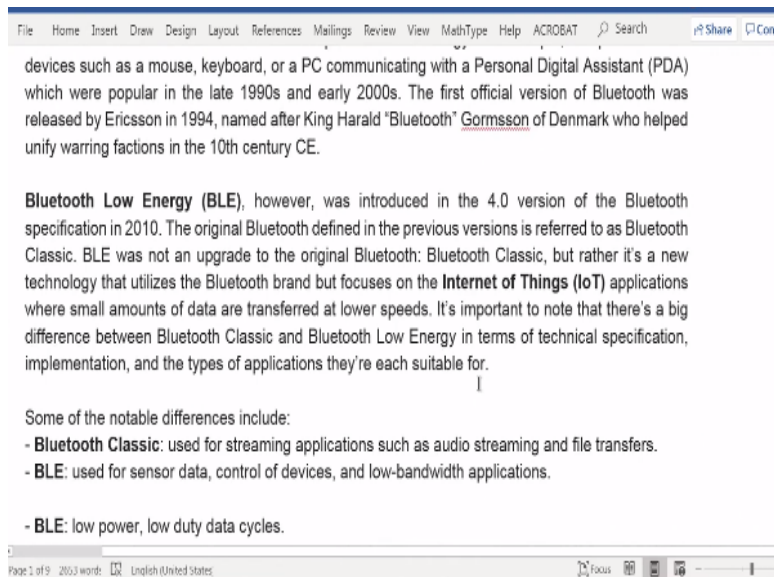
**Bluetooth Low Energy (BLE)** is a low power wireless technology used for connecting devices with each other. BLE operates in the **2.4 GHz ISM (Industrial, Scientific, and Medical)** band, and is targeted towards applications that need to consume less power and may need to run on batteries for longer periods of time—months, and even years.

Bluetooth started as a short-distance cable replacement technology. For example, to replace wires in



So for that, let me start with this document, which is actually a collection of some terms and names which we put together in this document. If required, we can share it with you. But I am sure you will find much superior documents out there, if you just Google for it. In fact, it is quite rich, the internet is quite rich in several documents. So you can download.

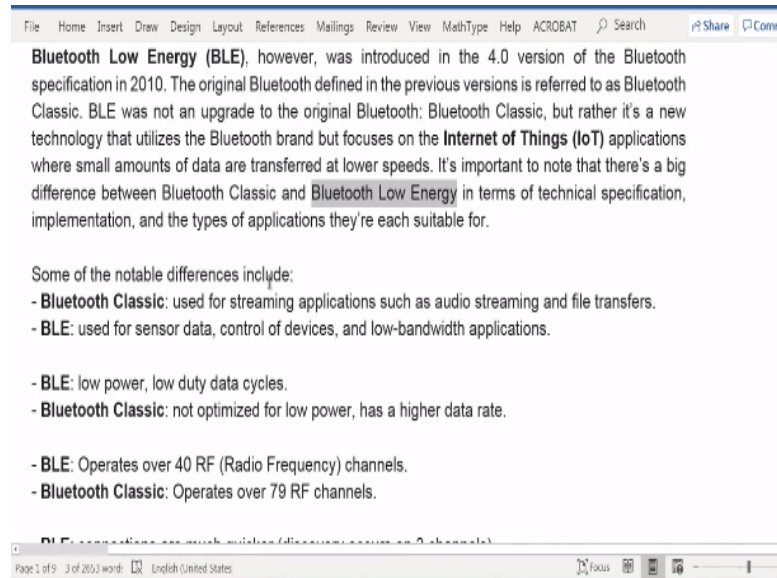
**(Refer Slide Time: 06:54)**



But let me get into some detailing here. Now I mentioned to you Bluetooth 5, which means Bluetooth 4 must have been there. And that is what is shown here. And that is as old as 2010, which is quite some more than a decade back, right. And then there are some names associated. There is something called Bluetooth Classic, and so on. So do not worry so much about these Bluetooth Classic and Bluetooth Low Energy and all

that. These are all terms which have evolved. But as engineers, as people who are using, you should know some nice things.

**(Refer Slide Time: 07:30)**



The screenshot shows a presentation slide with a menu bar at the top (File, Home, Insert, Draw, Design, Layout, References, Mailings, Review, View, MathType, Help, ACROBAT, Search, Share, Comment). The main text reads: "Bluetooth Low Energy (BLE), however, was introduced in the 4.0 version of the Bluetooth specification in 2010. The original Bluetooth defined in the previous versions is referred to as Bluetooth Classic. BLE was not an upgrade to the original Bluetooth: Bluetooth Classic, but rather it's a new technology that utilizes the Bluetooth brand but focuses on the Internet of Things (IoT) applications where small amounts of data are transferred at lower speeds. It's important to note that there's a big difference between Bluetooth Classic and Bluetooth Low Energy in terms of technical specification, implementation, and the types of applications they're each suitable for."

Some of the notable differences include:

- **Bluetooth Classic:** used for streaming applications such as audio streaming and file transfers.
- **BLE:** used for sensor data, control of devices, and low-bandwidth applications.

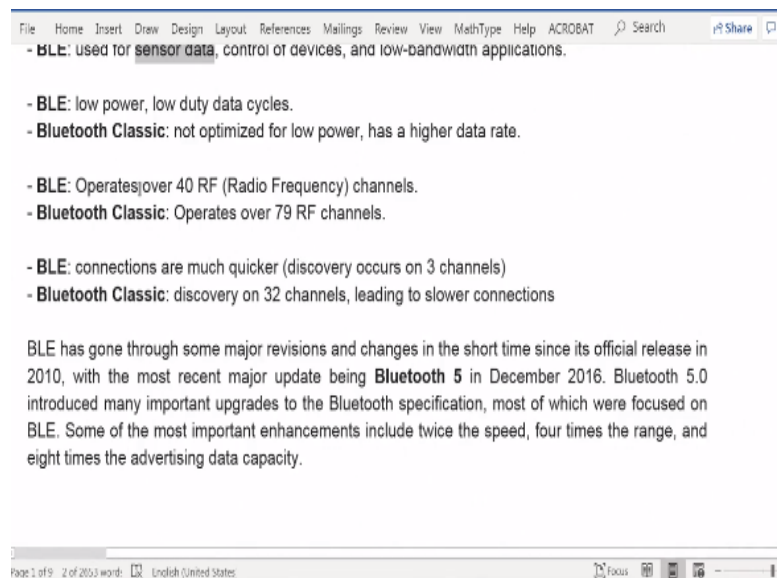
Additional differences listed below:

- **BLE:** low power, low duty data cycles.
- **Bluetooth Classic:** not optimized for low power, has a higher data rate.
- **BLE:** Operates over 40 RF (Radio Frequency) channels.
- **Bluetooth Classic:** Operates over 79 RF channels.

At the bottom, there is a search bar with the text "BLE connections are much quicker (discovery occurs on 3 channels)". The status bar at the bottom indicates "Page 1 of 9", "2 of 2013 words", and "English (United States)".

So if you look at Classic it is used for some streaming applications, audio streaming and file transfers. And Bluetooth Low Energy is what is of great interest to us because that is where sensor data communication comes in, in a big way. Control comes in a big way. And low bandwidth applications come in a big way, right?

**(Refer Slide Time: 07:50)**



The screenshot shows a presentation slide with a menu bar at the top (File, Home, Insert, Draw, Design, Layout, References, Mailings, Review, View, MathType, Help, ACROBAT, Search, Share, Comment). The main text reads: "- **BLE:** used for sensor data, control of devices, and low-bandwidth applications."

Additional differences listed below:

- **BLE:** low power, low duty data cycles.
- **Bluetooth Classic:** not optimized for low power, has a higher data rate.
- **BLE:** Operates over 40 RF (Radio Frequency) channels.
- **Bluetooth Classic:** Operates over 79 RF channels.
- **BLE:** connections are much quicker (discovery occurs on 3 channels)
- **Bluetooth Classic:** discovery on 32 channels, leading to slower connections

BLE has gone through some major revisions and changes in the short time since its official release in 2010, with the most recent major update being **Bluetooth 5** in December 2016. Bluetooth 5.0 introduced many important upgrades to the Bluetooth specification, most of which were focused on BLE. Some of the most important enhancements include twice the speed, four times the range, and eight times the advertising data capacity.

At the bottom, there is a search bar with the text "BLE connections are much quicker (discovery occurs on 3 channels)". The status bar at the bottom indicates "Page 1 of 9", "2 of 2013 words", and "English (United States)".

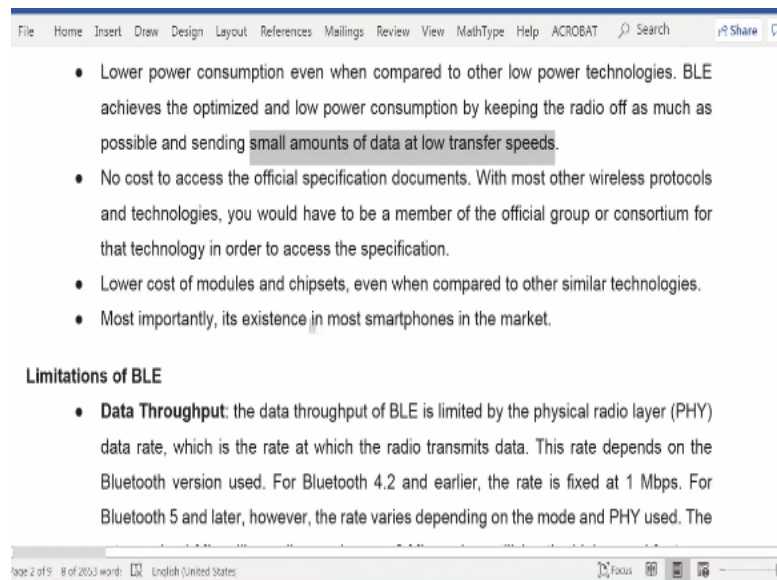
So although there are some differences of that nature, what comes to my mind is if you look at BLE, it offers you 40 RF channels as against in the classic which was 79 RF channels. Now folks, it is not difficult for you to break your head on how this is

becoming, you know so many channels here, at what compromise. Remember that the whole range is the ISM band, which we defined in the 2.4 gigahertz.

If you are getting 40 RF channels in BLE and 79 channels in Bluetooth Classic clearly the bandwidth of each channel must have been different. Roughly it must be twice, right? Okay. So in fact, that is the truth of the matter. You have a larger bandwidth in the BLE case, and you have a shorter bandwidth in the classic case. That is all the difference folks.

This you should not go, you should not think more than that in this whole thing. Now Bluetooth 5 came up in 2016 and that is what it says here. Alright.

**(Refer Slide Time: 09:00)**



The screenshot shows a presentation slide with a menu bar at the top (File, Home, Insert, Draw, Design, Layout, References, Mailings, Review, View, MathType, Help, ACROBAT, Search) and a 'Share' button. The slide content includes:

- Lower power consumption even when compared to other low power technologies. BLE achieves the optimized and low power consumption by keeping the radio off as much as possible and sending small amounts of data at low transfer speeds.
- No cost to access the official specification documents. With most other wireless protocols and technologies, you would have to be a member of the official group or consortium for that technology in order to access the specification.
- Lower cost of modules and chipsets, even when compared to other similar technologies.
- Most importantly, its existence in most smartphones in the market.

**Limitations of BLE**

- **Data Throughput:** the data throughput of BLE is limited by the physical radio layer (PHY) data rate, which is the rate at which the radio transmits data. This rate depends on the Bluetooth version used. For Bluetooth 4.2 and earlier, the rate is fixed at 1 Mbps. For Bluetooth 5 and later, however, the rate varies depending on the mode and PHY used. The

The footer of the slide indicates 'Page 2 of 19', '8 of 2653 words', 'English (United States)', and a 'Focus' button.

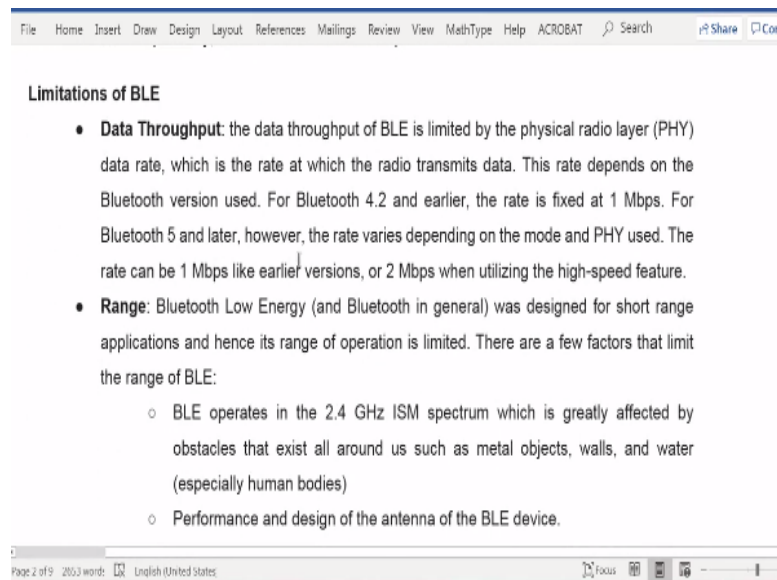
Now then you can go into and say that why did Bluetooth Classic go away and why did BLE come in, in a big way and all that. We mentioned that yeah, it is meant for sensor applications. Indeed, it is a low power consumption device technology. And it is meant for sending small amounts of data at low transfer speeds and all that. And the cost of these modules are also going to go down.

So it is interesting because you can integrate your technology well and make it work with the most ubiquitous device today existent and I am sure more than 7 billion population of the world, I am sure a large segment of them hold on to mobile phones.

And out of that, a significant number of them, a large, fairly significant amount of them actually have smartphones.

So if you build an IoT device with BLE, it is highly probable that you will be able to talk to your smartphone and that is perhaps the reason why you should use BLE.

**(Refer Slide Time: 10:02)**



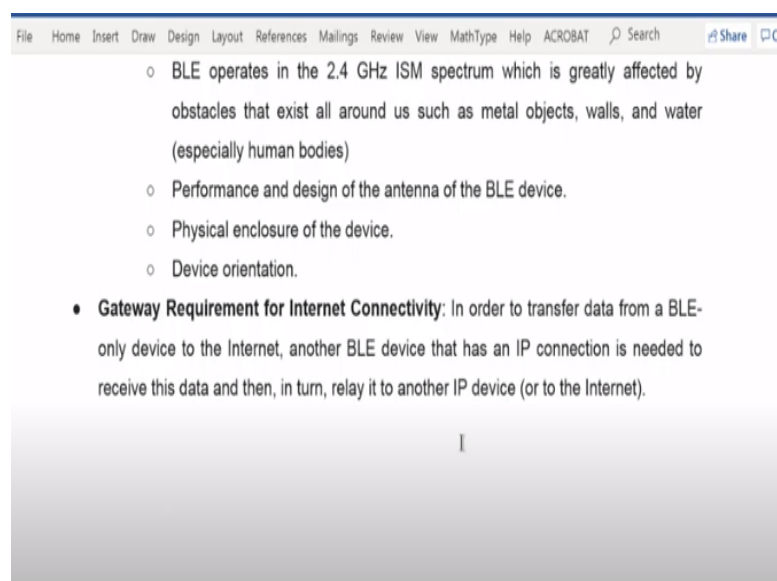
The screenshot shows a presentation slide with a menu bar at the top (File, Home, Insert, Draw, Design, Layout, References, Mailings, Review, View, MathType, Help, ACROBAT, Search, Share, Com). The slide title is "Limitations of BLE". It contains two main bullet points:

- **Data Throughput:** the data throughput of BLE is limited by the physical radio layer (PHY) data rate, which is the rate at which the radio transmits data. This rate depends on the Bluetooth version used. For Bluetooth 4.2 and earlier, the rate is fixed at 1 Mbps. For Bluetooth 5 and later, however, the rate varies depending on the mode and PHY used. The rate can be 1 Mbps like earlier versions, or 2 Mbps when utilizing the high-speed feature.
- **Range:** Bluetooth Low Energy (and Bluetooth in general) was designed for short range applications and hence its range of operation is limited. There are a few factors that limit the range of BLE:
  - BLE operates in the 2.4 GHz ISM spectrum which is greatly affected by obstacles that exist all around us such as metal objects, walls, and water (especially human bodies)
  - Performance and design of the antenna of the BLE device.

At the bottom of the slide, it says "Page 2 of 9 | 2603 words | English (United States) | Focus | [Navigation icons] | [Zoom icon] | [Volume icon] | [Refresh icon] | [Close icon]"

Now it is not that BLE is all that great. It does have certain limitations and those limitations are mentioned here like data throughput is something that you may want to look up and range okay, range is another compromise that you will have.

**(Refer Slide Time: 10:18)**



The screenshot shows a presentation slide with a menu bar at the top (File, Home, Insert, Draw, Design, Layout, References, Mailings, Review, View, MathType, Help, ACROBAT, Search, Share, Com). The slide contains a list of limitations:

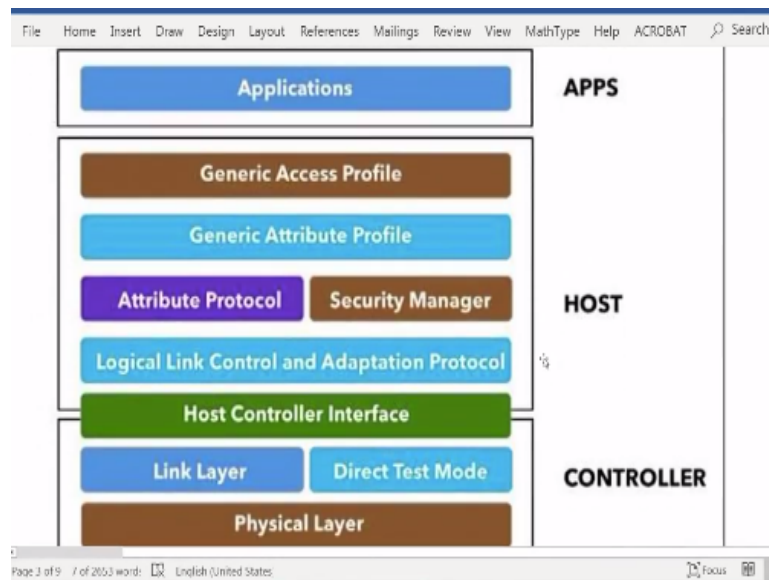
- BLE operates in the 2.4 GHz ISM spectrum which is greatly affected by obstacles that exist all around us such as metal objects, walls, and water (especially human bodies)
- Performance and design of the antenna of the BLE device.
- Physical enclosure of the device.
- Device orientation.
- **Gateway Requirement for Internet Connectivity:** In order to transfer data from a BLE-only device to the Internet, another BLE device that has an IP connection is needed to receive this data and then, in turn, relay it to another IP device (or to the Internet).

At the bottom of the slide, there is a large grey rectangular area with the letter "I" centered in it.



And then essentially Bluetooth Low Energy will definitely require you to connect to via an internet gateway in order to get it out on get the data that you acquired through Bluetooth on to the internet. So you need a BLE device that has an internet IP connection and then in turn, it will relay the data to other IP network connections. So that is something that you will have to take note of.

**(Refer Slide Time: 10:46)**



So this is a general picture. Again I would not like to go into the detail of this. Do spend time reading it yourself.

**(Refer Slide Time: 10:58)**

The good thing is that, as a developer looking to develop BLE applications, **you won't have to worry much about the layers below the Security Manager and Attribute Protocol**. But let's at least cover the definitions of these layers:

- **The physical layer (PHY)** refers to the physical radio used for communication and for modulating/demodulating the data. It operates in the ISM band (2.4 GHz spectrum).
- **The Link Layer** is the layer that interfaces with the Physical Layer (Radio) and provides the higher levels an abstraction and a way to interact with the radio (through an intermediary level called the HCI layer which we'll discuss shortly). It is responsible for managing the state of the radio as well as the timing requirements for adhering to the Bluetooth Low Energy specification.
- **Direct Test Mode:** the purpose of this mode is to test the operation of the radio at the

There is a five layer as I mentioned and this is obviously working in the ISM band. Remember if you are putting up channels you have to ensure that adjacent channel

interferences are reduced and therefore, you need super filters. Go back to the picture I drew about the filters at the output and at the output of the transmitter and the input of the receivers, right?

All of that will ensure that you are sticking to that particular channel in the most effective manner. Then you have link layer, you have higher levels of abstraction which you will see as you go along and read this, read this standard.

**(Refer Slide Time: 11:44)**

The screenshot shows a presentation slide with the following content:

File Home Insert Draw Design Layout References Mailings Review View MathType Help ACROBAT Search Share Comm

So, what is GAP anyway?

GAP stands for Generic Access Profile. It provides the framework that defines how BLE devices interact with each other. This includes:

- Roles of BLE devices
- Advertisements (Broadcasting, Discovery, Advertisement parameters, Advertisement data)
- Connection establishment (initiating connections, accepting connections, Connection parameters)
- Security

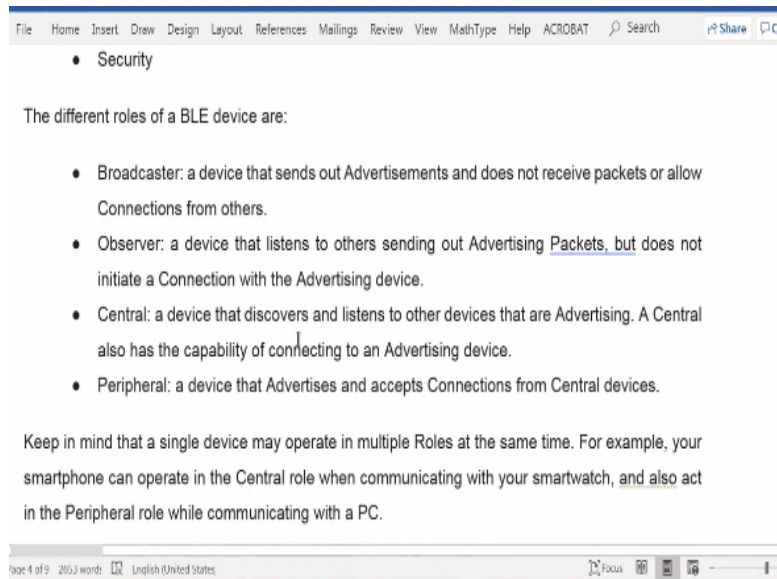
The different roles of a BLE device are:

- Broadcaster: a device that sends out Advertisements and does not receive packets or allow Connections from others.
- Observer: a device that listens to others sending out Advertising Packets, but does not

Page 4 of 9 3 of 2003 words English (United States) Focus

Then what is important is I think you should know this very well because this will actually connect to the picture I drew here, okay. Advertisements and connection establishment and security are important aspects and why because it connects to this picture very well, okay. So let us spend a little time understanding these advertisements in a little detail, connection establishment and so on.

**(Refer Slide Time: 12:10)**



Now BLE devices, if you take this device, this device, okay, this device has many roles, has many roles. And what are the roles? Look here are the roles which are defined here. You can see this many roles are defined. Now which is a clear indicator that you can have a broadcaster type of role, observer type of role, central type of role and peripheral type of role. Now as I mentioned to you, in this picture, peripheral was shown.

You can see this. We were talking about peripheral. Yeah. So you can see I wrote peripheral and central master and slave. So let us go back and look up what we mean by peripheral. See folks, of great importance is this role of this device as a peripheral. What this simply means is the device that advertises and accepts connections from a central device, okay. Now what is a central device?

Central device is a device that discovers and listens to other devices that are advertising. A central device has the capability of connecting to an advertising device. So let us put that picture. This guy says, I am available, I am available, I am available, I am available, I am available. This guy is listening. So he says, he gets it, right? If you do it this way, it is not going to reach.

So I am going to change this and show I am available, I am available, I am available. And like that, it goes and this node listens to that. And once it listens, it says great, I

know that so and so node exists. Like that, you can have other nodes, right? There can be another node 3, let us say. He will also advertise. So the way to start a connection is to first advertise.

So you need to if two devices have to locate and do a communication between themselves, they have to do what? They have to, they have to discover. If I do not know who you are, and you do not know where I am, how do we ever communicate? So therefore discover is an important step. This essentially is leading us to the fact that nodes try to advertise themselves.

Now when you talk about advertising, how often should they advertise is a question that will come to your mind immediately. Oh, how often? If I often, if I look at the whole course and the structure of the course if I do very often, I am going to burn up a lot of energy. If I do not advertise very often, I am not going to get a connection quickly. So you need to compromise between the two.

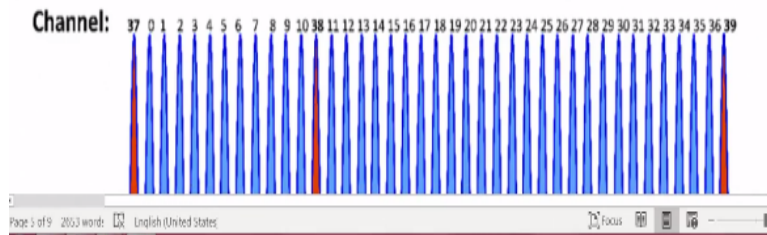
Therefore, the standard says, hey guys, you can advertise at an interval which I will tell you. The standard says the advertising interval can be anywhere from 20 milliseconds to 10.24 seconds, okay. And what is the gap? What is the step size? 0.625 milliseconds. That is it folks. Your next one if it is 20 you cannot set it to you can even set it to 20.625.

Then you add another 0.625 and go to the next one and so on and so forth. You can go up to 10.24 seconds. That is the beauty. This is a handle for what? This is a handle for what? This is a handle for battery, lifetime and so on. Please note that, okay. So this is what you must know that BLE peripheral devices, you can set the advertising interval. So we already know a little bit about that.

Now central is the device which listens and discovers and listens to other devices that are advertising. And it says, okay great, we can connect to each other and start communicating.

**(Refer Slide Time: 16:41)**

channels in BLE, each separated by 2 MHz (center-to-center), as shown in the following figure. Three of these channels are called the Primary Advertising Channels, while the remaining 37 channels are used for Secondary Advertisements and for Data Packet transfer during a Connection.



So that is essentially what was written here. Advertising state means device sends out a packet and there are 40 RF channels and each are separated by 2 megahertz. Some detailing is there. And then there are three primary advertising channels and the remaining are 37 channels. Again, folks go back. Do not worry so much about the advertising channels and all that.

What do you want to do? You are saying I am available. Some interval later, advertising interval later, I am available and I am available or I am so and so. Either available or so and so. Now you know that there is a possibility to play with the advertising interval. You can also the standard also says you should advertise on known channels. Otherwise, how will this guy ever receive it?

So what actually happens is this guy will sit on any one of the three channels. It is called 37 channel, 38 channel, and 39 channel, okay. Now when this guy is transmitting on 37, there is a very good possibility the receiver is on 38. It is possible because it is all unsynchronized at the moment, okay. And then when the advertiser moved from 37 to 38, it is possible that this node went to 39, okay.

Therefore, connection between the advertiser and the central may take a few amount, a little bit of time, before which they will be able to catch up and they will be able to

communicate among themselves. So do not expect the moment this guy sent out at 37 that the central actually got it, because it may be on a different channel.

Therefore, you have to ensure that wait for significant amount of time, not significant, I would say sufficient amount of time such that at some point in time the node, the receiver will be able to get the data, right. So this is important. And there are timers which you have to adjust such that even the central device. For instance, one simple thing you can do right?

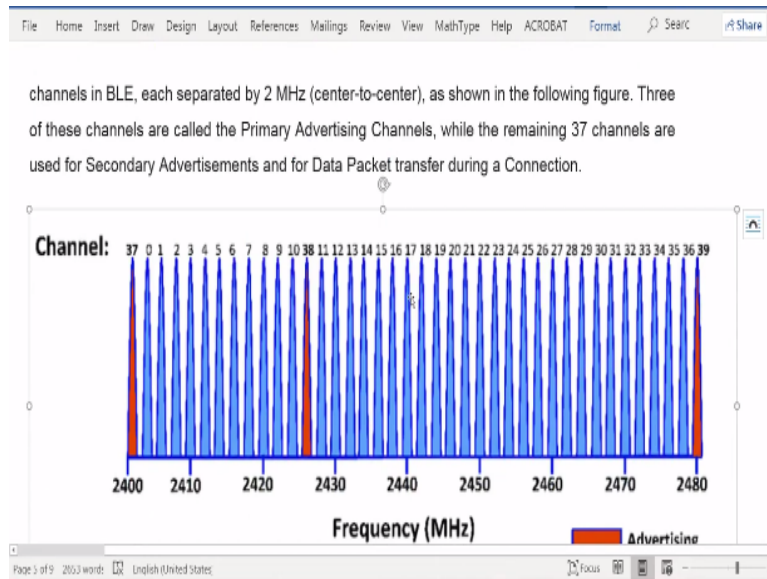
What you can do is supposing you are shifting every 20 milliseconds from one channel to the other. So if you start now, next 20 millisecond later, you are on 37. From 37 you move to, here you did on 37 right? At this point in time, the central was on 38. So it was on 38, okay. Now from 37 after 20 milliseconds, you move to 38. And after 20 milliseconds, you move to 37, 38 and 39.

And after 20 milliseconds, you went back to 37. Okay, now that means 20, 20 and then another 20, right. If you are able to adjust the central to be on that channel for 60 milliseconds, you will definitely catch it. That is all I am saying. Put the central to 60 milliseconds.

So you adjust in the manner. So these are things that will let you adjust in a manner that you will be able to ensure that the advertiser and the central will be able to communicate, will be able to transmit and receive successfully. So folks, these are things that you have to keep in mind. And you know all these things, you have to experiment and take the support of the developer, okay.

And you will be able to adjust these things. So do not think that all this will happen by magic. You have to put an effort and go back to the advertiser, I mean to the vendor of the device, and then accordingly, manage these things.

**(Refer Slide Time: 20:58)**



So that is what this actually this picture is actually saying that, advertising data is happening on 37, 38 and 39. There is a nice thing about this picture. What this is saying is 37 and 38 are number wise, they are one next to the other. But frequency wise they are not. Look where is 37, 2400. Where is 38? It is between 2420 and 2430. Similarly, where is 39? It is at 2480. It is 2480. So number wise it may be one next to the other.

But not necessarily adjacent frequencies. Look carefully, 0 to 9 is shown here. In fact, 0 to 10 are shown. And then you have 38 here. That is what you should make a note. So this is channel 38 reserved for, this frequency band is channel 38, okay. So that is very important and why is it done that way?

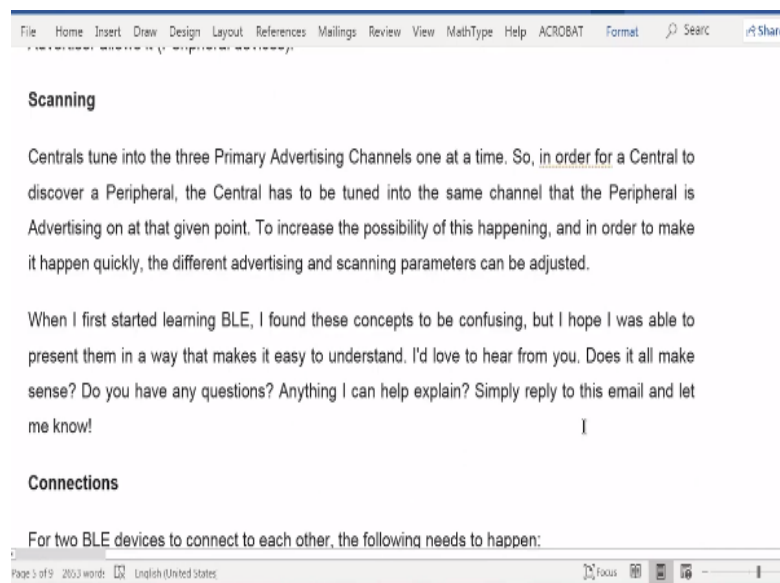
Because, if you do 37, 38 and 39 one next to the other, if there is a problem of you know interference in a given frequency band, if you put 37, 38 38 and 39 one next to the other and the interference is in that band, there is no chance that you will actually get the packet because all data one next to the other would be gone. Therefore, you keep them far apart, send one I am available here.

And after whatever advertising interval time send one here which is far apart in frequency band. And if there is indeed a fading, if there is indeed a problem on this channel, up to this point you may not receive anything, but then luckily you may be

able to get here. Similarly between 38 and 39 if there is a problem of interference across this full band, you do not need to worry because this is the frequency at which 39 is situated.

And therefore this becomes a big advantage for you to ensure that reliability of advertising packets are captured, improved because of this spacing of this system. So essentially it is that.

**(Refer Slide Time: 23:21)**



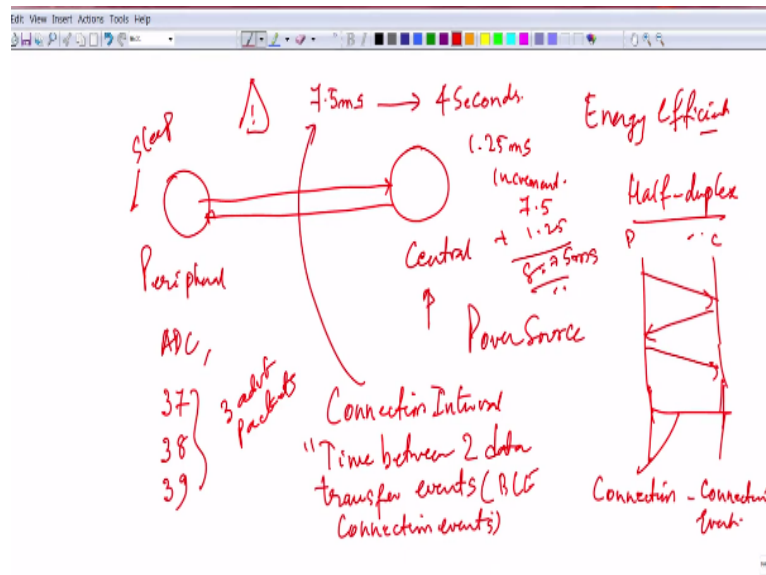
Yeah, then there is this issue of scanning. Centrals tune into the three primary digitizing channels one at a time. So the central, so in order for a central to discover a peripheral the central has to be tuned into the same channel, I mentioned this already. The question of to increase the possibility of this happening and in order to make it quickly happen advertising and scanning parameters can be adjusted.

I mentioned this already to you with an example, okay. So connections. Now that two or more devices have discovered themselves, you can establish a connection between the two. What is the advantage is the question. See, you can do data exchange in one direction, even without establishing a connection, okay. What does it mean?



It simply means this guy who is advertising can put some payload into it, which is temperature payload and give it to the central node. He is actually doing a data communication. It is not only saying I am available.

**(Refer Slide Time: 24:32)**



This is the advertiser. And this is the central. So you do not like the word advertiser. So let us call it the word peripheral, because that is what we defined, okay. I will go back to the same notion of yeah, we said peripheral and central, right. So let us do peripheral and central. Now this guy, when he is advertising on any one of the channels, he can actually also send the data.

If he is not going to, if he does not want to get anything back from the central, he can simply send that and say, okay take it. I am giving you my data, maybe I am giving you temperature data. So you can also do communication between peripheral and central, quick communication without establishing any connection by simply sending out the data as a part of the payload.

This becomes highly energy efficient. I will tell you why. Because central let us say is connected to a large power source, large power source, and he sits on the channel for a longer period of time, as we discussed last time. Peripheral can wake up from deep sleep, do a sensing on the ADC and use this BLE radio and push this data on 37. And he can also push the data on 38. He can also push the data on 39.

For sure, the central will be available on any one of them. So he can send out three advertising packets without connecting to the central. This is possible only when you are in the broadcasting mode or in the advertising mode, okay. But that is not always the case. You also want the central to give you back some data, which means you must do a half-duplex connection.

Basically, most of these radios that we discuss are all half duplex, half duplex connection radios. And therefore, you want bidirectional half duplex between the central and the peripheral. Now here is the another point that you should know. And that point that you should know is with respect to the important point with respect to that is after I have discovered the central may say why not we establish a connection.

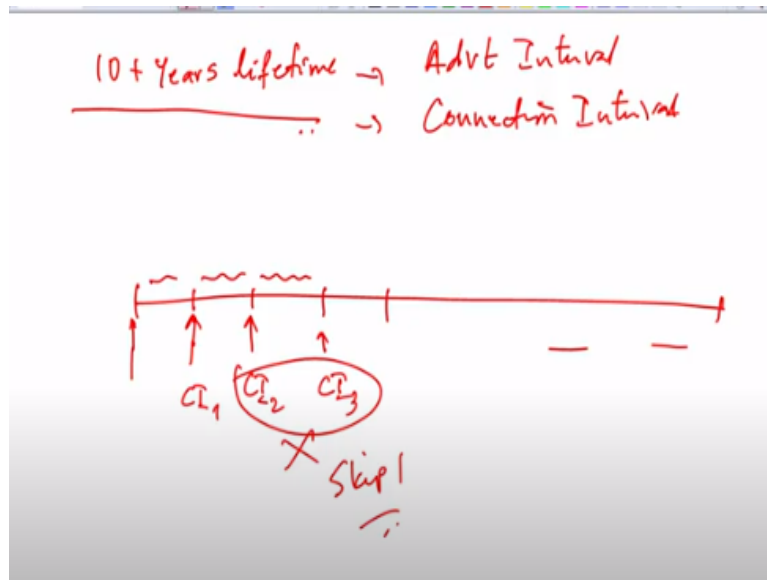
So they exchange a set of packets and then this is peripheral and this is central. They exchange a set of packets and then a connection is established. This is a connection, okay. Once a connection is established, this is called a connection event. This is called a connection event. Once a connection is made, you do not have to get into the issue of again trying to discover the central and all that, okay.

Once discovered is done, you do not have to worry about it. You can go and sleep and central can do whatever it wants. You can wake up next time and you can simply send it to the central because you have already established a connection earlier. But you must tell the central how often you are going to come back and transmit your data to the central. Now that is also configurable.

That is the beauty and that term is called connection interval. Connection interval. So how is it defined? It is the time between two data transfer events, okay. These are what events? BLE connection events. BLE connection events, okay. And this is configurable. What is that configuration? Great. You asked a good question. You can configure from 7.5 milliseconds to 4 seconds.

You can configure it from 7.5 milliseconds to 4 seconds, okay. And what is the increment of 1.25 milliseconds increment. What does it mean? 7.5 plus 1.25, 8.75, 10.0, 11.25, 12.5, 13.75, 15.0, 16.25, 17.5, 18.75, 20.0, 21.25, 22.5, 23.75, 25.0, 26.25, 27.5, 28.75, 30.0, 31.25, 32.5, 33.75, 35.0, 36.25, 37.5, 38.75, 40.0, 41.25, 42.5, 43.75, 45.0, 46.25, 47.5, 48.75, 50.0, 51.25, 52.5, 53.75, 55.0, 56.25, 57.5, 58.75, 60.0, 61.25, 62.5, 63.75, 65.0, 66.25, 67.5, 68.75, 70.0, 71.25, 72.5, 73.75, 75.0, 76.25, 77.5, 78.75, 80.0, 81.25, 82.5, 83.75, 85.0, 86.25, 87.5, 88.75, 90.0, 91.25, 92.5, 93.75, 95.0, 96.25, 97.5, 98.75, 100.0 milliseconds can be the next if you are putting it to the minimum. Maximum can be up to 4 seconds. Folks again this is important. Why is this important?

(Refer Slide Time: 30:17)



Because if you do not tune these parameters properly you will not get 10 years lifetime. 10 plus years lifetime. You can do lot of tricks here folks and Bluetooth Low Energy lets you do all those tricks, okay. So I will first write advertising interval and the second point is connection interval, okay. These are both very important parameters for 10 plus years lifetime, okay.

Folks, there are some tricks here. See, Bluetooth says all right, now let me draw the timeline and this is one CI. Second CI is here. Third CI is here. Fourth CI is here and so on, okay. In this CI what happens? We said that this is one CI, second CI and so on. This is the interval. These are the CIs. Actually, this is what I should draw. I think this is incorrect. Let me draw again. This is CI 1, CI 2, CI 3.

This period should be long enough such that you will be able to send something from the peripheral and get back an acknowledgement from the central and decently exchange data right in both directions. It should be managed, it should be set in such a manner that successful data communication on a half-duplex link is possible.

Now there are some nice feature in BLE, where the peripheral can actually skip a few CIs. You can skip, okay? Skipping CI means what? You are not incurring any overhead of energy, you do not have to wake up. You do not have to set up a connection, okay. Forget about advertisement. Advertisement is already finished, right? You finished all that. You discovered, discovery was over.

On an already established connection, you do not have to reestablish or you do not have to go back and say I have to connect back to you because I declared that I am going to connect to you every CI times. You do not have to do that. So you can skip a few of them. And that is what we meant by the definition of this whole thing related to master and slave.

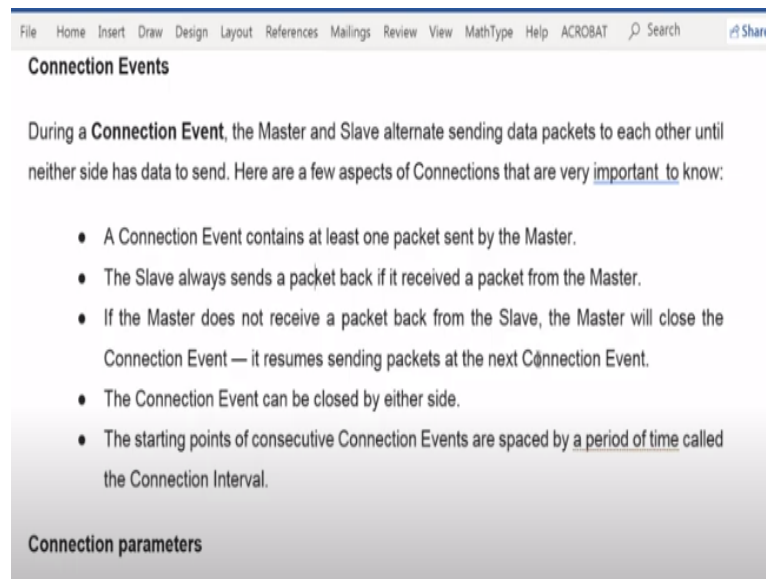
So master and slave notion comes after they have discovered and have connected. This is a term which you can use only after the peripheral and the central are connected. Now central becomes the master, peripheral becomes the slave. Why is it like that? Because master can connect to multiple slaves.

Of course, it will have to manage nicely between the different slaves which are there. It has to remember what is the connection interval of this slave and tune itself to that particular frequency and wait there. And then it has to go back and then get data from other slaves which are connected and all that. So the management routines on the central are quite complicated, okay.

So you have to note all of that. So therefore, this is another important tuning parameter which I wanted to tell you. So now let us go back. So here you have two Bluetooth devices connected to each other and central device needs to be in scanning mode so that it will be able to listen to advertisement coming from the peripherals, okay. And so it is in the scanning mode.

Can you get data in a scanning mode? The answer is big yes. The advertiser can give away some data and so on.

**(Refer Slide Time: 34:31)**



The screenshot shows a presentation slide with a menu bar at the top containing 'File', 'Home', 'Insert', 'Draw', 'Design', 'Layout', 'References', 'Mailings', 'Review', 'View', 'MathType', 'Help', 'ACROBAT', 'Search', and 'Share'. The slide title is 'Connection Events'. The main text reads: 'During a Connection Event, the Master and Slave alternate sending data packets to each other until neither side has data to send. Here are a few aspects of Connections that are very important to know:'. Below this is a bulleted list of five points. At the bottom of the slide, the text 'Connection parameters' is visible.

File Home Insert Draw Design Layout References Mailings Review View MathType Help ACROBAT Search Share

### Connection Events

During a **Connection Event**, the Master and Slave alternate sending data packets to each other until neither side has data to send. Here are a few aspects of Connections that are very important to know:

- A Connection Event contains at least one packet sent by the Master.
- The Slave always sends a packet back if it received a packet from the Master.
- If the Master does not receive a packet back from the Slave, the Master will close the Connection Event — it resumes sending packets at the next Connection Event.
- The Connection Event can be closed by either side.
- The starting points of consecutive Connection Events are spaced by a period of time called the Connection Interval.

Connection parameters

Connection events I discussed this already, right. Connection events what should it contain, at least one packet sent by the master. Slave always sends a packet back if it received a packet from the master, okay. If the master does not receive a packet back from the slave, the master will close the connection event. It resumes sending packets at the next connection event.

Because that was what was agreed upon between the master and the slave. The starting points of consecutive connection events are spaced by a period of time called the connection interval. The starting points of consecutive connection events are spaced by a period of time called the connection interval, okay.

**(Refer Slide Time: 35:20)**

The screenshot shows a presentation slide with a menu bar at the top (File, Home, Insert, Draw, Design, Layout, References, Mailings, Review, View, MathType, Help, ACROBAT, Search, Share). The main content is a bulleted list:

- If the Master does not receive a packet back from the Slave, the Master will close the Connection Event — it resumes sending packets at the next Connection Event.
- The Connection Event can be closed by either side.
- The starting points of consecutive Connection Events are spaced by a period of time called the Connection Interval.

**Connection parameters**

The most important parameters that define a Connection include:

- **Connection Interval:** the interval at which two connected BLE devices wake up the radio and exchange data (at each Connection Event).

I

The interval at which two connected BLE devices wake up. So here is the definition, right? The interval at which two connected BLE devices wake up the radio and exchange data at each connection event; that is what I draw in the picture.

**(Refer Slide Time: 35:39)**

The screenshot shows a presentation slide with a menu bar at the top (File, Home, Insert, Draw, Design, Layout, References, Mailings, Review, View, MathType, Help, ACROBAT, Search, Share). The main content is a bulleted list:

- **Slave Latency:** this value allows the Peripheral to skip a number of consecutive Connection Events and not listen to the Central at these Connection Events without compromising the Connection. For example, a Slave Latency of 3 allows a Slave to skip 3 Connection Events and a value of 0 means that the Slave has to send data to the Master at every Connection Event.
- **Supervision Timeout:** the maximum time between two received data packets before the Connection is considered lost.

The main reason to connect two BLE devices to each other is to transfer data between them. Without a *Connection*, it is not possible to have a bidirectional data transfer between two BLE devices.

I

Which brings us to the concept of **GATT**... so what the heck is GATT?!

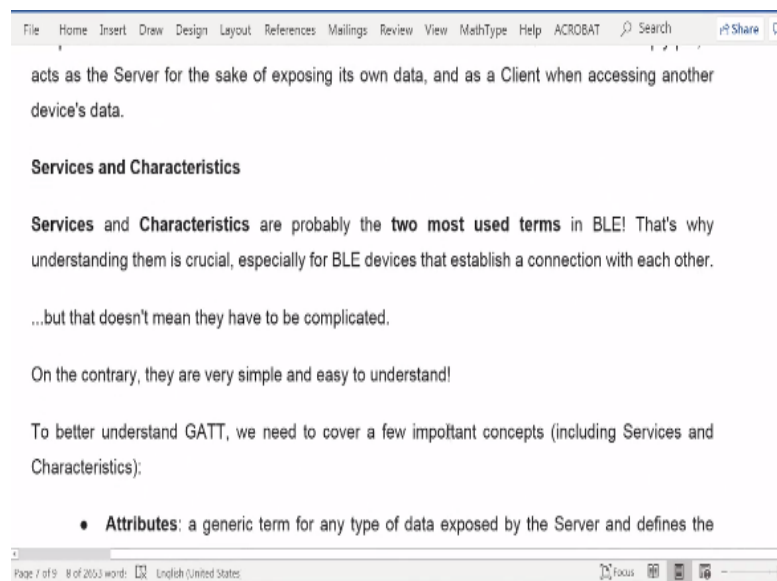
Now this is what I was trying to tell you about slave latency. Look at this parameter. It says the peripheral can skip a number of consecutive connection events because it did not want to wake up. Maybe it did not have sufficient battery. Maybe it did not have data to send. So many reasons why it did not want to wake up. So what it will do is it will not listen to the central at these connection events.

It will just say forget it, I do not have energy, I do not have data, I do not want to listen whatever be the reason. And then say I will allow three skips. So you will do a slave latency of three skips. Connection events which have a value of 0 means slave has to send data to the master at every connection event.

Otherwise, if you say slave latency of 3 means you can skip the in between connection events and then send out the data packet. So here is where the master will have to take stock of everything. Supervision timeout. The maximum time between two received data packets before the connection is considered as lost. So there are some terms associated with it.

All right, so client server, master, slave and so on. These are all important terms. Please read the document. So you will get to know much better, okay.

**(Refer Slide Time: 37:05)**



Now whenever you talk about BLE, you will be talking about services and characteristics. You have to see these are the most used terms as this document says. And so you must understand them. And that is crucial, especially for BLE devices that establish a connection with each other. All right folks, question of different roles of BLE nodes we have understood now.

And we did talk about the connection interval as well, right? We also mentioned about the connection interval and the advertisement interval and so on. What is its impact on energy consumption is the question, right. Also one important point I wanted to say is there is this without connecting between having established a full-fledged connection between the peripheral and the central, there is still a possibility to transfer data in either direction.

Basically, it uses scanned request and scanned response, okay. These are things which are commands which are easily possible, where data can be exchanged between them. So setting that aside, the impact of connection interval, and the battery life of the system, is a very important calculation to do, okay. And therefore, we must see how to connect these bullets.

Because we said that one can easily overwhelm and configure it in a manner that you completely destroyed the ability of the node to provide you this 10 plus years lifetime, okay. So that is a very important aspect. And let us take an example to demonstrate how the impact actually comes up.

**(Refer Slide Time: 38:46)**

---

***Application Note AN092***

**Measuring Bluetooth® Low Energy Power Consumption**  
By Sandeep Kamath & Joakim Lindh

---

**Keywords**

<i>Bluetooth Low Energy</i>	<i>CC2540</i>
<i>BLE</i>	<i>CC2541</i>
<i>Power Consumption</i>	<i>CC2540DK-MINI</i>
<i>Battery Life</i>	

1 Introduction

---

So for that what I did was, I took a application note, which is beautifully written by these two people. This is from TI, okay. Measuring Bluetooth Low Energy power consumption. I think you should read this article very well.



(Refer Slide Time: 39:02)

## 1 Introduction

The Bluetooth® low energy (BLE) standard was developed with long battery life in mind, allowing for devices that can last anywhere from several months to several years while operating on a single coin-cell battery. This application note describes the setup and procedures to measure power consumption on a CC2541 device operating as a GAP "Peripheral" in a BLE connection.

It is assumed the reader of this application note has knowledge about the BLE standard and the CC2541. The example is based on the Texas Instruments CC2541 running version 1.2 of the BLE stack. Please refer to [1] and the [2] for further information.

In addition, it is assumed that the reader

engineering concepts, and understands how to use laboratory test equipment such as an oscilloscope and DC power supply.

The current consumption measurements are presented, and battery life time is calculated for an example application. An accompanying Excel sheet is provided so that users can estimate their battery life based on their own custom usage scenario.

Note that the results presented in this document are intended as a guideline only. A variety of factors will influence the battery life calculation and final measurements. Measurements should be performed on customer hardware, in a controlled environment, and under the target application scenario.

I will give you an overview of this article, but I am not going to spend every line trying to explain this. Therefore, read them read this article very, you can just download it, okay. It is available freely, you can just download it. This is the title of the article. Just download and read it. So the goal is the same. You want 10 plus years of operating and you are talking about a coin cell battery.

Typically, something like CR2032, which gives you some 240 or 250 milliampere hour. It is a small coin milliampere hour battery, right? Now author is going to giving you a beautiful feel for the problem. This is what we already know very well.

(Refer Slide Time: 39:41)

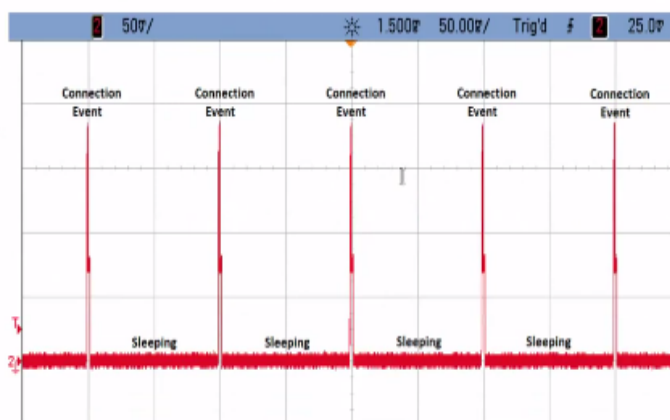


Figure 1- Current Consumption versus Time during a BLE Connection

In addition to transmitting, a BLE device will most likely go through several other states, such as receiving, sleeping, waking-up from sleep, etc... Even if a device's current consumption in each different state is known, this is still not enough information to determine the total power

What is happening here? You have connection event. Then you have sleeping. Then you have another connection event, sleeping, connection event and sleeping. Now our problem is how is this how is this connection event, what is its energy consumption in this connection event. Because you can see significant amount of time it is sleeping, but then this is typical of any sensor node, right you will in IoT node.

Whenever it wants to transmit it will establish a connection event with the central. The peripheral will establish with the central and then it will move on. Only first time as I mentioned it has to do all that broadcasting and so on and then establish a connection between the two of them and repeated connections can be simply used this connection event essentially. So this is the same picture that you know very well.

**(Refer Slide Time: 40:34)**

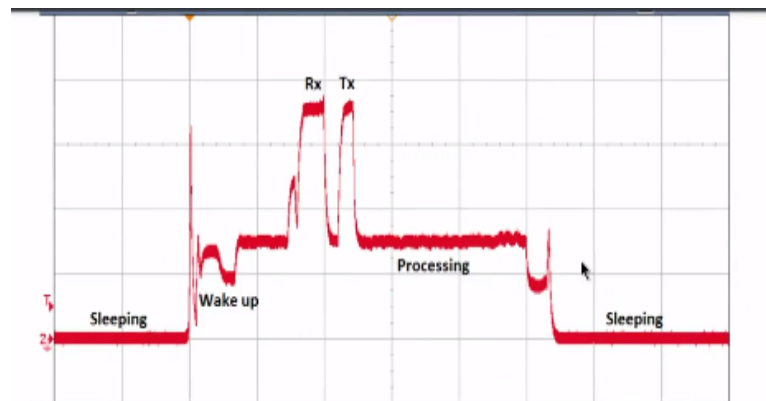


Figure 2- Current Consumption versus Time during a single Connection Event

Now every time the node wakes up and it wants to transmit something, this is what will happen in the node. Kindly recall the sleeping is the same as this sleeping. Now we are interested in expanding this part. How does this part look like? This will simply look like wake up. RX reception mode of the radio. This is the transmission mode of the radio.

Some processing right and then doing something additional and then going back to sleep. Broadly this is what it is. There are no further details here, okay. And we are talking about a Keyfob okay, and Bluetooth enabled Keyfob. And you want to

estimate the lifetime of that Keyfob. So this whole story is about that. AN092 application node is about that. So figure 2 is very straightforward.

This is something that we know very well and x axis is time, y axis is the current consumption, okay. This is how it looks, the waveform.

(Refer Slide Time: 41:32)

---

***Application Note AN092***

In a typical application, a device running the BLE stack will spend most of the time in a sleeping state between connection events. When the CC2541 goes into "Power mode 2" (PM2) between connection events, the internal digital voltage regulator is turned off, along with the 16 MHz RCOSC and 32 MHz crystal oscillator. The 32 kHz sleep timer remains active while the RAM and registers are retained. The only way that the device will wake up is if an I/O interrupt or sleep timer interrupt occurs.

The primary metric that takes all of these other time and current measurements into account is the "average current". It is this value that can be used to determine the battery life of a device running the BLE stack. Note that a single "average current" value cannot be given for a device in its datasheet or in the device's specifications, as the average current is highly dependent on the connection parameters used. Anytime an "average current" specification is given, it is very important to understand the exact use-case during which the measurement was made.

For a complete system-on-a-chip such as the CC2541, it is important to understand that the MCU is typically not only running the BLE protocol stack, but it is also running profiles and an

---

Now then the article goes on to talk about how you get these spikes, okay. For example this spike. How did this spike come is a question. So they write a beautiful expansion here. The digital voltage regulator is turned off along with some oscillators. Then the only way the device will wake up is if an IO interrupt or sleep timer occurs, interrupt occurs and all that.

Then he says that the real you know energy consumption is all about trying to get to this average current, okay. Average current into time will actually give you milliampere or whatever into time. And if the voltage is known you actually know the energy, okay. Note that a single average current value cannot be given. You cannot actually say what is the average current, it is not possible.

How can you find out the average current? You have to take some sort of time over which this amount of current was consumed for this time, this amount of current for this much time and so on. And then you add up everything, then you may get one

average current value, right. So that is what he has written here. And this is something that we know very well, okay.

**(Refer Slide Time: 42:45)**

---

The primary metric that takes all of these other time and current measurements into account is the "average current". It is this value that can be used to determine the battery life of a device running the BLE stack. Note that a single "average current" value cannot be given for a device in its datasheet or in the device's specifications, as the average current is highly dependent on the connection parameters used. Anytime an "average current" specification is given, it is very important to understand the exact use-case during which the measurement was made.

For a complete system-on-a-chip such as the CC2541, it is important to understand that the MCU is typically not only running the BLE protocol stack, but it is also running profiles and an application. The application not only uses the MCU on the device, but it may also be using peripherals on the chip, such as an ADC or op-amp. In addition, other devices on the circuit board, aside from the device running the BLE protocol stack, may be drawing current as well. This document will focus on strictly measuring current consumed as a result of the BLE protocol stack; however it is important to be aware of other sources of current consumption, as they will affect the battery life.

#### 4 Test Setup

This section describes the general setup required for performing power testing.

##### 4.1 System Overview

---

Now he also says that for the complete system-on-a-chip like CC2541 and all that you have to understand that the MCU is typically not only running the radio stuff, but it is doing so many other things. It is operating the ADC for sensing and the MCU may be in some other state, it has to wake up. Acquire data from the sensor data via the ADC so on.

So it is important to be aware that other sources of current consumption will also affect the battery. So whatever you measure is not going to be straightforward.

**(Refer Slide Time: 43:19)**

---

protocol stack; however it is important to be aware of other sources of current consumption, as they will affect the battery life.

#### 4 Test Setup

This section describes the general setup required for performing power testing.

##### 4.1 System Overview

In order to properly measure average current consumption, the current must be measured with respect to time. Therefore, a basic multimeter is not sufficient, and an oscilloscope is required. The simplest way to measure current with an oscilloscope is to use a current probe and directly monitor the current going into the system. If you do not have a current probe available, an easy alternative is to use a small resistor in line with the power supply input to the system. You can then use a standard oscilloscope voltage probe to measure the voltage across the resistor, and effectively measure the current by dividing the voltage by the resistance. A good resistor value to use is  $10\ \Omega$ , as this value is small enough that it shouldn't affect the existing circuitry, and large enough to provide a voltage that can be measured with decent precision. In addition, using a value of  $10\ \Omega$  makes the calculations very easy.

When performing measurements, it is best to use a regulated DC power supply as opposed to an actual battery. This eliminates variables that might be caused by a defective or low battery. Figure 3 below shows the full setup.

---

Now this test setup is pretty straightforward. If you read it carefully it only simply boils down to just this.

**(Refer Slide Time: 43:24)**

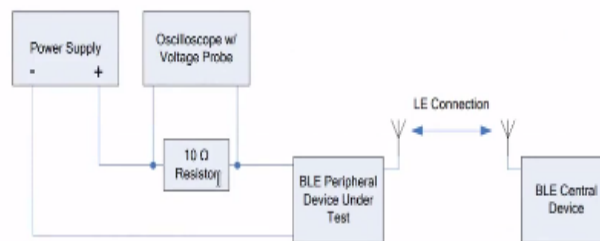


Figure 3- Test Setup using Oscilloscope with Voltage Probe

##### 4.2 Hardware Modifications

In the CC2540DK-MINI kit, the peripheral device is the "Keyfob" board. A few simple hardware modifications are required to implement the setup in Figure 3.

1. Solder a wire on the Keyfob PCB to ground. An easy location to do this is at pin 1 of the DEBUG connector on the board, as shown in Figure 4.
  2. Solder a  $10\ \Omega$  resistor to the VDD on the Keyfob PCB. An easy location to do this is at the left side (with the board oriented so that the text is read properly) of the unpopulated resistor R1, as shown in as shown in Figure 4.
- 

That is he wants you to put a 10 ohm resistor in series. It is like a shunt. And this is your BLE peripheral of interest. You connect it to a oscilloscope probe, a voltage you get it directly into voltage you can measure the voltage which is already you know the current profile because you know the current passing through at fixed 10 ohm resistor. Very accurate 10 ohm resistor is already good enough. So essentially he is talking to you about that setup here, okay.

**(Refer Slide Time: 43:52)**



---

You should now be able connect the DC power supply and oscilloscope voltage probe to the board, with the hardware configured properly for current measurements.

### 4.3 Embedded Software Modifications

Before testing can begin, the Keyfob software must be properly configured. To properly measure current consumption resulting from the BLE stack, additional application processing must be turned off. The SimpleBLEPeripheral project, included with the BLE stack, is simple in that the device immediately starts advertising upon power-up, and will accept any connection request from a master device. A couple of modifications to the software can help to reduce unnecessary power consumption. [

#### 4.3.1 Remove Periodic Event

As long as no buttons the Keyfob are pressed, the only regular application processing that occurs should be the periodic event, which is set to occur once every five seconds. This reading may throw off the power measurements, and therefore must be removed.

To eliminate the periodic event from the application, simply comment out the following line of source code from the SimpleBLEPeripheral\_ProcessEvent function in the file simpleBLEPeripheral.c:

```
// osal_start_timerEx( simpleBLEPeripheral_TaskID, PERIODIC_EVT, PERIODIC_EVT_PERIOD );
```

By commenting out this line, the OSAL timer for the first periodic event will never get set. By preventing the first timer from being set, all subsequent OSAL timers are set after the initial

---

Now embedded software modifications have to be done because you are connecting it to a power supply. You may want to do a few things. Device immediately starts advertising upon power-up and will accept any connection request from a master device is something that you may have to change in your software stack. And you do some other couple of modifications that can help you reduce the unnecessary power consumption, okay.

**(Refer Slide Time: 44:17)**

---

As long as no buttons are pressed, the only regular application processing that occurs should be the periodic event, which is set to occur once every five seconds. This reading may throw off the power measurements, and therefore must be removed.

To eliminate the periodic event from the application, simply comment out the following line of source code from the SimpleBLEPeripheral\_ProcessEvent function in the file simpleBLEPeripheral.c:

```
// osal_start_timerEx( simpleBLEPeripheral_TaskID, PERIODIC_EVT, PERIODIC_EVT_PERIOD );
```

By commenting out this line, the OSAL timer for the first periodic event will never get set. By preventing the first timer from being set, all subsequent OSAL timers are set after the initial event. This will stop the application from performing unnecessary processing. Once you have implemented this change, rebuild the project and download to the keyfob.

#### 4.3.2 Configure General-Purpose Input / Output (GPIO) Pins

If the GPIO pins are not configured properly, unnecessary current draw may occur. Ideally, each GPIO pin will be unconnected, and therefore no leakage of current will occur. In the case of the keyfob, as with most boards, many of the GPIO pins are connected to peripheral devices, such as the LEDs, the buzzer, the accelerometer, and the buttons. To maximally reduce the current, all GPIO pins must be set to outputs at a low level. Note that this has already been implemented in the SimpleBLEPeripheral\_Init function in version 1.2 of the BLE stack.

```
P0SEL = 0; // Configure Port 0 as GPIO
```

---

Now remove periodic events. This is another important step and you configure anything that will periodically occur you just remove because you are only interested in the power consumption of that particular radio because that is what your whole

interest is. Because if you recall, the title of this article is you are interested in measuring Bluetooth Low Energy power consumption.

You are not interested in any other periodic events which will keep bothering you, right. So that is your focus. All right. So having said that part, you may also want to check the peripherals of the SoC.

**(Refer Slide Time: 44:56)**

---

By commenting out this line, the OSAL timer for the first periodic event will never get set. By preventing the first timer from being set, all subsequent OSAL timers are set after the initial event. This will stop the application from performing unnecessary processing. Once you have implemented this change, rebuild the project and download to the keyfob.

#### 4.3.2 Configure General-Purpose Input / Output (GPIO) Pins

If the GPIO pins are not configured properly, unnecessary current draw may occur. Ideally, each GPIO pin will be unconnected, and therefore no leakage of current will occur. In the case of the keyfob, as with most boards, many of the GPIO pins are connected to peripheral devices, such as the LEDs, the buzzer, the accelerometer, and the buttons. To maximally reduce the current, all GPIO pins must be set to outputs at a low level. Note that this has already been implemented in the SimpleBLEPeripheral\_Init function in version 1.2 of the BLE stack.

```
POSEL = 0; // Configure Port 0 as GPIO
P1SEL = 0; // Configure Port 1 as GPIO
P2SEL = 0; // Configure Port 2 as GPIO

P0DIR = 0xFC; // Port 0 pins P0.0 and P0.1 as input (buttons),
              // all others (P0.2-P0.7) as output
P1DIR = 0xFF; // All port 1 pins (P1.0-P1.7) as output
P2DIR = 0x1F; // All port 2 pins (P2.0-P2.4) as output
```

---

TEXAS

If there are other peripherals like GPIOs, which are on you keep them off so that no leakage of current will occur. That is another important thing, okay.

**(Refer Slide Time: 45:05)**

---

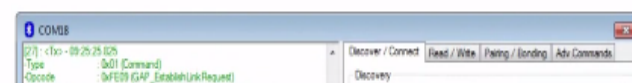
initialization processes complete.

#### 4.4 Central Device and BTool Setup

You will also need to have a central device in order to form a connection with the Keyfob. The simplest way to do this is to use the USB Dongle in the CC2540DK-MINI kit running the standard HostTestRelease application, with BTool used to control the dongle.

With dongle connected to the PC and the Keyfob powered by the DC power supply, open BTool. Press the right button on the Keyfob to make it discoverable, and then press the "Scan" button in BTool to verify that the dongle and the Keyfob are able to talk to each other. The advertisement and scan response data from the Keyfob should show up in the BTool log window. You are now ready to form a connection between the devices.

Before forming the connection, the proper connection parameters should be used. This will be dependent on the application that is being considered. The supervision timeout setting should not affect the power measurements; however be sure to use a legal value as per the Bluetooth 4.0 specification. For our first example, you will use a connection interval of 1 second, with zero slave latency. Therefore, use the values as shown in Figure 6. Be sure to hit the "Set" button after entering in the values.

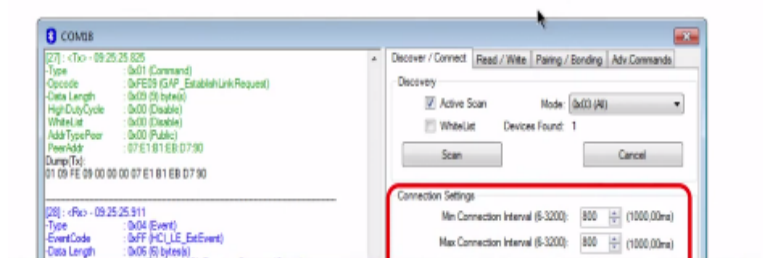


Now the advertisement and scan response data which is a two way communication between the peripheral and the central without connecting, very important, without establishing a connection can also be done and that in order to show you those message exchanges you need a tool and they are asking if you are interested you can go and look up this BT Tool log window and now you will be, you are ready to form a connection between the devices.

**(Refer Slide Time: 45:35)**

With dongle connected to the PC and the Keyfob powered by the DC power supply, open up BTool. Press the right button on the Keyfob to make it discoverable, and then press the "Scan" button in BTool to verify that the dongle and the Keyfob are able to talk to each other. The advertisement and scan response data from the Keyfob should show up in the BTool log window. You are now ready to form a connection between the devices.

Before forming the connection, the proper connection parameters should be used. This will be dependent on the application that is being considered. The supervision timeout setting should not affect the power measurements; however be sure to use a legal value as per the Bluetooth 4.0 specification. For our first example, you will use a connection interval of 1 second, with zero slave latency. Therefore, use the values as shown in Figure 6. Be sure to hit the "Set" button after entering in the values.



So essentially all the messages which are passed before making connection establishment, you can actually see with this BT Tool, okay. Now here is the problem setting. He says, these author say let us take CI of 1 second with zero slave latency. Fantastic, so you actually know what this means. Zero slave you know and you also know that the CI is set to 1 second.

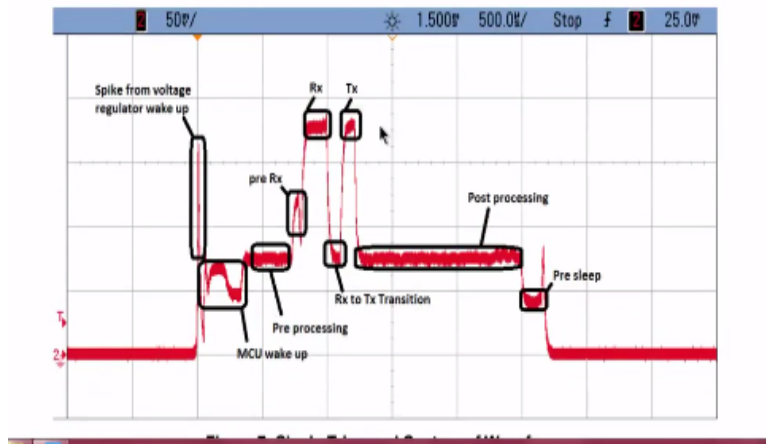
So you do all kinds of settings here and then you use the BT tool and then you configure it and then you will end up with this picture.

**(Refer Slide Time: 46:10)**



### 5.1 Capturing a Waveform during a Connection Event

In Figure 7, the voltage waveform is captured using a rising-edge trigger, set to trigger each time that the CC2541 wakes up from power mode 2 for a connection event.



This is single triggered capture of a waveform, okay. x axis is time y axis is the current consumption because we already have the current shunt resistor in series, you can actually measure it, okay.

(Refer Slide Time: 46:28)

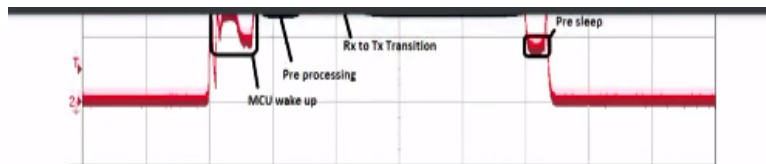


Figure 7- Single-Triggered Capture of Waveform

Because the voltage is measured across a 10 Ω resistor, the current level can easily be calculated by dividing the voltage by 10. One of the first things that you may notice when looking at the capture is a large spike in current at the moment when the MCU wakes up from sleep. This spike is caused by the digital voltage regulator inside the CC2541 re-powering up. The regulator contains capacitors that must be re-charged, and thus quickly draw current when the device wakes up. This spike normally would not appear with capacitor C7 still populated on the board; and therefore while testing power consumption this spike can be ignored.

In addition to the voltage spike, you will notice that the current draw changes as the CC2541 goes through several different states as a part of the connection event:

**MCU wake-up** – upon waking up, the current level drops slightly

**Pre-processing** – the BLE protocol stack prepares the radio for sending and

Now he writes here, because the voltage is measured across a 10 ohm resistor, the current level can easily be calculated by dividing the voltage by 10, correct. One of the first things that you may notice when looking at the capture is a large spike in current at the moment when MCU wakes up from sleep. This spike is caused by the digital voltage regulator inside the CC2541 re-powering up.

This is the thing. Whenever you talk about SoCs you have regulators inside. The regulator kicks up. Whenever there is a regulator kick, I showed you this already. No one really knew at that time why it came but you actually know now that this kick must have is coming because of the voltage regulator waking up, okay. So this is an important thing.

**(Refer Slide Time: 47:15)**

---

calculated by dividing the voltage by  $I_0$ . One of the first things that you may notice when looking at the capture is a large spike in current at the moment when the MCU wakes up from sleep. This spike is caused by the digital voltage regulator inside the CC2541 re-powering up. The regulator contains capacitors that must be re-charged, and thus quickly draw current when the device wakes up. This spike normally would not appear with capacitor C7 still populated on the board; and therefore while testing power consumption this spike can be ignored.

In addition to the voltage spike, you will notice that the current draw changes as the CC2541 goes through several different states as a part of the connection event:

- MCU wake-up** – upon waking up, the current level drops slightly
  - Pre-processing** – the BLE protocol stack prepares the radio for sending and receiving data
  - Pre-Rx** – the CC2541 radio turns on in preparation of Rx and Tx
  - Rx** – the radio receiver listens for a packet from the master
  - Rx-to-Tx transition** – the receiver stops, and the radio prepares to transmit a packet to the master
  - Tx** – the radio transmits a packet to the master
- 

So you have MCU wake up pre-processing which essentially means the BLE protocol stack prepares the radio for sending and receiving data. Then you have pre-reception. Radio turns on the preparation of Rx to Tx. Now it is basically a state machine transition from Rx to Tx. Rx essentially means the radio receiver is listens to packet from the master.

Rx-to-Tx transition is essentially the receiver stops and the radio prepares to transmit a packet to the master and so on. Tx is radio transmits a packet to the master. This whole view, this whole article is written with a view of a Keyfob. That is important okay, it is with respect to the peripheral. Because the peripheral is the one that has that CR2032 battery. So you are interested in estimating the lifetime of that peripheral.

Think about your car key and the battery inside that. This calculation actually matches that. It is your car key, Keyfob that is what you are interested in. So all these things are with respect to the MCU on the Keyfob and so on.

(Refer Slide Time: 48:20)

---

## Application Note AN092

**Post-processing** – the BLE protocol stack processes the received packet and sets up the sleep timer in preparation for the next connection event.

**Pre-Sleep** – the BLE protocol stack prepares to go into sleep mode.

A similar waveform can be captured during every connection event; however the amount of time for each state will vary depending on the circumstances (the size of the PDUs being transmitted / received, the amount of processing time required by the stack, etc.). In addition, if more than one packet is transmitted or received, there will be additional Rx, Tx, and transition states.

While the capture in Figure 7 appears like it can be used to start performing measurements, it does not tell us the entire story. In Figure 8 below, the voltage waveform is captured again using a rising-edge trigger, set to trigger each time that the CC2541 wakes for a connection event. This time, the oscilloscope's "persistence" feature is used, in which many captures are overlaid on top of one another.



Alright, a similar waveform can be captured during every connection event. However, the amount of time for each state will vary depending on the circumstances. All that he writes. In addition, if more than one packet is transmitted or received, there will be additional Rx, Tx and transition states and all that. Right now you can restrict it to just one reception and one transmission perhaps so that you can understand the complete energy consumption much more clearly, right?

Now this is figure 7. What you saw here is indeed figure 7. And now he writes a sentence about figure 7.

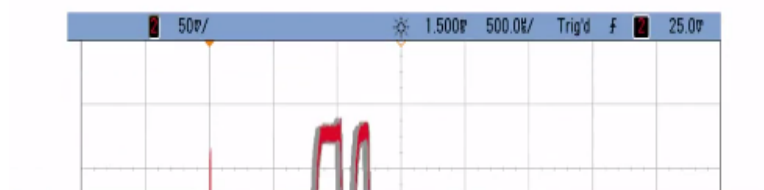
(Refer Slide Time: 48:59)

---

**Pre-Sleep** – the BLE protocol stack prepares to go into sleep mode.

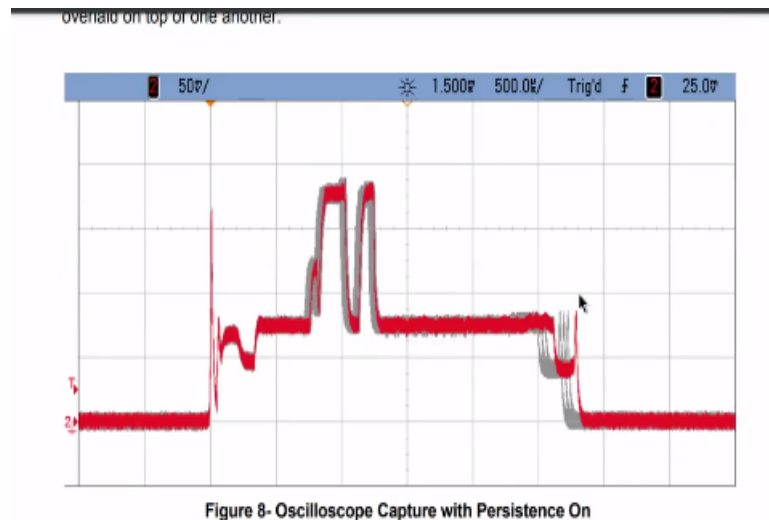
A similar waveform can be captured during every connection event; however the amount of time for each state will vary depending on the circumstances (the size of the PDUs being transmitted / received, the amount of processing time required by the stack, etc.). In addition, if more than one packet is transmitted or received, there will be additional Rx, Tx, and transition states.

While the capture in Figure 7 appears like it can be used to start performing measurements, it does not tell us the entire story. In Figure 8 below, the voltage waveform is captured again using a rising-edge trigger, set to trigger each time that the CC2541 wakes for a connection event. This time, the oscilloscope's "persistence" feature is used, in which many captures are overlaid on top of one another.



Now finger 7 appears like it can be used to start performing measurements. It does not tell us the entire story. That is the problem. Now in figure 8, you see the entire story, okay. Figure 8 below the voltage waveform is captured again, using a rising edge trigger. Set the trigger each time that the CC2541 wakes for a connection event. This time the oscilloscope's persistent feature is used in which many captures are overlaid on top of one another.

**(Refer Slide Time: 49:32)**



From the Figure 8, you can deduce a few interesting facts about the current consumption

Look at this line, it is so thick. One after the other after the other after the other. You go on tracing one after the other. So one thin line is one measurement. Second line on top of it is like second measurement. So you can see how many measurements must have done. But what you find is this very interesting thing folks. It is now a bit of a spread here, right? Because of this additional measurement.

So you look at 7, look at 7 and then look at 8. You see 7 is like this, okay. Single trigger capture of waveforms. It is just one line, one capture trace. Think of what would happen if you did it again? Like this and like this. Again you do it goes on top of this, that is what you do you get here, right? This is what you are trying to say.

**(Refer Slide Time: 50:25)**

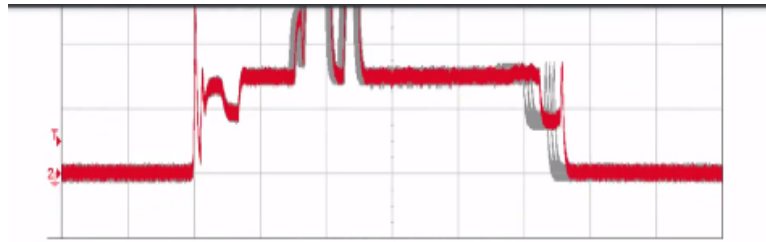


Figure 8- Oscilloscope Capture with Persistence On

From the Figure 8, you can deduce a few interesting facts about the current consumption during connection events. The first point is that the total processing time for each connection event is not always exactly the same. This means that a single scope capture is not sufficient to perform power measurements.

The next point to notice is that even though there is variance in the processing time from event-to-event, the total processing time is not completely random, but rather falls into "slots". Figure 9 below shows a closer view of these slots in which the device completes processing.

---

Now you see from figure 8, you can deduce some interesting things, okay. One of the things is about the current consumption during connection events. First of all, the processing time for each connection event is not exactly the same. That is the one problem that he notices. This means a single scope capture is not sufficient, because there is a range over which the current is varying.

In fact, the time is varying. And because time varies, the current consumption will vary and therefore energy will vary. This is the big takeaway. The next point to notice is that even though there is variance in processing time, from event to event, the total processing time is not completely random, but rather falls into some sort of slots, okay.

**(Refer Slide Time: 51:10)**

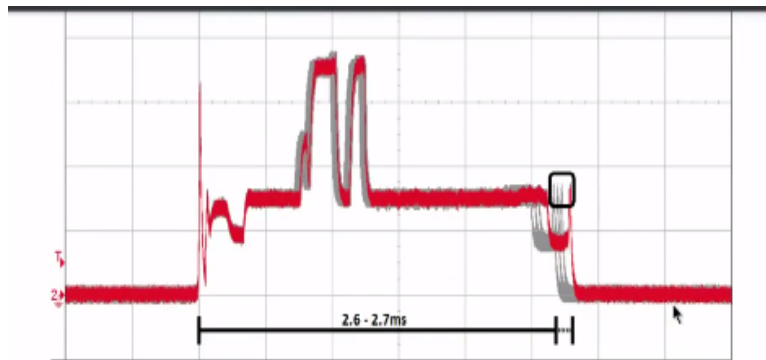


Figure 9- Processing Time "Slots"

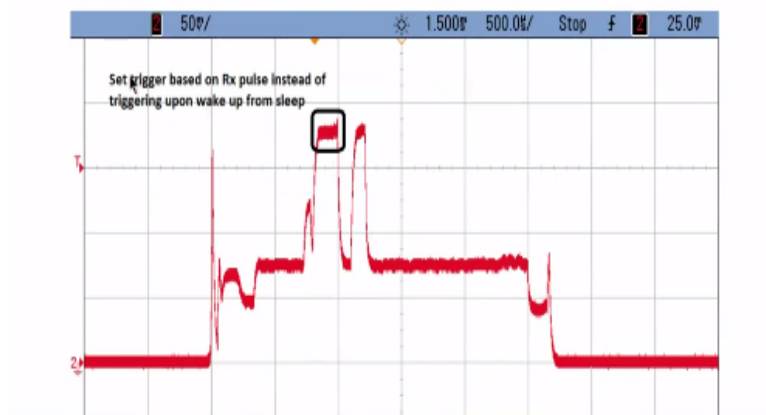
Events that fall into the shortest slot take less time to process, and therefore less power is consumed. In this case, you can see using the oscilloscope's cursors that the CC2541 is awake for approximately 2.6 ms. Events that fall into the longest slot take more time to process, and therefore more power is consumed. In this case, the CC2541 is awake for approximately 2.7 ms. In order to get an accurate calculation of the power being consumed, you must take into account the fact that the processing time for connection events is not always the same. A little bit of statistical analysis will be required to get accurate values.

Now this is what he is talking about, processing time slots. Events that fall into shortest slot, take less time to process and therefore, less power is consumed, okay. And that he says is 2.6 milliseconds. This is the shortest slot, take less time. So this is the shortest slot. The longer slot is the 2.7 milliseconds. So essentially, he shows it here. From here to here you can have either 2.6 or you can have 2.7.

So this is what you should account for. And now he says, there are some measurement artifacts in this picture. So let us correct those measurement artifacts.

**(Refer Slide Time: 51:53)**

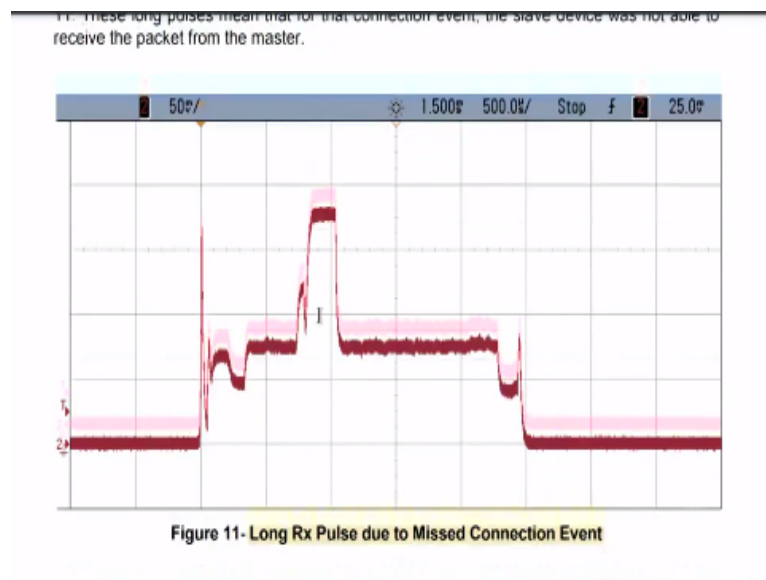
### ***Application Note AN092***





The measurement artifact is this. You should set the trigger based on Rx pulse instead of triggering upon wake up from sleep, do not do this. Do it based on the Rx pulse. So therefore, this picture again changes and this is figure 10.

**(Refer Slide Time: 52:08)**



Now whenever you okay, this 11 is not all that critical, but anyway, I think I will skip it you can read it yourself.

**(Refer Slide Time: 52:17)**

---

### 5.2 Determining Variance in the Length of Connection Events

As mentioned in the previous section, the CC2541 is awake variously for approximately 2.6 ms to 2.7 ms during each connection event. To improve the power consumption estimate, you must determine the percentage of the time that each of these cases occurs, or use the shortest and longest of these slots to calculate an average. Some oscilloscopes contain embedded software for statistical analysis, while others have PC applications that can be used to interface with a scope and perform analysis. If these tools are available this task can be done fairly easily; however even if these options are not available there is a way to estimate this percentage. By simply randomly triggering (be sure to use the rising-edge trigger upon wake-up from sleep) connection events and keeping a count of events in each slot, a good estimate can be made.

In the case of our example, the connection interval is long enough (one second) that you can just leave the trigger running on auto, and keep a tally of each event that falls into each slot. The more events that are watched, the more accurate the calculation will be. By doing this type of analysis, you will find that the average of the shortest and longest slot, 2.65ms, will be accurate enough. For a better overview we will use the longest and shortest slot time and state that they occur equal amount of times, that is, 50% each. Now we need to look closer on these two slots.

### 5.3 Performing Measurements for Connection Events

---

Now what you should do is, you have variously for approximately what is the time which we said in each time in the connection interval event, 2.6 milliseconds and 2.7 milliseconds, both are there. So to improve our consumption estimate, you must determine the percentage of time that each of these cases occur. How many times did

you have 2.6 milliseconds and how many times did you have 2.7 milliseconds? You have to get to that point, okay.

**(Refer Slide Time: 52:50)**

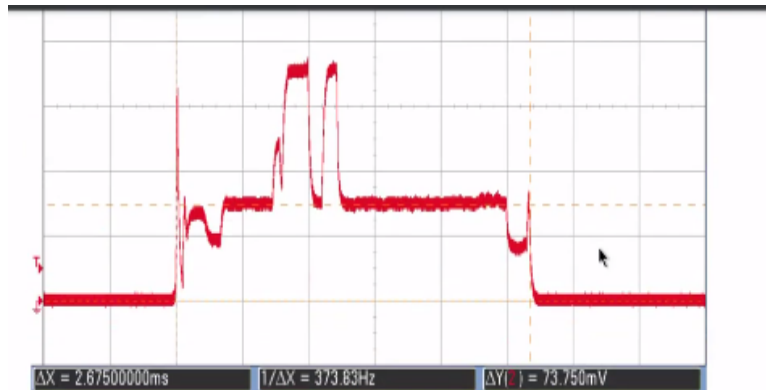


Figure 12- Typical Connection Event with CC2541 Awake for 2.6 ms

To perform measurements, the sections of the waveform must be divided up, with current and timing measured for each state. Figure 13 shows this division of sections for each state.

Now performing measurements for connection events. Now typical connection event with CC2541 awake for 2.6 milliseconds. This is a waveform that you get. And you see the line here right, this is what it is.

**(Refer Slide Time: 53:03)**

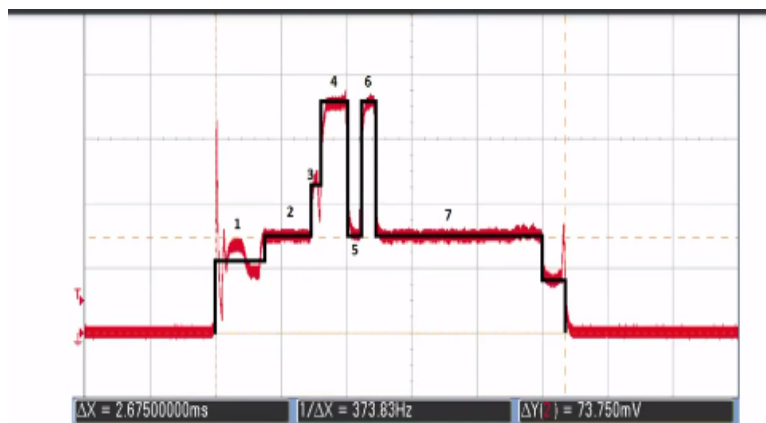


Figure 13- Current Waveform Split into Sections

The oscilloscope cursors can be used to get accurate timing and current measurements for each state. For some states, such as state 1 in Figure 13, the current draw is not steady. For very accurate measurements, the section could be split up into smaller sections; however

Now if you take the current waveform split into different sections, you have 1, 2, 3, 4, 5, 6, 7. You start naming them because here, you really have not labeled anything. So now you start labeling. What is 1, what is 2, what is 3, what is 4, 5 and all that.

**(Refer Slide Time: 53:20)**



The oscilloscope cursors can be used to get accurate timing and current measurements for each state. For some states, such as state 1 in Figure 13, the current draw is not steady. For very accurate measurements, the section could be split up into smaller sections; however using the divisions shown and guessing an average current value should provide fairly accurate estimates. Table 1 below shows the measurements from Figure 13:

	Time [ $\mu$ s]	Current [mA]
State 1 (wake-up)	400	6.0
State 2 (pre-processing)	340	7.4
State 3 (pre-Rx)	80	11.0
State 4 (Rx)	190	17.5
State 5 (Rx-to-Tx)	105	7.4
State 6 (Tx)	115	17.5
State 7 (post-processing)	1280	7.4
State 8 (pre-Sleep)	160	4.1

Table 1- Measurements from Capture in Figure 13

Note that the exact timings of the pre-processing and post-processing may differ; however the sum of the two values will always be the same. Since the current draw is the same during pre-processing and post-processing, these differences will not affect the average current.

Next, similar measurements need to be made using a capture in which the CC2541 is awake

It turns out that 1 is wakeup, 2 is pre- processing. This is pre Rx, then is Rx, Rx-to-Tx, Tx, post-processing and pre-sleep. This is same set of states that I showed you right in the beginning of this article, okay.

**(Refer Slide Time: 53:35)**

State 3 (pre-Rx)	80	11.0
State 4 (Rx)	190	17.5
State 5 (Rx-to-Tx)	105	7.4
State 6 (Tx)	115	17.5
State 7 (post-processing)	1280	7.4
State 8 (pre-Sleep)	160	4.1

Table 1- Measurements from Capture in Figure 13

Note that the exact timings of the pre-processing and post-processing may differ; however the sum of the two values will always be the same. Since the current draw is the same during pre-processing and post-processing, these differences will not affect the average current.

Next, similar measurements need to be made using a capture in which the CC2541 is awake for the longest slot, which measures 2.775 ms. Once you have the capture, use the same process as before to take measurements. Doing so will result in the following values:

	Time [ $\mu$ s]	Current [mA]
State 1 (wake-up)	400	6.0
State 2 (pre-processing)	315	7.4
State 3 (pre-Rx)	80	11.0
State 4 (Rx)	275	17.5
State 5 (Rx-to-Tx)	105	7.4
State 6 (Tx)	115	17.5

This is the note of the exact timings of pre-processing and post-processing may differ. However, the sum of the two values will always be the same, right. Since the current draw is the same during pre-processing and post-processing these differences will not affect the average current. So essentially, he is interpreting from these above pictures a few things.

Next, similar measurements need to be made using a capture in CC2541 is awake for the longest slot. This is for 2.6 millisecond; that is for shorter. This is for longer, which measures 2.775 milliseconds. So you get these numbers.

**(Refer Slide Time: 54:12)**

---



---

**Application Note AN092**

State 7 (post-processing)	1325	7.4
State 8 (pre-Sleep)	160	4.1

**Table 2- Measurements from Capture with Longest Awake Time**

**5.4 Performing Measurements of Sleep Current**

In addition to the active current, a very important metric for calculating the battery life is the sleep current. This is important for battery life, because in most use-cases the CC2541 will spend the majority of the time in PM2 while connected, waiting for the next connection event.

The easiest way to measure the PM2 current is to use an ammeter or a digital multimeter (DMM), with the setup shown in Figure 14.

---

Now what you do, these are the measurements for the longest awake.

**(Refer Slide Time: 54:15)**

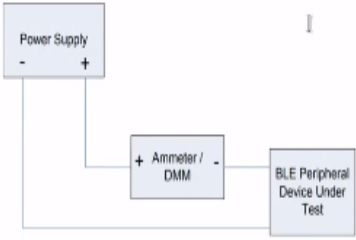
---

**Table 2- Measurements from Capture with Longest Awake Time**

**5.4 Performing Measurements of Sleep Current**

In addition to the active current, a very important metric for calculating the battery life is the sleep current. This is important for battery life, because in most use-cases the CC2541 will spend the majority of the time in PM2 while connected, waiting for the next connection event.

The easiest way to measure the PM2 current is to use an ammeter or a digital multimeter (DMM), with the setup shown in Figure 14.



**Figure 14- Test Setup using Ammeter for Sleep Current Measurement**

---

Then what you do performing measurements on sleep current, you connect an ammeter and then you make a few measurements. Then you find out that. You get Keyfob will still be around 13.5 microamp higher than it should be. And you add that also and then put everything together. I will show you what exactly he does in terms of putting everything together.

(Refer Slide Time: 54:40)



Figure 17- 1  $\mu$ A PM2 Sleep Current Measured on Keyfob

### 5.5 Formulas and Calculations

Before taking into account the current consumed while the device is sleeping, you can first calculate the average current draw during the connection event. For this, you can use the following formula:

$$\text{Average current during connection event} = \frac{[(\text{State 1 time}) \times (\text{State 1 current}) + (\text{State 2 time}) \times (\text{State 2 current}) + \dots]}{(\text{Total awake time})}$$

It does not matter whether units of  $\mu$ s or ms are used in the time calculation, as long as they are used consistently for each state and for the total. In the case of the shortest wake time, you can use values from Table 1 in the formula:

$$\frac{[(400 \mu\text{s}) \times (6 \text{ mA}) + (340 \mu\text{s}) \times (7.4 \text{ mA}) + (80 \mu\text{s}) \times (11 \text{ mA}) + (190 \mu\text{s}) \times (17.5 \text{ mA}) + (105 \mu\text{s}) \times (7.4 \text{ mA}) + (115 \mu\text{s}) \times (17.5 \text{ mA}) + (1280 \mu\text{s}) \times (7.4 \text{ mA}) + (165 \mu\text{s}) \times (4.1 \text{ mA})]}{(2675 \mu\text{s})} = 8.2463 \text{ mA}$$

The average current consumption during a single connection event with the shortest slot, a wake time of 2.675ms, is calculated to be approximately 8.2463 mA.

By repeating the calculation using the values from Table 2, we can get the average current

Very simple folks. This is state 1 into current of that state. State 2 into the current. Take all of this add up all this and divide it by the total awake time. So what is state 1? You already know what state 1 is. For the long it is one number, for the short it is another number. This is for long right, this is for long. I suppose that is what you would have written here. Yeah, this is the longest. So state 1, 2, 3, 4, 5, 6 you know all that.

So you take those states. And you multiply that into the current consumption of that. See this how long that state is alive. How long the state 1 is in terms of time, multiply it into the corresponding current, add the next state, add the next state next state and all that and arrive at this number after you divide by the total awake time. You will get 8.2463 milliamperes. Now this is for the shortest.

(Refer Slide Time: 55:39)

---

It does not matter whether units of  $\mu\text{s}$  or  $\text{ms}$  are used in the time calculation, as long as they are used consistently for each state and for the total. In the case of the shortest wake time, you can use values from Table 1 in the formula:

$$[(400 \mu\text{s})(6 \text{ mA}) + (340 \mu\text{s})(7.4 \text{ mA}) + (80 \mu\text{s})(11 \text{ mA}) + (190 \mu\text{s})(17.5 \text{ mA}) + (105 \mu\text{s})(7.4 \text{ mA}) + (115 \mu\text{s})(17.5 \text{ mA}) + (1280 \mu\text{s})(7.4 \text{ mA}) + (165 \mu\text{s})(4.1 \text{ mA})] / (2675 \mu\text{s}) = 8.2463 \text{ mA}$$

The average current consumption during a single connection event with the shortest slot, a wake time of 2.675ms, is calculated to be approximately 8.2463 mA.

By repeating the calculation using the values from Table 2, we can get the average current consumption for the connection event in the case of the longest slot:

$$[(400 \mu\text{s})(6 \text{ mA}) + (315 \mu\text{s})(7.4 \text{ mA}) + (80 \mu\text{s})(11 \text{ mA}) + (275 \mu\text{s})(17.5 \text{ mA}) + (105 \mu\text{s})(7.4 \text{ mA}) + (115 \mu\text{s})(17.5 \text{ mA}) + (1325 \mu\text{s})(7.4 \text{ mA}) + (160 \mu\text{s})(4.1 \text{ mA})] / (2775 \mu\text{s}) = 8.5312 \text{ mA}$$

The average current consumption during a single connection event with the longest slot, a wake time of 2.775ms, is calculated to be approximately 8.5312 mA.

---

If you take 2.775, which is the longer one, how much do you get? You get 8.5312 milliamperes. So you got two numbers. Now obviously, you have to take an average of both of them, right? So he does an average of both of them.

**(Refer Slide Time: 55:53)**

---

Once again, be sure to use consistent time units throughout the formula. You must also once again perform this calculation twice; once for each of the shortest and longest wake time. Using the values from the previous calculations and the sleep current measurements, you can calculate the average current for the shortest slot, a wake time of 2.675 ms:

$$[(1000 \text{ ms} - 2.675 \text{ ms})(0.001 \text{ mA}) + (2.675 \text{ ms})(8.2463 \text{ mA})] / (1000 \text{ ms}) = 0.0230 \text{ mA}$$

Repeating the calculation for the longest slot, a wake time of 2.775ms,:

$$[(1000 \text{ ms} - 2.775 \text{ ms})(0.001 \text{ mA}) + (2.775 \text{ ms})(8.5312 \text{ mA})] / (1000 \text{ ms}) = 0.0247 \text{ mA}$$

The final step to getting the average current while connected is to take the weighted average of the two averages that were previously calculated. Previously it was determined that the shortest slot, a wake time of 2.675 ms occurred 50% of the time, while the longest slot, a wake time of 2.775ms, occurred 73% of the time. The weighted average is then calculated as follows:

$$(0.0230 \text{ mA})(0.50) + (0.0247 \text{ mA})(0.50) = 0.0239 \text{ mA}$$

The average current consumption while the device is in a connected state is approximately 0.024 mA (24  $\mu\text{A}$ ). You can now use this value to calculate the amount of time that you can expect the battery last while running continuously in a connection. The total hours of battery life can be calculated using the following simple formula:

Expected battery life running continuously in a connected state =

---

And then he arrives at something which is essentially, see he takes 0.5 and he gives a weight of this current into 0.5 and he gives this current into 0.5, weightage of these two currents, right? If you do all that, you will end up with 0.0239 milliamperes. This is with proper waiting for the short and the long as well. Okay. Now because this is the average current, okay.

Basically you are taking connection interval minus total wakeup time into average sleep current plus total awake time into average current during connection interval divided by the connection interval. Apply this formula, you will get this number, okay?

**(Refer Slide Time: 56:41)**

---

follows:

$$(0.0230 \text{ mA}) \cdot (0.50) + (0.0247 \text{ mA}) \cdot (0.50) = 0.0239 \text{ mA}$$

The average current consumption while the device is in a connected state is approximately 0.024 mA (24 uA). You can now use this value to calculate the amount of time that you can expect the battery last while running continuously in a connection. The total hours of battery life can be calculated using the following simple formula:

$$\text{Expected battery life running continuously in a connected state} = \frac{\text{Battery capacity}}{\text{Average current while connected}}$$

If you assume that the battery capacity is 230 mAh (a common capacity value for a CR2032 coin cell battery) and use the average current calculated from before, you can calculate the expected battery life:

$$(230 \text{ mAh}) / (0.024 \text{ mA}) = 9583 \text{ hours}$$

The battery can be expected to last for 9583 hours, or approximately 400 days, while running continuously in a connected state with a 1 second connection interval and zero slave latency.

#### 5.6 Using Excel Spreadsheet for Calculations

An Excel spreadsheet is provided along with this application note that can be used to perform simple calculations. The first "EXAMPLE\_CC2541" worksheet in the spreadsheet features the

---

Now it is so straightforward now. Once you get this number, you know that your CR2032 is 230 milliampere hour battery. Once this battery, you know its capacity, it is from the datasheet if you buy this battery, this is its current consumption. If you divide this 0.0239 milliampere, which is 0.24 divided by this 230, you will get some number. This is 9583 hours. What does it mean? It simply means 400 days.

That means it is a little over 1, a little over 1 here. But this calculation is for what connection interval? Go back and look at the original picture. It was for 1 second. Now folks, you now make a call. How many years do you want your Keyfob to survive, and then accordingly, play with this connection interval, you will get to know the lifetime of the system. That is all. Thank you very much.

I hope you understand the importance of CI advertising interval, the importance of CI and so on. That will give you a rounded view of why these parameters are important in BLE. So this module is about arriving at how to use BLE effectively so that you can give the a good lifetime for your Keyfobs or systems which use BLE in their for

their communication units. So that is all I have for the moment. I hope you enjoyed going through this article. Thank you very much.