ECEN 5023-001, -001B, -740

Mobile Computing & IoT Security

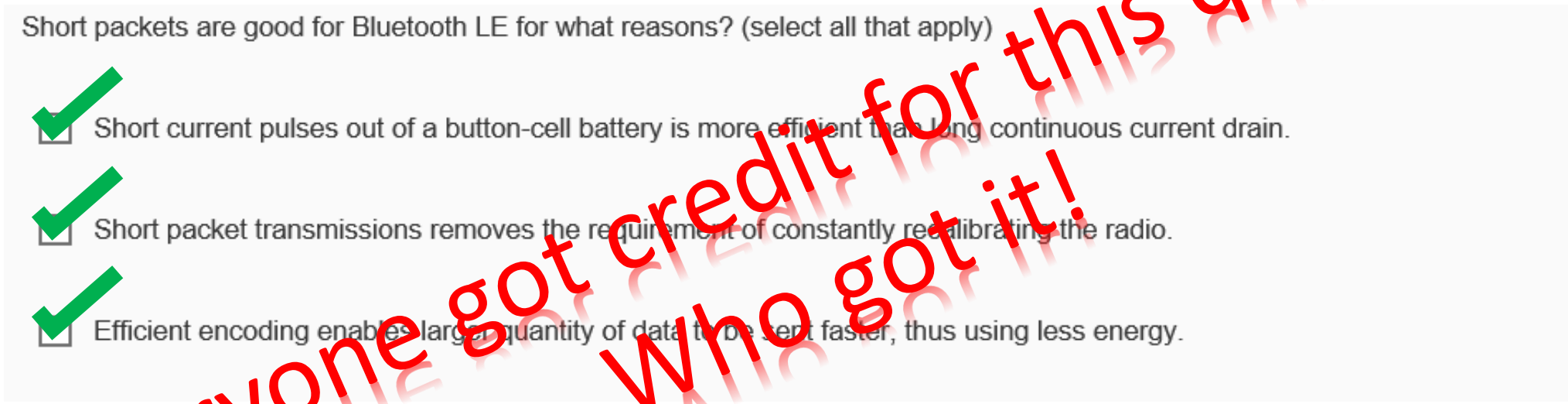
Lecture #12

23 February 2017

Be Boulder.

University of Colorado Boulder

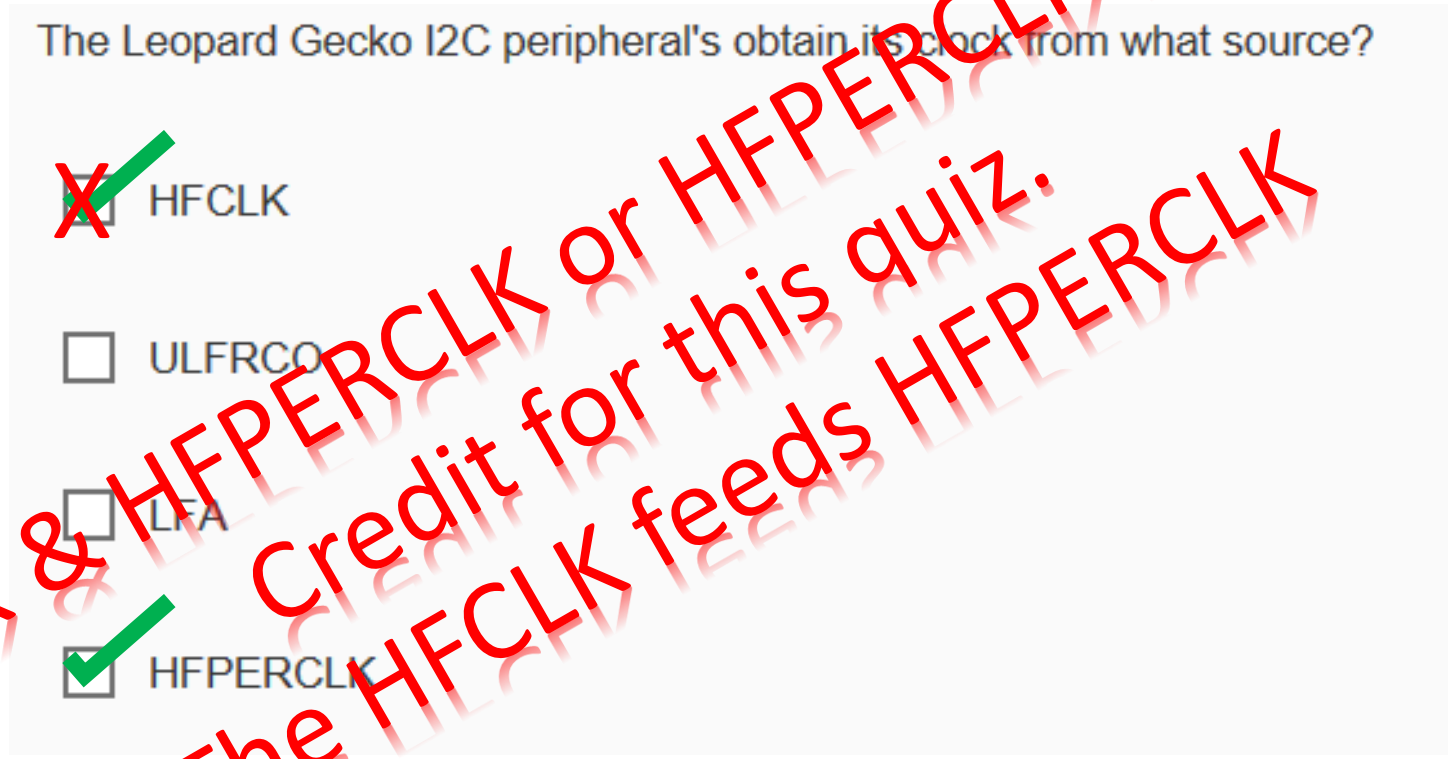
Quiz Review



Everyone got credit for this question

Who got it!

Quiz 5 Review



HFCLK & HFPERCLK or HFPERCLK received Credit for this quiz. The HFCLK feeds HFPERCLK

Helpful I2C hints from AN0011SW application note

```
/* Initializing I2C1 ports for TSL2651 sensor */
/* Output value must be set to 1 to not drive lines low... We set
*/
/* SCL first, to ensure it is high before changing SDA. */
GPIO_PinModeSet(I2C1_SCL_Port, I2C1_SCL_Pin, gpioModeWiredAnd, 1);
GPIO_PinModeSet(I2C1_SDA_Port, I2C1_SDA_Pin, gpioModeWiredAnd, 1);

/* Toggle I2C SCL 9 times to reset any I2C slave that may require
it */
for (int i=0;i<9;i++) {
   GPIO_PinOutClear(I2C1_SCL_Port, I2C1_SCL_Pin);
   GPIO_PinOutSet(I2C1_SCL_Port, I2C1_SCL_Pin);
   }
```

PARAMETER MEASUREMENT INFORMATION

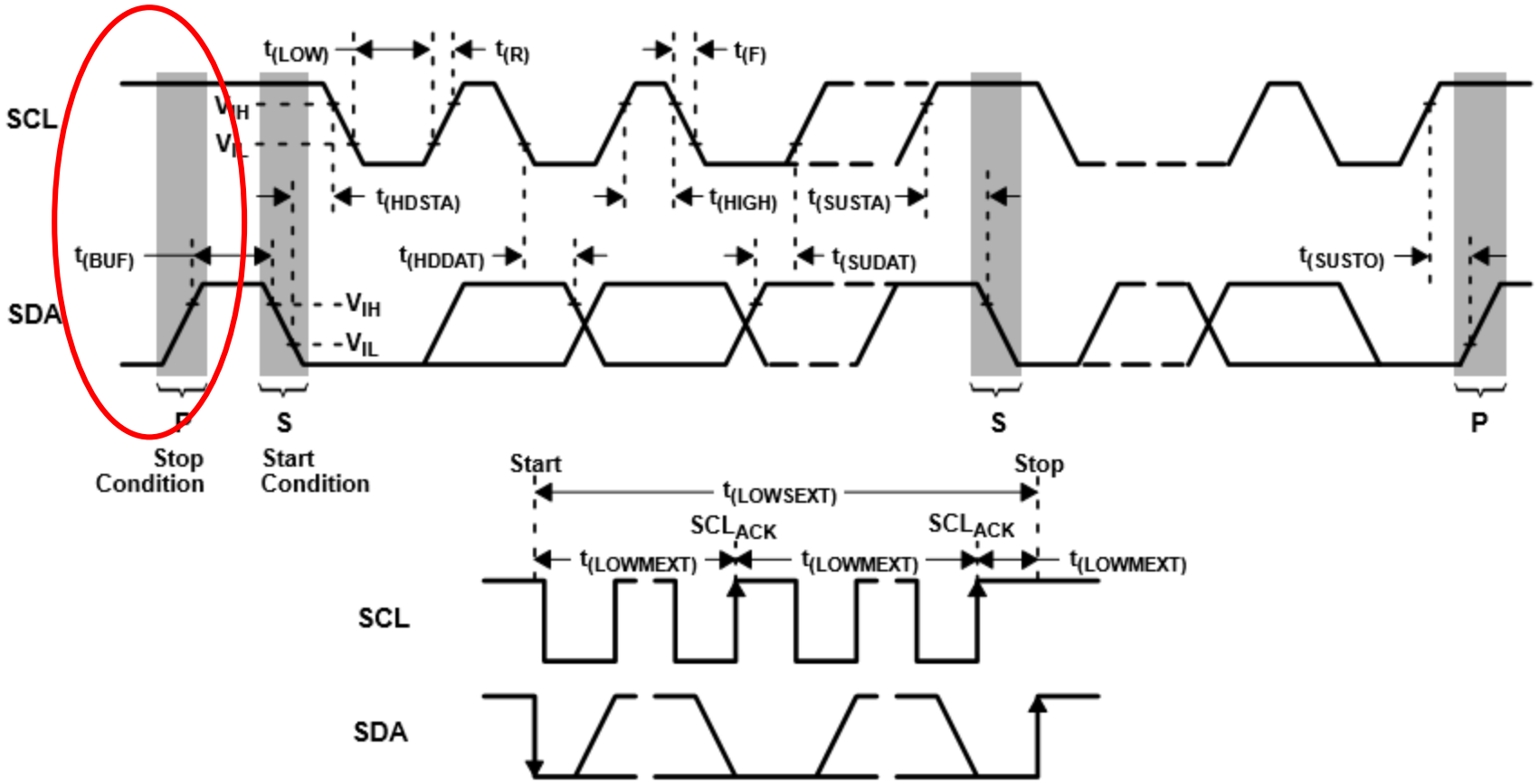


Figure 1. Timing Diagrams

Helpful I2C hints from AN0011SW application note

```
/* Initializing I2C1 ports for TSL2651 sensor */
/* Output value must be set to 1 to not drive lines low... We set
*/
/* SCL first, to ensure it is high before changing SDA. */
GPIO_PinModeSet(I2C1_SCL_Port, I2C1_SCL_Pin, gpioModeWiredAnd, 1);
GPIO_PinModeSet(I2C1_SDA_Port, I2C1_SDA_Pin, gpioModeWiredAnd, 1);

/* Toggle I2C SCL 9 times to reset any I2C slave that may require
it */
for (int i=0;i<9;i++) {
   GPIO_PinOutClear(I2C1_SCL_Port, I2C1_SCL_Pin);
   GPIO_PinOutSet(I2C1_SCL_Port, I2C1_SCL_Pin);
   }
```

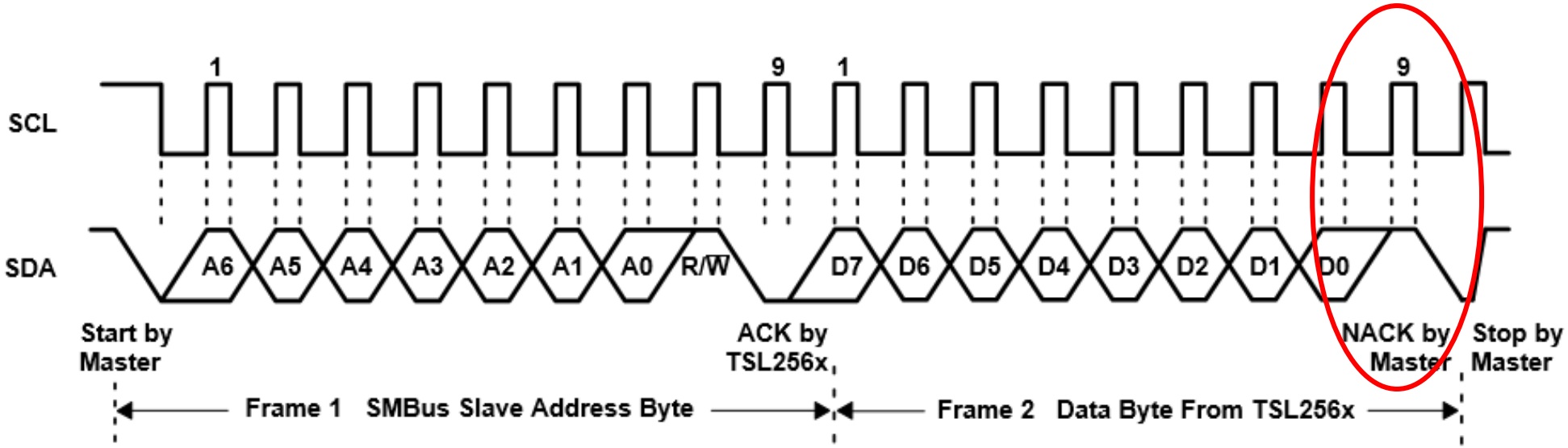


Figure 3. Example Timing Diagram for SMBus Receive Byte Format

Agenda

- Class Announcements
- Atmel ATSAMB11 tutorial assigned
- Bluetooth Smart / Low Energy

Class Announcements

- Quiz #6 is due at 11:59pm on Sunday, February 26th, 2017
- Implementing an I2C Sensor assignment is now due Friday, February 24th, at 11:59pm
- Atmel ATSAMB11 dev kits will be distributed at the end of class today which will be required for the ATSAMB11
- Atmel tutorial assignment due before the March 7th
  - Questions regarding the tutorial will be on the mid-term
- Mid-term will be held in class on Tuesday, March 7th, at 6:30 in class
  - For on campus students, you must be in class for the exam
  - For distant learners, the mid-term will be due by 6:00pm on Thursday, March 9th

Atmel ATSAMB11 tutorial

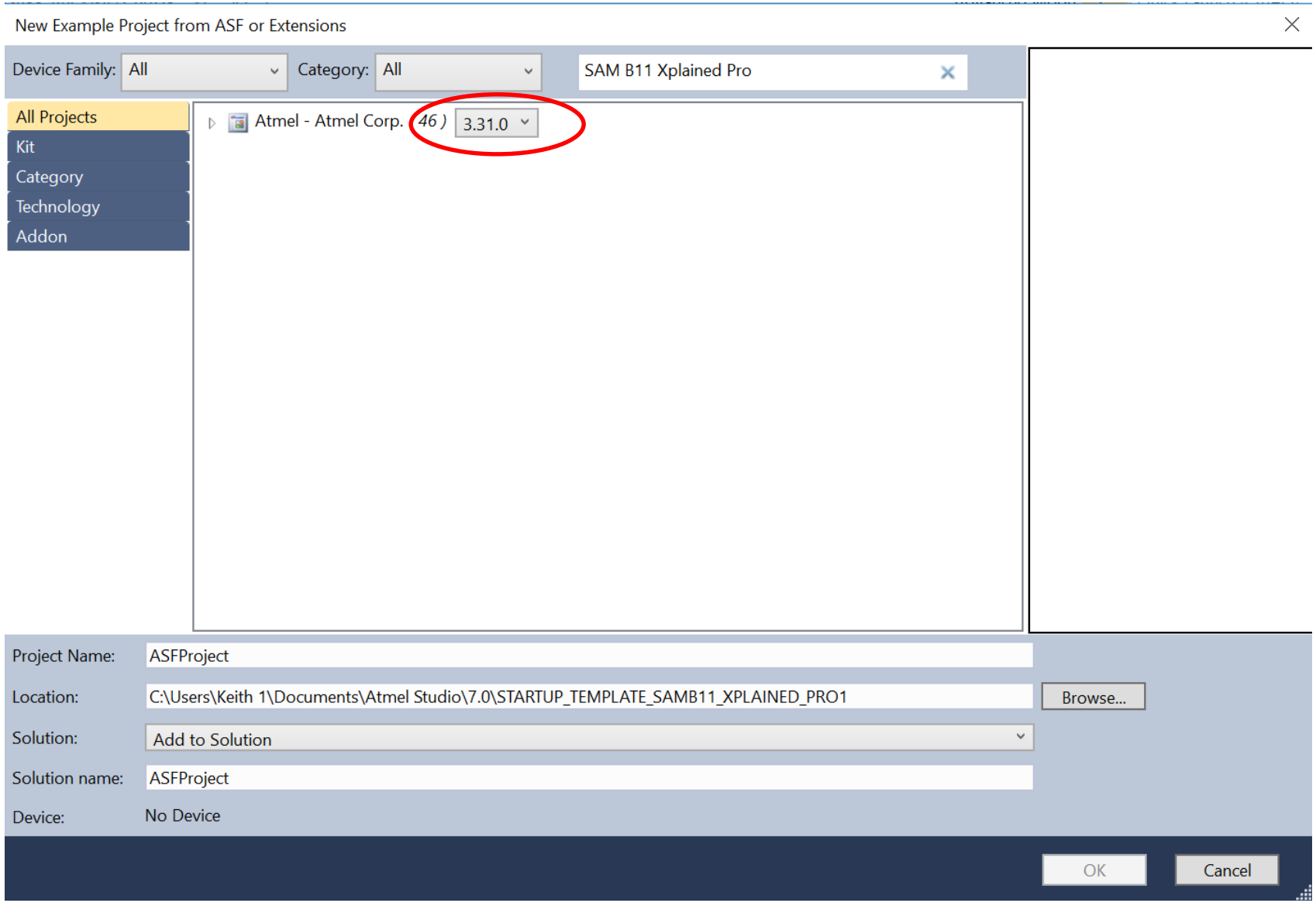
- Steps through the ATSAMB11 being used in two applications
  - A beacon
  - A thermometer
- It should take around 5-6 hours to go through the tutorial
- The tutorial is a very good review of how Bluetooth operates
- No work to hand in, but there will be questions on the Mid-Term from this tutorial

Atmel ATSAMB11 tutorial

- To perform the tutorial:
  - Download and install Atmel Studio 7
    - It only works on Microsoft's Window operating system
    - Atmel Studio 7 is installed on the computers in the ESE lab
  - Download and install the Atmel SMART and Beacon applications to a smart phone or tablet
- The tutorial will be distributed via Slack
  - Slack channel: #atmeltutorial

Atmel ATSAMB11 tutorial

- When asked for the example project, you must select ASF 3.31 to match the material in the tutorial



BLE: Time is Energy

Energy = Power x Time

- Optimizing a number of important and repetitive action is a must
  - Discovering devices
  - Connecting to devices
  - Sending data

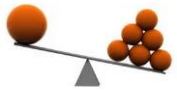
➢Reducing the time for these activities

➢Reduces the energy consumed for these activities

➢Lengthening the battery life

BLE: Time is Energy (Advertising)

- Advertising to be discovered requires a device to transmit a very short message 3 times per second and listen immediately afterwards if it wants to connect
- Three transmits are done, one per advertising frequency channel, for robustness.
  - If the advertising channels was just one frequency band, if that frequency became blocked or a lot of interference, devices would not be able to connect
  - If the number of channels was much higher than 3, such as 16, then the device would spend more time and energy by having its radio on 5x+
- Searching for a device that is transmitting requires the radio to be on a long period of time which requires more energy than the advertising device
  - Thus, the device that typically has more resources, larger battery, is listening and will become the master
  - The advertiser will normally be the smaller device with the smaller battery and become the slave

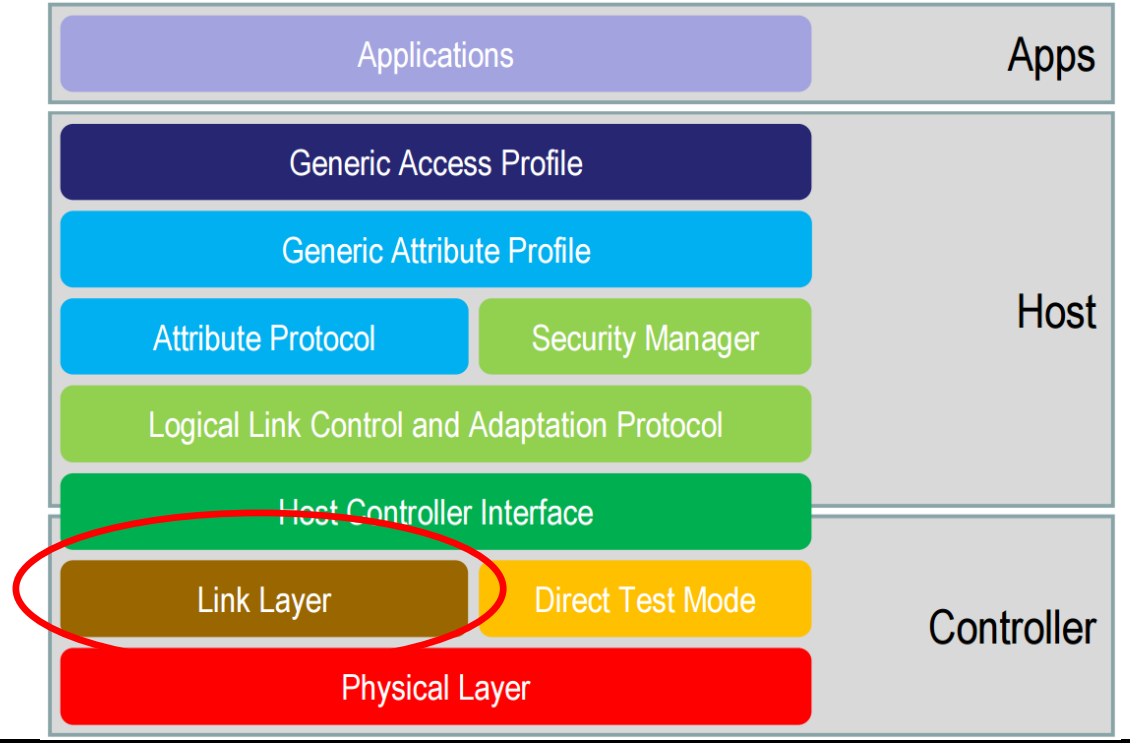


BLE: Time is Energy (Packet size)

- Short packets are good for three reasons:
  - Efficient encoding allows short packets to transmit as much as larger packets faster using less energy
  - Restricting the radio devices to only use short packets removes the requirement of constantly recalibrating the radio within the controller due to internally heating while the radio is operational
  - Short packets reduces peak power consumption which enables more energy to be taken out of the battery

BLE: Link Layer

- Two types of Link Layer Channels:
  - Advertising channels
    - Broadcast data
    - Advertise that they are connectable and discoverable
    - Scan
    - Initiate connections
  - Data channels
    - Only used once a connection has been established
    - And, only when data needs to flow



CSR: Bluetooth 4.0 Low Energy

http://chapters.comsoc.org/vancouver/BTLER3.pdf

Bluetooth Low Energy: The Developer's Handbook By Robin Heydon

BLE: Link Layer packet structure

- Basic packet structure is the same for both advertising channels and data channels
  - A minimum of 80 bits of addressing, header, and check information for every packet
- The packets are optimized to increase their robustness by using an 8-bit preamble that is sufficiently large to allow the receiver to synchronize bit timing and set the radio's automatic gain control
- A 32-bit access address that is fixed for advertising packets, but can be completely random and private for data packets
- An 8-bit header packet to describe the contents of the packet
- An 8-bit length field to describe the payload length
- 0-296 bit payload
- And, a 24-bit cyclic redundancy check (CRC) value to ensure that there are not bit errors in the received packet

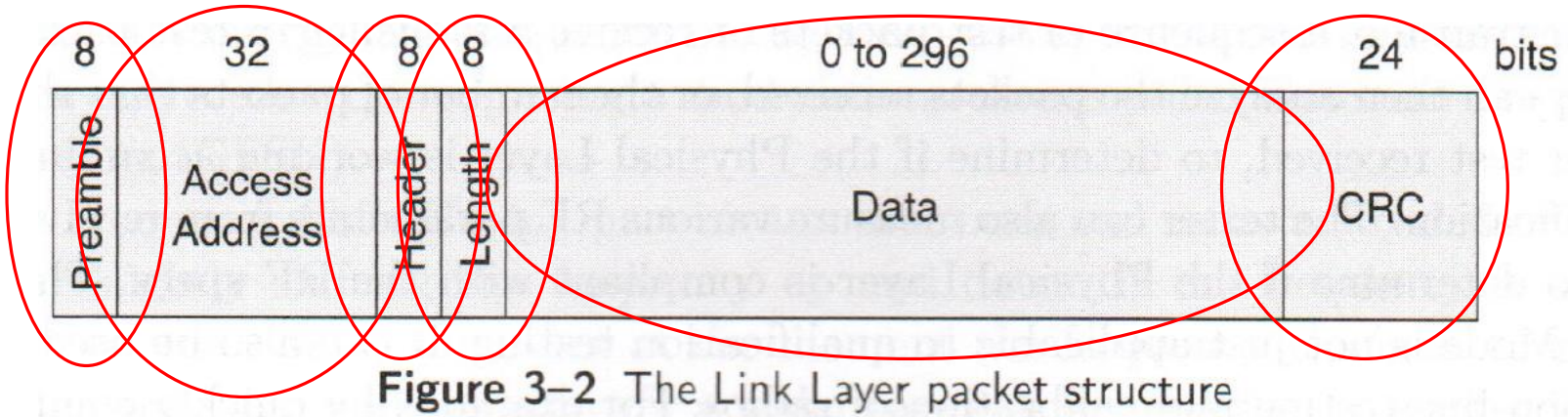
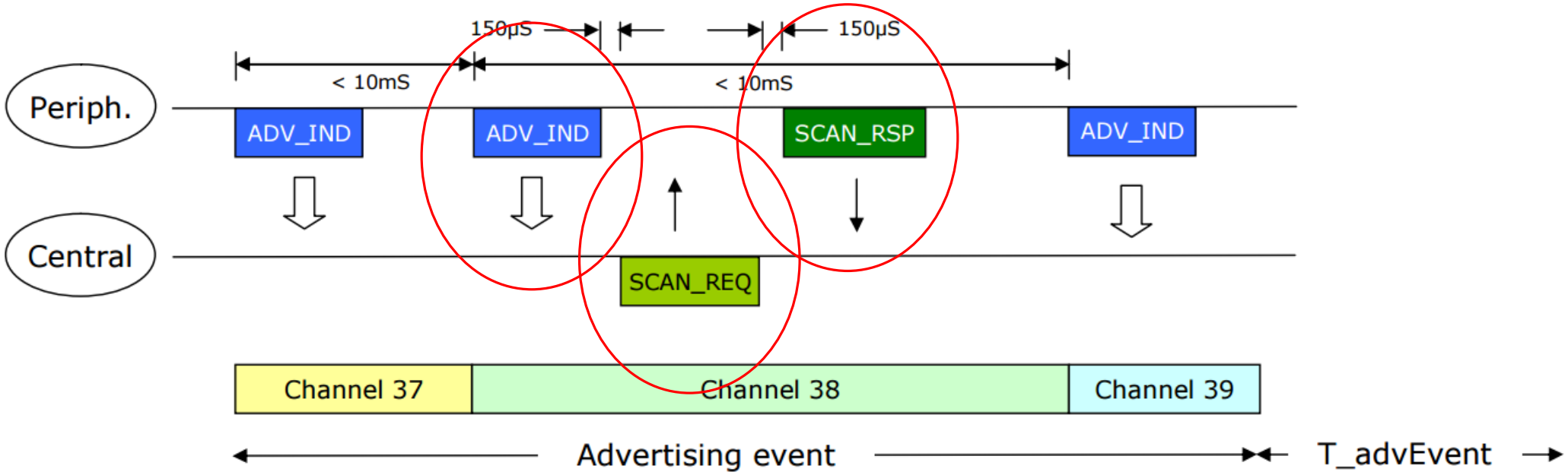


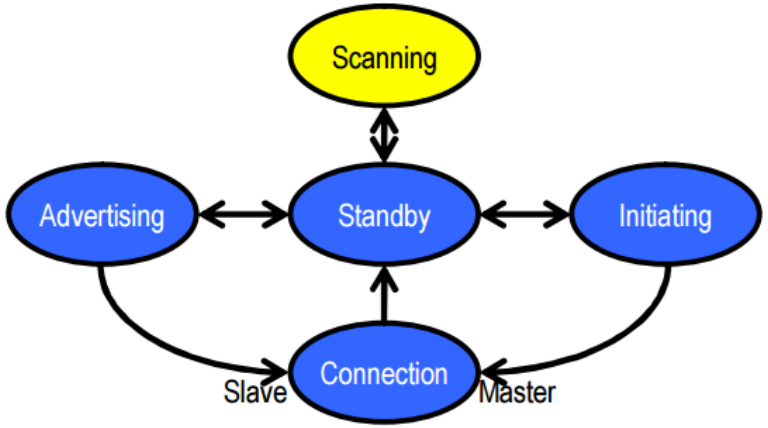
Figure 3–2 The Link Layer packet structure

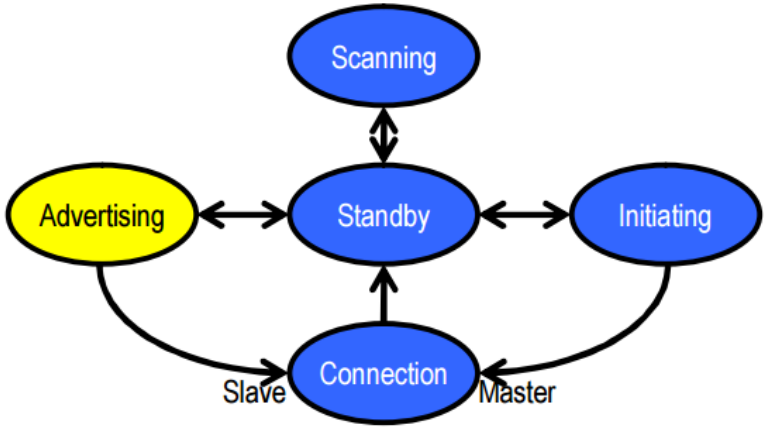
BLE: Advertising

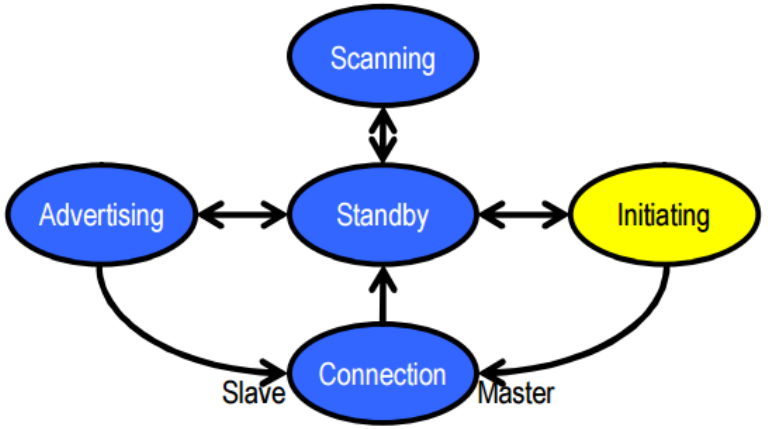


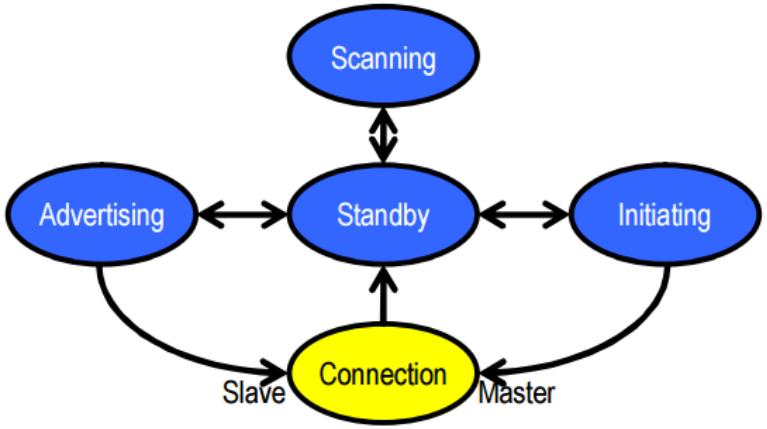
- Devices can advertise for a variety of reasons:
  - To broadcast promiscuously
  - To transmit signed data to a previously bonded device
  - To advertise their presence to a device wanting to connect
  - To reconnect asynchronously due to a local event

BLE: The 4 active states during an advertising event

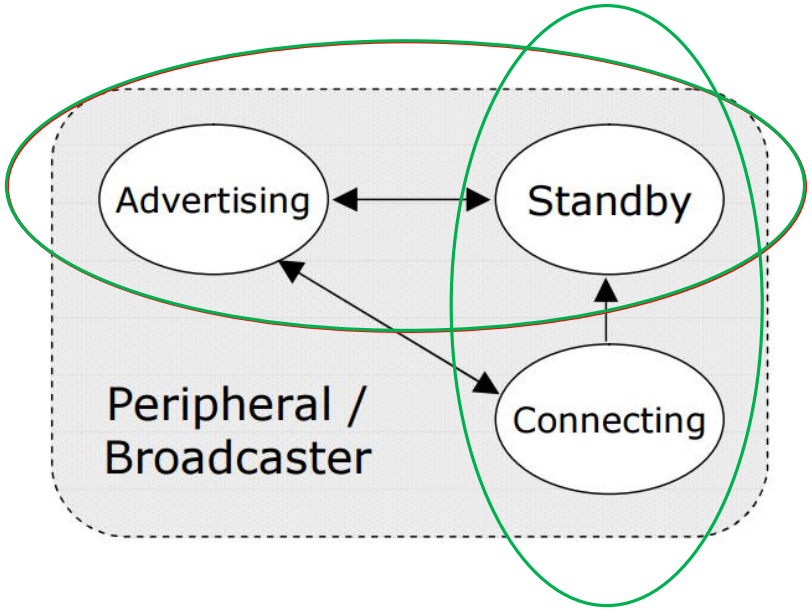




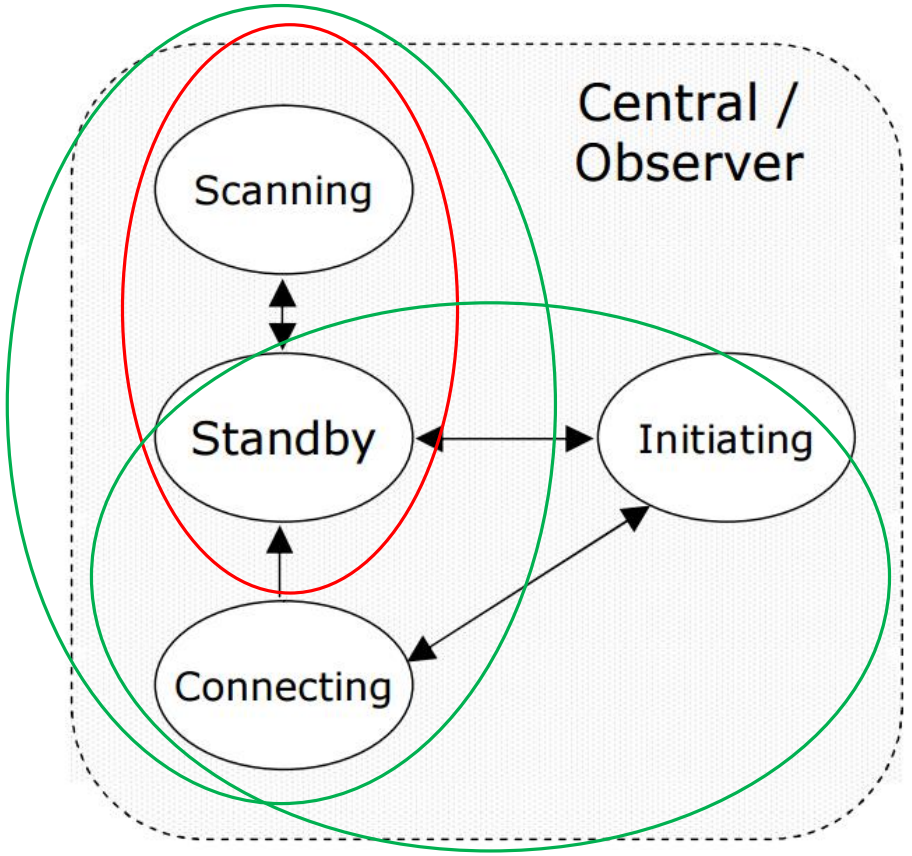




BLE: Advertising (Peripheral and Central States)

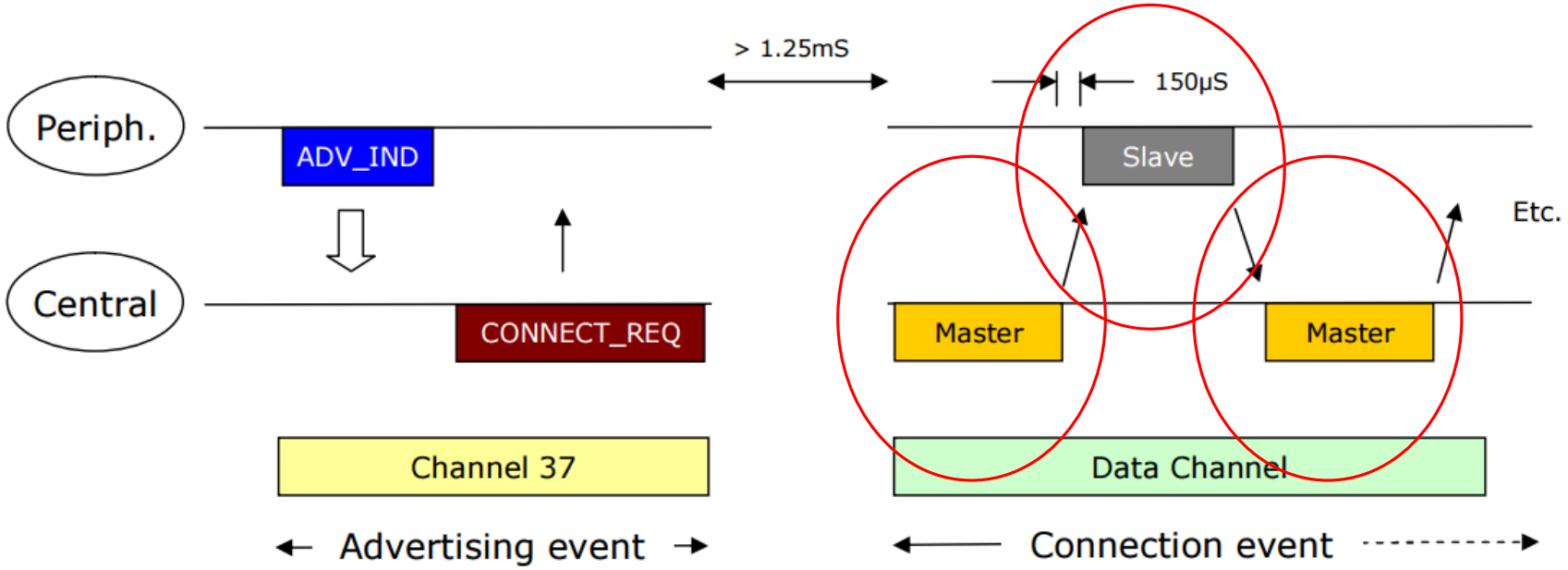


A Broadcaster cannot enter the Connecting state.



An Observer cannot enter the Initiating State.

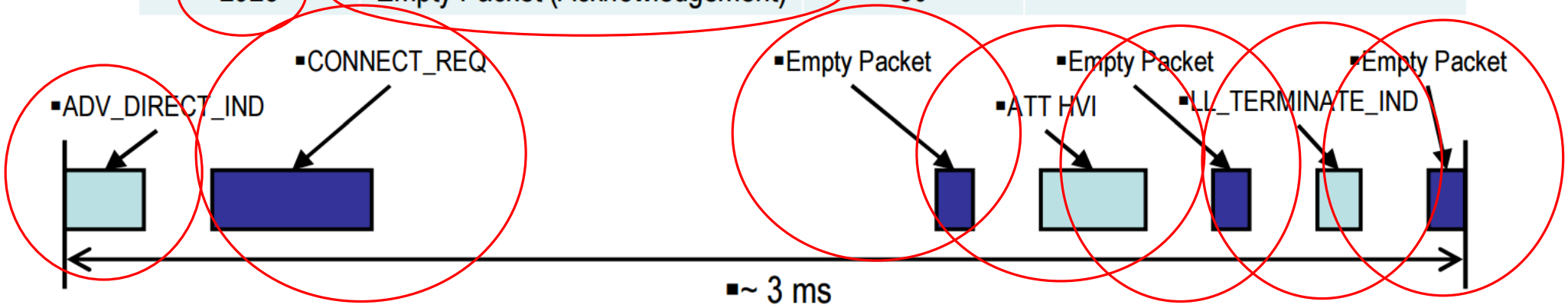
BLE: Data transactions



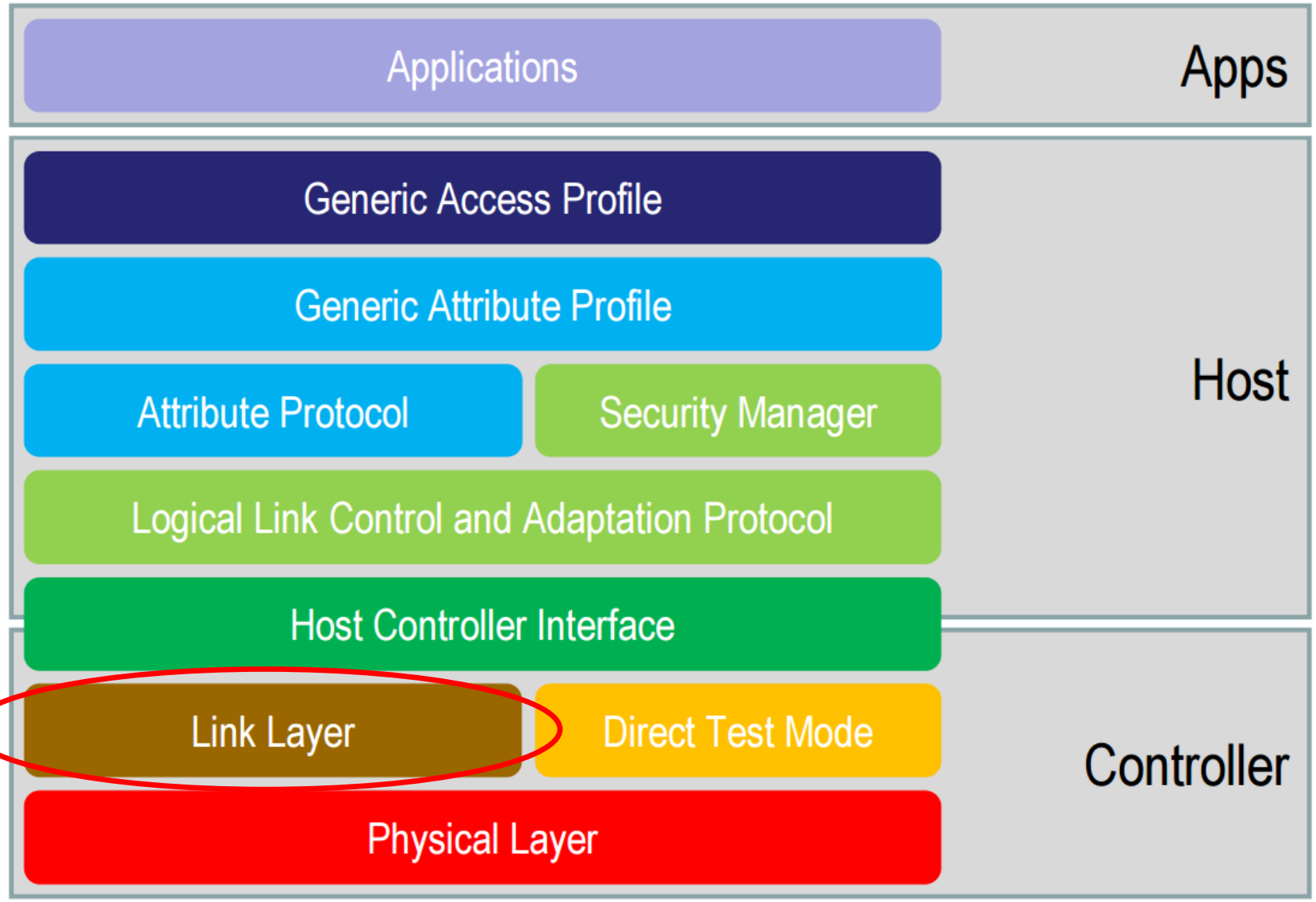
- Once a connection is made:
  - Master informs slave of hopping sequence and when to wake
  - All subsequent transactions are performed in the 37 data channels
  - Transactions can be encrypted
  - Both devices can go into deep sleep between transactions

BLE – Minimum time for data transaction

Time (us)	Master Tx	Radio Active (us)	Slave Tx
0		176	ADV_DIRECT_IND
326	CONNECT_REQ	352	
1928	Empty Packet	80	
2158		144	Attribute Protocol Handle Value Indication
2452	Empty Packet (Acknowledgement)	80	
2682		96	LL_TERMINATE_IND
2928	Empty Packet (Acknowledgement)	80	



BLE: Stack


Applications
Apps
Generic Access Profile
Generic Attribute Profile
Attribute Protocol
Security Manager
Logical Link Control and Adaptation Protocol
Host Controller Interface
Host
Link Layer
Direct Test Mode
Physical Layer
Controller

BLE: Optimizing for Low Power/Energy

- Primary methods to reducing power are:
  - Keeping the packets short
  - Using a high physical bit rate
  - Providing low overhead
  - Optimized acknowledgement scheme
  - Single-channel connection events
  - Using offline encryption (encrypting when the radio is off)
- Two types of power consumption that are critical for lower power consumption
  - Low peak-power consumption to optimize the use of button-cell batteries ($P=I^2R$)
  - Low power-per-application bit to enable a device to be used a long time sending a defined quantity of application data

BLE: Short Packets

- Low power designer dilemma:
  - To make a radio more stable, more circuity is required that increases cost and power consumption
  - The BLE radio solves this dilemma by making the packet length significantly small that heating effects are minimized
    - The heating effect does not require a very long packet to cause this heating problem
    - The 3 millisecond packets in Bluetooth Classic are long enough to cause this heating issue
  - The BLE specifications take into account the physical properties of the semiconductor that are used to create the radios
  - The goal is to keep packets no more than a few hundredths of milliseconds in length to prevent any semiconductor heating problems
    - Resulting in no requirement for calibration or stabilization circuitry for the radio

BLE: Short Packets

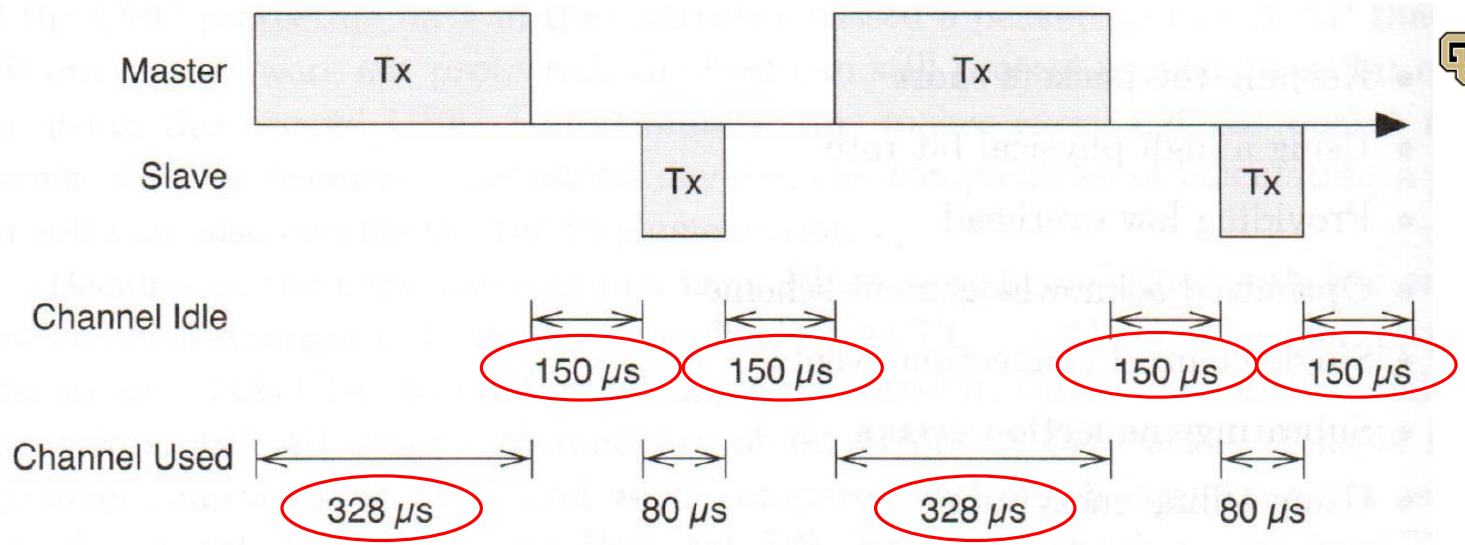
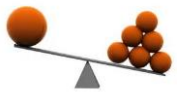


Figure 7–31 Short packets

- The packets are short enough that any drift of the frequency due to heating will not be outside the BLE radio specifications
  - The longest packet in an Advertising Event is 378uS
  - The longest packet in a Connection Event is 328uS
- To further reduce the issue of silicon heating, the specification requires a 150uS gap between "very long" packets to enable the silicon to cool down between packets



  - Removing the requirement of calibrating of the frequencies between transmitting and receiving or receiving and transmitting packets.

BLE: Short Packets

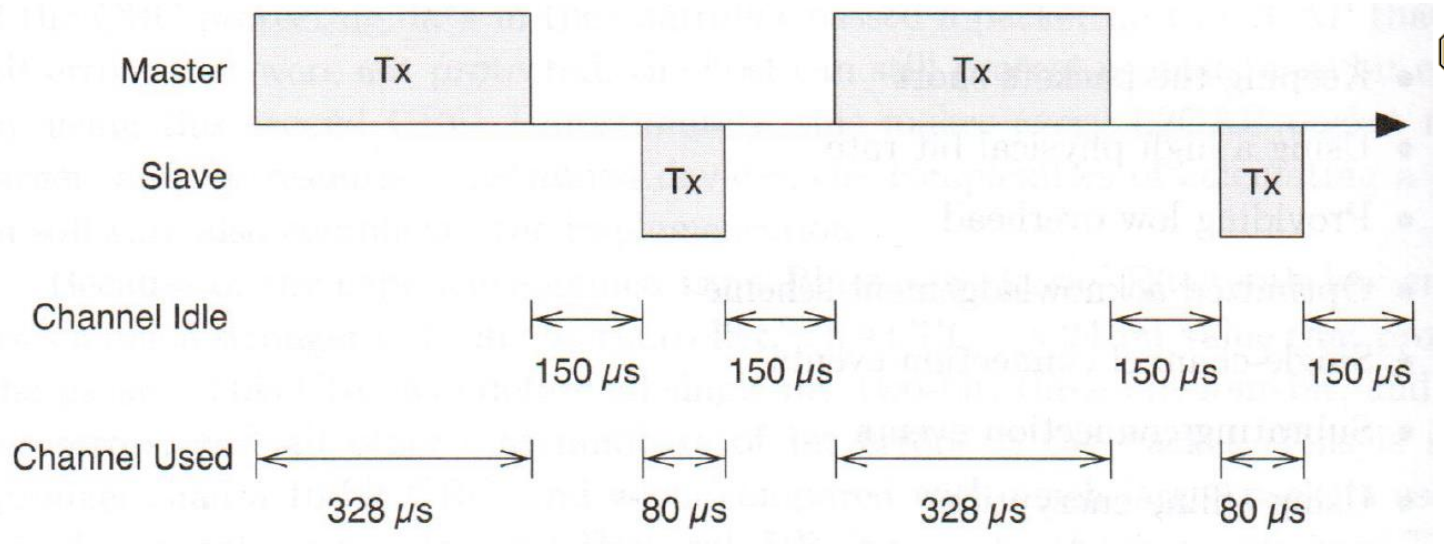


Figure 7–31 Short packets

- The addition of a 150uS cool down periods reduces the maximum duty cycle of transmitting data in one direction on an encrypted link
  - $\frac{maximum\ size\ packet + acknowledge\ packet}{total\ time\ to\ send\ and\ acknowledge\ data}$
  - $\frac{(328+80)}{(328+150+80+150)} = \frac{408}{708}$ = ~58%
- 58% is a very low duty cycle for wireless technology
  - For example, Bluetooth classic is 72% (Increase packet length and time has more effect on the numerator than the denominator)

Reducing bandwidth

BLE: High Bit Rate

- CMOS technology is optimized for gates that do not change state since in digital systems, most gates do not change state
  - Running a 2.4GHz oscillator used for the radio modulation is contrary to what CMOS is optimized, thus consuming a significant amount of energy
  - Since basic technology of the CMOS semiconductor defines a base amount of current/energy for the radio, the efficiency of the modulated signal becomes significant.
    - The quicker that a given amount of data can be transmit, the more efficient the radio
    - The BLE radio is designed for 1Msps transmit rates
    - If BLE could only transmit 250Ksps, the radio power would be 4x
  - Note, modulations schemes that are more complex and transmit data at higher data rates can result in more power/energy due to the complexity of the radio

BLE: Low Overhead

- As discussed, the short the packet, the more energy efficient the radio
  - Reducing the time that the radio is enabled and the time generating the 2.4GHz oscillator
  - BLE overhead includes:
    - Preamble
    - Access address
    - Header
    - Length
    - CRC
    - And optional MIC value

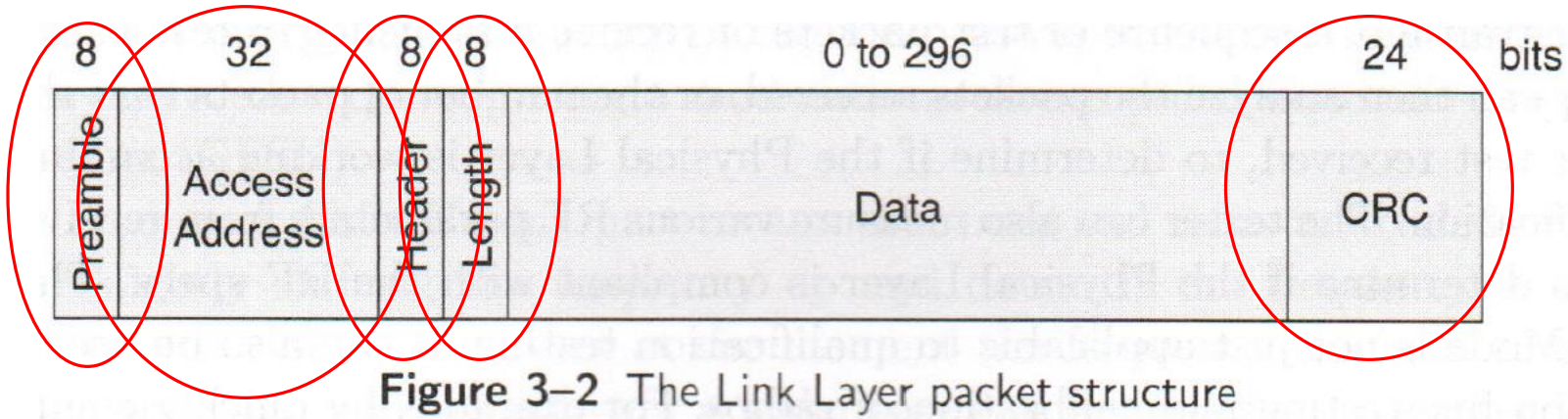


Figure 3–2 The Link Layer packet structure

BLE: Overhead

- Efficiency is measured as:
  - For an unencrypted packet, size of the application data compared with the total packet size required to transmit the application data
  - For an encrypted packet, the efficiency is lower, primarily because of the additional 4 octets of MIC included in each packet

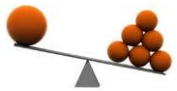
- Compared to other low energy radio technologies, the BLE efficiency is very good
  - For example, ZigBee has a packet overhead of 15 to 31 octets compared to BLE's 10 for unencrypted
  - With ZigBee transmitting 4x slower physical data rates than BLE, a short 4 octet application data in ZigBee could require up to 10 times more energy to transmit than BLE

Table 7–6 The Overhead for Application Data

Packet Type	Application Data Size (octets)	Overhead (octets)	Efficiency (%)
Unencrypted	4	10	29%
Unencrypted	8	10	44%
Unencrypted	16	10	62%
Unencrypted	27	10	73%
Encrypted	4	14	22%
Encrypted	8	14	36%
Encrypted	16	14	53%
Encrypted	27	14	66%

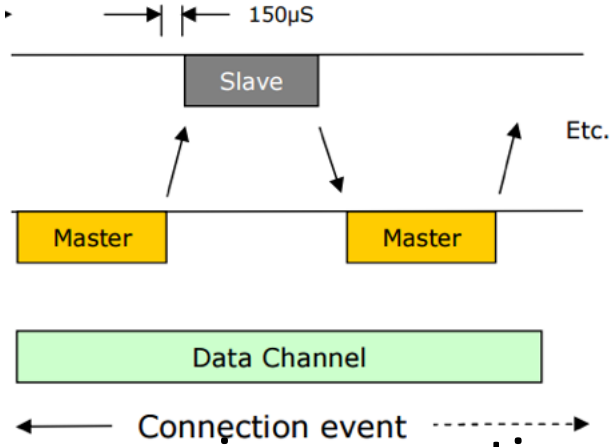
BLE: Acknowledgement Scheme

- The Link Layer acknowledgement scheme does not require an acknowledgement of a packet to be performed or even delivered immediately
  - This non-requirement of an acknowledgement is a departure from Bluetooth Classic
  - In Bluetooth Classic, the receiver must acknowledge the packet at the next opportunity it has to transmit
    - If the acknowledgement is not received immediately, the receiver must signal a negative acknowledgement in the next transmit packet



  - In BLE, every packet sent can acknowledge the last packet transmitted, even if this was transmitted some time ago
    - The BLE scheme enables the transmit acknowledgement to occur when convenient such as when it is ready to transmit for some other reason or allow it to finish transmitting a large amount of data quickly

BLE: Single-Channel Connection Events



- All communication between a master and a slave occur in connection events
  - A connection event is a packet transmitted by the master, followed by the slave, followed by a series of alternating packets send by the slave and master
  - During a connection event, the master and slave stay on the same frequency
  - The assumption is that if the master packet was successfully transmitted, the interference is low enough to enable a good channel for the slave response

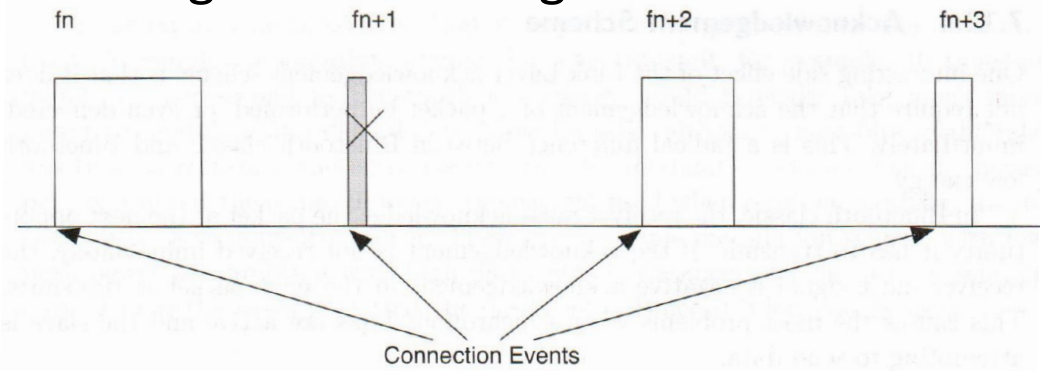


Figure 7–32 Single-channel connection events

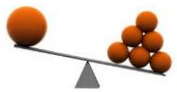
CSR: Bluetooth 4.0 Low Energy
http://chapters.comsoc.org/vancouver/BTLER3.pdf
Bluetooth Low Energy: The Developer's Handbook By Robin Heydon

BLE: Single Channel Connection Events

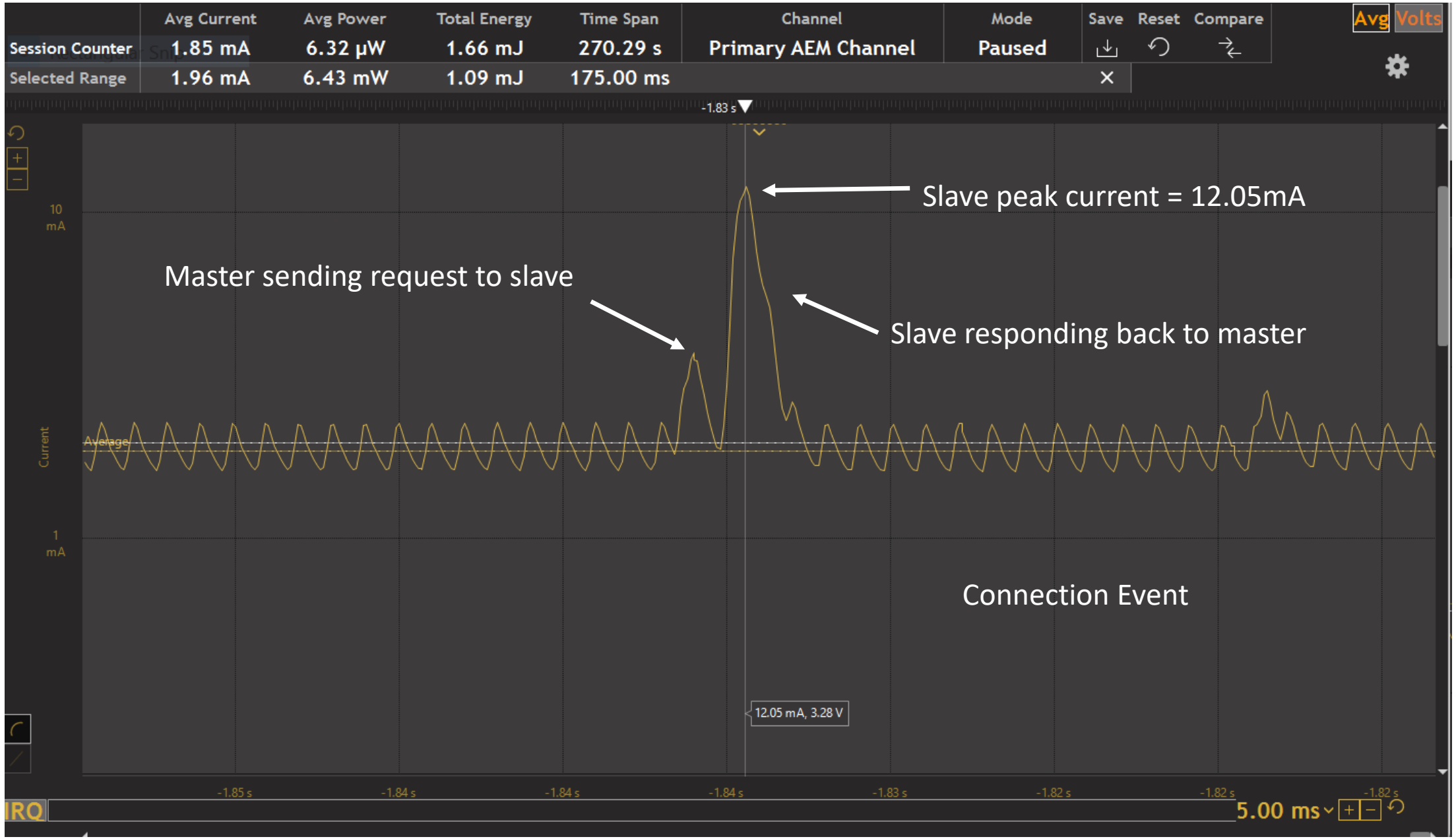
- If the assumption to use Single Channel Connection Events is that the channel is clear for the slave if open for the master, why not use this frequency channel all the time?
  - The master cannot signal this change to all the slaves
  - Consumes a lot of power to resynchronize the single frequency once the channel begins to begins to fail
  - With interference having a “bursty” nature, such as a WiFi packet occurring, staying on one channel continually begins to fail
  - The one-channel model also reduces the number of co-located networks because each will naturally drift to a clean frequency, and once filled, no more co-located networks would be able to be established without interfering on an existing network
- Frequency-hopping algorithm distributes network traffic in both time and frequency allowing for many more simultaneous networks to be active

BLE: Subrating Connection Events

- Low latency
  - Low latency means frequent contact between the slave and the master
  - Frequent contact means high use of the radio
  - High use of radio results in higher energy use and lower battery life
- Low power/energy
  - Low power means the slave is listening to the master infrequently
  - If the master is continually polling the slave and the slave has to listen to each of the polling connection events, it would not be low power
- How to balance between low latency and low power/energy?



  - Allow the slave to ignore most connection events from the master


Avg Current
Avg Power
Total Energy
Time Span
Channel
Mode
Save
Reset
Compare
Avg
Volts
Session Counter
1.85 mA
6.32 µW
1.66 mJ
270.29 s
Primary AEM Channel
Paused
Selected Range
1.96 mA
6.43 mW
1.09 mJ
175.00 ms
-1.83 s
10 mA
1 mA
Current
Average
Slave peak current = 12.05mA
Master sending request to slave
Slave responding back to master
Connection Event
12.05 mA, 3.28 V
-1.85 s
-1.84 s
-1.84 s
-1.84 s
-1.83 s
-1.82 s
-1.82 s
-1.82 s
IRQ
5.00 ms

BLE: Subrating Connection Events

- The feature of allowing the slave to ignore X number of connection events is called slave latency.
- The more connection events that a slave can miss, the lower power the slave can be.
- The limit to slave latency is that it cannot be longer than the supervision timeout of the connection.

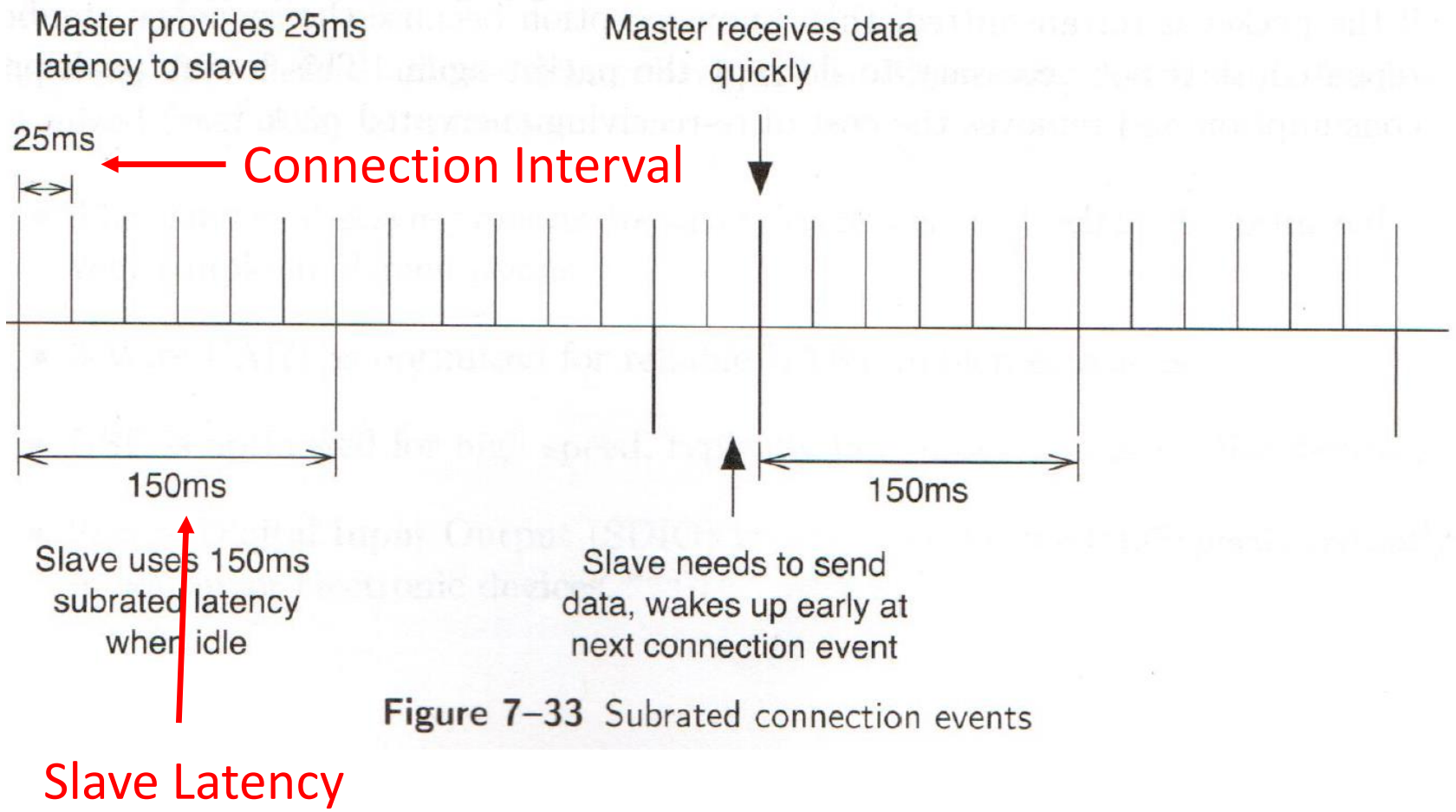
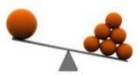


Figure 7–33 Subrated connection events

BLE: Subrating Connection Events

- It is not recommended to have a slave latency that gives fewer than 6 opportunities for the slave to resynchronize before the supervision timeout
- For example, if the supervision timeout is 600mS and the connection interval is 25mS, the slave latency should not be longer than 450mS
- Resulting in the slave to transmit data in an average of 25mS, connection interval, but only needing to connect 1 out of 24 connection events



Optimizing to the slave or less resource rich device

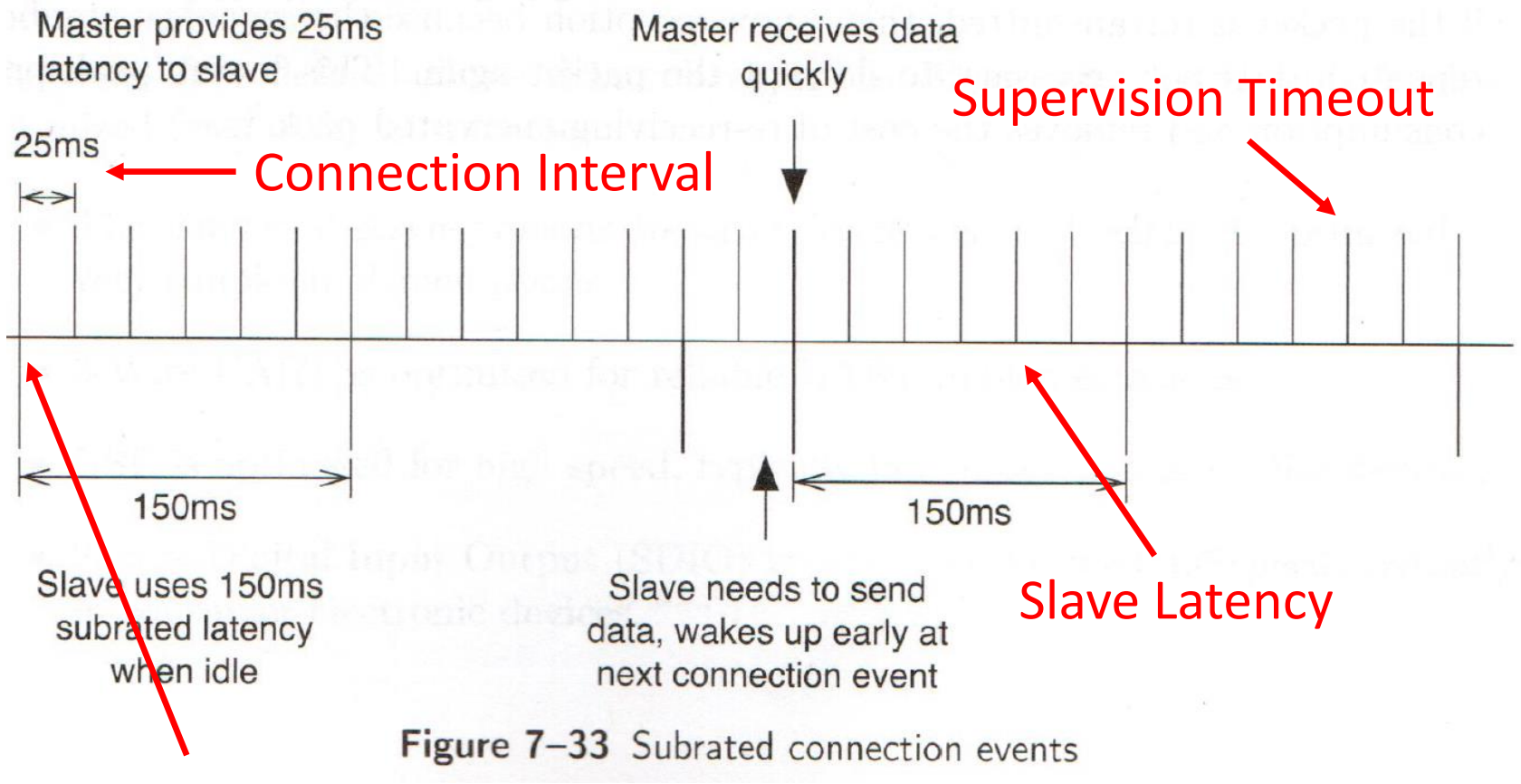
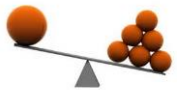


Figure 7–33 Subrated connection events

BLE: Offline Encryption

- Encrypting data requires a significant amount of computational time and energy
  - Compute cipher blocks for the payload
  - Message authentication codes
  - Compute the authentication code itself
- For a maximum BLE packet, this would require seven iterations through the AES-128 encryption block consuming a large amount of energy
  - If the encryption had to occur real-time while the radio was on, the peak current would be significantly higher
  - Resulting in higher peak currents from the button-cell battery and reducing the battery life



BLE: Offline Encryption

- Bluetooth Low Energy enables the encryption of the data and authentication code to be computed in the background
  - Before a packet is transmitted, the encryption of the data can be performed when the radio is still off
  - The encryption of the data does not depend on the sequence of the data, so it can be encrypted at anytime
    - The data can be retransmitted any number of times, and the encryption and the authentication code will not have to change
  - When receiving encrypted data, the CRC value is computed real time and is the only value that determines whether the data was received correctly.
    - The encrypted data can then remain in the Link Layer until the radio activity has stopped to decrypt the data

BLE: Peripherals

- For peripherals to operate extended time and possibly years on a button-cell battery, the states that the peripheral enters must be optimized
- This includes determining the optimal:
  - Advertising Interval
  - Connection Interval
  - Slave Latency
  - Access to attributes
  - Deciding to stay connected or to disconnect/reconnect

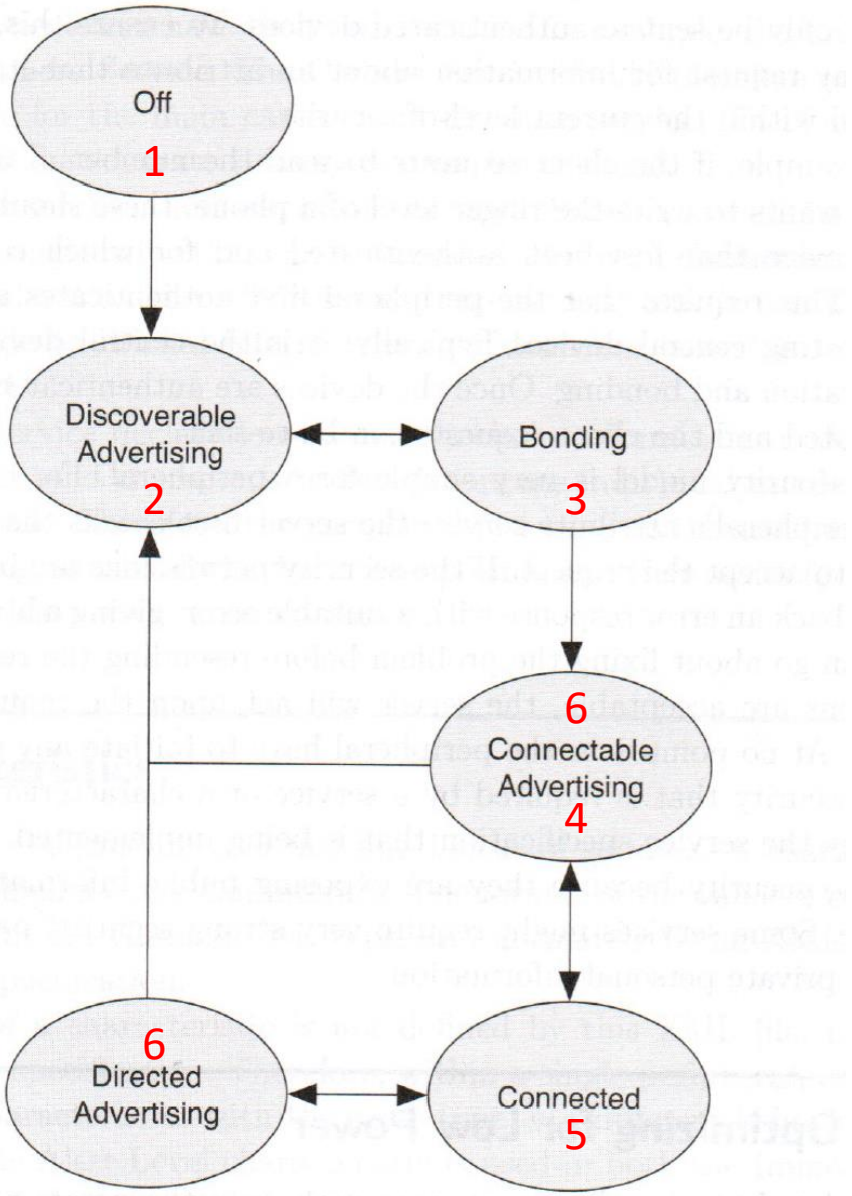


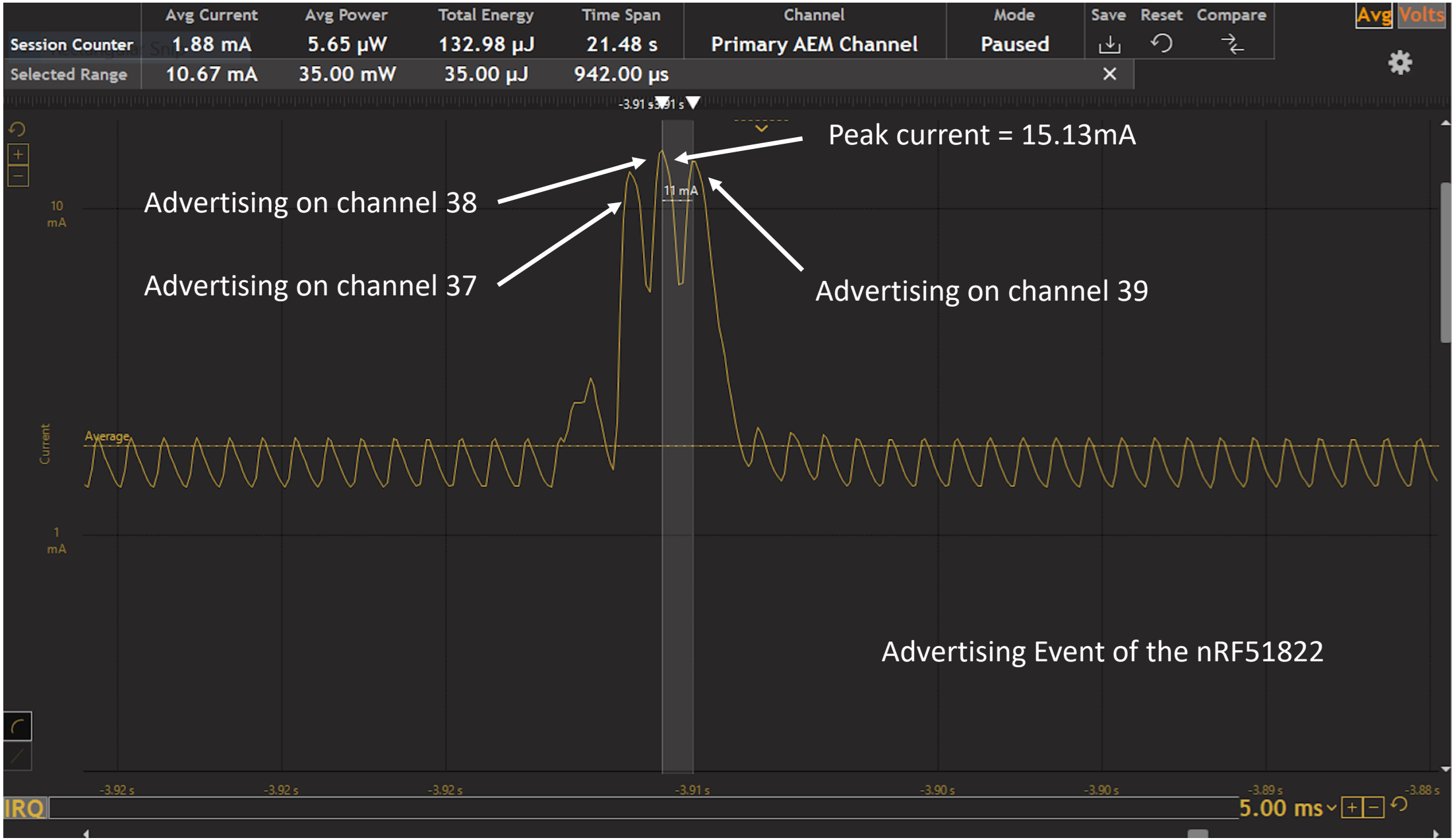
Figure 14–1 The typical states of a peripheral device

BLE: Peripherals (Discoverable Advertising)

- One of the fundamental ways to optimize for low power is to appropriately choose the intervals for advertising and connection intervals
  - The choice could be the difference between a few weeks of operation to years
- Typically, the first state that a peripheral performs is discoverable advertising so that a central device can find it
  - The time that a peripheral is in this state of operation is typically very short in the lifetime of the device since in most applications, the user will want to connect a device shortly after installation
  - Once bonded to a central device, it will move into connectable advertising

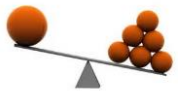
BLE: Peripherals (Discoverable Advertising)

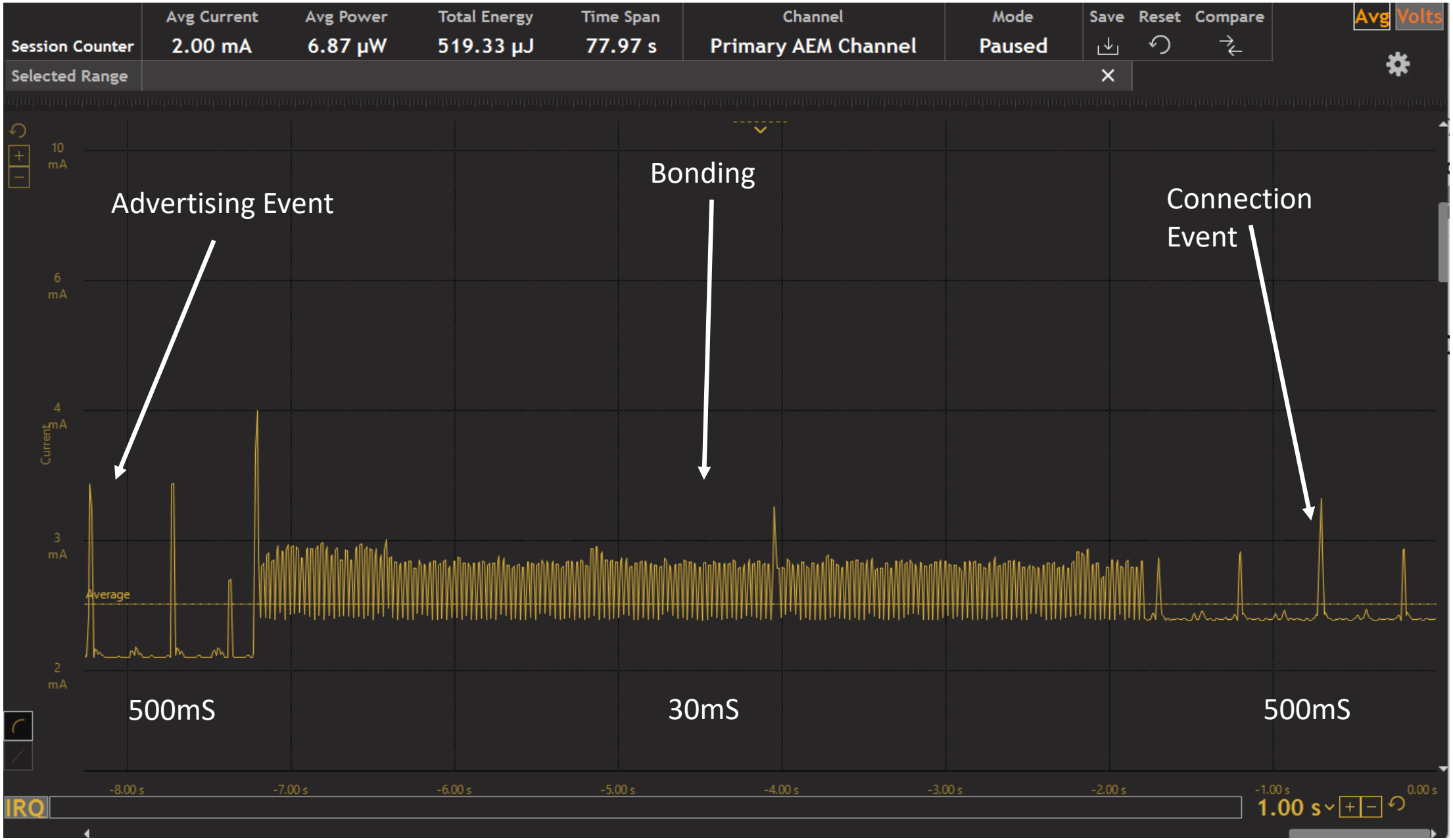
- With the idea that the peripheral will be in the discoverable advertising state for a very short period of its life cycle and to provide a good user experience of a quick connect, the advertising interval of 250mS would be a good compromise between speed of discovery and power savings


Avg Current
Avg Power
Total Energy
Time Span
Channel
Mode
Save
Reset
Compare
Avg
Volts
Session Counter
1.88 mA
5.65 µW
132.98 µJ
21.48 s
Primary AEM Channel
Paused
Selected Range
10.67 mA
35.00 mW
35.00 µJ
942.00 µs
Peak current = 15.13mA
Advertising on channel 38
Advertising on channel 37
Advertising on channel 39
11 mA
10 mA
1 mA
Current
Average
Advertising Event of the nRF51822
IRQ
5.00 ms

BLE: Peripherals (Bonding)

- After the peripheral makes a connection from the discoverable advertising state, the device enters bonding and the connection interval can be set very short (or fast)
  - A fast connection interval of 7.5mS to 25mS can use a lot of power, but it also allows the central device to discover the set of services and characteristics that the peripheral has to offer as well as provide prompt feedback to the user about how it can interact with the peripheral
  - If the connection interval is very slow, 1 to 4S, it can be an extremely long time before a central device can determine how to utilize the peripheral
- Once the appropriate bonding information has been exchanged, the connection interval timing can be long (or slow) to conserve energy




Session Counter
Avg Current
2.00 mA
Avg Power
6.87 µW
Total Energy
519.33 µJ
Time Span
77.97 s
Channel
Primary AEM Channel
Mode
Paused
Save
Reset
Compare
Avg
Volts
Selected Range
10 mA
6 mA
4 mA
3 mA
2 mA
Current
Average
Advertising Event
Bonding
Connection Event
500mS
30mS
500mS
-8.00 s
-7.00 s
-6.00 s
-5.00 s
-4.00 s
-3.00 s
-2.00 s
-1.00 s
0.00 s
IRQ
1.00 s

BLE: Peripherals (Connectable Advertising)

- After a peripheral has bonded with a central device, it can disconnect and at a later time advertise to allow the central device to reconnect
- The advertising interval used in this state is a compromise between how fast at central device can reconnect to the peripheral and the power consumption that the peripheral will use when disconnected
  - Example: A heart-rate belt used by a runner might only be connected for the 3 hours a week while the runner is using it while the remaining 165 hours a week it remains in the connectable advertising state. It also does not need to connect instantaneously, so using a longer connectable advertising interval would be advisable or even disconnect when the belt is not worn
  - A connectable advertising interval of 1 second would allow a central device to be able to connect within a few seconds, and if the peripheral requires a faster connection time, than a connectable advertising interval of 0.5s or less would be required

BLE: Peripheral (Directed Advertising)

- Directed Advertising is used when a peripheral needs to connect directly to a central device usually based on an event happening
- And, when the time between the event and sending notification to the central device must be as short as possible
  - Example: A fire or smoke alarm
- A peripheral burns a lot of power/energy when executing Directed Advertising because the peripheral transmit lots of advertising packets very quickly to a single central device
  - If the central device is available to initiate a connection to this peripheral, it will immediately connect, allowing the peripheral to send its data quickly

BLE: Peripheral (Directed Advertising)

- Directed advertising is the quickest way for a peripheral to make a connection to a central device
  - Connection times + sending the required data can be less than 3 milliseconds
- There is no interval configuration in Directed Advertising
  - Every 3.75mS, the peripheral will send its advertising packets on each of the 3 advertising channels
  - Resulting in one directed advertising packet every 1.25mS or 800 packets per second
  - If there is no central device to respond, directed advertising could consume 250+ times more energy per second compared to a device sending connectable advertising packets once per second
- Directed Advertising could be very useful for peripherals that rarely need to be connected, but when the need arises, connected very quickly

BLE: Peripheral (Connected)

- When connected, the central device has complete control over the connection intervals and latency used by the peripheral
- The peripheral does not have away to signal to the central device that the current connection values being used are appropriate
  - But, the peripheral can request or suggest alternative settings after connection has been made
- There are two configurable values for the connection parameters that relate to power consumption
  - Connection Interval (connInterval): The time that determines how often the central will transmit and synchronize with the peripheral. The connInterval is a multiple of 1.25mS
  - Slave Latency (slavelatency): The number of master connection intervals that the slave can ignore. This value can be 0 to 500

BLE: Peripheral (Connection)

- Example 1:
  - connInterval is set to 12.5mS (10 times 1.25mS)
  - slaveLatency is set to 0
  - This would result in the slave required to listen every 12.5mS to determine if the master has a request and consuming a lot of energy
- Example 2:
  - connInterval is set to 12.5mS (10 times 1.25mS)
  - slaveLatency is set to 1
  - This would enable the slave to skip one of the reserved connection events for the peripheral, but must listen to the second connection event
  - This would result in a 50% power reduction due to halving the time that the peripheral listens and transmits data to the central device
  - The peripheral could listen and transmit to the first 12.5mS connection event if required

BLE: Peripheral (Connection)

- IMPORTANT: Slave Latency also sets the latency of the central device to the peripheral
  - For example: If a keyboard has an LED indicator that needs to be turned on within 500mS for the desired user experience, the connInterval X slavelatency must be less than 500mS
  - The slave has the option of jumping onto a connection event early, but the master does not!