Answer Key

Chapter 1

Exercise 1.3

1. What is the organization of the spec?

   Volumes, Parts, Sections

2. On what page does the Attribute protocol specification start?

   2169

Chapter 2

Exercise 2.1

1. Which chip GPIO is used for the I2C SCL and SDA pins?

   SCL: WICED_P26
   SDA: WICED_P28

2. Are the button pins pulled up or down? Where is that specified?

   Buttons are pulled up. This can be found in the file wiced_platform_pin_config.c in the platform folder on lines 85 and 96.

Exercise 2.2

1. What is the name of the first user application function that is executed? What does it do?

   APPLICATION_START. It just initializes the Bluetooth stack and registers the callback.

2. What is the purpose of the function bt_cback? When does the BTM_ENABLED_EVT case occur?

   It is the Bluetooth stack management callback function. It is called whenever there is a management event from the stack.

3. What controls the rate of the LED blinking?

   The first parameter to the RTOS delay function wiced_rtos_delay_milliseconds specifies the delay which controls the rate of the LED blinking.

Exercise 2.6

1. What happens if you don't remove the *while(1)* loop from the function that blinks the LED? Why?

   The LED will appear to stay on all the time (in fact, it is blinking on/off rapidly) so it appears dim. The reason is that as soon as the timer executes the LED blinking function once, it never exits so it continually blinks the LED with almost no delay.

Exercise 2.7

1. How many bytes does the NVRAM read function get before you press the button the first time?

   0

2. What is the return status value before you press the button the first time?

   The return value is 40 (0x28).

   1. What does the return value mean?

      The return value of 40 (0x28) means ERROR. This is defined in the file wiced_result.h.

Chapter 3

Exercise 3.1

1. Do you need *wiced_rtos_delay_millisecon*ds() in the LED thread? Why or why not?

   No, because the wiced_rtos_get_semaphore will cause the thread to suspend each time through the infinite loop while it waits for another button press.

Exercise 3.2

1. What happens if you forget to unlock the mutex in one of the threads? Why?

   The thread that has the lock will keep running but the other thread will stay suspended because it can never get access to the mutex. Therefore, only one of the threads will cause the LED to blink.

Chapter 4A

Exercise 4A.3

1. How many bytes is the advertisement packet?

   The advertisement packet is19 bytes. They are:

   - Flags (3)
     - Length (1)
     - Type (1)
     - Data (1)
   - Local Name (9)
     - Length (1)
     - Type (1)
     - Data (7)
   - Appearance (4)
     - Length (1)
     - Type (1)
     - Data (2)
   - Manufacturer Specific Data (3)
     - Length (1)
     - Type (1)
     - Data (1)

Exercise 4A.4

1. What function is called when there is a Stack event? Where is it registered?

   The function is *ex03_ble_con_management_callback*. It is registered using *wiced_bt_stack_init* in *application_start*.

2. What function is called when there is a GATT database event? Where is it registered?

   The function is *ex03_ble_con_event_handler*. It is registered using *wiced_bt_gatt_register* in *ex03_ble_con_app_init*.

3. Which GATT events are implemented? What other GATT events exist? (Hint: right click and select Open Declaration on one of the implemented events)

   Implemented:

   GATT_CONNECTION_STATUS_EVT
   GATT_ATTRIBUTE_REQUEST_EVT

   Others:

   GATT_OPERATION_CPLT_EVT
   GATT_DISCOVERY_RESULT_EVT
   GATT_DISCOVERY_CPLT_EVT
   GATT_CONGESTION_EVT

4. In the GATT "GATT_ATTRIBUTE_REQUEST_EVT", what request types are implemented? What other request types exist?

   Implemented:

   GATTS_REQ_TYPE_READ
   GATTS_REQ_TYPE_WRITE

   Others:

   GATTS_REQ_TYPE_PREP_WRITE
   GATTS_REQ_TYPE_WRITE_EXEC
   GATTS_REQ_TYPE_MTU
   GATTS_REQ_TYPE_CONF

Chapter 4B

Exercise 4B.3

1. How long does the device stay in high duty cycle advertising mode? How long does it stay in low duty cycle advertising mode? Where are these values set?

   High: 30 seconds

   Low: 60 seconds

   These are specified in the wiced_bt_cfg.c file in *wiced_bt_cfg_settings.ble_advert_cfg.high_duty_duration* and *wiced_bt_cfg_settings.ble_advert_cfg.low_duty_duration*

Exercise 4B.4

1. What items are stored in NVRAM?

   Hostinfo (Remote BDADDR and Button CCCD state)
   Local Keys (Privacy Information)
   Paired Device Keys (Encryption Information)

2. Which event stores each piece of information?

   Hostinfo is stored during BTM_PAIRING_COMPLETE_EVT and in ex03_ble_bond_set_value if the Button CCCD value was written

   Local Keys are stored during BTM_LOCAL_IDENTITY_KEYS_UPDATE_EVT

   Paired Keys are stored during BTM_PAIRED_DEVICE_LINK_KEYS_UPDATE_EVT

   All three are cleared out (i.e. reset) in the button_cback function to allow re-pairing.

3. Which event retrieves each piece of information?

   Hostinfo is retrieved by BTM_ENCRYPTION_STATUS_EVT (if the device was previously bonded

   Local Keys are retrieved by BTM_LOCAL_IDENTITY_KEYS_REQUEST_EVT

   Paired Keys are retrieved by ex03_ble_bond_app_init (at startup) and by BTM_PAIRED_DEVICE_LINK_KEYS_REQUEST_EVT

4. In what event is the privacy info read from NVRAM?

   BTM_LOCAL_IDENTITY_KEYS_REQUEST_EVT

5. Which event is called if privacy information is not retrieved after new keys have been generated by the stack?

   BTM_LOCAL_IDENTITY_KEYS_UPDATE_EVT

Exercise 4B.5

1. Other than BTM_IO_CAPABILITIES_NONE and BTM_IO_CAPABILITIES_DISPLAY_ONLY, what other choices are available? What do they mean?

   BTM_IO_CAPABILITIES_DISPLAY_AND_YES_NO_INPUT

   > Device can display values (e.g. 6-digit numbers) and can accept a Yes/No input from the user.

   BTM_IO_CAPABILITIES_KEYBOARD_ONLY

   > Device can accept input (e.g. numbers) but cannot display any values.

   BTM_IO_CAPABILITIES_BLE_DISPLAY_AND_KEYBOARD_INPUT

   > Device can display values (e.g. 6-digit numbers) and can accept input (e.g. numbers).

2. What additional stack callback event occurs compared to the previous exercise? At what point does it get called?

   BTM_PASSKEY_NOTIFICATION_EVT

   This event is called between BTM_PAIRING_IO_CAPABILITIES_BLE_REQUEST_EVT and BTM_ENCRYPTION_STATUS_EVT.

Chapter 4C

Exercise 4C.1

1. Which lines in the code are used to configure sleep?

   Lines 177 – 188:

   ```
   /* Configure and initialize sleep */
    sleep_config.sleep_mode                = WICED_SLEEP_MODE_NO_TRANSPORT;
    sleep_config.device_wake_mode          = WICED_SLEEP_WAKE_ACTIVE_LOW;
    sleep_config.device_wake_source        = WICED_SLEEP_WAKE_SOURCE_GPIO;
    sleep_config.device_wake_gpio_num      = WICED_GPIO_PIN_BUTTON_1;
    sleep_config.host_wake_mode            = WICED_SLEEP_WAKE_ACTIVE_LOW;
    sleep_config.sleep_permit_handler      = low_power_sleep_callback;

    if(WICED_BT_SUCCESS != wiced_sleep_configure(&sleep_config))
    {
        WICED_BT_TRACE("Sleep Configure failed!\n\r");
    }
   ```

2. When in the code is sleep configured (i.e. after which event)?

   Sleep configuration is done right after the BT stack initialization (wiced_bt_stack_init).

3. What is used as a wakeup source?

   WICED_GPIO_PIN_BUTTON_1 is used as a wakeup source.

4. What is the name of the sleep permit handler function?

   low_power_sleep_callback

5. What character is printed to the UART for:
   a. The PMU requests sleep permission and it is allowed without shutdown: __$__
   b. The PMU indicates that it is time to sleep: __e__

6. When are the connection interval min, max, latency, and timeout values updated and what values are used?

   The values are updated in the GATT connect callback function when a connection is established.

   The values are set to:
   Min Interval: 100ms, Max Interval: 100ms, Latency: 0, Timeout: 5120ms.

Exercise 4C.7

1. Which variable is used to control sleep permissions? What are its states?

   The variable is sleep_type. It can be:

   WICED_SLEEP_NOT_ALLOWED
   WICED_SLEEP_ALLOWED_WITHOUT_SHUTDOWN
   WICED_SLEEP_ALLOWED_WITH_SHUTDOWN

2. What is printed to the UART when:

a.	A sleep request is denied:	x
b.	A sleep request is allowed without shutdown:	$
c.	A sleep request is allowed with shutdown:	.
d.	Sleep is entered:	e
e.	A fast boot occurs:	f

3. When is sleep not allowed? When is sleep allowed but without shutdown (PDS)? When is sleep allowed with shutdown (SDS)?

   Sleep is not allowed from powerup until advertising is started.

   PDS is allowed once the GATT connection is made (GATT_CONNECTION_STATUS_EVT with status of connected) until paring is complete (BTM_ENCRYPTION_STATUS_EVT with result of WICED_SUCCESS).

   SDS is allowed during advertising and during a connection once pairing is complete.

4. When does SDS occur (Hint: you can determine that SDS occurred when you see a fast boot)?

   SDS occurs during low duty cycle advertising (because the timeout is set to 0) and during a connection once pairing is complete.

5. What is done differently for a cold boot vs. a fast boot? Why?

   For a cold boot, the GPIO for button 1 is configured. That isn't needed for a fast boot since the GPIO configuration is retained. Also, more debug information is printed during a cold boot to get initial startup information without flooding the screen with values during fast boot.

   For a fast boot, we check the connection_id. If the device is connected, then we read hostinfo (paring information) from the NVRAM, restore the previous CapSense button value and CCCD value in the GATT database. This allows the firmware to determine if a notification must be sent the next time a CapSense scan is done (i.e. if the CapSense value has changed and notifications are enabled).

6. What is AON RAM? What values is it used for and why?

   AON is Always ON RAM. It is RAM that is retained during SDS. It is used for connection_id, prevVal, and advert_mode.

   connection_id allows us to tell if we were connected when we went to sleep so that the bonding information can be restored on a fast boot.
   prevVal is used to restore the CapSense button value. This is done so that the firmware can tell if the button value changed while the device was asleep.
   Advert_mode is saved so that on a fast boot advertising can start up where it left off. If this wasn't done, the device would start high duty cycle advertisements again as soon as it did a fast boot.

Chapter 4D

Exercise 4D.1

1. What is the cause of “Unhandled Bluetooth Management Event: 0x16 (22)”? How could you fix it?

   The "Unhandled Bluetooth Management Event: 0x16 (22)" occurs when the scan state changes. The default configuration in wiced_bt_cft.c has the high duty and low duty scan duration both set to 5 seconds. So, after 5 seconds it goes from high to low duty scan and then after another 5 seconds it stops scanning.

   To fix this, I could change the high duty scan duration to 0 which would prevent scanning from timing out. In this case, I would still get one call to this event when scanning first starts. To eliminate that, I could just implement a case for that event and print out an appropriate message or do nothing.