Answer Key

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.

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.

2. What happens if you use a value of 100 for the semaphore timeout? Why?

   The LED will blink every 100ms because the semaphore times out. This will happen even without pressing the button.

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 buttons will cause the LED to blink (the one that has the lock).

Exercise 3.4

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.

Chapter 4A

Exercise 4A.1

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.2

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

   The function is *ex02_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 *ex02_ble_con_event_handler*. It is registered using *wiced_bt_gatt_register* in *ex02_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.2

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.3

1. What items are stored in NVRAM?

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

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

Exercise 4B.4

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.