PSoC4 BLE Authentication - I/O Capabilities

Objective

This example demonstrates the authentication process based on different I/O capabilities of the device

Overview

This code example, Authentication - I/O Capabilities, uses the BLE component to configure PSoC 4 BLE in the GAP Peripheral role and enable authentication and encryption. After the BLE stack is ON, the user has to select one of the possible I/O capabilities of the device. As per I/O capabilities of the Peripheral and Central, the BLE component automatically selects the suitable pairing process. The I/O capability of the BLE-USB bridge, acting as the GAP Central, when using with Cysmart 1.0 tool is **Keyboard and Display**.

Requirements

Tool: PSoC Creator 3.1, CySmart 1.0

Programming Language: C (GCC 4.8.4)

Associated Parts: All PSoC4 BLE parts

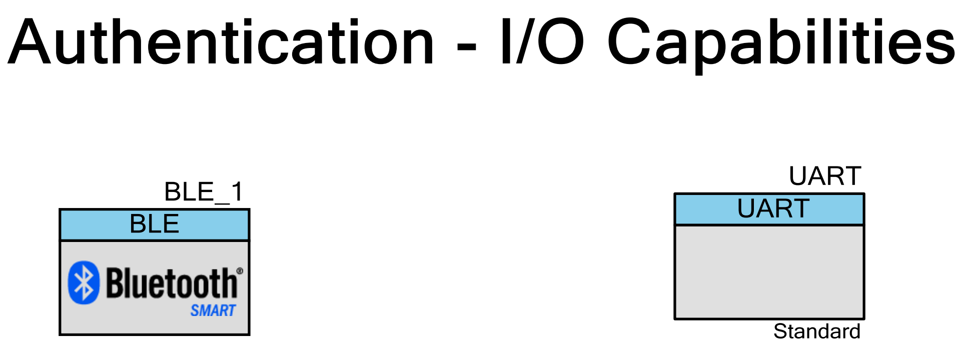
Related Hardware: CY8CKIT-042-BLE Bluetooth® Low Energy (BLE) Pioneer Kit

Hardware Setup

BLE Pioneer Kit has the necessary hardware connections required for this lab. If you are using your own hardware connect UART RX and UART TX to P1[4] and P1[5].

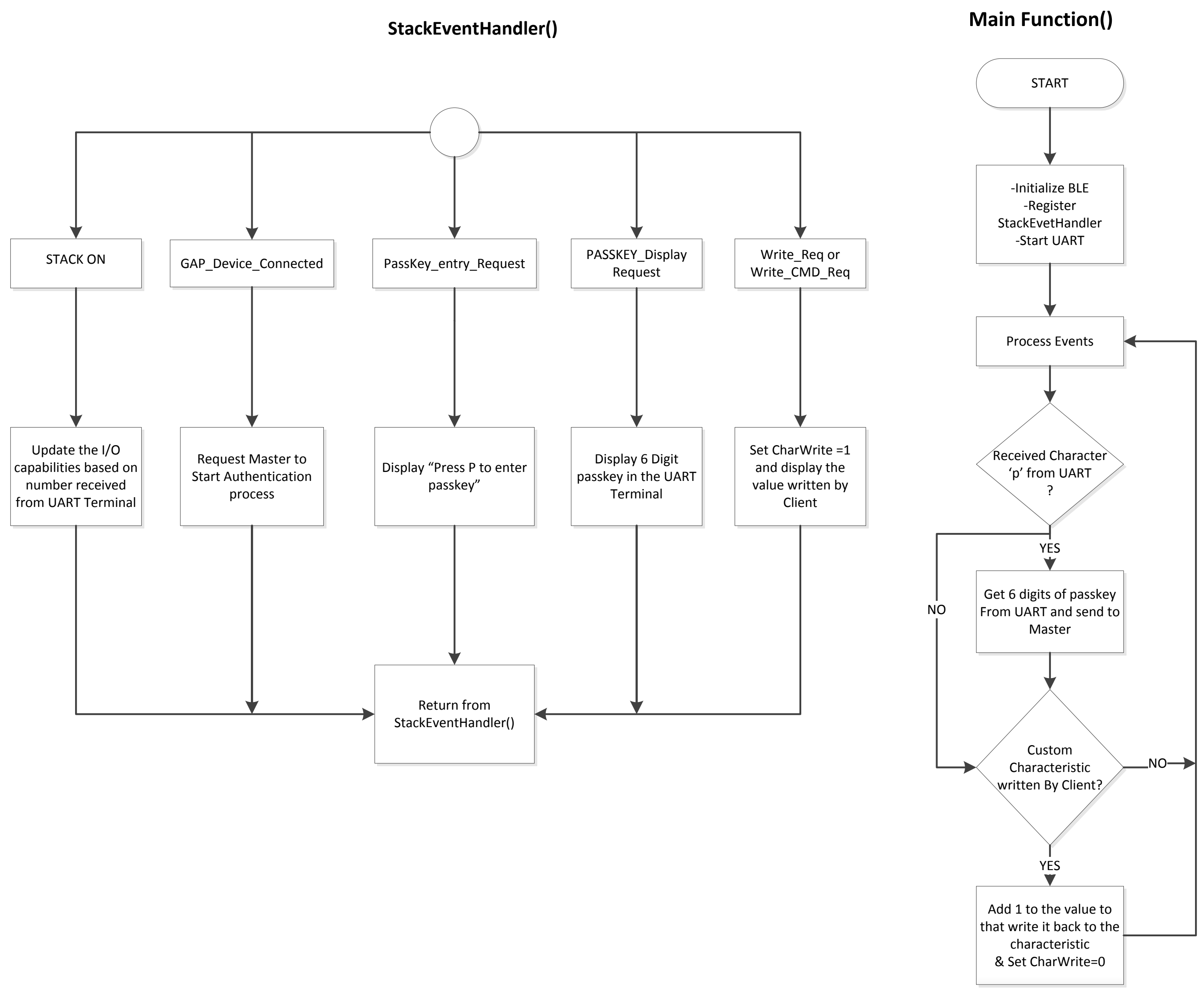
PSoC Creator Schematic

Figure 1. PSoC Creator Schematic



Operation :

Figure 2. Firmware Flow


StackEventHandler()
Main Function()
STACK ON
GAP_Device_Connected
PassKey_entry_Request
PASSKEY_Display Request
Write_Req or Write_CMD_Req
Update the I/O capabilities based on number received from UART Terminal
Request Master to Start Authentication process
Display "Press P to enter passkey"
Display 6 Digit passkey in the UART Terminal
Set CharWrite =1 and display the value written by Client
Return from StackEventHandler()
START
-Initialize BLE -Register StackEvetHandler -Start UART
Process Events
Received Character 'p' from UART ?
YES
NO
Get 6 digits of passkey From UART and send to Master
Custom Characteristic written By Client?
NO
YES
Add 1 to the value to that write it back to the characteristic & Set CharWrite=0

1. **main() function**: This is the central function which performs the initialization of the BLE stack, UART and executes the necessary routines to process the BLE events and maintain the connection. In the initial section of the *main()* function, the API *CyBle_Start(StackEventHandler)* is called to start the BLE Component and register a callback to the stack event handler. Note that the callback function can assume any name – in this project, we used *StackEventHandler*. Once the system is initialized, *main()* function continuously operates in a *for(;;)* loop executing *CyBle_ProcessEvents()*. CyBle_ProcessEvents processes the events received by the BLE stack and enables application layer to use them and take the appropriate action. When slave has to respond with 6 digit passkey user has to press 'p' followed by 6digit.This is handled in for(;;).

2. **Ble_Event_Handler() function**: This function handles the events generated for the BLE Stack.

   I. CYBLE_*EVT_STACK_ON*: When BLE STACK is ON it will wait for the user to update the I/O capabilities.
      a) 0 → Display
      b) 1 → Display Yes/No
      c) 2 → Keyboard
      d) 3 → No Input No Output
      e) 4 → Keyboard and Display

   II. CYBLE_EVT_GAP_DEVICE_CONNECTED: Request Master to start Authentication process
   III. CYBLE_EVT_GAP_AUTH_REQ: Based on the I/O capabilities of the master and client pairing process will be selected automatically.
   IV. CYBLE_EVT_GAP_PASSKEY_DISPLAY_REQUEST: Display the 6 digit passkey.
   V. CYBLE_EVT_GAP_KEYINFO_EXCHNGE_CMPLT: Display the keys that are exchanged during pairing process.
   VI. CYBLE_EVT_GAP_ENCRYPT_CHANGE: It will display whether Encryption is ON or OFF for the current connection
   VII. CYBLE_EVT_GAP_AUTH_COMPLETE: Displays the information about completed authentication process.
   VIII. CYBLE_EVT_GATTS_WRITE_REQ: If the Client writes to the characteristic set CharWrite=1 and display the value in the characteristic.

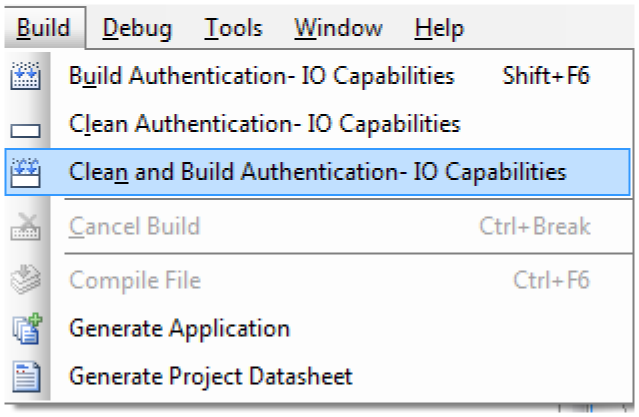
Build and Program the device

This section shows how to build the project and program the PSoC4 BLE device. If you are using a development kit with a built-in programmer (BLE Pioneer Kit, for example), connect the BLE Pioneer Baseboard to your computer using the USB Standard-A to Mini-B cable. For other kits, refer to the kit user guide.

If you are developing on your own hardware, you need a hardware debugger, for example, a Cypress CY8CKIT-002 MiniProg3.

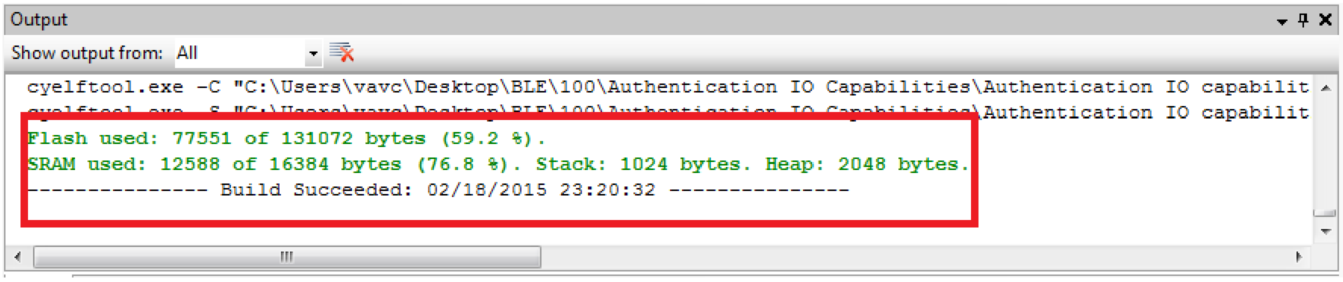
1. On PSoC Creator, select **Build** > **Clean and Build Authentication - IO Capabilities**, as shown in Figure3.

Figure3. Build Project


Build
Debug
Tools
Window
Help
Build Authentication- IO Capabilities Shift+F6
Clean Authentication- IO Capabilities
Clean and Build Authentication- IO Capabilities
Cancel Build Ctrl+Break
Compile File Ctrl+F6
Generate Application
Generate Project Datasheet

2. On successful build, total flash and SRAM usage is reported, as shown in Figure4.

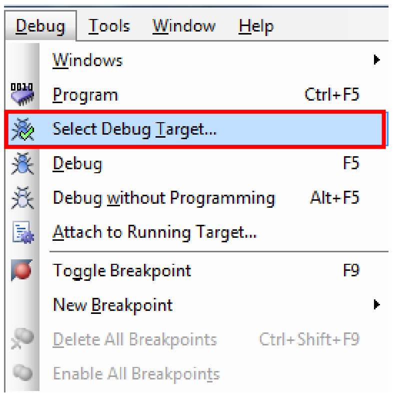
Figure4. Build Succeeded



```
cyelftool.exe -C "C:\Users\vavc\Desktop\BLE\100\Authentication IO Capabilities\Authentication IO capabilit
cyelftool.exe -S "C:\Users\vavc\Desktop\BLE\100\Authentication IO Capabilities\Authentication IO capabilit
Flash used: 77551 of 131072 bytes (59.2 %).
SRAM used: 12588 of 16384 bytes (76.8 %). Stack: 1024 bytes. Heap: 2048 bytes.
--------------- Build Succeeded: 02/18/2015 23:20:32 ---------------
```

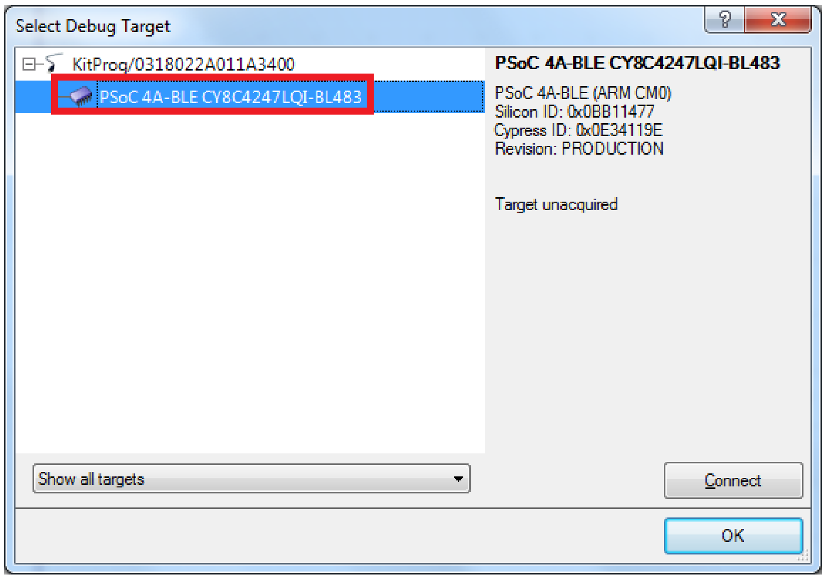
3. Select **Debug** > **Select Debug Target**, as shown in Figure5.

Figure5. Selecting Debug Target



4. In the Select Debug Target dialog box, click Port Acquire, and then click Connect, as shown in Figure 6. Click OK to close the dialog box.

Figure 6. Connecting to a Device



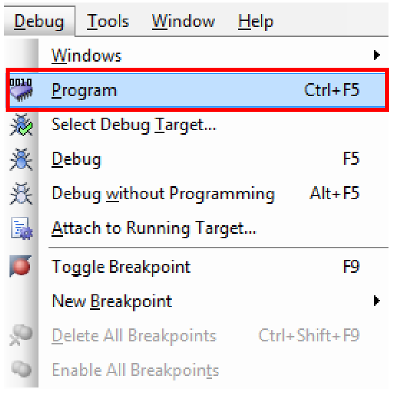
PSoC 4A-BLE CY8C4247LQI-BL483
PSoC 4A-BLE (ARM CM0)
Silicon ID: 0x0BB11477
Cypress ID: 0x0E34119E
Revision: PRODUCTION

Target unacquired

If you are using your own hardware, make sure the Port Setting configuration under Select Debug Target window for your programming hardware is configured as per your setup.

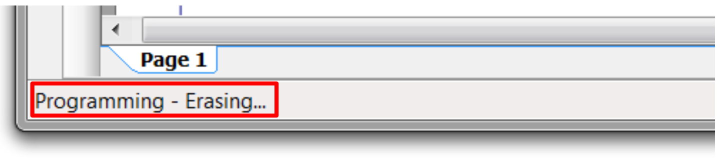
5. Select **Debug** > **Program** to program the device with the project, as shown in Figure 7.

Figure 7. Programming the Device



You can view the programming status on the PSoC Creator status bar (lower-left corner of the window), as shown in Figure 8.

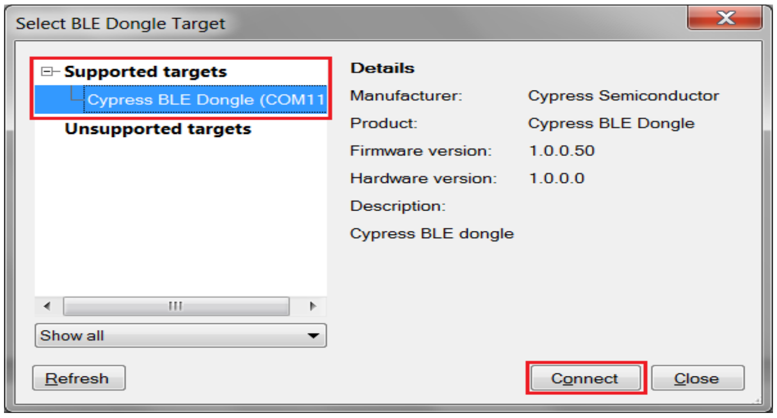
Figure 8. Programming Status



Testing

1. Plug the BLE-USB Bridge (included with the BLE Pioneer Kit) in your computer's USB port.
2. On your computer, launch **CySmart 1.0**. It is located in the **All Programs -> Cypress -> CySmart** folder in the Windows start menu. The tool opens up and asks you to **Select BLE Dongle Target**. Select the **Cypress BLE Dongle (COMxx)** and click **Connect**, as shown in Figure .

Figure 9: CySmart 1.0: Select BLE Dongle Target



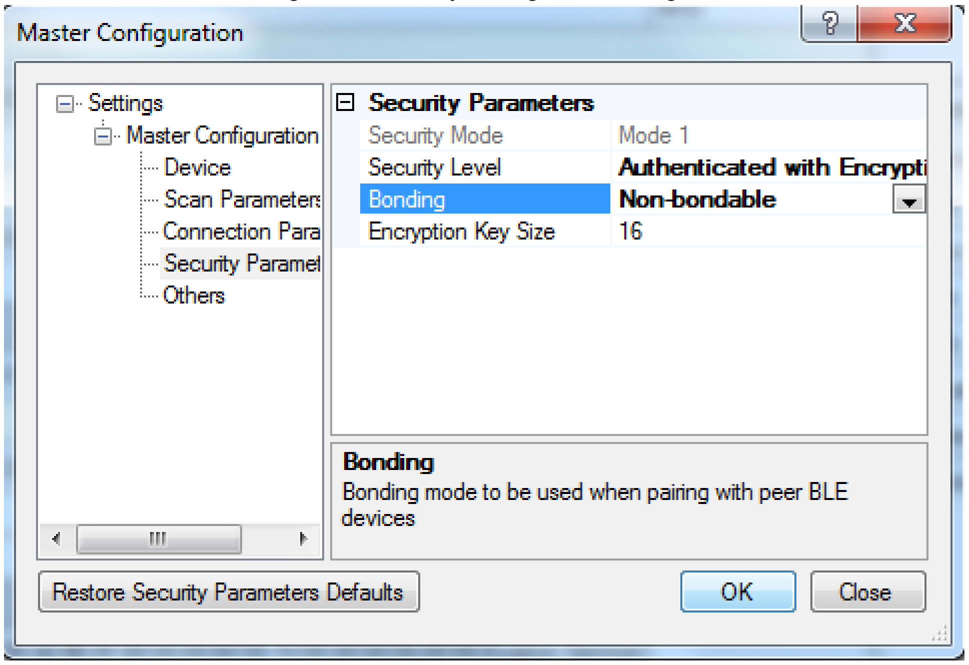
Details

Manufacturer:	Cypress Semiconductor
Product:	Cypress BLE Dongle
Firmware version:	1.0.0.50
Hardware version:	1.0.0.0
Description:	

Cypress BLE dongle

3. When BLE-USB Bridge is connected, update the security setting. Go to "**Configure master settings**" and select the options as shown in the Figure 10

Figure 10.Security settings of the dongle



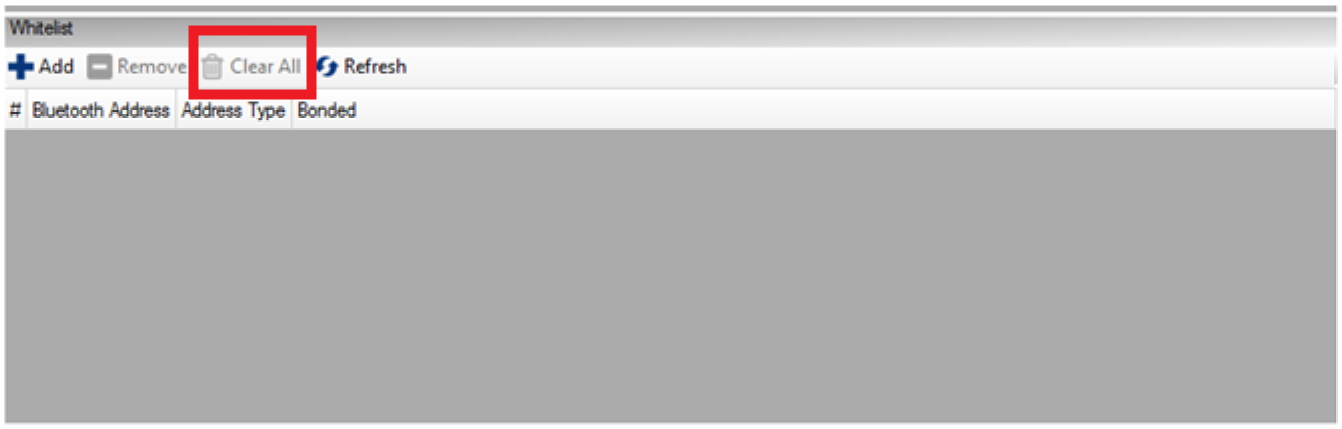
Security Parameters

Security Mode	Mode 1
Security Level	**Authenticated with Encrypti**
Bonding	**Non-bondable**
Encryption Key Size	16

Bonding
Bonding mode to be used when pairing with peer BLE devices

Before starting to scan, make sure that the current device is not bonded already. If it is already bonded then select **Clear All** to remove the all devices from the bonded list or select the device and click **Remove** to remove a selected device from the bonded list, as shown in Figure 11.

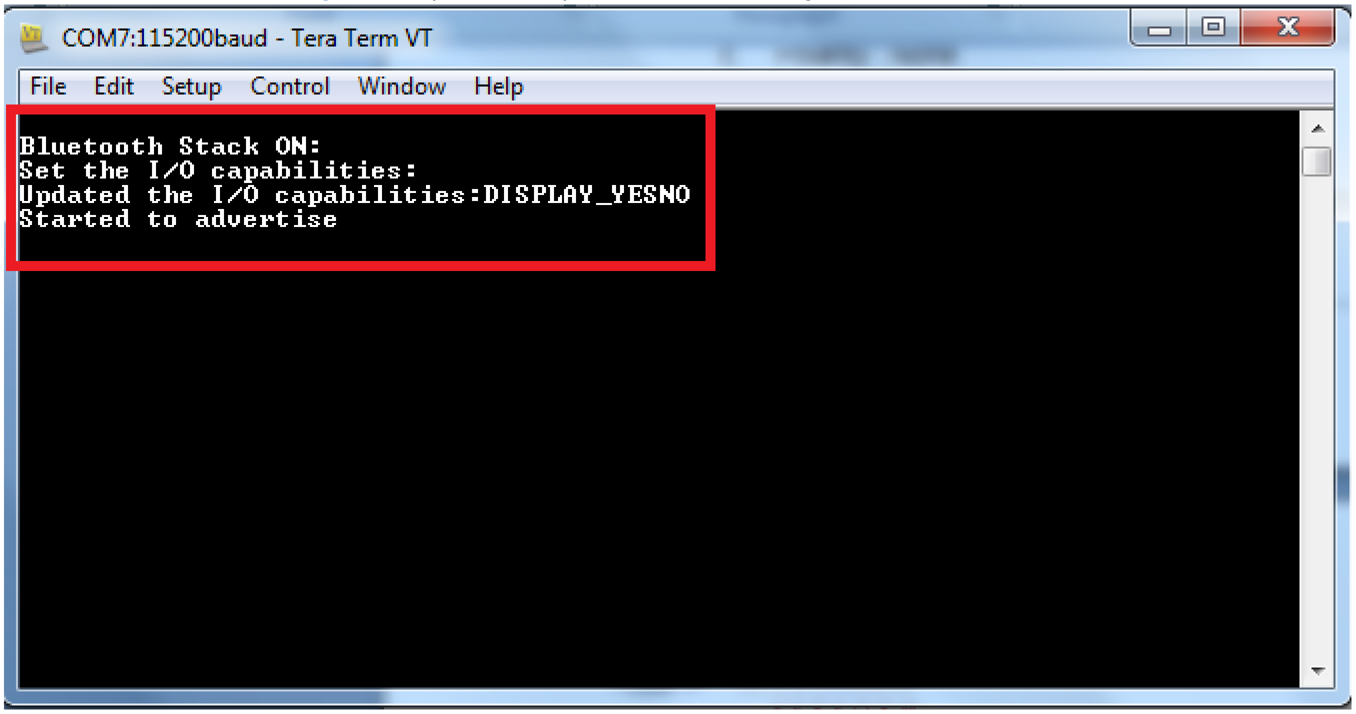
Figure11. Clearing Bonded device list



4. Program the PSoC4 BLE module. Use UART Terminal with the following settings
   a. Baudrate:115200 bps
   b. Data bits: 8
   c. Polarity : None
   d. Stop bits: 1bit
   e. Flow control: NO

5. Send one of the numbers from ‘0’ to ‘4’ to select the I/O capability of the PSoC 4 BLE.
   a. 0 → Display
   b. 1 → Display Yes/No
   c. 2 → Keyboard
   d. 3 → No Input No Output
   e. 4 → Keyboard and Display

Figure 12. Update I/O capabilities before starting to Advertise


COM7:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
Bluetooth Stack ON:
Set the I/O capabilities:
Updated the I/O capabilities:DISPLAY_YESNO
Started to advertise
```

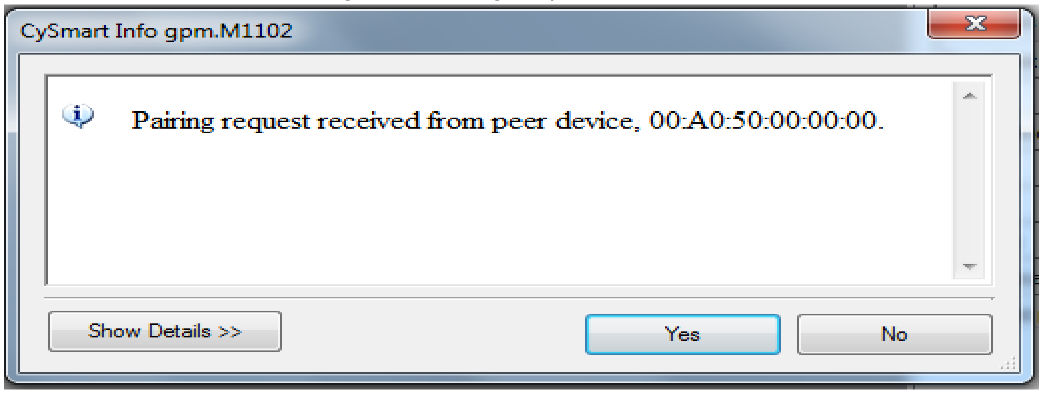
6. After selecting the I/O capability, device starts to advertise.
7. Scan for device in CySmart and select the device **Authentication** in the list and **Connect** to it.
   Based on the I/O capability selected for the PSoC4 BLE device, pairing process will be selected

Expected results:

I/O capability - Display or Display YES/NO:

1. After the device get connected a popup windows appears in Cysmart as shown in Figure 13.

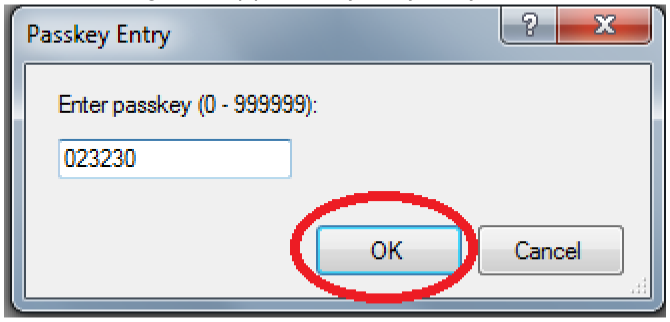
Figure 13.Pairing Request from slave



Pairing request received from peer device, 00:A0:50:00:00:00.

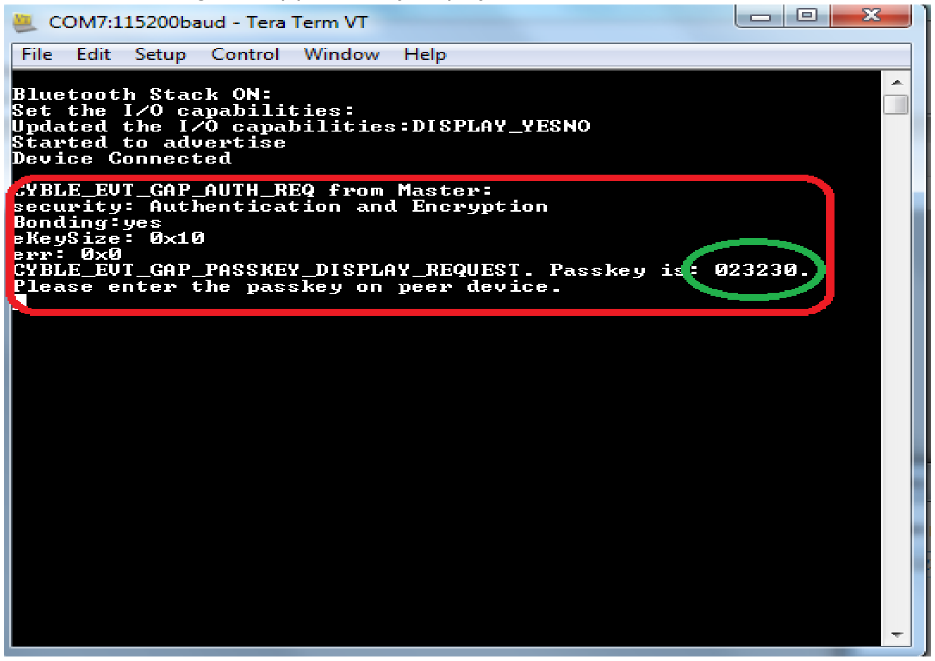
2. Select **Yes** and then type the passkey that is displayed in the UART Terminal and press **OK** as shown in Figure 14.

Figure 14 (a).Passkey entry in Cysmart



Enter passkey (0 - 999999): 023230

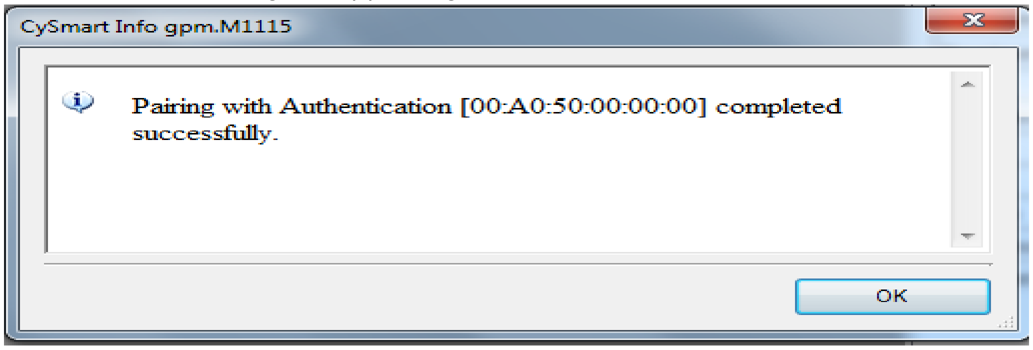
Figure 14(b). Passkey displayed in UART



```
Bluetooth Stack ON:
Set the I/O capabilities:
Updated the I/O capabilities:DISPLAY_YESNO
Started to advertise
Device Connected
CYBLE_EVT_GAP_AUTH_REQ from Master:
security: Authentication and Encryption
Bonding:yes
eKeySize: 0x10
err: 0x0
CYBLE_EVT_GAP_PASSKEY_DISPLAY_REQUEST. Passkey is: 023230.
Please enter the passkey on peer device.
```

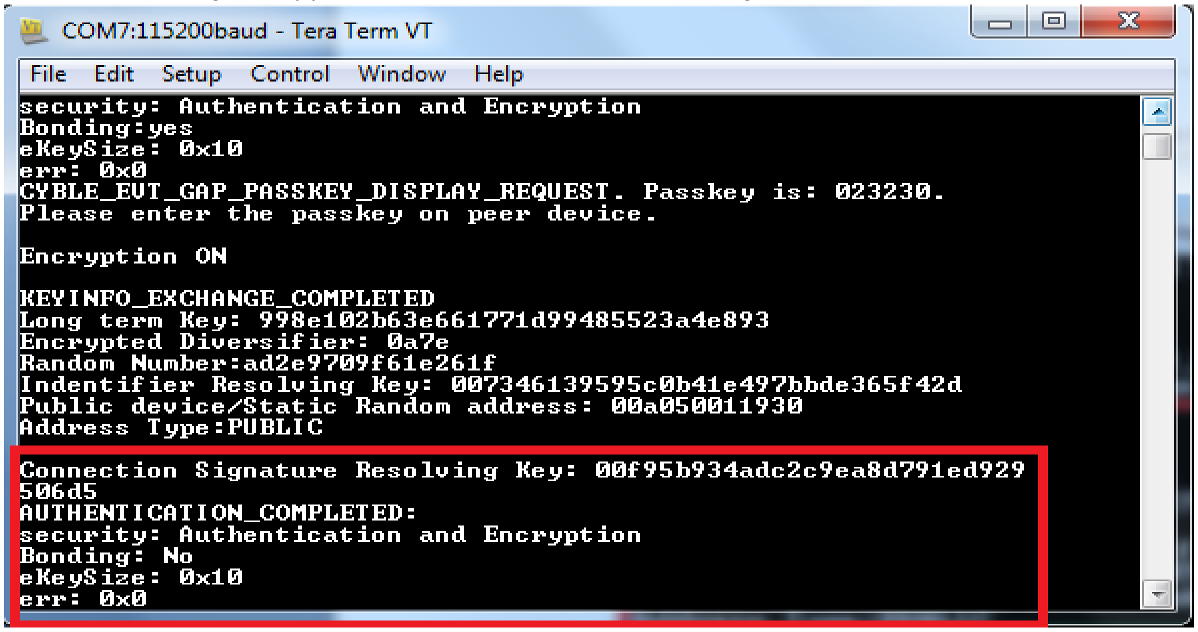
3. After pairing is successful as shown in Figure 15. Information about the keys that are exchanged during authentication process will be displayed in UART.

Figure 15(a).Pairing and Authentication successful



Pairing with Authentication [00:A0:50:00:00:00] completed successfully.

Figure 15(b). Authentication Successful and Exchanged KEY information in UART

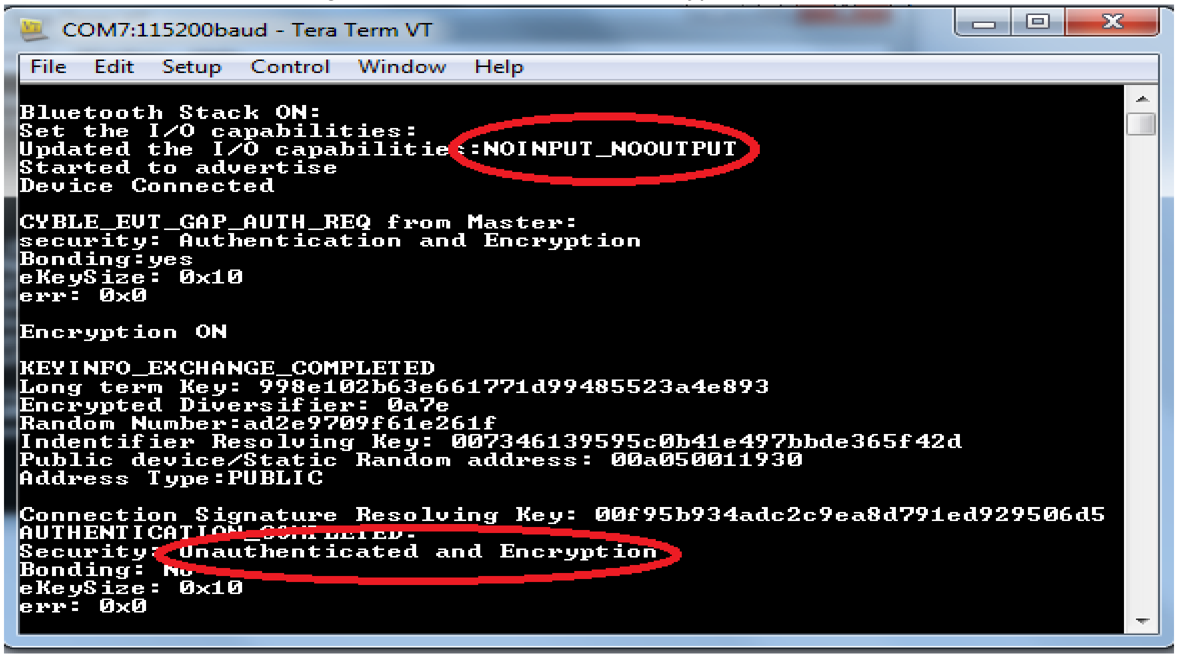

COM7:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
security: Authentication and Encryption
Bonding:yes
eKeySize: 0x10
err: 0x0
CYBLE_EVT_GAP_PASSKEY_DISPLAY_REQUEST. Passkey is: 023230.
Please enter the passkey on peer device.

Encryption ON

KEYINFO_EXCHANGE_COMPLETED
Long term Key: 998e102b63e661771d99485523a4e893
Encrypted Diversifier: 0a7e
Random Number:ad2e9709f61e261f
Indentifier Resolving Key: 007346139595c0b41e497bbde365f42d
Public device/Static Random address: 00a050011930
Address Type:PUBLIC

Connection Signature Resolving Key: 00f95b934adc2c9ea8d791ed929
506d5
AUTHENTICATION_COMPLETED:
security: Authentication and Encryption
Bonding: No
eKeySize: 0x10
err: 0x0
```

I/O Capability – NO INPUT and NO OUTPUT:

1. After the device get connected a popup windows appears in Cysmart as shown in Figure 13.
2. After selecting **Yes,** pairing successful message will as shown in Figure 15(a). As there is NO INPUT and OUTPUT capability of the PROC BLE the pairing process will be JUST WORKS. The established connection would be an unauthenticated encryption connection. Figure 16 shows the UART terminal message log.

Figure16. Unauthenticated and encrypted connection


COM7:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
Bluetooth Stack ON:
Set the I/O capabilities:
Updated the I/O capabilities:NOINPUT_NOOUTPUT
Started to advertise
Device Connected

CYBLE_EVT_GAP_AUTH_REQ from Master:
security: Authentication and Encryption
Bonding:yes
eKeySize: 0x10
err: 0x0

Encryption ON

KEYINFO_EXCHANGE_COMPLETED
Long term Key: 998e102b63e661771d99485523a4e893
Encrypted Diversifier: 0a7e
Random Number:ad2e9709f61e261f
Indentifier Resolving Key: 007346139595c0b41e497bbde365f42d
Public device/Static Random address: 00a050011930
Address Type:PUBLIC

Connection Signature Resolving Key: 00f95b934adc2c9ea8d791ed929506d5
AUTHENTICATION_COMPLETED:
Security: Unauthenticated and Encryption
Bonding: No
eKeySize: 0x10
err: 0x0
```

I/O capability - Keyboard Only or Keyboard display:

1. After the device get connected a popup windows appears in Cysmart as shown in Figure 13..
2. After selecting **Yes**, passkey will be displayed in Cysmart as shown in Figure 17(a).Enter the passkey in the UART terminal by typing ‘p’ first followed by 6 digit numbers. After the 6 digit number is entered it will be sent automatically Now select close in the pop up window, pairing successful message will be displayed shown in Figure 14.

Figure 17(a).Passkey number displayed in Cysmart.

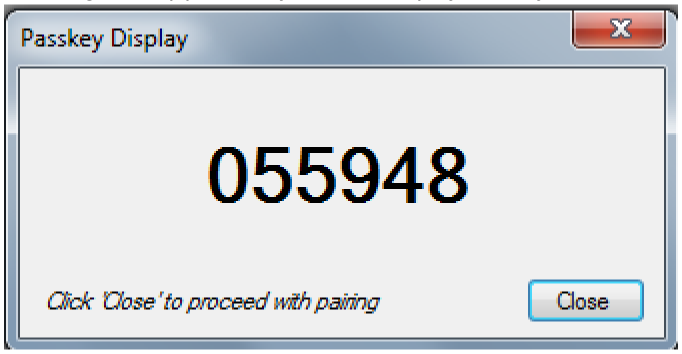
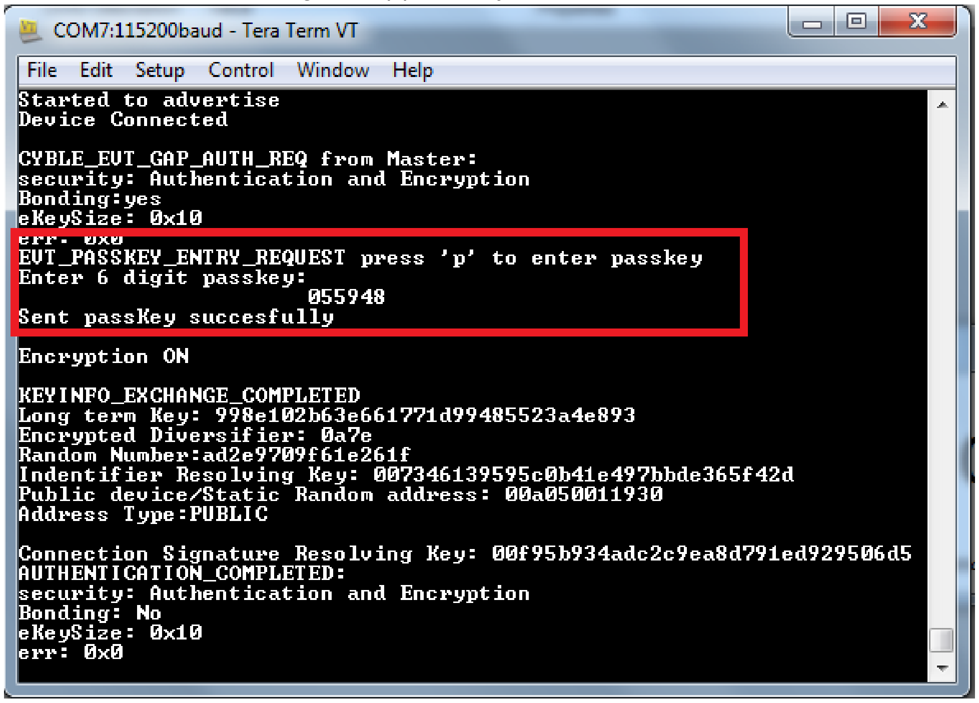


Figure 17(b) Passkey entered in UART



```
Started to advertise
Device Connected

CYBLE_EVT_GAP_AUTH_REQ from Master:
security: Authentication and Encryption
Bonding:yes
eKeySize: 0x10
err: 0x0
EVT_PASSKEY_ENTRY_REQUEST press 'p' to enter passkey
Enter 6 digit passkey:
                                            055948
Sent passKey succesfully

Encryption ON

KEYINFO_EXCHANGE_COMPLETED
Long term Key: 998e102b63e661771d99485523a4e893
Encrypted Diversifier: 0a7e
Random Number:ad2e9709f61e261f
Indentifier Resolving Key: 007346139595c0b41e497bbde365f42d
Public device/Static Random address: 00a050011930
Address Type:PUBLIC

Connection Signature Resolving Key: 00f95b934adc2c9ea8d791ed929506d5
AUTHENTICATION_COMPLETED:
security: Authentication and Encryption
Bonding: No
eKeySize: 0x10
err: 0x0
```

Reading or writing from server:

1. If the authenticated and encrypted link is created then only client can read or write to the server.
2. Write some number to the custom characteristic and read it again. The value that was written by client is incremented by 1 in the firmware and writes it back to the characteristic. So when client reads the value it will be "value written by client"+ 1.

Related Documents

Table 1 lists all relevant application notes, code examples, knowledge base articles, device datasheets, and Component / user module datasheets.

Table 1. Related Documents

Document	Title	Comment
AN91267	Getting Started with PSoC4 BLE	Provides an introduction to PSoC4 BLE device that integrates a Bluetooth Low Energy radio system along with programmable analog and digital resources.
AN91445	Antenna Design Guide	Provides guidelines on how to design an antenna for BLE applications.

Note:

1. To know the details about the relation between pairing procedure and IO capability refer Table 2.5 BLUETOOTH SPECIFICATION Version 4.1 [Vol 3] PART H