PSoC BLE 101: 1. Configure a Find Me Profile

My name is Alan Hawse. I'm Vice President of Technical Staff for Solutions and Software at Cypress Semiconductor. Welcome to my video series on PSoC BLE, our revolutionary new Bluetooth solution.

When I started thinking about these lessons I took a look at how most people introduce new engineers to Bluetooth and I really, really did not like what I found. So I am going to do things a little differently.

I am not going to show a picture of a stack and drone on and on and on about what each layer does. When you teach someone to drive you don't start by teaching them the intricacies of internal combustion engines. It's just too painful and nobody will pay attention.

So, I am going to start with the simple example of a Find Me tag. These tags are the little battery powered things that you attach to car keys Ð or really whatever. Then, when you lose them, you can trigger the tag to flash or beep or whatever using your cell phone.

I'll talk about all of the pieces of this example as I add functionality to it. And then, in future lessons I'll build onto that example. If you watch these lessons, then repeat what I do here with your kit, then you should understand all of the BLE concepts required to get started with your product.

For this first project, I will use the Pioneer Kit with the red radio module. The red LED on the baseboard will serve as the visual indication that the Find Me service has been triggered.
First, let's create a new project. Make sure you select PSoC 4100 BLE/PSoC 4200 BLE in the new project dialog. Next add the BLE component to your schematic. As with all PSoC Creator components, you just double-click to open the customizer dialog.

All BLE devices are implemented with Profiles. A profile is just the BLE word for a Bluetooth SIG specified collection of data and functionality, which are called "Services". Some examples of these profiles include things like heart rate monitors, bike power meters and running speed indicators. It also possible for you to create your own custom profiles, which I will talk about in the last lessons. When you choose a standard profile, PSoC Creator automatically selects the correct Services and characteristics for it. We programmed PSoC Creator to know all about the standard requirements for each of these profiles, which will greatly simplify your work.

A Service is, quite simply, the things that the device can do and the Characteristics is all the information that can be shared. Services and Characteristics are also specified by the Bluetooth SIG. But, once again you could create your own Services and Characteristics. However, for your first design we're just going to use the standard Services. All right, let's get started.

First, select the standard Find Me profile. PSoC Creator will automatically create a GAP Peripheral that implements the standard GATT Server Find Me profile. These two acronyms, GAP and GATT are the fundamental concepts to understanding BLE.

The Acronym GAP stands for Generic Access Profile. You can think of GAP as the mechanism by which two devices get connected, your cell phone and your project.

GATT stands for Generic Attribute Profile. You can think of GATT as the way that connected devices exchange the actual data.

So GAP is what enables connections and GATT is how you exchange data.

As we move forward, I will talk about these concepts in more detail.
Now back to our project. The Find Me profile can be used to either send or receive alerts. In our application we will be receiving alert requests from a cell phone.

For this example, you will be implementing a device that serves in the role of a GATT server. BLE was modeled after the world-wide-web client server model. The device that has the data is called the "server" or more specifically and more accurately the GATT Server. The device that wants to read or write the data is called the client- or more specifically the GATT Client. For this application the GATT client will be your cell phone-, which will write to the Pioneer board GATT server.

The connection between BLE devices is established using GAP. The two main roles in GAP are the GAP Central (which is most often played by a cell phone) and the GAP Peripheral (which is most often the thing you're building). Our device is capable of serving in both of these roles, and in fact switching between them, however, that is pretty complicated and is outside of the scope of these getting started lessons.

For this application we choose use the most common setup, where your device is a GATT Server and a GAP peripheral. In the "Profiles" tab you can see the definitions of all of the profiles (as denoted with a little P). You can see the Services, they are denoted with a little S, and the Characteristics and they are as denoted with a little C. All of this configuration data is held automatically for you in a database- called the GATT database. This database is provided for you by Cypress and is part of the BLE Stack.
All BLE peripherals are required to implement a GAP and a GATT service. These are the first two Services you see in your configuration and they are both handled for you automatically by PSoC Creator.

The last Service you see is the immediate alert service, which contains the alert level characteristic. The Alert Level Characteristic is the actual thing that will be changed by the cellphone GAP central to cause the LED to blink. You can see on the configuration screen that it can be one of three values, no alert, mild alert or crazy, high alert.

For now accept all of the defaults, and then move on to the GAP settings.

Remember, I said earlier, that GAP is a protocol that defines how devices connect. In this next section we will configure the connection parameters of your device. Specifically, how often and what your device advertises.

First, assign a name to your device. This string characters will be used when the device advertises its presence and makes connections with other devices.

I’ll also ask the device to generate its own unique ID number by checking this little box.

Every Bluetooth device is required to have a unique ID (also known as Universally Unique ID or UUID). When I check this box the device will generate a unique ID based on information that is embedded in every PSoC 4 BLE chip.

To make it easier to find the device when I look for it on my phone, I need to ask the component to include the name of the device in the advertising packet.

I also ask to advertise the Service UUID, because that prevents the phone from showing my device as having an “unknown service”. We’ll see that working for us in the next lesson.

That’s all I need to do to set up a BLE peripheral that I can control with my phone or tablet.

In the next lesson we will build the rest of the project and implement the firmware required for the Find Me Profile.

I hope this lesson helps explain some of the basic BLE terminology and gives you an idea of what you can do with the component. Please try this out for yourself. Create a project and add the BLE component, then explore the dialog and the profiles that it allows you to configure.

As always you are welcome to email me at alan_hawse@cypress.com with your comments, suggestions, criticisms and questions.

Watch all PSoC Creator 101 lessons at www.cypress.com/creator101