REVERb

Introduction

As our lives become more hectic by the day, we tend to forget certain trivial stuff. This is where REVERB comes into play. REVERB is our idea of a personal artificial butler. A home automation system which is smart enough to give you quirky and fun responses and uses Internet of Things to make your house Smart.

Unlike the present “Smart Home Devices” which requires a lot of Human Machine/Computer Interaction, Amazon Echo helps us avoid that by the power of speech. For instance, if you have a meeting at 8:30 AM and you woke up at 8:00 AM, you might not have enough time to get ready for work and also have a cup of freshly brewed coffee. Suppose you want to switch on the heater in your bathroom so that you can take bath in hot water. None of these are possible with the current smart home devices without the User to do some physical interaction. Imagine someone who could do all this for you, just by your instructions. Imagine your own personal butler. This butler could be REVERB. Your coffee gets ready while you take bath for work. You get notified of the room ambient parameters when prompted.

**Intel Edison**

General Overview

Intel Edison is a SoC which contains a dual core Microprocessor [Intel Atom] and a single core Microcontroller [Quark]. This small chip [in the size of an SD Card] is also embedded with a Broadcomm Wi-Fi module and a dual mode Bluetooth module [BLE and Classic Bluetooth] with BlueZ Stack [stack supported by Linux]. Since Edison also has 1GB of memory and 4GB of storage, the processor can be uploaded with an OS by Linux called Yocto without the need of an external SD Card.

**Bluetooth:**

Intel Edison has a dual mode Bluetooth module with both Bluetooth Low Energy and Classic Bluetooth capabilities. These modules use an application by Linux named BlueZ. This BlueZ provides various tools like hcitool, gatttool, bluetoothctl to control the Bluetooth module. But sometimes this BlueZ application may not be installed or it might be an older version. As of now the latest BllueZ version supported by Edison is BlueZ-5.24. You can download and install that using the following link.

<https://software.intel.com/en-us/articles/using-the-generic-attribute-profile-gatt-in-bluetooth-low-energy-with-your-intel-edison>

Once the BlueZ folder is extracted, there will be a folder named “test” where all test codes for BlueZ have been written. These codes can be used to receive data through the different profiles available. These test codes can be found on GitHub. The test codes contains codes for some profiles like Heart Rate Profile, Thermometer Profile, and Cycling Speed Profile etc. It also has a code name test-device which can be used to perform the basic functions of BLE module like connect, disconnect, pair, trust etc.

Note: - If you want to pair to a particular Bluetooth module, you will have to first register an agent and then try to pair. Else pairing won’t work.

**Wi-Fi:**

In Edison we have a wpa\_cli interface for connecting to a network. Peer to Peer connection [p2p connect] or Wi-Fi Direct connection between two Edison’s or Edison and an App is also possible. More details are given in the Wi-Fi guide.

To sync data to the cloud, Intel provides a cloud for Edison. This cloud can be accessed by creating an account. The data synced to the cloud is plotted on a graph. The date and time that is synced with the data is the date and time during sync and not the date and time during measurement.

***FOR MORE DETAILS ON BLUETOOTH AND WIFI REFER TO “Bluetooth and WiFi GUIDE.doc”.***

Moreover Python scripts for a server and client can be developed and run on the Edison to communicate with different devices and servers.

**Edison’s Application in Reverb**

The Intel Edison acts as a gateway through which sensors can store data to the cloud. The cloud that we’ve used is ThingSpeak. In this project the Intel Edison communicates with a BMP180 Barometric and Temperature sensor. This sensor has been mounted on to a RedBear Nano module which uses Nordic nRF1822 as the BLE module. This RedBear Nano also has an ARM Cortex M0 micro-controller. The BMP180 is a digital sensor which provides the Temperature and Pressure values when requested through the I2C ports available on the RedBear Nano. The RedBear Nano reads these values and sends them over BLE to the Intel Edison. The Intel Edison reads the values through the Profile and then syncs these values to the ThingSpeak cloud database using the appropriate API key and Channel Number. We tried to create our own custom profile for the communication with the RedBear Nano, but due to some minor errors in the characteristic handlers we were getting errors that we weren’t willing to spend time on the short amount of time given for the competition. So instead we used an already existing profile for transmitting the Temperature and Pressure Values.

The Intel Edison being a powerful platform to act as a gateway, could perform multiple tasks in parallel. The Edison was automated, in the sense the Edison wouldn’t need any Human Intervention for its functioning. It just needs to be turned on and the Edison is ready to be used. Shell scripts were created which would be run during boot up of the board. The Edison automatically connects to the Wi-Fi network and then starts scanning the BLE devices in the vicinity to find whether any devices are available. If the Reverb Home Automation devices are present, the Edison detects them and automatically connects to them. Once connected, the corresponding profiles are run to receive the readings from the sensor module. These values are then instantaneously synced to the ThingSpeak cloud on a corresponding channel. This is then accessed by Amazon Echo which reads the latest values when requested by the User.

**RedBear Nano**

The RedBear Nano is BLE device that uses Nordic nRF51822 module on it with an ARM Cortex M0 microcontroller. It has ports which can be used for communicating using different protocols like UART, SPI and I2C. It also has a 12bit ADC which can be used to read from an analog values. We have used the RedBear Nano interfaced with a BMP180 sensor to send values to the Intel Edison as explained earlier.

**Raspberry Pi 3**

Rasperberry Pi 3 is used to run the EC2 server to generate a Public IP address. As the USB Wireless doesn’t allow us to create a Public IP and ThingSpeak requires a public IP to trigger commands. We are also using it as the main visualization base of the Internet of Things. It displays all the connected sensors and their updated values.

Users can visualize this Output and it would help them know their devices and triggers in detail and would also estimate some useful statistics like mean room temperature, highest pressure to ensure it meets safety standards, even power estimates to help them save electricity.

The Pi 3 also connects to the Actuator shield we have built, this responds to Alexa’s trigger requests. For example, when you ask Alexa to brew your coffee, the relay on the board goes on.

**Amazon Echo**

The Echo is a great and intuitive tool. We all have fallen in love with Alexa over the past 3 hours of hacking. Alexa serves as the main component of this IoT based smart home based system. Alexa can tell you alerts, brew your coffee, wake you up and even open the door before you come in. We live in a technologically thriving world and time has started to become our major obstacles. With your work and busy life, we believe it should be essential for all of us to be walking in a Home that cares for you. A Home that can be prepared in the time of need and adapt to you and your family. So just a Normal home automation system is never going to cut it. Alexa makes you feel that you’re home by adding that very human touch she has.

It was easy to fall in love with Alexa but so was to get irritated when she didn’t work the way you’d expect her too.

It was hard figuring out the Amazon Echo API, but the very helpful volunteers and the most amazing mentors helped us get thought with this, without there help it was not possible.

**Photon Kit**

Some of us had used Arduino before but the Particle Photon was still very new to all of us. The Particle IDE is great with such flexibility to upload over Wi-Fi, import and fork Libraries from GitHub directly, create and use IFTTT recipes. We have built a stand-alone battery powered Sensor Hu using the Photon. It has the sensors like Light, Temperature, Humidity, Fire Detection etc. These sensors post data to the ThingSpeak channel using their awesome API. Also these serve as the trigger points for the Smart Home system. These sensors can update their values to ThingSpeak (at a min. frequency of once per 15 seconds.)

The particle’s API support and easy to use IDE made it a pleasure to work with. The weirdest thing we encountered was not able to get the Interrupts working on the Photon, and apparently a lot of other users have experienced that and it’s still not resolved completely.

References  
Intel Edison

* <https://software.intel.com/en-us/articles/intel-edison-board-getting-started-with-bluetooth>
* <http://www.intel.com/content/dam/support/us/en/documents/edison/sb/edisonbluetooth_331704007.pdf>
* <http://download.intel.com/support/edison/sb/edison_wifi_331438001.pdf>
* <https://www.youtube.com/watch?v=BlJv7Ctgc9c> [Tutorial by SprakFun]
* <https://communities.intel.com/>
* <https://docs.python.org/2/library/urllib2.html>
* <https://github.com/aguedes/bluez>
* <https://www.arduino.cc/en/Guide/IntelEdison#toc1>
* <https://learn.adafruit.com/reverse-engineering-a-bluetooth-low-energy-light-bulb/control-with-bluez>
* <https://developer.bluetooth.org/TechnologyOverview/Pages/Profiles.aspx>
* <https://software.intel.com/en-us/articles/using-the-generic-attribute-profile-gatt-in-bluetooth-low-energy-with-your-intel-edison>
* <http://www.tektyte.com/docs/docpages/edison-reference/runonstartup.html>