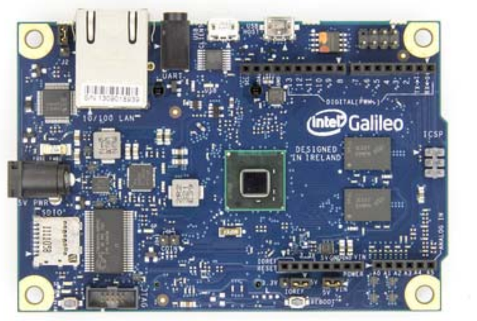
Intel Galileo



Arduino-Compatible Development Board featuring Intel Architecture

Emmanuel Ademuwagun

EECE 494: Computer Bus and SoC Interfacing

Electrical and Computer Engineering

Howard University

Instructor: Dr. Charles Kim

Intel Galileo

- Introduction
- Galileo Relevant Spec
- Hardware Setup
- Software Setup
- Talking Arduino
- Getting into Galileo Via Terminal
- Intel Quark SoC X1000
- Extended Intel Galileo’s Reach
- Bigger Linux Image Installation
- Using SSH for Remote Access
- Mini-Project: Bluetooth Device Discovery

Introduction

- Arduino is a well known open-source hardware designed around a **32-bit Atmel ARM** Processor
- To bring some perspective to the innovation behind the Intel Galileo, Intel and Arduino collaborated to design the first Arduino-compatible Intel microcontroller based on **Intel Quark SoC X1000 processor**

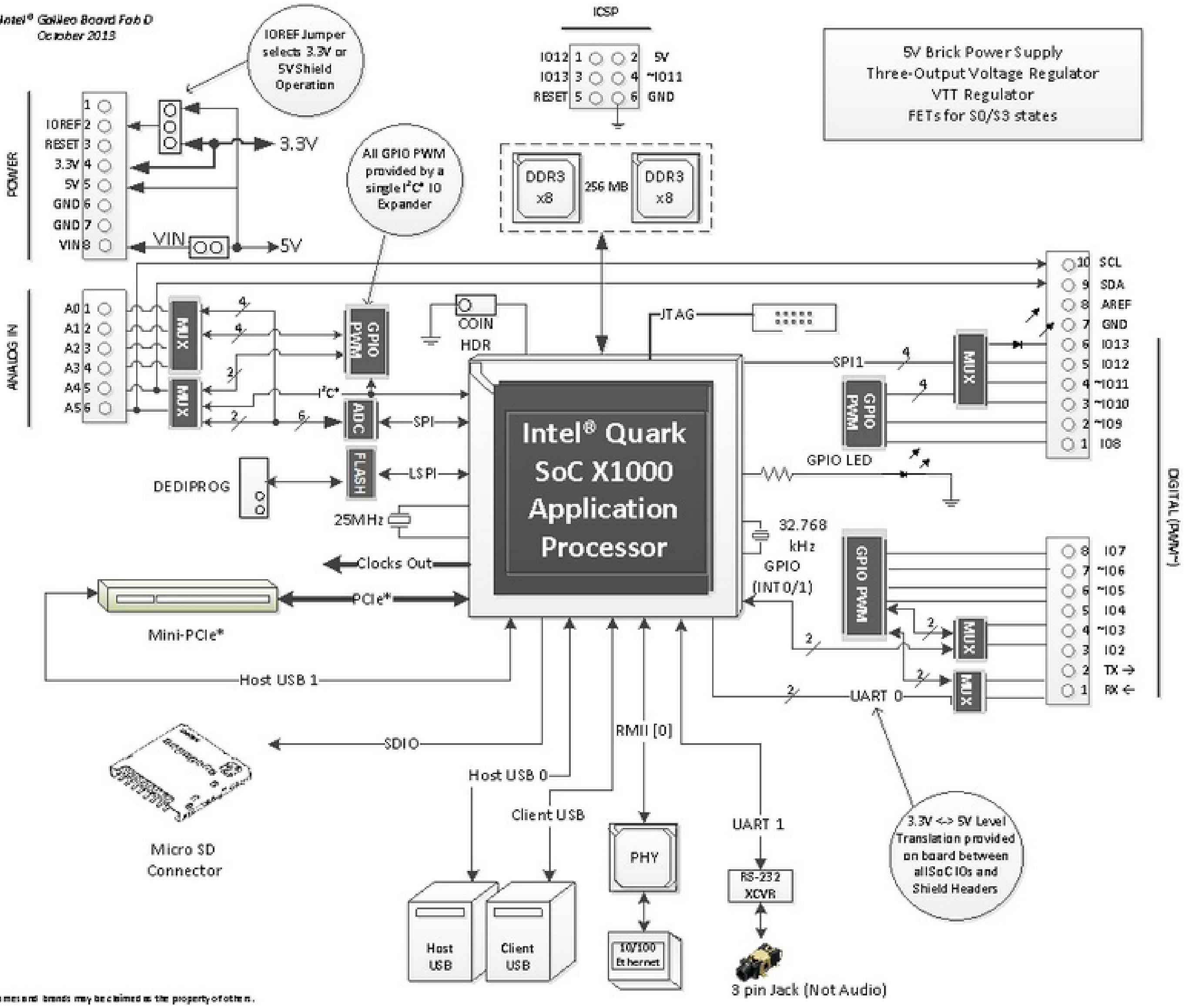
Galileo Relevant Spec

- Intel Quark SoC X1000
  - 400MHz 32-bit Intel Pentium ISA-compatible Processor
- 10/100 Ethernet Connector
- Full PCI Express mini-card slot
- USB 2.0 Host Connector
- USB 2.0 Client Connector
- 10-pin standard JTAG header for debugging
- Reboot button
- Reset Button

Galileo Relevant Spec

- Storage Options
  - 8MB Legacy SPI Flash (store firmware or bootlooader/latest Sketch)
  - 512KB embedded SRAM
  - 256MB DRAM
  - Optional micro SD card offers up to 32GB Storage

Intel® Galileo Board Fab D
October 2013



* Other names and brands may be claimed as the property of others.

Hardware Setup

Development Platforms

- PC: Windows XP and above
- Mac
- Linux

Connect a **USB micro cable** to your development workstation using the USB Client port on the Galileo Board

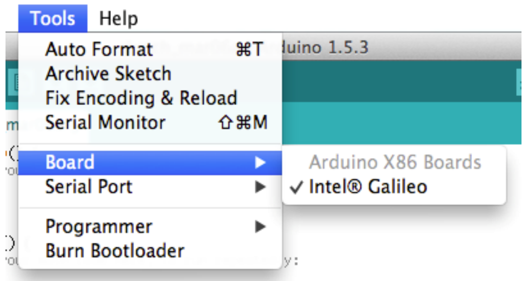
NOTE: Do not turn on the Galileo Board with a connected USB micro cable; you will risk damaging the Board

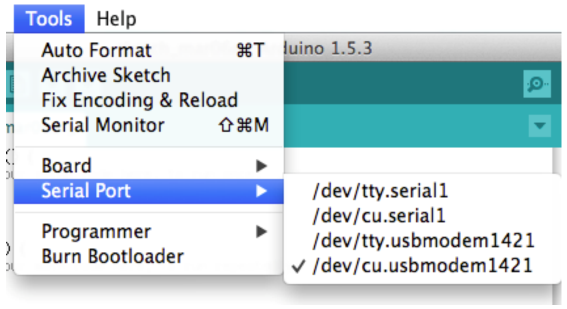
Software Setup

- Download latest IDE and firmware files
  - https://communities.intel.com/community/makers/drivers
- NOTE: If you are using Windows, ensure the name of the path your files are stored does not have spaces. **Galileo_Arduino was built on Linux, and does not like file paths with spaces**
- Install Galileo USB Driver using the **Intel Galileo Getting Started Guide** for your specific Development Platforms (Windows, Mac or *nix)
  - Mac Users do not have to install any drivers

Software Setup

- After USB drivers are installed, launch the Arduino application and ensure that the correct Board and Serial Ports are selected.





- Now, you should be ready to update the firmware on the Galileo Board

Do not upload any sketches without updating the firmware. This could potentially damage the board

Setup Reference

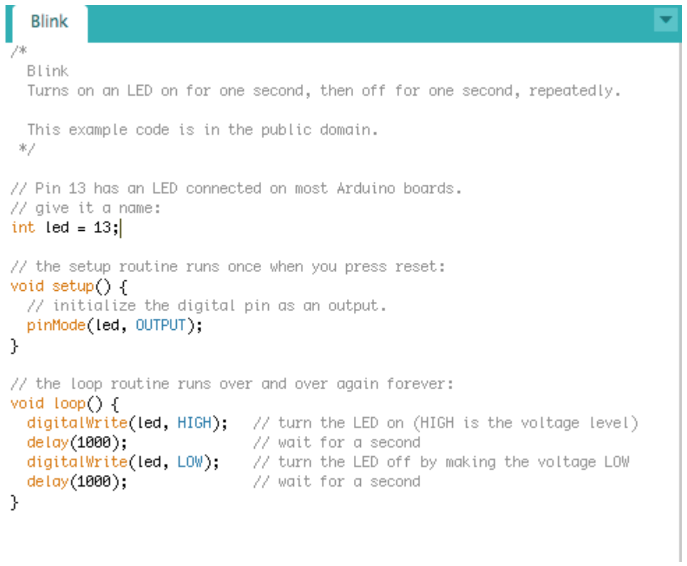
- Intel Galileo Getting Started Guide

https://communities.intel.com/servlet/JiveServlet/downloadBody/21838-102-7-25423/Galileo_GettingStarted_329685_005.pdf

- Debugging Resource: BING!

Talking Arduino

- The Intel Galileo operates just like every other Arduino Board
- For example, you can upload a Blink sketch on the board, and get the same result as you would on any other Arduino Board



Blink

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

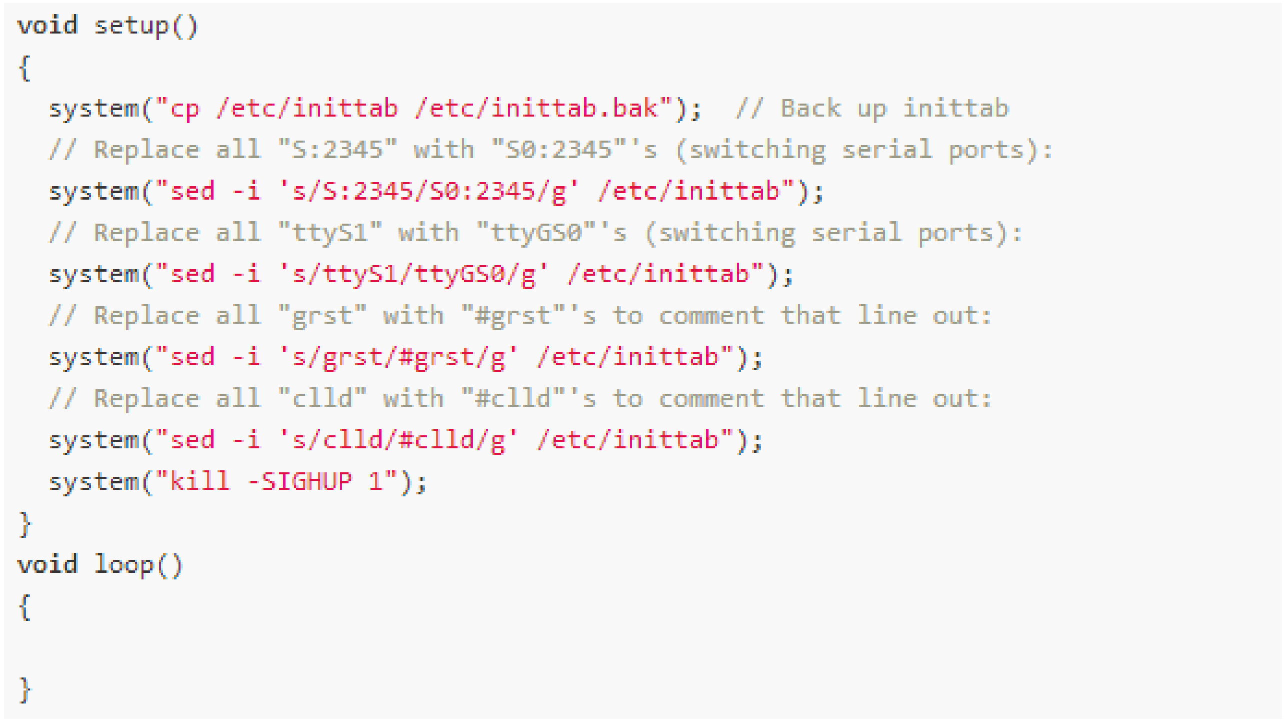
Getting into Intel Galileo via Terminal

- The terminal interface on the Galileo is a **3.5mm stereo jack**, like what you might plug headphones into
- A specialized **3.5mm to DB9 RS-232 cable** can help move the interface to a more common connector, and you may need a **RS-232 to USB cable** on top of that to interface the board with your computer.
- Open a terminal program and change the Baud Rate to 115200 bps
  - For Windows Users, you can use programs like Tera Term
  - For Mac and Linux Users, you can use the **screen** command on Terminal

Some fun hacking! – Not for the weak-hearted

- It is still very possible to access the Intel Galileo via Terminal without **the RANDOM 3.5mm stereo jack**
- You can upload a sketch that will allow access to the Terminal
- **WARNING:** The sketch messes up with some inner workings of Galileo, so you will not be able to upload another sketch while the "hack" sketch is running
- When you are done with the terminal, you can run a few commands to reset the Galileo Board to its original state

Run this sketch to access Terminal

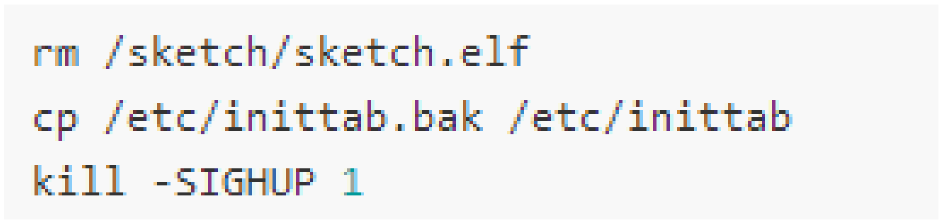


```
void setup()
{
  system("cp /etc/inittab /etc/inittab.bak");  // Back up inittab
  // Replace all "S:2345" with "S0:2345"'s (switching serial ports):
  system("sed -i 's/S:2345/S0:2345/g' /etc/inittab");
  // Replace all "ttyS1" with "ttyGS0"'s (switching serial ports):
  system("sed -i 's/ttyS1/ttyGS0/g' /etc/inittab");
  // Replace all "grst" with "#grst"'s to comment that line out:
  system("sed -i 's/grst/#grst/g' /etc/inittab");
  // Replace all "clld" with "#clld"'s to comment that line out:
  system("sed -i 's/clld/#clld/g' /etc/inittab");
  system("kill -SIGHUP 1");
}
void loop()
{

}
```

Reset Galileo to Regular State

- Before exit, run these commands in order when you are in the Galileo Terminal



```
rm /sketch/sketch.elf
cp /etc/inittab.bak /etc/inittab
kill -SIGHUP 1
```

Intel Quark SoC X1000 Processor

- Very small and designed for very low power consumption
- They only support embedded operating systems
- Clanton – codename for Linux flavor running on the processor
- Official Specs
  - http://ark.intel.com/products/79084/Intel-Quark-SoC-X1000-16K-Cache-400-MHz

Extending Intel Galileo’s Reach

- Intel Galileo, out of the box, has amazing applications, but it is limited in space and function
  - The flavor of Unix on the device is very limited in terms of the number of commands it comprehends
  - The size of the internal storage is not enough for running a lot of applications
- This problems are by design, there is a solution: an optional (but mandatory in a sense) SD storage option (up to 32GB)

“Bigger” Linux Image Installation

- You can download the “Bigger” Linux Image from:
  - http://downloadmirror.intel.com/23171/eng/LINUX_IMAGE_FOR_SD_Intel_Galileo_v0.7.5.7z
- **Format** a micro SD card with FAT or FAT32 partition
  - You can access the micro SD card
- Unzip the downloaded image and store the contents at the **root** of the micro SD card

“Bigger” Linux Image Installation

- Ensure the Galileo Board is turned off
- Slide the card in the SD slot on the Galileo Board
- Connect the board to Power
  - You should notice some activity by the LED on the right of the SD card slot
  - If not, the board is not booting from micro SD card – look over steps again; you might be missing something
  - Still no success…

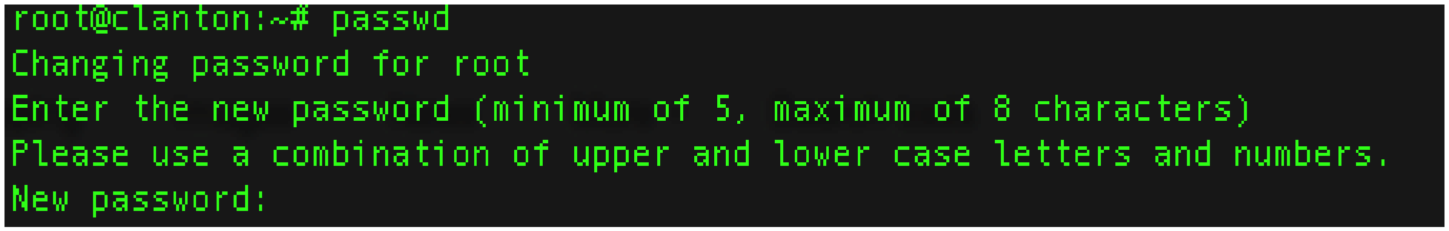
“Bigger” Linux Image Installation

- This Linux Image has the following in-built utilities (among others):
  - Wifi Drivers
  - Python
  - Node.js
  - SSH
  - openCV
  - ALSA – Advanced Linux Sound Architectures
  - V4L2 – Video4Linux2
  - BlueZ tools

Using Secure Shell (SSH) for Remote Access

Precondition: Connection to an existing network

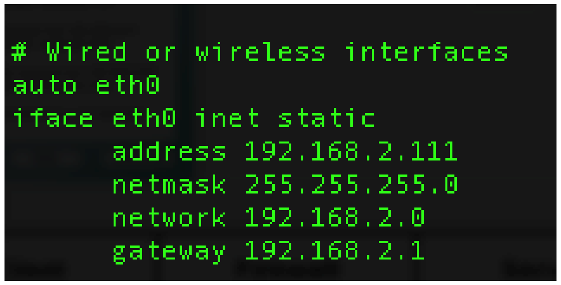
- Hack into Galileo (using the sketch method) via terminal
- Set a password for **root** (login user). This will be your password for remote access



```
root@clanton:~# passwd
Changing password for root
Enter the new password (minimum of 5, maximum of 8 characters)
Please use a combination of upper and lower case letters and numbers.
New password:
```

Using Secure Shell (SSH) for Remote Access

- By default, the eth0 interface uses a DHCP server to get its IP address. This is disadvantageous because the IP address when its lease has expired
- Assign a static IP address to the `eth0` interface so that you can access it anytime with the same IP. Type the command: `vi /etc/network/interfaces` to edit Network Configuration File. For example:



```
# Wired or wireless interfaces
auto eth0
iface eth0 inet static
        address 192.168.2.111
        netmask 255.255.255.0
        network 192.168.2.0
        gateway 192.168.2.1
```

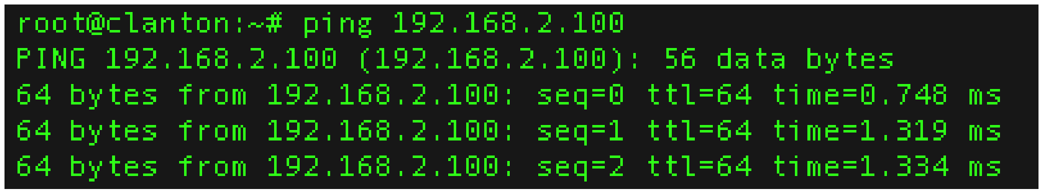
Enter your network information, and ensure the assigned IP address is unique to your network.

Using Secure Shell (SSH) for Remote Access

- Run the following command to restart the network interface to activate your new configuration

```
/etc/init.d/networking restart
```

- To test connectivity, you can ping your development machine. You should get a response

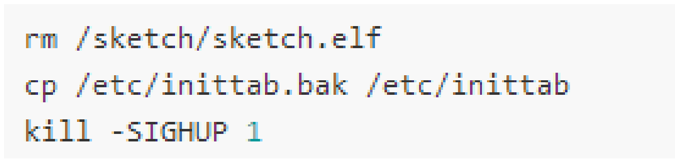


```
root@clanton:~# ping 192.168.2.100
PING 192.168.2.100 (192.168.2.100): 56 data bytes
64 bytes from 192.168.2.100: seq=0 ttl=64 time=0.748 ms
64 bytes from 192.168.2.100: seq=1 ttl=64 time=1.319 ms
64 bytes from 192.168.2.100: seq=2 ttl=64 time=1.334 ms
```

- Your Galileo is now ready for SSH-ing

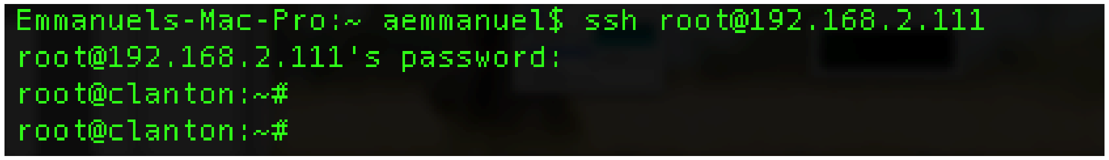
Using Secure Shell (SSH) for Remote Access

- Reset Galileo to its original state:



```
rm /sketch/sketch.elf
cp /etc/inittab.bak /etc/inittab
kill -SIGHUP 1
```

- Exit the Terminal
- You should now be able to SSH to Galileo using your development machine



```
Emmanuels-Mac-Pro:~ aemmanuel$ ssh root@192.168.2.111
root@192.168.2.111's password:
root@clanton:~#
root@clanton:~#
```

More fun with Protocols

- The connection possibilities are limitless
- Another interesting protocol to play around with is TFTP (Trivial File Transfer Protocol)
  - You can transfer files from your development machine to the Galileo Board via TFTP after some configuration setup have been done on both other ends

Mini-Project: Bluetooth Device Discovery

Project Description: Find Bluetooth Devices in the area

Precondition(s):

Access to Galileo via Terminal

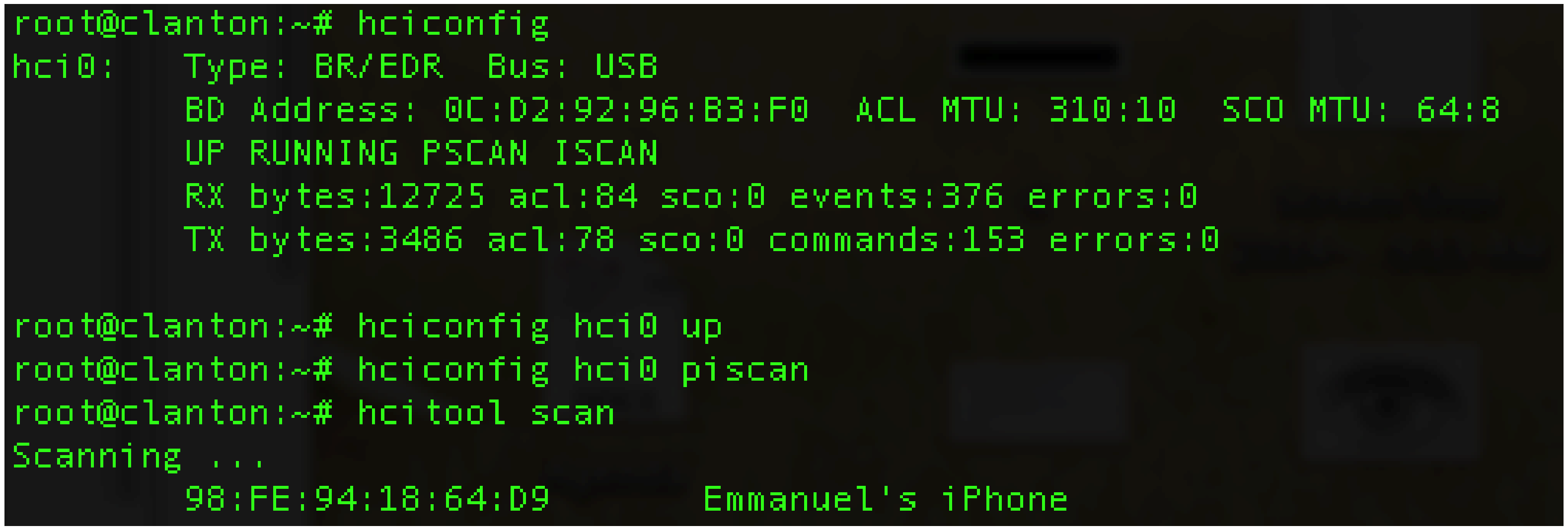
NOTE: You can also write a sketch for Bluetooth device discovery for automation

- The SD card image contains BlueZ tools with in-built features for Bluetooth connectivity
- If you do not have the SD card, you can try to download and compile the source files for BlueZ tools into the internal storage of Galileo
  - Good luck! I did not have much success on this, because I hit the storage limit on compilation

Mini-Project: Bluetooth Device Discovery

- Get an Intel Wireless Card that is compatible with the Galileo Board and attach to the PCIe-Mini Slot
  - I am currently using **Intel Wireless Card N-135**
- To test that the device exists, run `hciconfig`
- To turn on Wireless Interface, run `hciconfig hci0 up`
- To turn on Discovery mode, run `hciconfig hci0 piscan`
  - Conversly, to turn off Discovery mode, run `hciconfig hci0 pscan`
- To find a Bluetooth device, run `hcitool scan`

Mini-Project: Bluetooth Device Discovery



```
root@clanton:~# hciconfig
hci0:   Type: BR/EDR  Bus: USB
        BD Address: 0C:D2:92:96:B3:F0  ACL MTU: 310:10  SCO MTU: 64:8
        UP RUNNING PSCAN ISCAN
        RX bytes:12725 acl:84 sco:0 events:376 errors:0
        TX bytes:3486 acl:78 sco:0 commands:153 errors:0

root@clanton:~# hciconfig hci0 up
root@clanton:~# hciconfig hci0 piscan
root@clanton:~# hcitool scan
Scanning ...
        98:FE:94:18:64:D9       Emmanuel's iPhone
```

Mini-Project: Bluetooth Device Discovery

- You can do more by with this utility:
  - Pairing with a Bluetooth device
  - Transferring files via Bluetooth

Thank you for your time!