



Spade

Chengyun Liu, Romi Phadte, Lawrence Supian

Product Summary

- Payment device
- Vendor-side
- Gets data from cell phone
- Emulates card swipe

Components

iOS



Nordic



Arduino



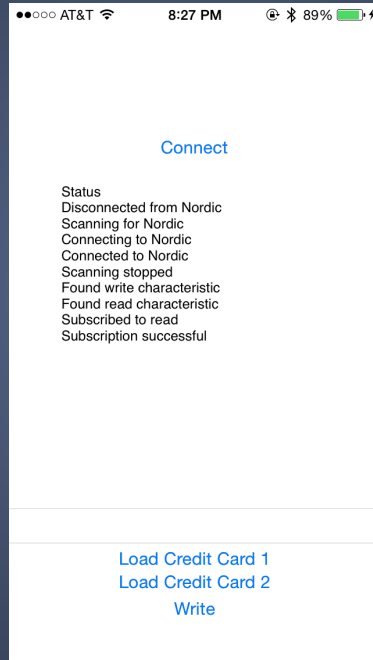
Electromagnet coil

iOS Development

- Language: Objective C
- Tools: CoreBluetooth library

- Function
- App sends credit card data to Nordic board

iOS Development



iOS Development

The screenshot displays the Xcode development environment for an iOS application named "Wallet".

- File Explorer (Left):** Shows the project structure for "Wallet" (iOS SDK 8.1), including files like AppDelegate.h, AppDelegate.m, ViewController.xib, ViewController.h, ViewController.m, Images.xcassets, LaunchScreen.xib, Supporting Files, UUID.h, WalletTests, and Products.
- Storyboard (Center-Left):** Shows a storyboard with a "Connect" button and a "Status" label. Below the button are two "Load Credit Card" buttons and a "Write" button. A "wAny" constraint is visible.
- Code Editor (Center-Right):** Shows Objective-C code in ViewController.m. The code implements a method to connect to a Bluetooth peripheral. It logs discovered peripherals, saves a local copy of the peripheral, and attempts to connect. If successful, it updates the status view. If it fails, it logs the error and cleans up.
- Console (Bottom):** Shows system logs. A log entry indicates that scanning stopped. Another log entry shows the discovery of a Bluetooth service with UUID 8B8B3220-5A0D-11E4-93E8-0002A5D5C51B. A third log entry shows the discovery of a Bluetooth characteristic with UUID FB71BC00-5A0C-11E4-91AE-0002A5D5C51B.
- Right Panel:** Shows the "Outlets" and "Referencing Outlets" sections. The "Outlets" section includes a "delegate" outlet. The "Referencing Outlets" section includes a "writeField" outlet.

Nordic Development

- Language: C++

- Tools: mbed platform

- Function?

- Receives credit card data and sends it in binary as digital signal to electromagnet coil at a desired frequency

Nordic Development



Nordic Development

The screenshot shows the mbed IDE interface. The top bar displays the file path `/BLE_API_DEMO/main.cpp`. The left sidebar shows a project tree with folders like `BLE_API_DEMO`, `BLE_API`, `common`, `public`, `services`, `Classes`, `Structs`, `Groups`, `nRF51822`, `btle`, `common`, `nordic`, `Files`, `Structs`, `Groups`, and source files like `nRF51Gap.cpp`, `nRF51Gap.h`, `nRF51GattServer.cpp`, `nRF51GattServer.h`, `nRF51822n.cpp`, `nRF51822n.h`, `projectconfig.h`, `main.cpp`, and `mbed`.

The main editor shows the following C++ code in `main.cpp`:

```
120 }
121 set_low();
122 }
123
124
125 const uint8_t LED1_UUID[LENGTH_OF_LONG_UUID] = {
126     0xdb, 0x71, 0xbc, 0xcd, 0x5a, 0x0c, 0x11, 0xe4,
127     0x91, 0xae, 0x00, 0x02, 0xa5, 0xd5, 0xc5, 0x1b
128 };
129 const uint8_t BUTTON_UUID[LENGTH_OF_LONG_UUID] = {
130     0x7a, 0x77, 0xbe, 0x20, 0x5a, 0x0d, 0x11, 0xe4,
131     0xa9, 0x5e, 0x00, 0x02, 0xa5, 0xd5, 0xc5, 0x1b
132 };
133 const uint8_t TEST_SERVICE_UUID[LENGTH_OF_LONG_UUID] = {
134     0xb0, 0xdb, 0x58, 0x20, 0x5a, 0x0d, 0x11, 0xe4,
135     0x93, 0xee, 0x00, 0x02, 0xa5, 0xd5, 0xc5, 0x1b
136 };
137
138 const static char DEVICE_NAME[] = "Nordic";
139 static volatile bool is_button_pressed = false;
140 static volatile uint16_t led1_handler;
141
142 uint8_t led_state, button_state;
143
144 void disconnectionCallback(Gap::Handle_t handle, Gap::DisconnectionReason_t reason)
145 {
146     ble.startAdvertising(); // restart advertising
147 }
148
149 void changeLED(const GattCharacteristicWriteCParams *eventDataP) {
150     // eventDataP->charHandle is just uint16_t
151     // it's used to dispatch the callbacks
152     // if (eventDataP->charHandle == led1_handler) {
153     //     led1 = eventDataP->data[0] & 2;
154     // }
```

Compile output for program: BLE_API_DEMO

Errors: 0 Warnings: 2 Infos: 1

Description	Error Number	Resource	In Folder	Location
Variable "count" was declared but never referenced "int count = 0;"		main.cpp		Line: 257, Col: 9
Variable "TEST_SERVICE_UUID" was declared but never referenced "const uint8_t TEST_SERVICE_UUID[LENGTH_OF_LONG_UUID] = {"		main.cpp		Line: 133, Col: 15
Success!		Build Details		

Compile Output Find Results Notifications

Ready.

In 108 col 16 279 | INS

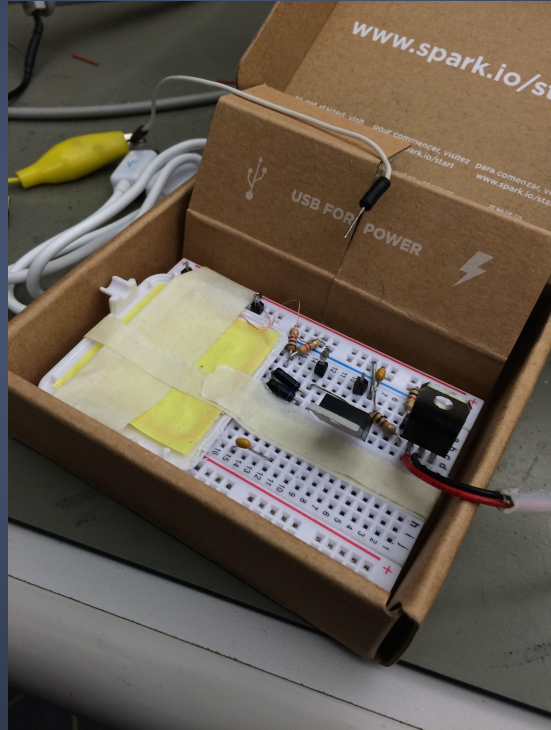
Electromagnet Coil

-Hardware: Multimeter,
oscilloscope, Square reader

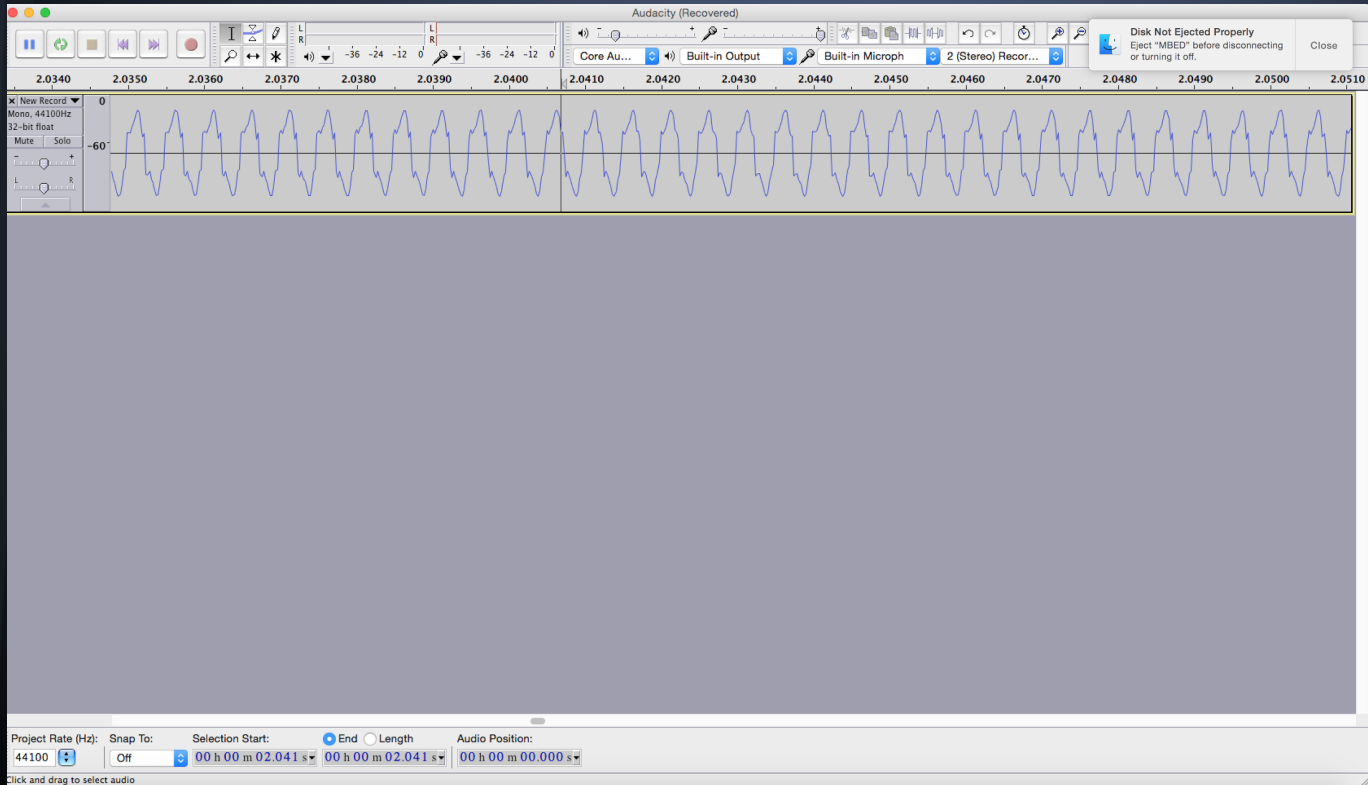
-Software: Audacity, Square app

-Function: Translate digital signal into
electromagnetic pulses in emulation of credit
card magnetic swipe

Electromagnet Coil



Electromagnet Coil



Theory: BLE

Bluetooth 4.0

- central connects to peripheral, peripherals may have services, services have characteristics, i.e. read, write, notify
- could be useful for communication protocol
- up to 100m range, can detect strength of signal

Embedded Systems: BLE

- communication between devices
- devices in states depending on messages received
- wait and receive next message to “change to next state”
- waiting, sending, acknowledging important because separate, concurrent systems

Testing: BLE

- testing communication
- automatic ack, notification for packets
- large packets failed
- discovered “recommended” limit of 20 data bytes per packet

Theory: Security

Ensure it is the Spade of the vendor we are purchasing from?

- register Spade online with vendor name, public key, location

MITM?

- use nonce

Theory: NFC

Bypass need for hardware with NFC EM?

- not feasible

- 13.56 MHz crystal oscillator soldered, not interfaceable

- crystal needed for intended function at NFC 13.56MHz

- thus doesn't work for us because we need the frequency to be KHz in order to correspond with swipe speed

- source: Texas Instruments NFC Info Sheet <http://www.ti.com/lit/sg/slyt493/slyt493.pdf>

Testing: Nordic Frequency

- checking Nordic board frequency
- need somewhat better than 1KHz so we can toggle pin at 1KHz
- got >100MHz, definitely more than enough
- delay toggling of pin if necessary to adjust to 1KHz

Theory: Credit Card (Data Level)

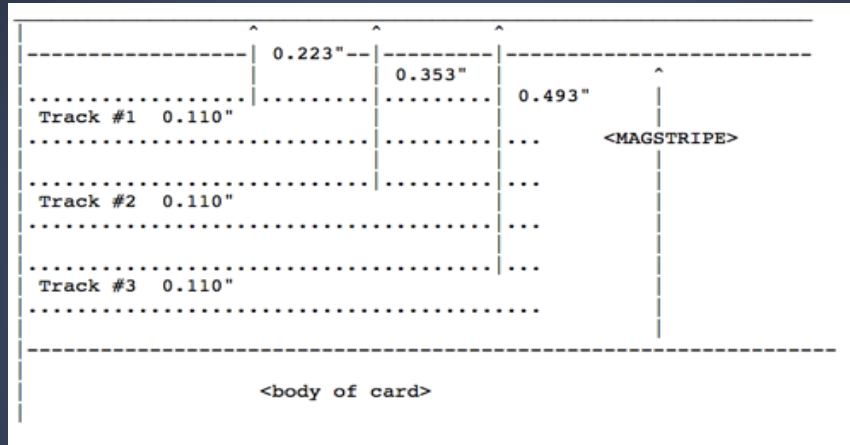
Track 1:

%B4465411255386057^PHADTE/ROMI^
1601101100001000000000494000000?;

Track 2:

4465411255386057=1601101100001494?

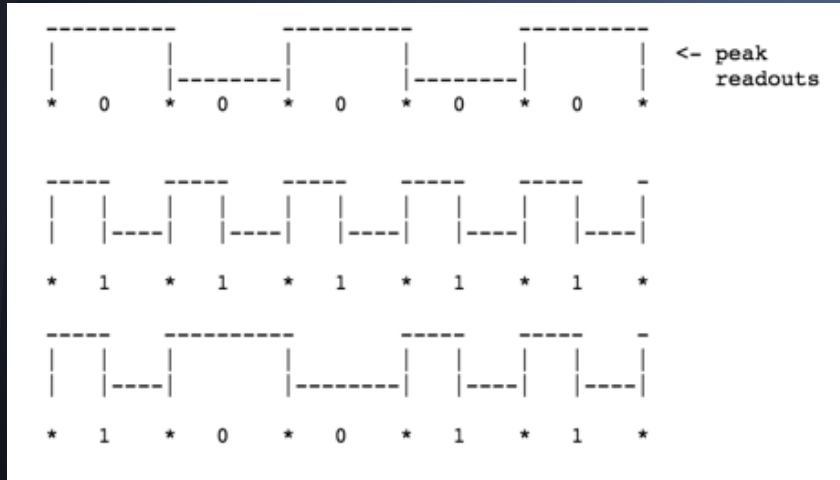
Theory: Credit Card (Protocol Level)



Track	Name	Density	Data-Rate	Format	Characters	Use
1:	IATA	210 bpi	1.05kHz-10.5kHz	ALPHA	79	Name+Account
2:	ABA	75 bpi	375Hz-3.75kHz	BCD	40	Read Account
3:	Thrift	210 bpi	1.05kH-10.5kHz	BCD	107	Encode

Each begins and ends with a sentinel. Humans swipe with speed of 5-50 in/s. We used this to settle on frequencies of about 2-3kHz. This modeling was verified to work in real world tests.

Theory: Credit Card (Byte Level)



--Data Bits--					Parity	Character	Function
b1	b2	b3	b4	b5			
0	0	0	0	1	0 (0H)	Data	
1	0	0	0	0	1 (1H)	"	
0	1	0	0	0	2 (2H)	"	
1	1	0	0	1	3 (3H)	"	
0	0	1	0	0	4 (4H)	"	
1	0	1	0	1	5 (5H)	"	
0	1	1	0	1	6 (6H)	"	
1	1	1	0	0	7 (7H)	"	
0	0	0	1	0	8 (8H)	"	
1	0	0	1	1	9 (9H)	"	
0	1	0	1	1	: (AH)	Control	
1	1	0	1	0	; (BH)	Start Sentinel	
0	0	1	1	1	< (CH)	Control	
1	0	1	1	0	= (DH)	Field Separator	
0	1	1	1	0	> (EH)	Control	
1	1	1	1	1	? (FH)	End Sentinel	

***** 16 Character 5-bit Set *****
 10 Numeric Data Characters
 3 Framing/Field Characters
 3 Control Characters

Theory: Credit Card (Physical Level)

N-S.N-S.N-S.N-S.N-S.N-S.N-S.N-S <-particles in stripe

```

      ||| ||| <-flux lines
N-----N-N-S-S-----S
flux lines -> ||| |||
    
```

magstripe----> -----NN-----SS-----NN-----SS-----

voltage---->+.....-.....+.....-.....

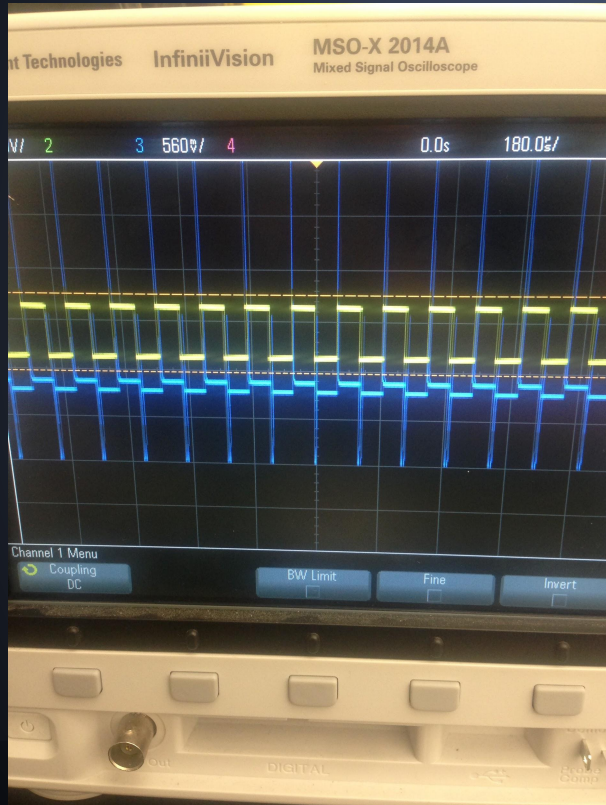
```

peak readout-->
                |-----|
                |-----|
                |-----|
                |-----|
    
```

```

      | | <----wires leading to solenoid
      | | (wrapped around ring)
    /-|-|-\\
    |       | <----solenoid (has JUST changed polarity)
    \ N S / <---gap in ring.. NS polarity across gap
N-----SS-N-----S
      ^^
<<<<<<-direction of stripe movement
    
```

Testing: Electromagnet Emulation



$$v(t) = L \frac{di}{dt}$$

$$V = IR.$$

Next Steps

Create custom PCB for circuit.

Miniatimize device.

Improve app

Remove Arduino.