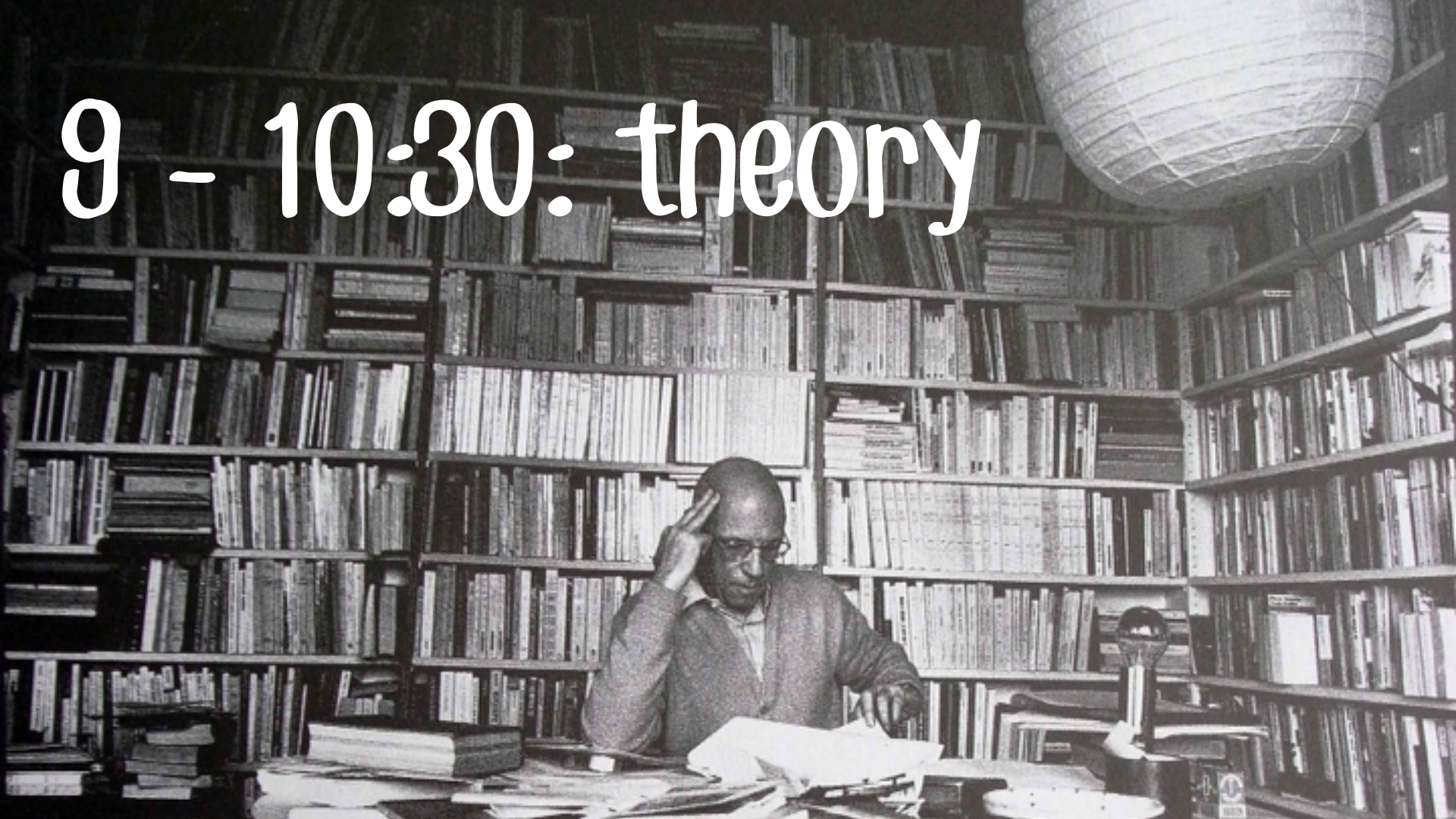
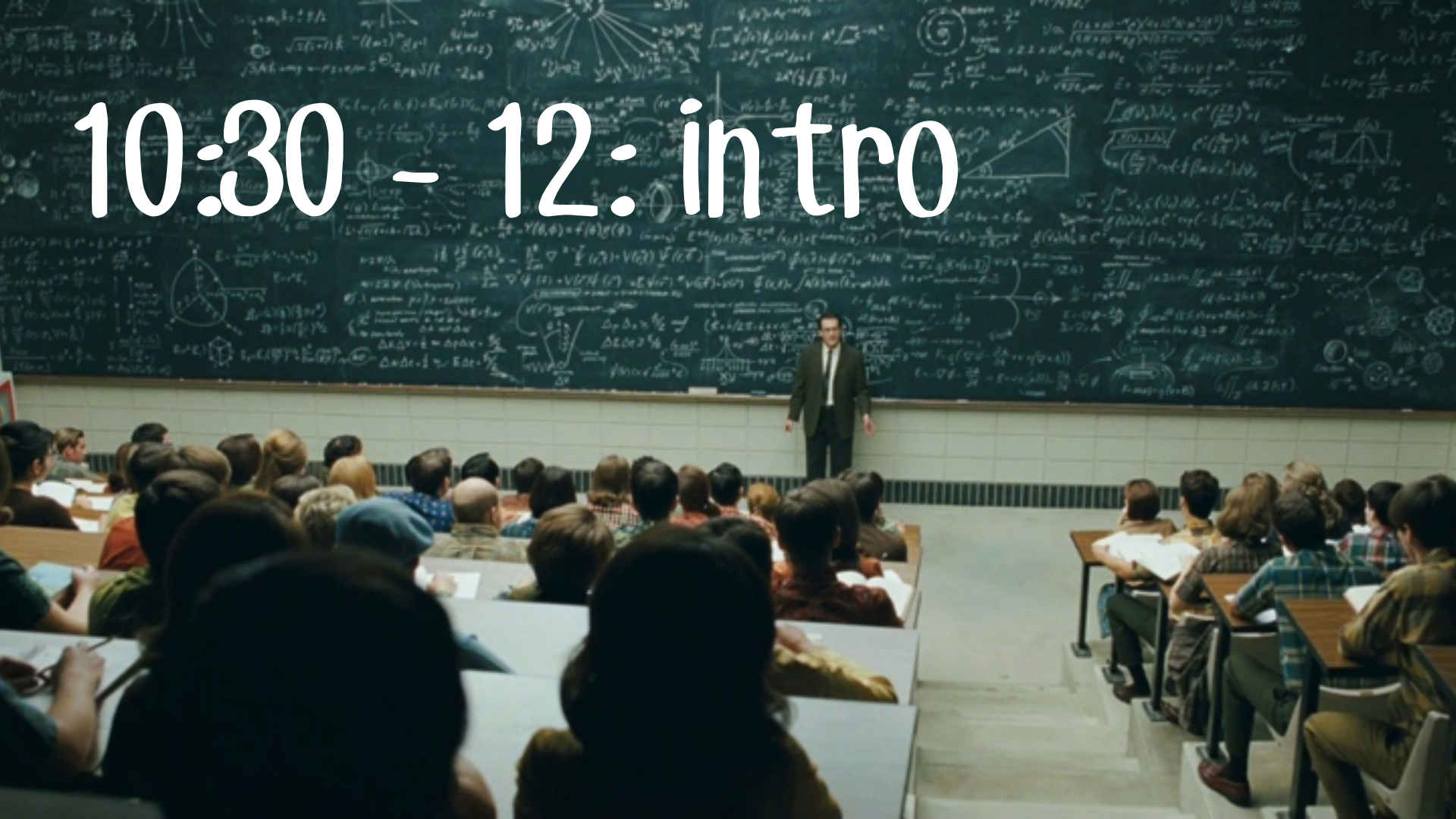
Helping Tiny Things Talk

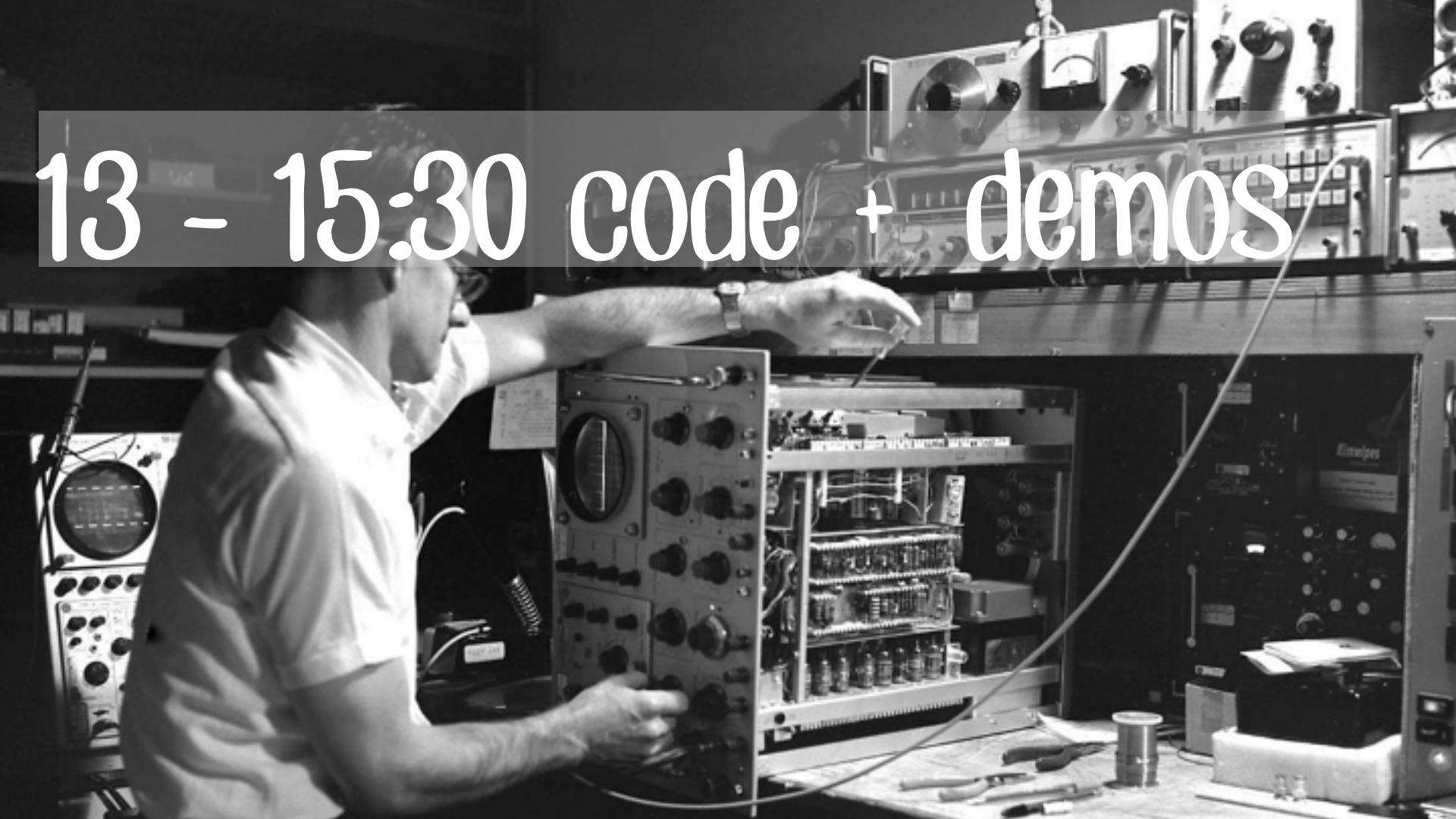
inst-int 2014

9 - 10:30: theory



10:30 - 12: intro





13 - 15:30 code + demos


15:30 - 16:30
troubleshooting

Why am I teaching this course?

frog: I work there

CIID: I teach there

Seattle: I live there

I'm kind of a designer & I'm also kind of a programmer but mostly I think about how things should work

Let things be themselves
Let things that are supposed to
be computers be computers
Let people use things they know

Things don't need to be screens if they can just talk to a screen

Things that can talk to each other without us needing to intervene are fascinating

talking is hard

Electricity over wires, Electricity through air, Light through air, Light through wires, Sound wave, Sound pattern, Vibration, Color,

Why are you taking this course?

caveats

This workshop is insane

Things might not work

We might run out of time

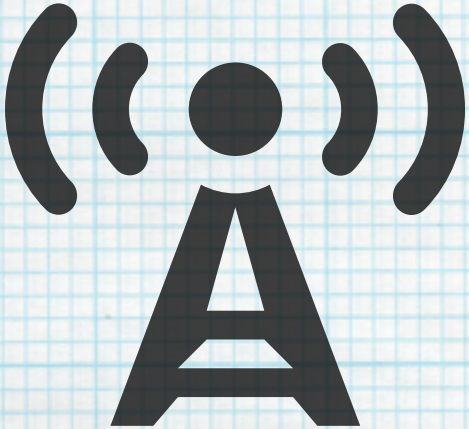
Everything could go wrong

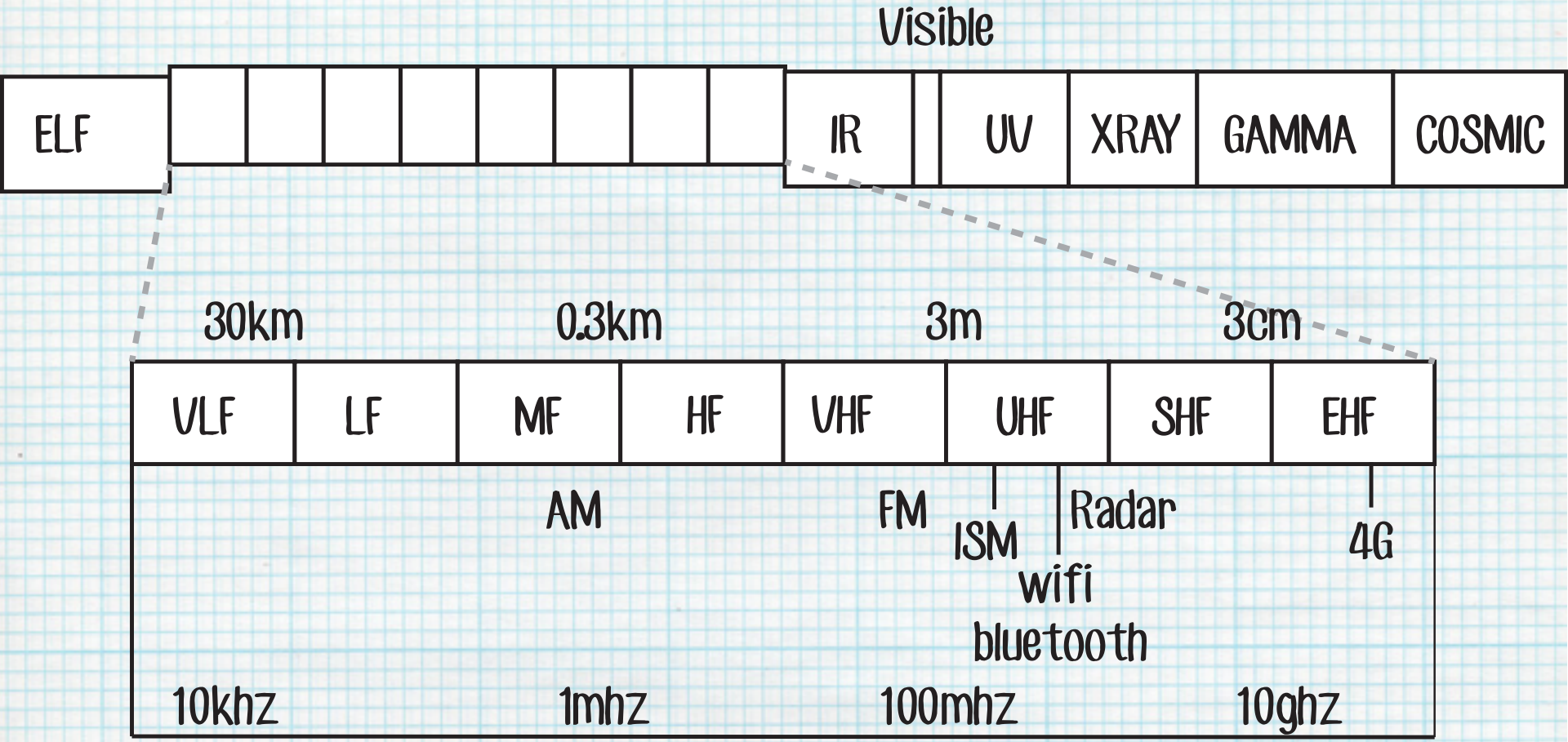
You may not learn one thing you wanted to

general principles

This is about 1 tool that uses 1 technology *but* it's also more generally about the internet of things (yeah yeah I know).

What's radio?




Visible
ELF
IR
UV
XRAY
GAMMA
COSMIC
30km
0.3km
3m
3cm
VLF
LF
MF
HF
VHF
UHF
SHF
EHF
AM
FM
ISM
Radar
wifi
bluetooth
4G
10khz
1mhz
100mhz
10ghz

ISM frequency = we can use it

315 mhz

434 mhz

915 mhz

2.4 ghz (Bluetooth!)

5.8 ghz

One way (aka simplex)

Half duplex

Full duplex

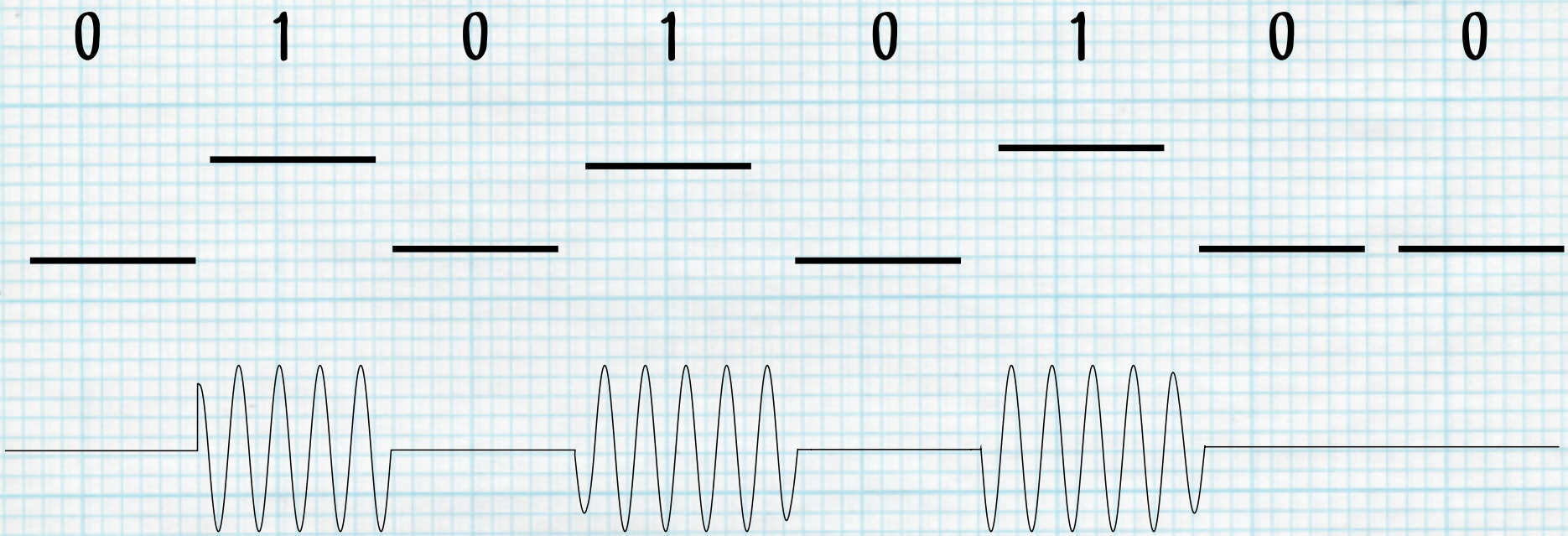
Radio frequency communication

Duplex implies data not just sound.

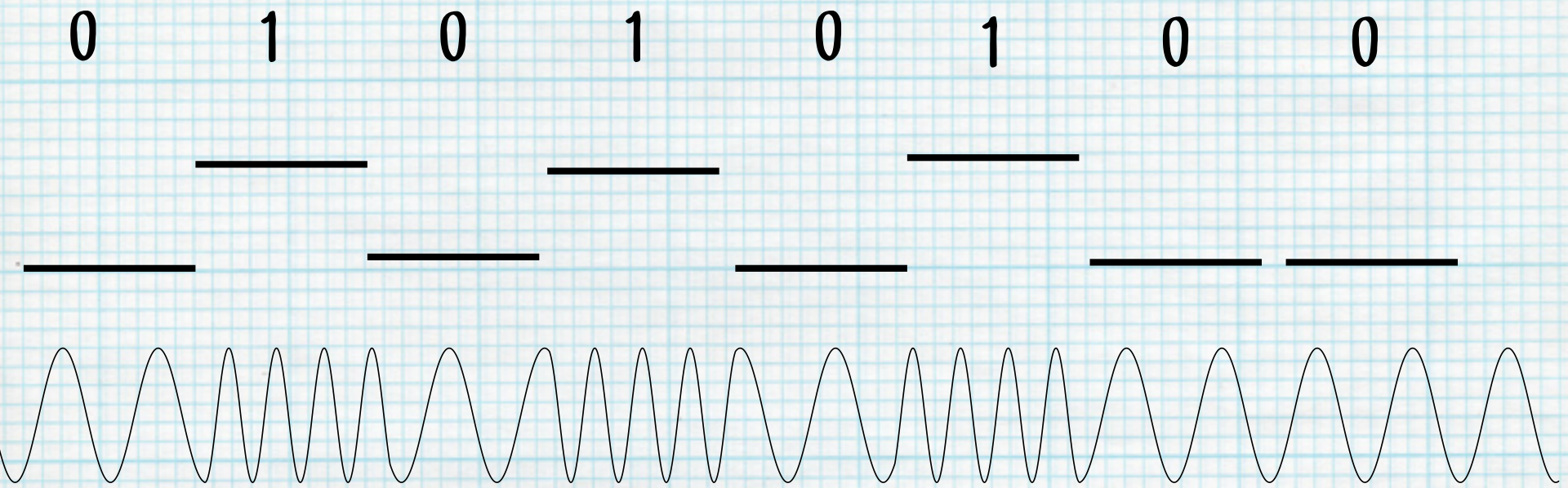
Data implies 1 and 0s.

So how do you turn waves into 1s and 0s?

Sending some data ASK



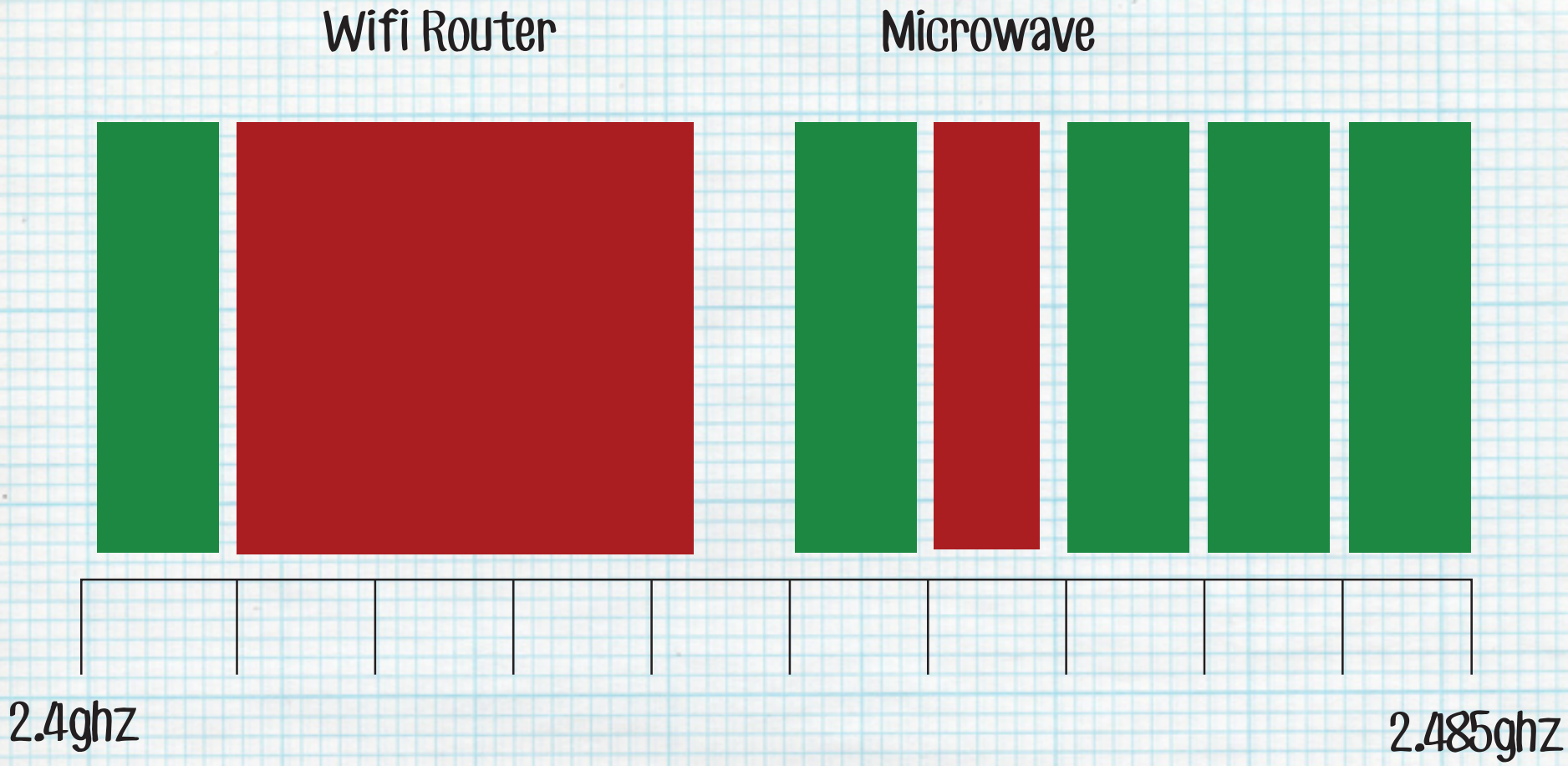
Sending some data FSK



What happens if there's noise?

What's noise? Something on the same frequency. A wifi router, microwave, wireless mouse, a remote control car, garage door, etc.

Well, that depends. Bluetooth uses frequency


Wifi Router
Microwave
2.4ghz
2.485ghz

What is Bluetooth?

2.4 to 2.485 GHz

frequency hopping spread spectrum

full-duplex signal at 1600 hops/sec

adaptive hopping among 79 freqs

What does Bluetooth need?

A transceiver

An antennae

A mC

Some power

How does Bluetooth work?

caveat: this is not how BLE works

1. frequency agreement
2. communication agreement
3. communication

How does BLE work?

I'm a device!

Yeah! What kind of device are you?

I'm an X. What are you?

I'm an Y. Let's connect!

Ok. Do you have any services?

Yep, I've got XX and YY

What's characteristics does XX have?

XX has AAA and BBB

Can I read AAA?

Yep. You can read it.

Can I read BBB?

Nope. You can only write to it.

Ah, well I'll subscribe to AAA. What's it at now?

It's 0x93c8d0

How does BLE work?

Device: a device

Service: a set of characteristics that other devices can be read and written

Characteristic: a data point in a service that can be read/written

nrf51822: a Transceiver + ARM mC

nrf51822

ARM M0 core

Bluetooth transceiver

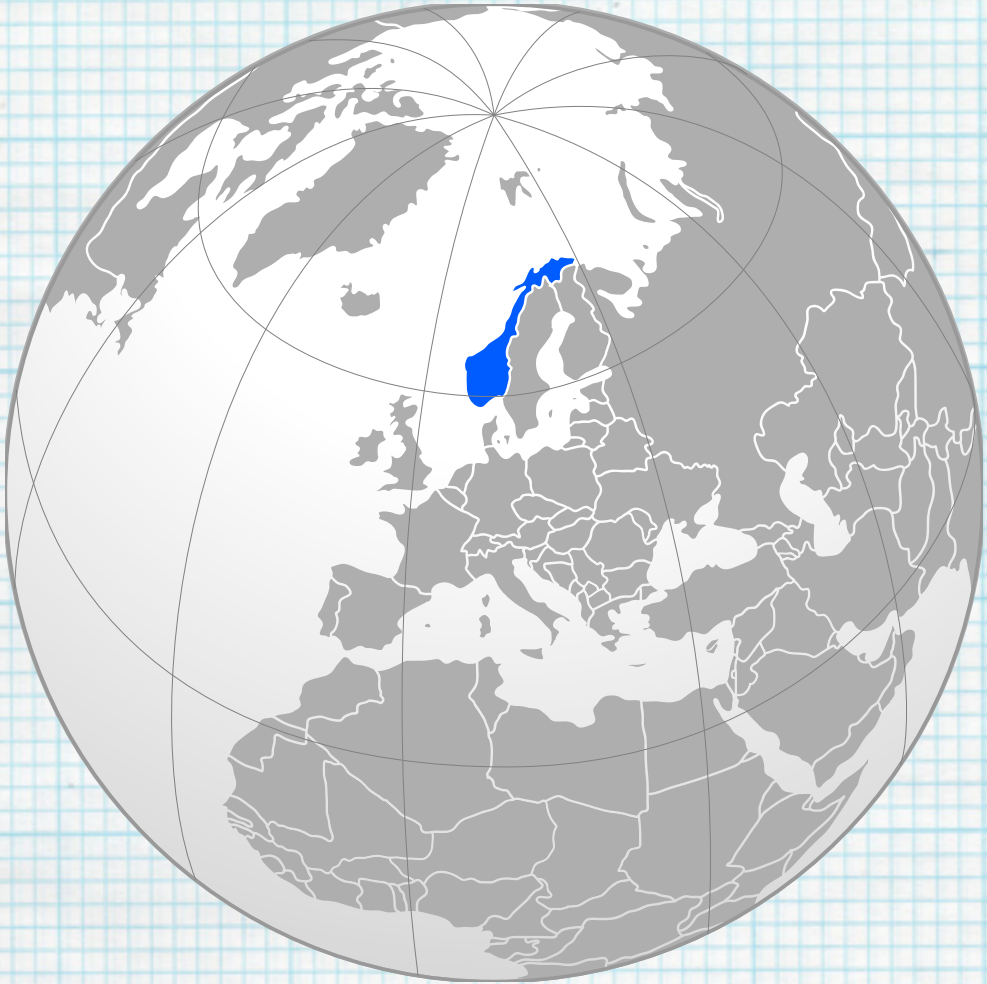
256KB flash 16KB RAM

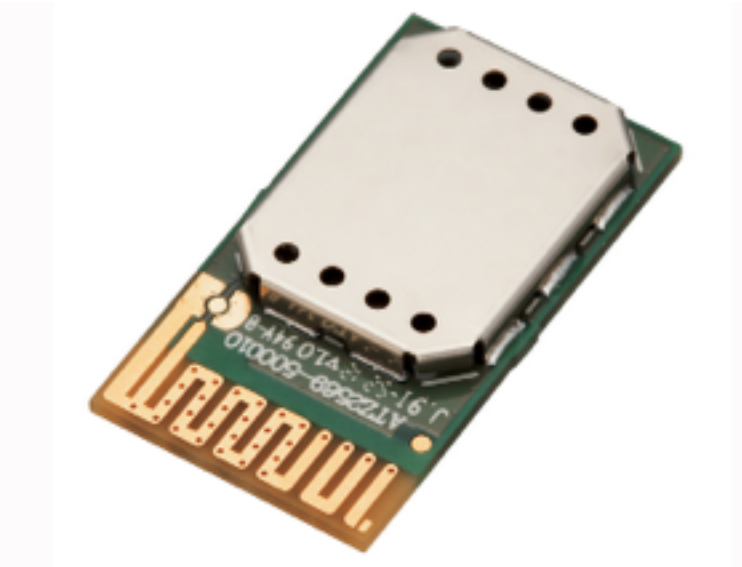
3 data rates (2Mbps/ 1Mbps/250kbps)

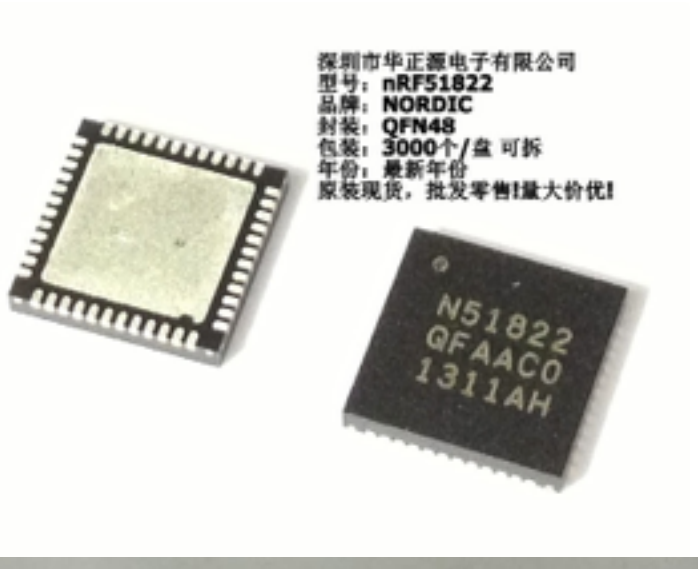
31 GPIO

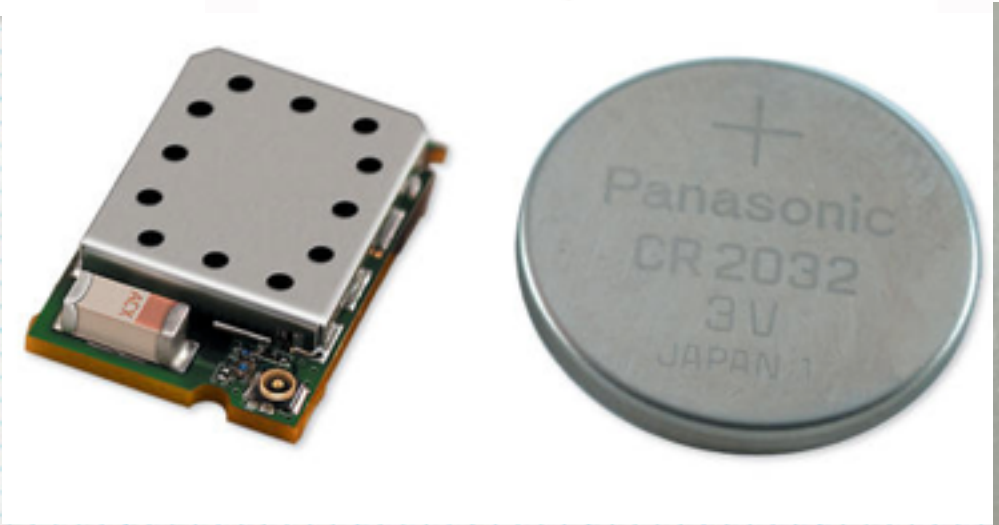
Up to 4 PWM

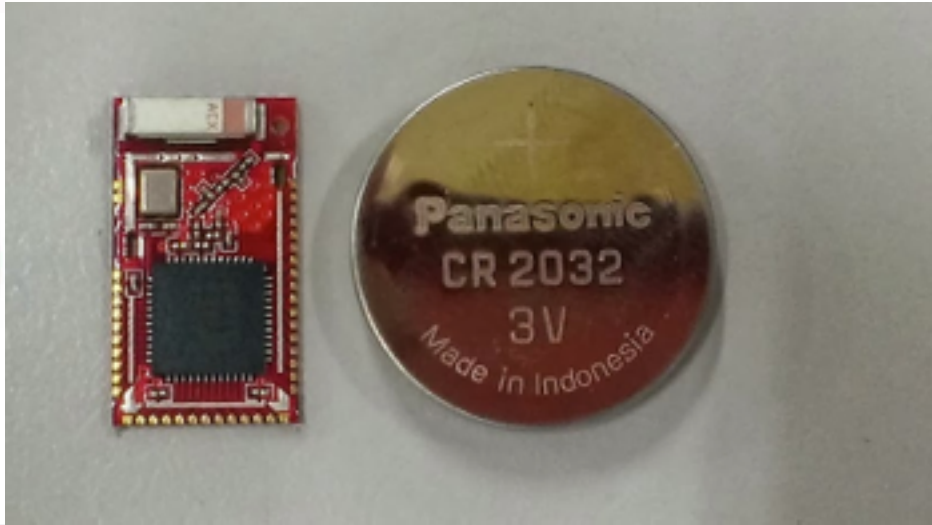
NORDIC
SEMICONDUCTOR





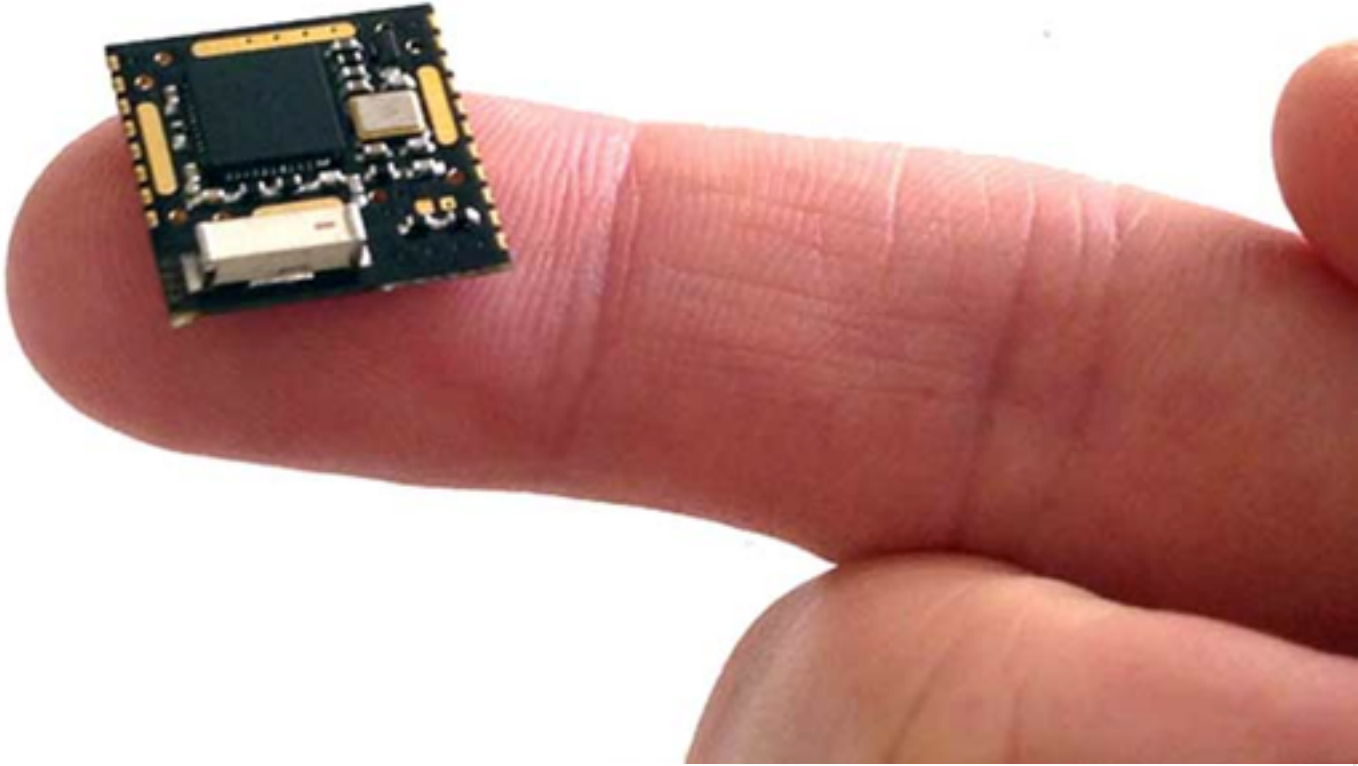

深圳市华正源电子有限公司
型号：nRF51822
品牌：NORDIC
封装：QFN48
包装：3000个/盘 可拆
年份：最新年份
原装现货，批发零售!量大价优!
N51822
QFAAC0
1311AH

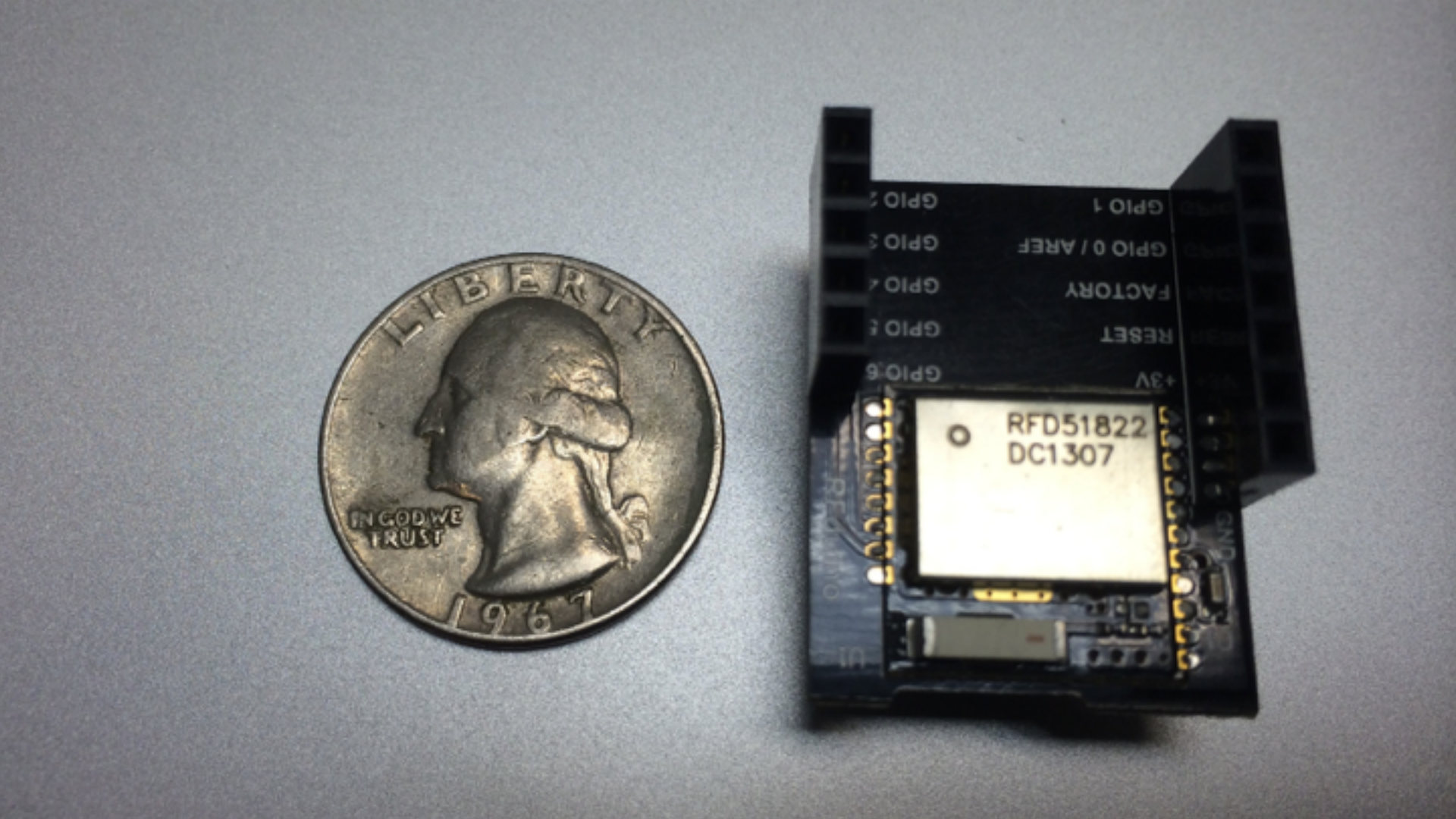

Panasonic
CR 2032
3V
JAPAN


Panasonic
CR 2032
3V
Made in Indonesia

RFDuino!

nrf58122 made friendly!




LIBERTY
IN GOD WE TRUST
1967
GPIO 1
GPIO 0 / AREF
FACTORY
RESET
+3V
RFD51822
DC1307

What's great about RFDuino?

nrf51822 in Arduino ready form!

It's got a nice Arduino-y API for turning on BLE, sending/receiving data, all that stuff!

Has a ceramic antennae on it

Uses SoftDevice 120 (icyc)

What's not great about RFDuino?

Power consumption

Closed source bootloader (more on this later)

Much more expensive

Limited in some respects

Installation on your computer

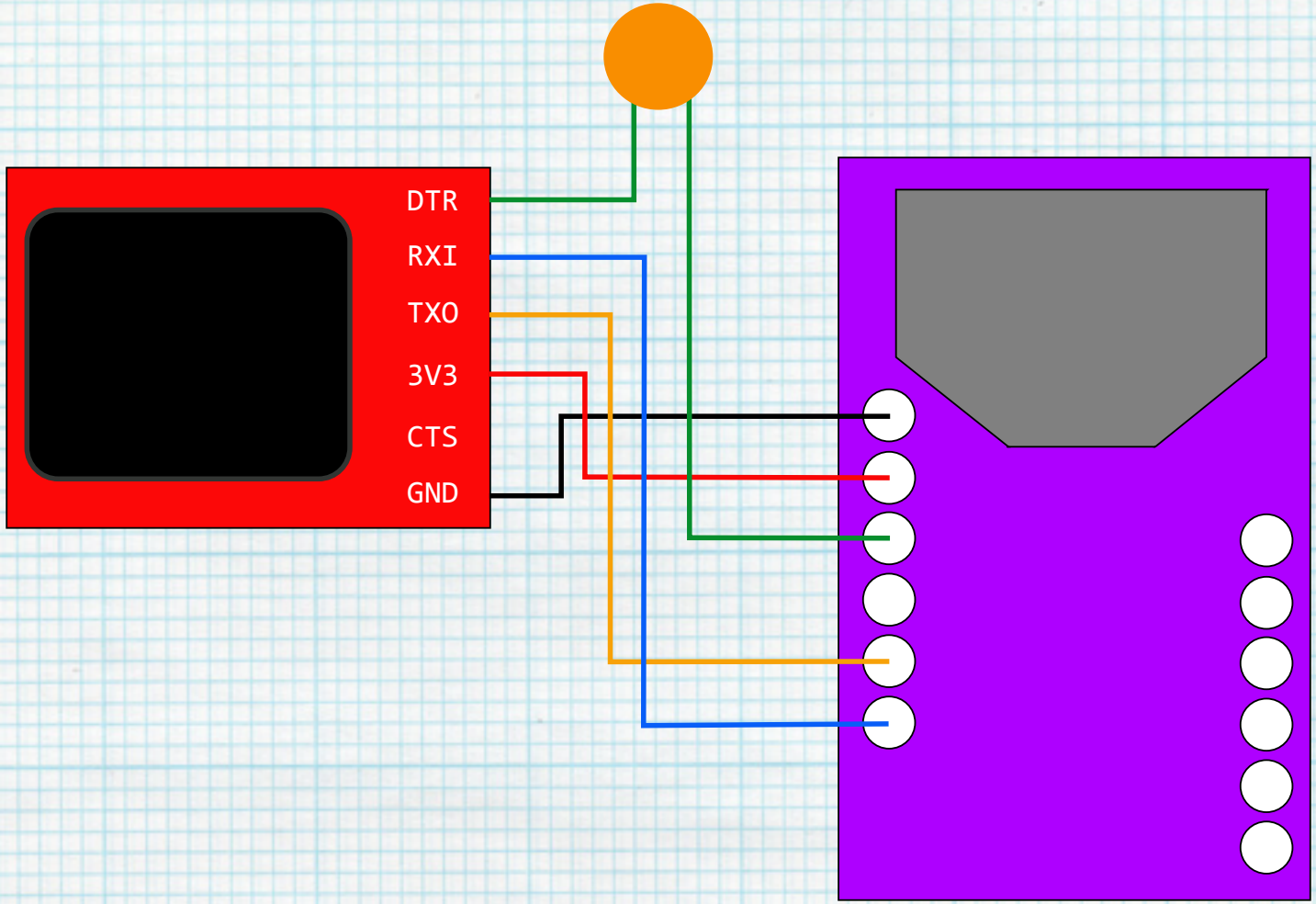
Arduino IDE version 1.5 beta (!)

RFDuino library

An RFDuino

A programmer

A 10 mF Capacitor


DTR
RXI
TXO
3V3
CTS
GND

What are you going to talk to?

Talk to an iOS device

caveats apply!

openFrameworks

C++

Creative Code

OSX, Windows, Linux, Android, iOS

Not for the faint of heart but also not really that scary

cinder

C++

Creative Code

OSX, Windows, iOS

Not for the faint of heart

Talk to an Android device

caveats apply!

processing

Java

Creative Code

OSX, Windows, Linux, Android, Browsers-ish

Not scary but can get messy

Talk to another RFDuino

no caveats apply!

GZLL (gazelle)

1 host connects to up to 8 slaves in a star

Host must be "always on" (i.e. draws lots of power).

Slaves are power-efficient.

Host always listens, slave initiates

GZLL (gazelle)

A host has to wait for a packet from a slave before it can send data to it.
One slave can talk to several hosts and devices can switch between host and slave, enabling more complicated networks.

Lets make stuff

this might get messy

What can go wrong?

Your computer isn't set up right

You don't have the right version of Arduino

You don't have the right version of Android

20+ RFDuinos in the same room

Something else...

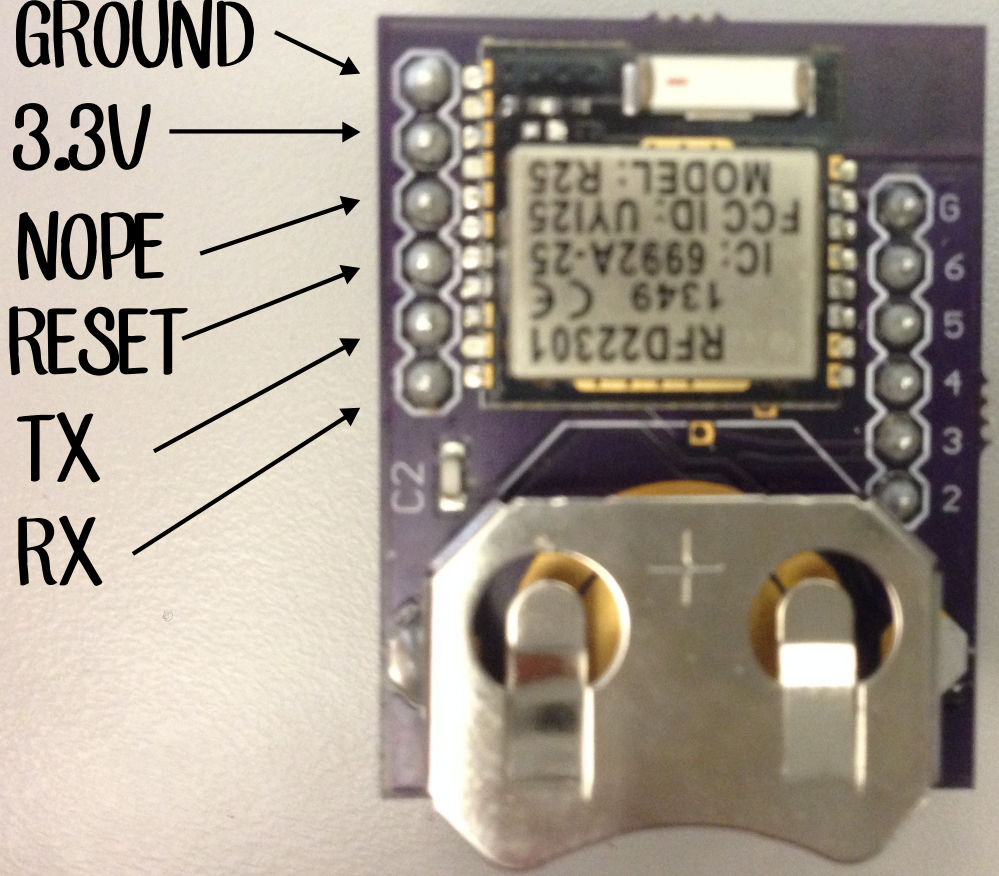
Are you ready?

Do you have an RFDuino board?

Do you have something to talk to it?

Do you have either Processing or Xcode set up properly?

Do you have Arduino ready?


GROUND
3.3V
NOPE
RESET
TX
RX
G
6
5
4
3
2
C2
RFD22301
1349
IC: 6992A-23
FCC ID: UV125
MODEL: R25

GROUND

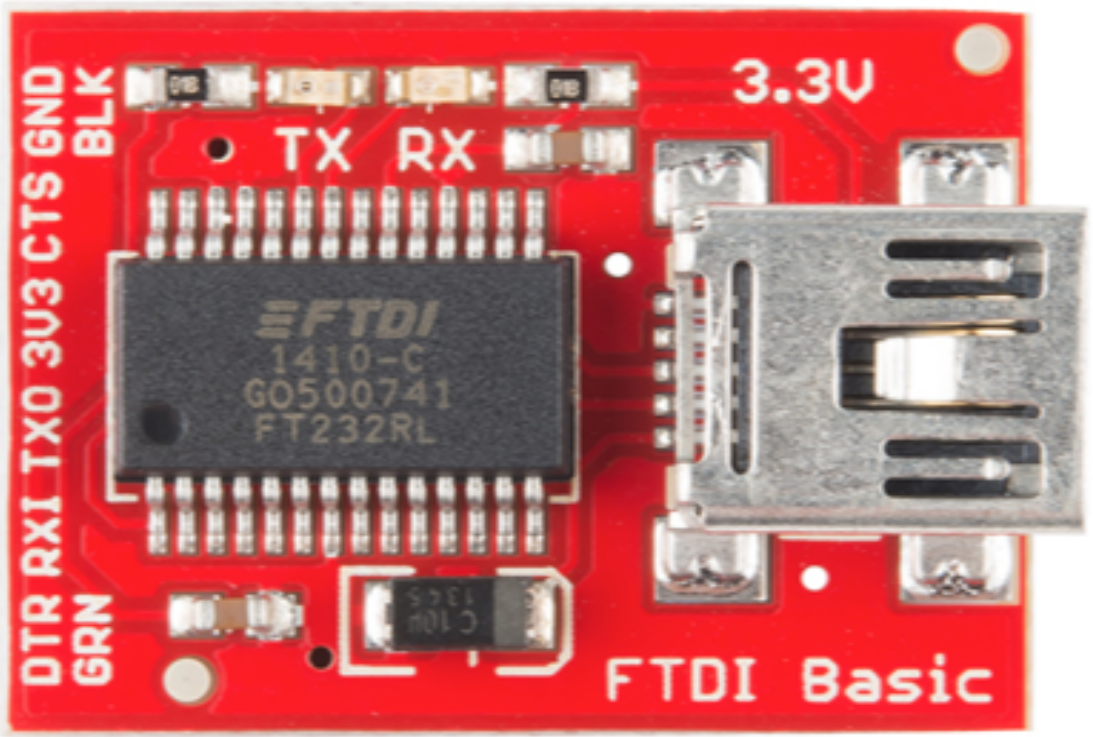
NOPE

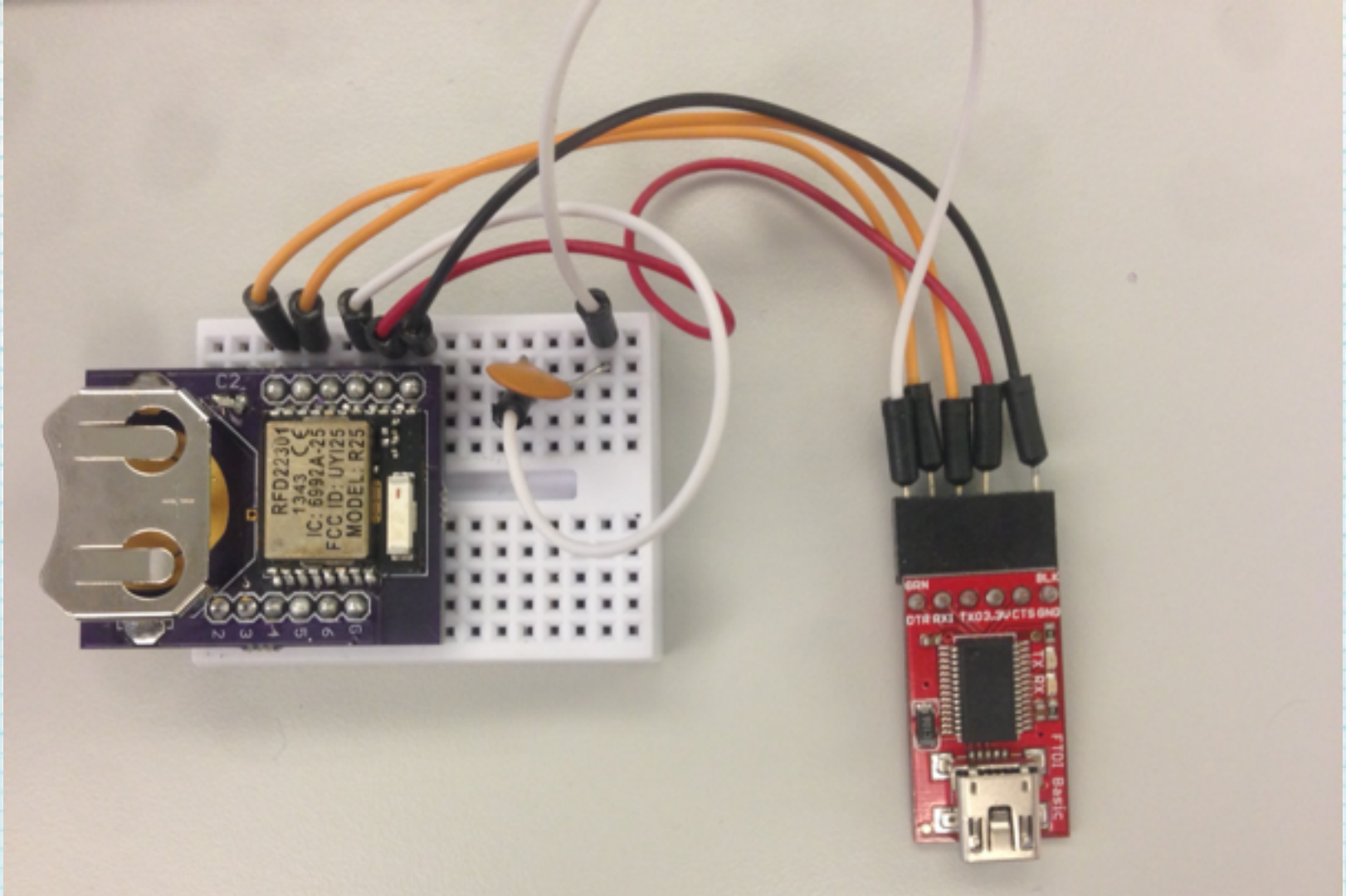
3.3V

TX

RX

RESET




RFD22301
1343
IC: 6992A-25
FCC ID: UY125
MODEL: R25
C2
GRN
BLK
DTR RXI TXO 3.3V CTS GND
TX
RX
FTDI Basic

Blinking an LED!

Let's not worry about talking just yet, let's just make sure everybody can program their RFDuino first.

Blinking an LED!

Look in Examples->Basics.

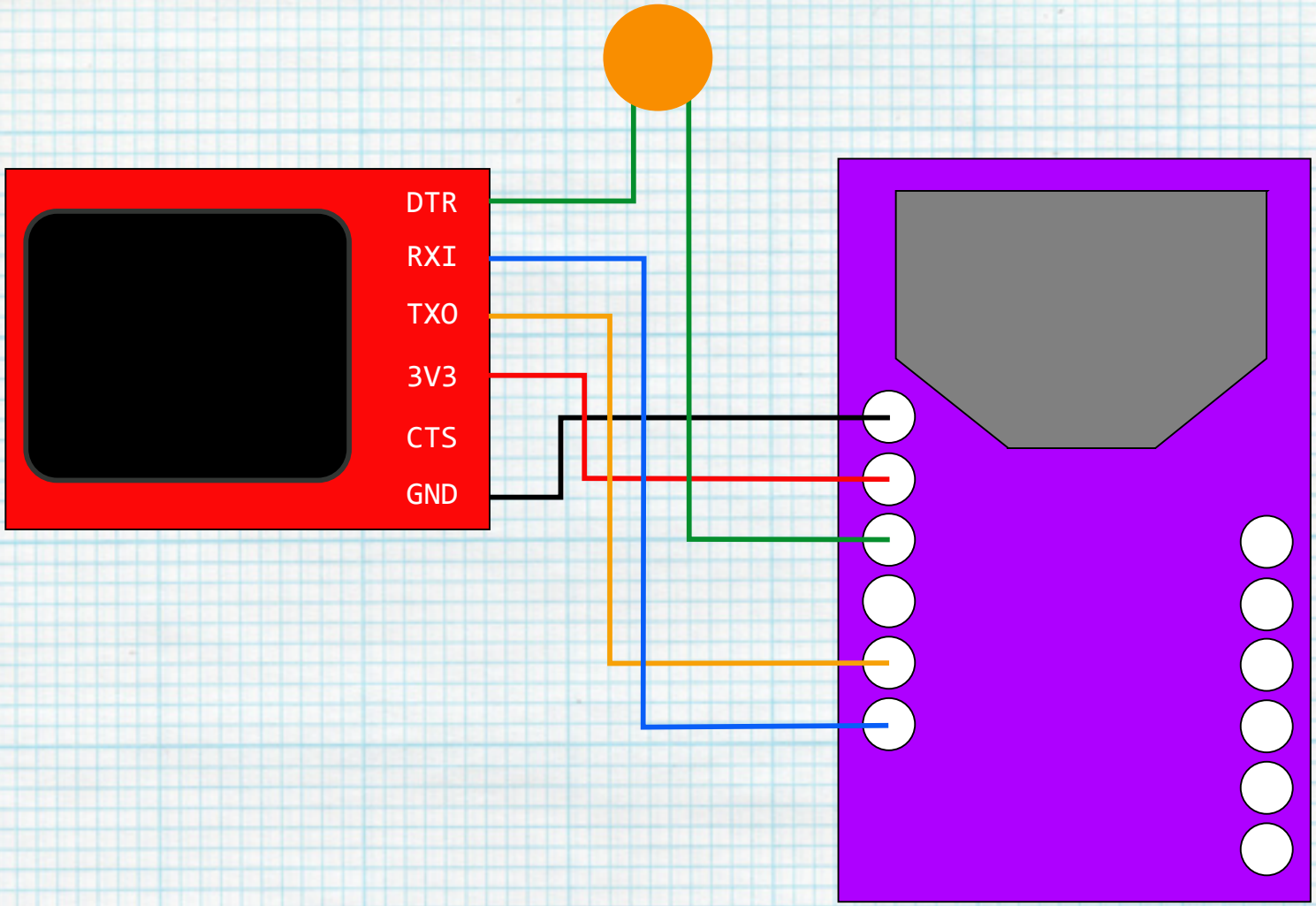
Change the pin to a pin you actually have.

Connect an LED to a pin.

Does it work?

Saying 'hi'

1. Start up the bluetooth stuff
2. Start up a service
3. Send some data over that service


DTR
RXI
TXO
3V3
CTS
GND

Saying ‘hi’

```
RFduinoBLE.send();
RFduinoBLE.send(1); // sends ‘1’
RFduinoBLE.send(“hi”); // sends
‘hi’
```

Hearing a 'hi'

(this excludes all the platform related stuff)

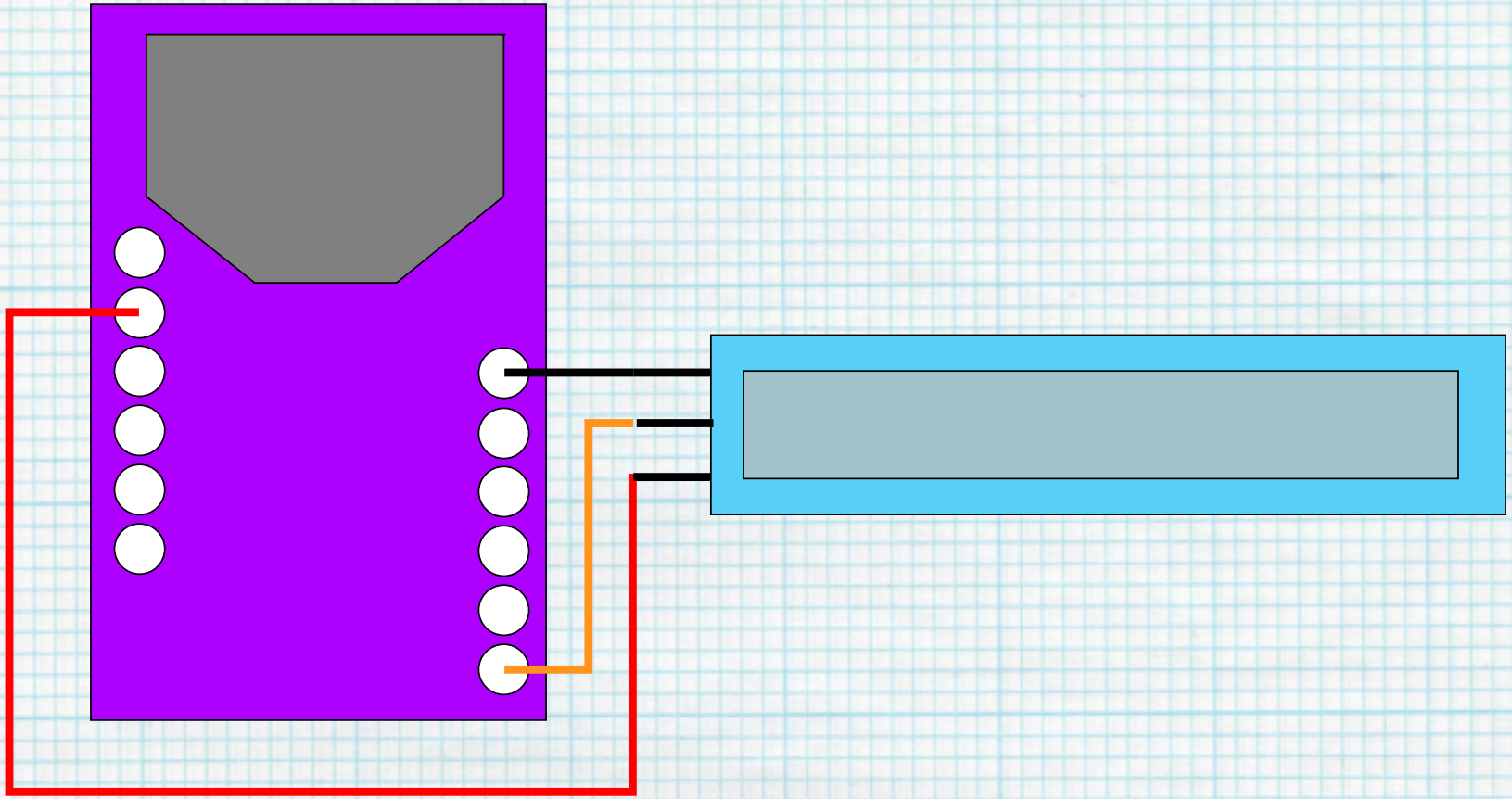
1. start up the bluetooth stuff
2. look for your device
3. Subscribe to a characteristic on a service
3. listen for changes on the characteristic

SO MANY DEVICES!

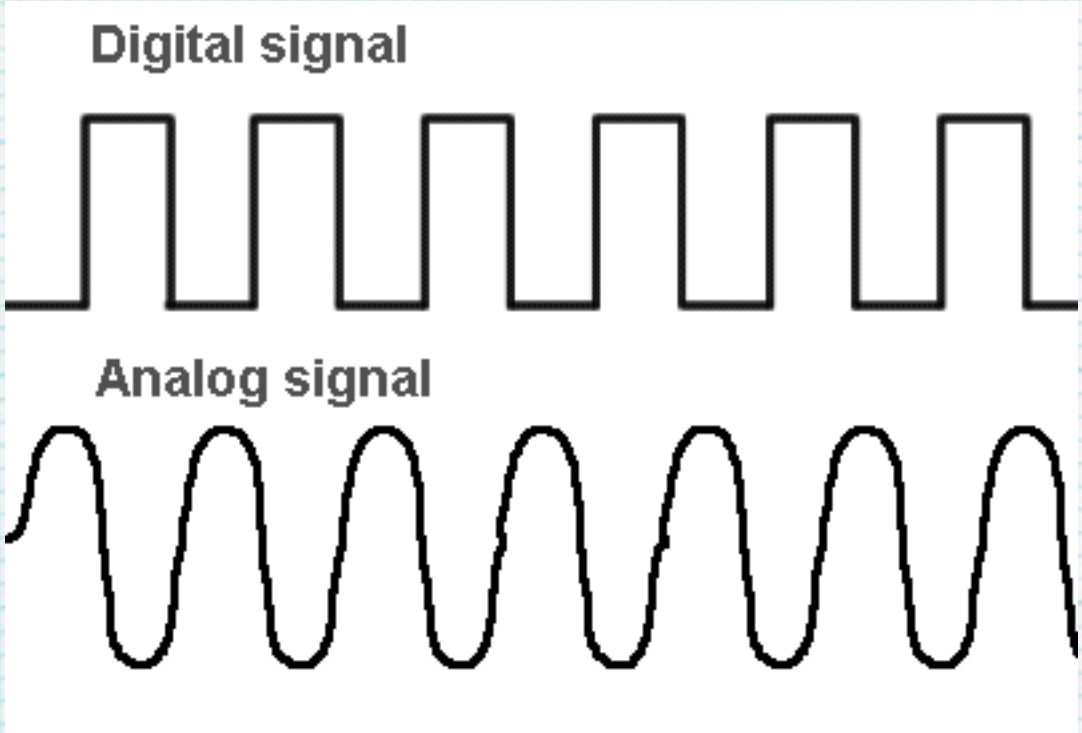
There's 25 of you, so make sure you give your device a name that your program can find. 25 devices called RFDuino isn't gonna work.

Reading a slider

1. Make a pin an input
2. Check for changes on the pin
3. Send some data out!



analog vs digital



0 or 1

0 to 1024

Hearing a slider

(this excludes all the platform related stuff)

1. start up the bluetooth stuff
2. look for your device
3. Subscribe to a characteristic on a service
3. listen for changes on the characteristic

Changing an LED (RFD)

1. Set a device name
2. Start up the BLE
3. Wait for RFduinoBLE_onReceive() to get triggered
4. Do something with the data you get

Changing an LED (App)

1. Look for your device name
2. Connect to it
3. Connect to the service
4. Send some data at some interval

Reading a potentiometer

First we need to read the potentiometer, so:

```
analogRead();
```

Then we need to send the value, so:

```
RFduinoBLE.sendInt();
```

SPI

What's SPI?

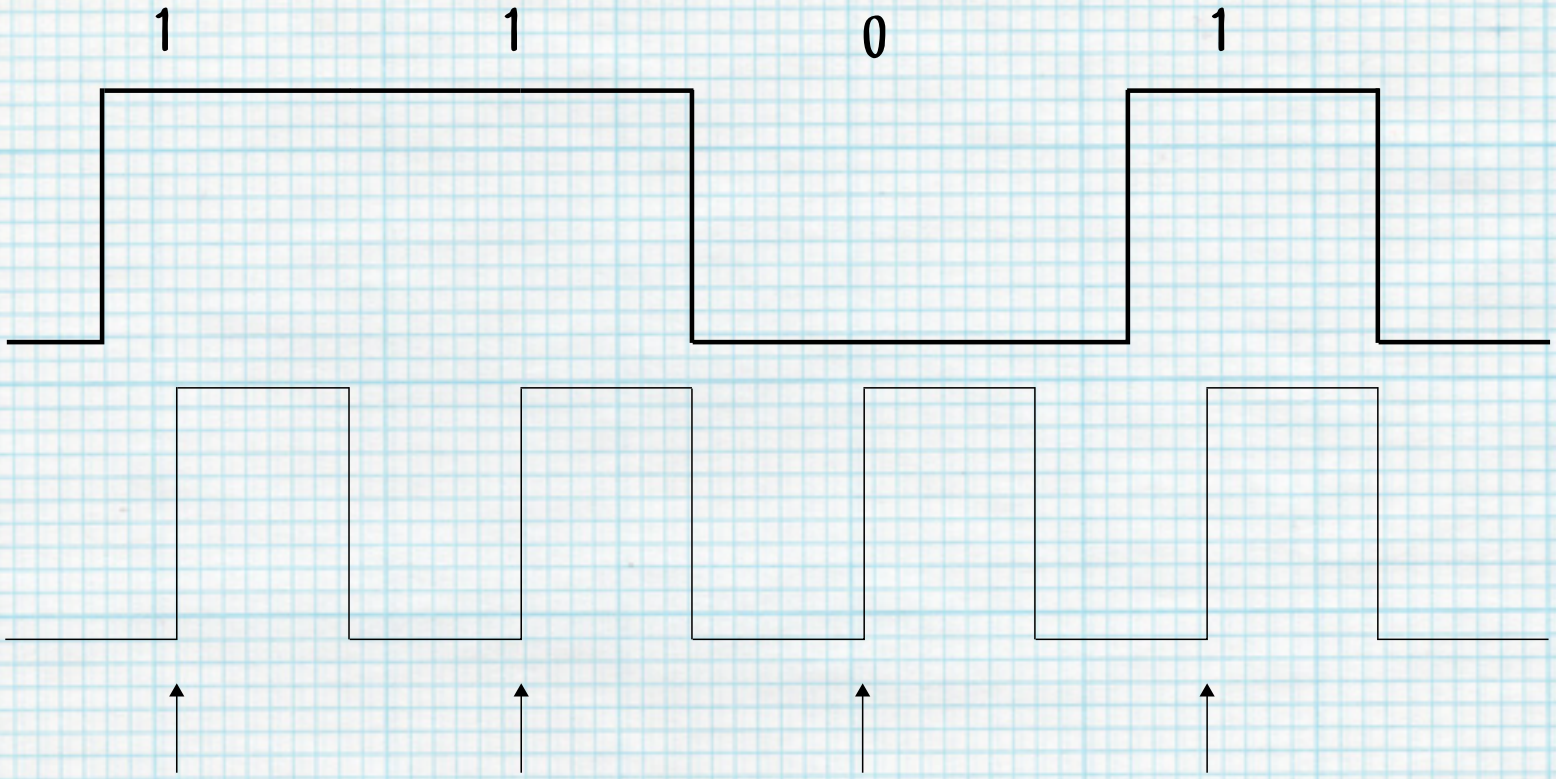
Three lines: one TX, one RX, one clock

Clock says: send a bit

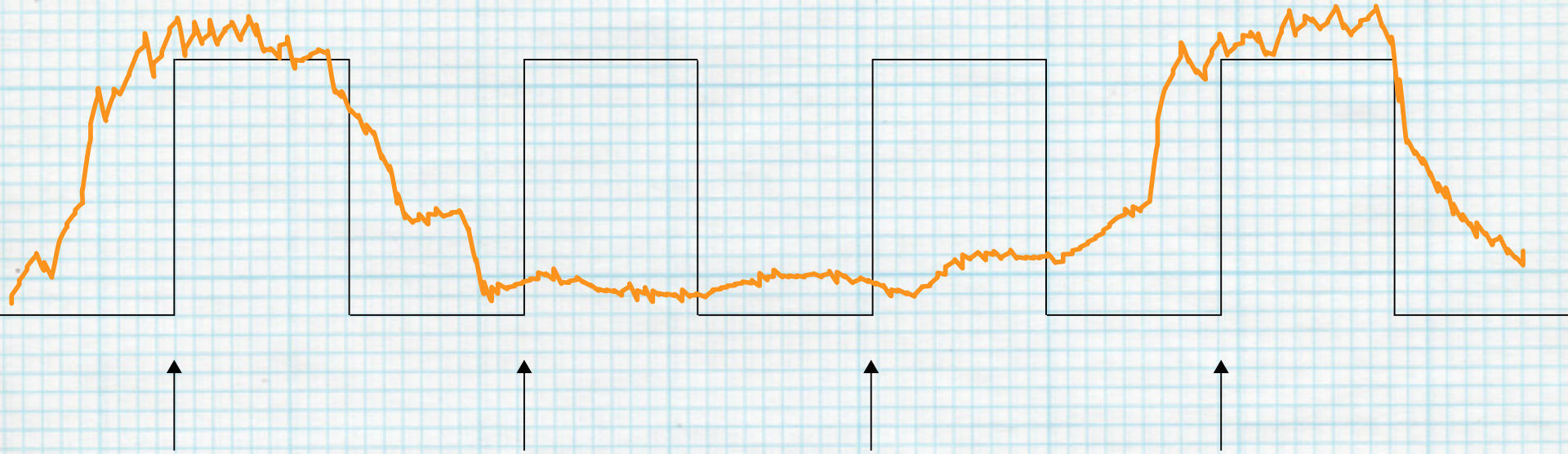
TX says: send with this one

RX says: listen with this one

SPI



SPI



Reading an accelerometer

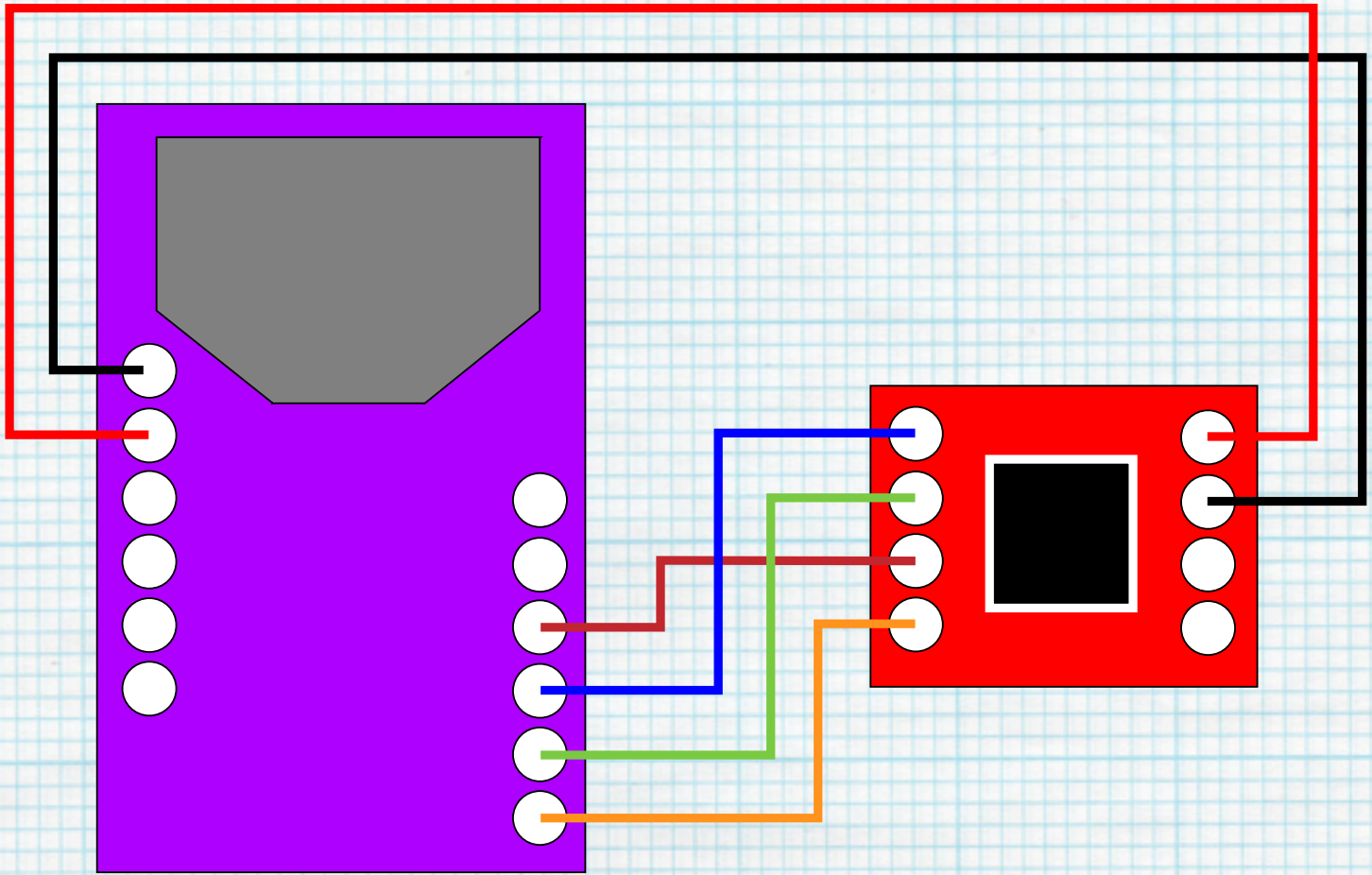
Make an SPI

Start the service

Get the accelerometer data

Send it to the listening device

Take a break



Hearing an accelerometer

Connect to device

Start the service + characteristic listening

Get some data at an odd interval

Do something with it when you get it

Reading multi-byte

The gnarliest of method calls:
`memcpy()`

take some memory and put it in another place and convert to something else in that other place

Types types types!

"1" isn't 1

uint16_t = 16 bits of int

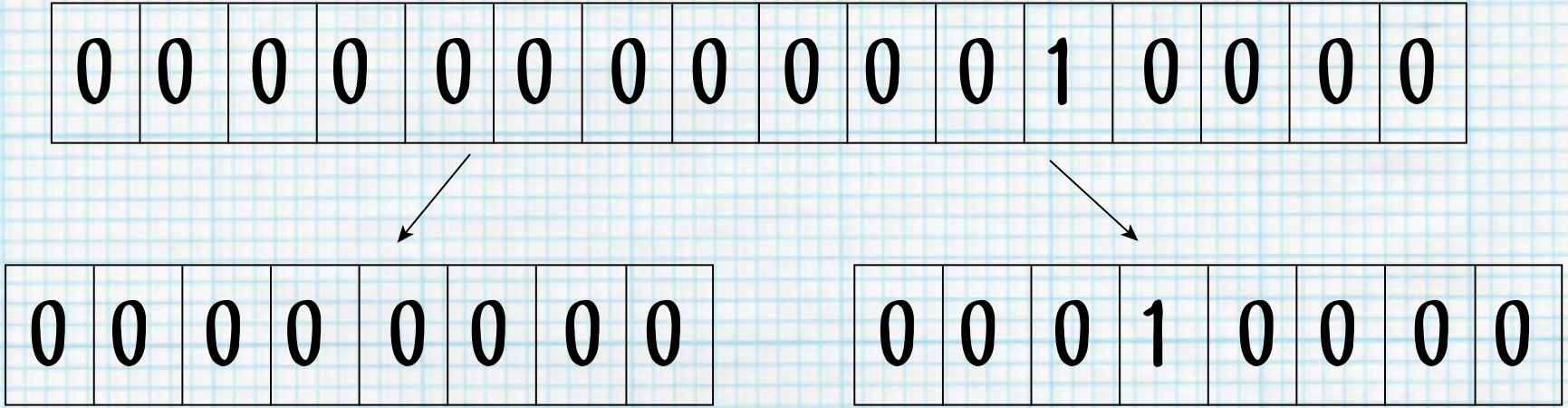
char = 8 bits of character

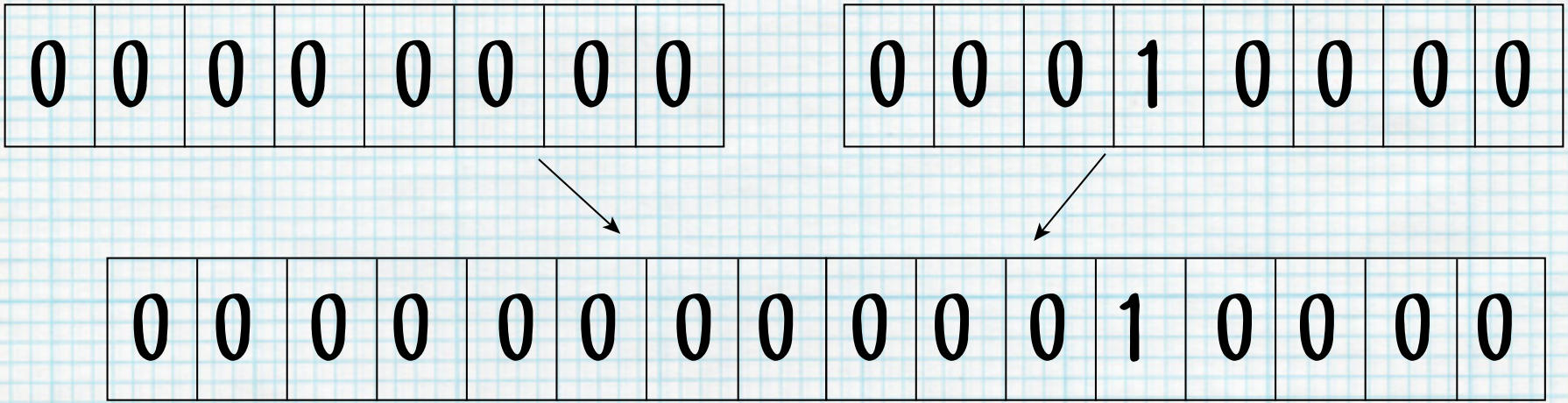
Sending multi-byte

```
int16_t XValue = 12;
char buffer[sizeof(int16_t) * 3];
memcpy(&buffer[0],
       &XValue,
       sizeof(int16_t));
```

Reading multi-byte

```
const char *data;
int16_t x;
size_t sz = sizeof(int16_t)
memcpy(&x, &data[0], sz);
```


0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0


0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0

Sending other data

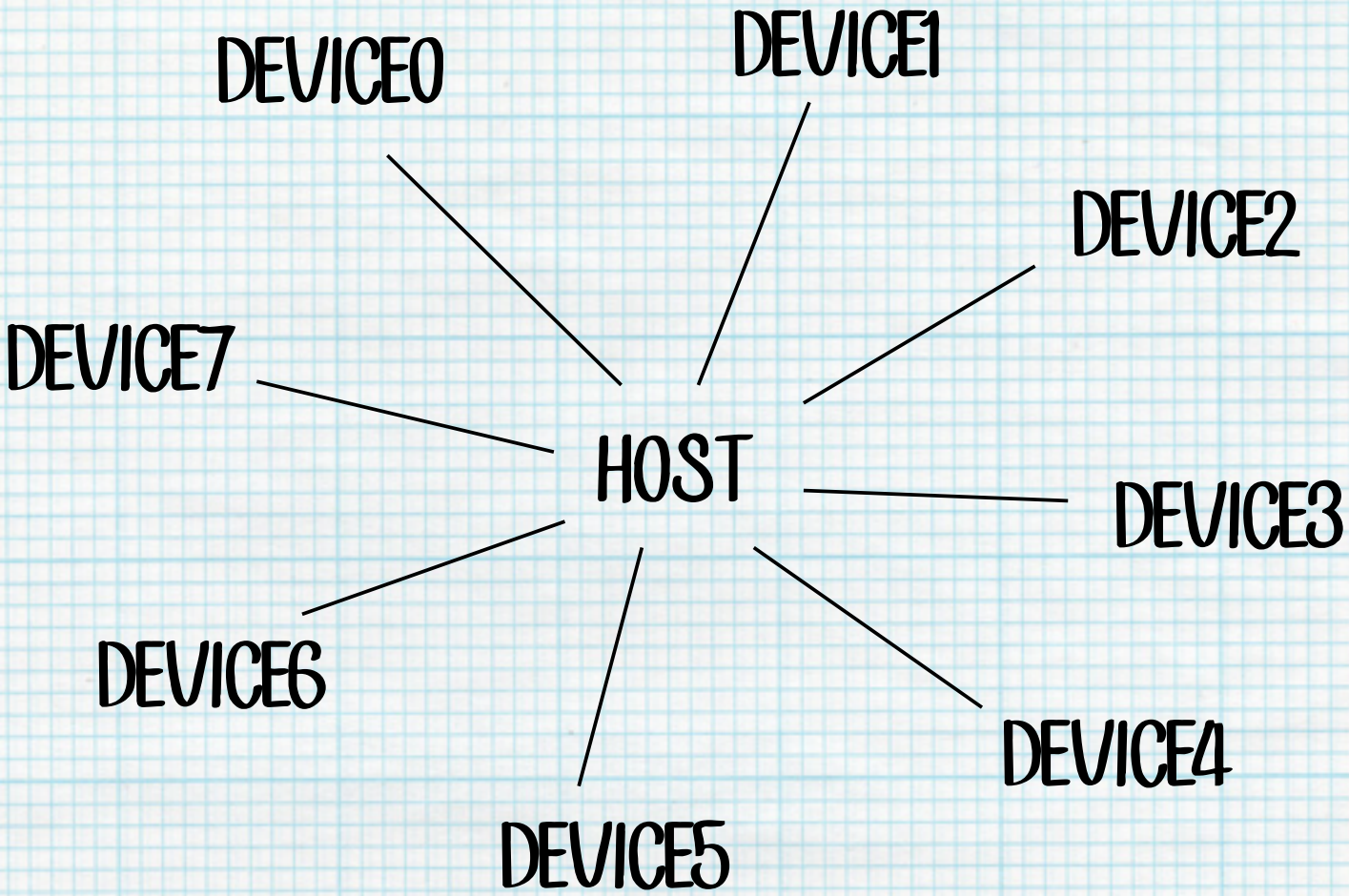
You can send all kinds of stuff: float, int, byte

Keep what you send simple

Don't send things sequentially, send them in arrays

GZLL

hot rfduino on rfduino action


DEVICE0
DEVICE1
DEVICE2
DEVICE7
HOST
DEVICE3
DEVICE6
DEVICE4
DEVICE5

Starting it up

What role is the GZLL?

HOST,DEVICE0,DEVICE1,DEVICE2...

Wait for some data (on either side)

Receiving

```
RFduinoGZLL_onReceive(
        device_t device,
        int rssi,
        char *data,
        int len );
```

Hubs

Get some data from a spoke

```
RFduinoGZLL_onReceive()
```

Send some data to a spoke

```
RFduinoGZLL.sendToHost()
```

Spokes

Get some data from a hub

```
RFduinoGZLL_onReceive()
```

Send some data to a hub

```
RFduinoGZLL.sendToDevice(device,
"OK");
```

Can we try it out?

Sure. We have to hack the library but then we can say what device we want to talk to.

```
RFduinoGZLL_host_base_address =
0x0D0A0704;
```

Can we try it out?

Sure. We have to hack the library but then we can say what device we want to talk to.

Both devices and hosts set their ID.

Giving out names

What's the device name?

RFduinoGZLL_device_base_address

What's the host name?

RFduinoGZLL_host_base_address

What's next?

Oh man what to do now?

Play with GZLL!

Send stuff to other BLE devices

Create wearable device systems

Make tiny little mesh networks

Play w/non-RFDuino

The nrf51822 is pretty cool & you can play with it outside of RFDuino.

There's lots of other BLE devices: nRF8001, ble112, ble113, the list goes on

Play w/non-Bluetooth

nrf2401 is pretty rad: 315, 443, and 915mhz

Cheap little 315mhz radios are awesome

I work on a series of libraries for Arduino + radio

Great resources on the Arduino forum

Other ways things talk

RFID { meh }

NFC { kinda awesome }

wifi { pain in the butt }

IR { awesome }

Sound { awesome but hard }

Make some things

Check out the RFDuino forums

Check out the Arduino forums

Talk to each other

Email me, I'll halp