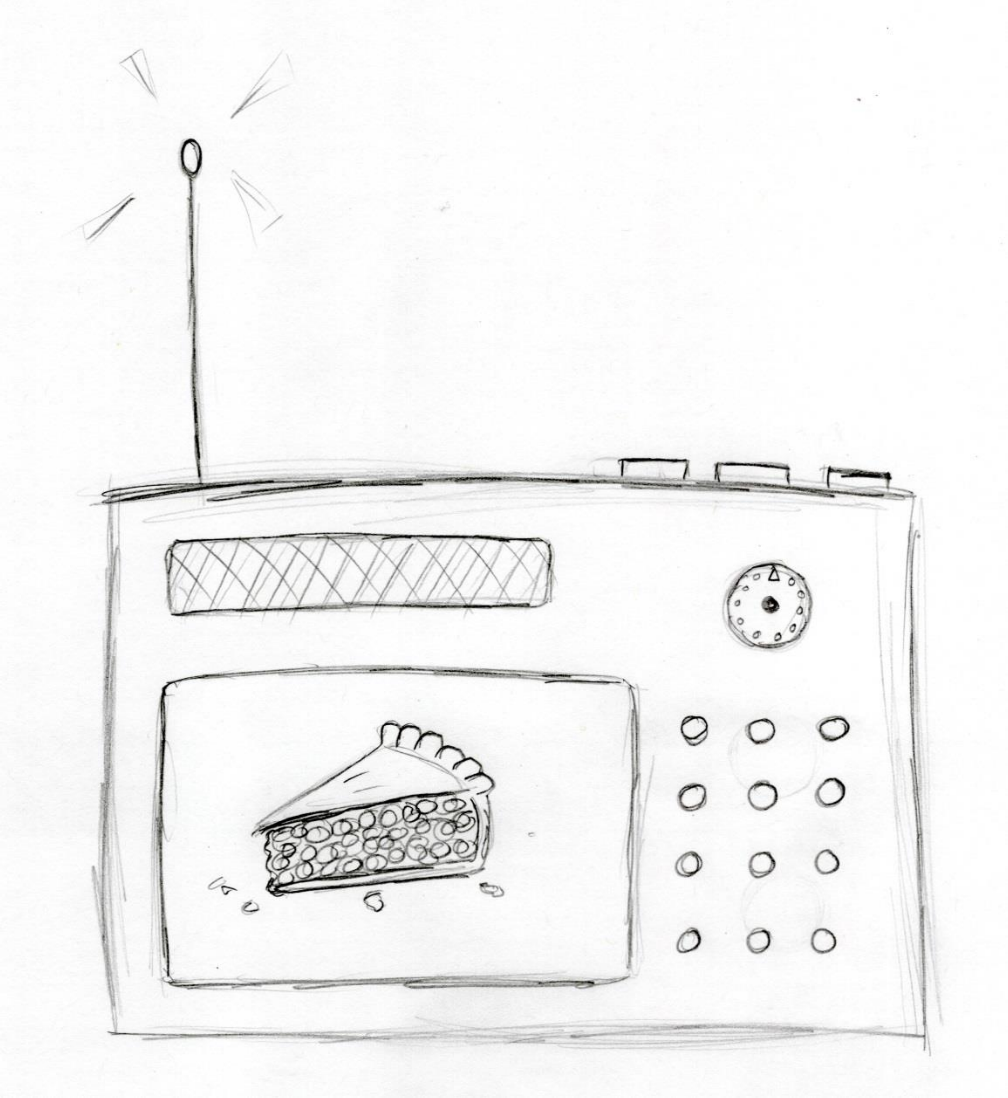
RASPBERRY PI
IN
AMATEUR RADIO

Matthew Miller

KK4NDE

http://kk4nde.com



WHAT IS RASPBERRY PI

Raspberry Pi	Conventional PC
Runs Linux operating system	Runs Linux, Windows, MacOS, etc
ARM Processor	Intel x86 or x86-64 Processor
Single SD card for storage	One or more hard-drives for storage
Fixed RAM for working memory	RAM modules for working memory
USB ports for peripherals	USB ports for peripherals
Network interface (depends on model)	Network cards
Expandable with “HATs”	Expandable with PCI cards
Configurable Input/Output connections	Audio card

WHAT IS RASPBERRY PI

Raspberry Pi

- Runs Linux operating system
- ARM Processor
- Single SD card for storage
- Fixed RAM for working memory
- USB ports for peripherals
- Network interface (depends on model)
- Expandable with “HATs”
- Configurable Input/Output connections

Arduino

- Single-purpose programming
- Programmable Microcontroller
- Integrated EEPROM storage
- Integrated RAM for working memory
- Expandable with “Shields”
- Configurable Input/Output connections

OVERVIEW OF VERSIONS

	Model B	Model A+	Pi 2	Pi 3	Pi Zero	Pi Zero W
Cost	Outdated	$20	$35	$35	$5	$10
USB	2x USB-A	1x USB-A	4x USB-A	4x USB-A	1x OTG	1x OTG
Network	10/100M		10/100M	10/100M		
WiFi				2.4GHz		2.4GHz
Bluetooth				Yes		Yes
RAM	512MB	256MB	1GB	1GB	512MB	512MB
CPU	1 core 700MHz	1 core 700MHz	4 cores 900MHz	4 cores 1.2GHz	1 core 1GHz	1 core 1GHz
GPIO	Male Pins	Male Pins	Male Pins	Male Pins	Thru-Hole	Thru-Hole

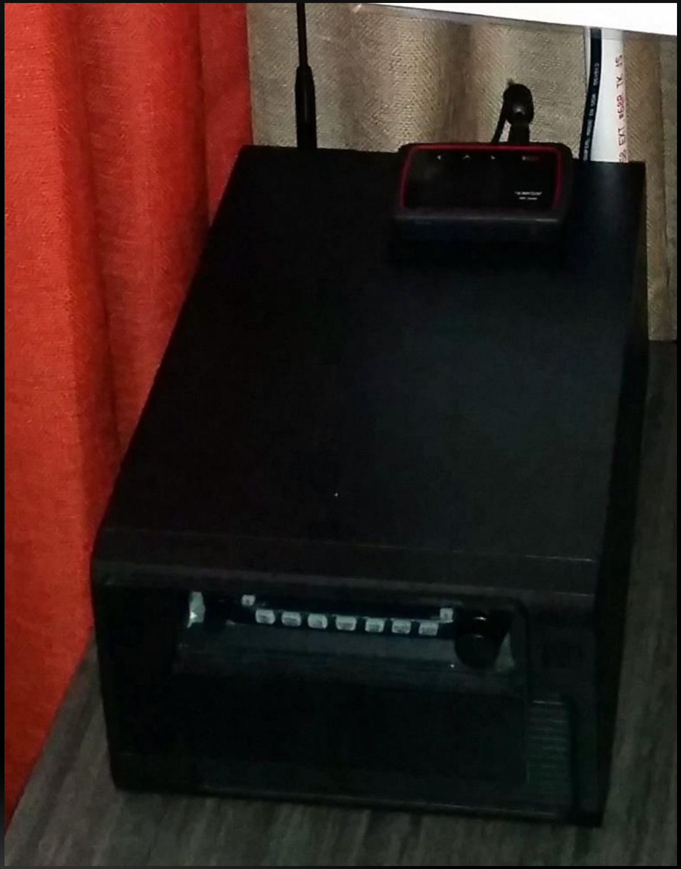
EXAMPLE PROJECTS

- General purpose computer
    - Mini Desktop/TV Computer
    - Mini Server
    - Streaming box
    - …ok, so what? This is an Amateur Radio group.

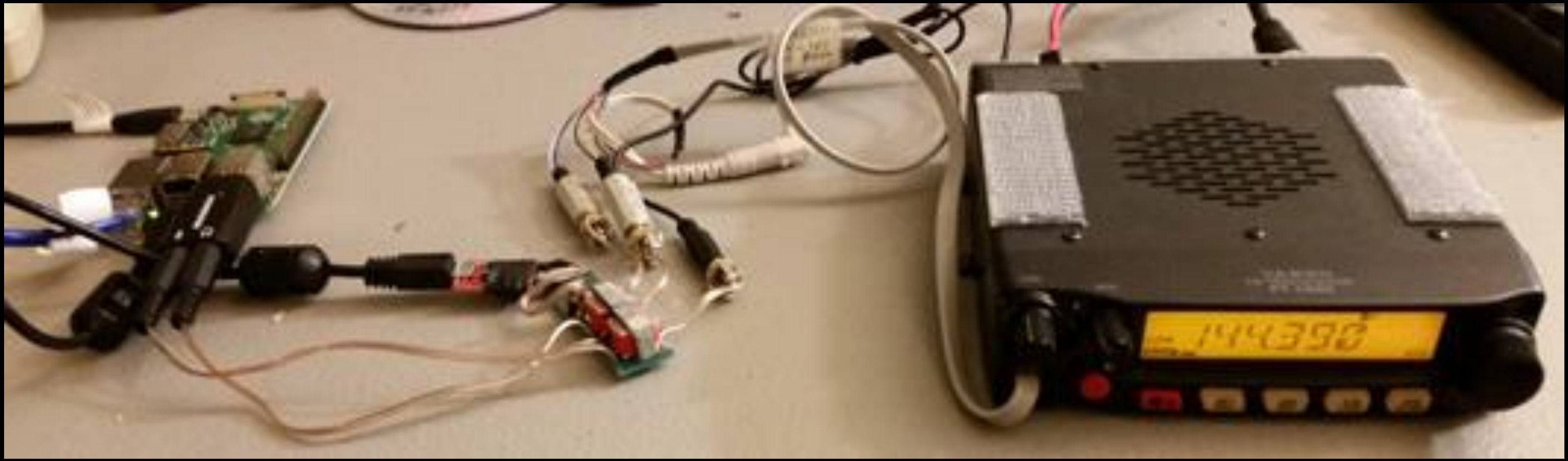
EXAMPLE PROJECTS

- Connect to radio
    - Remote rig operation
    - Repeater controller
    - Echolink controller
    - APRS digipeater
    - APRS i-gate
    - …and more!
- Use SDR instead of radio
    - Mostly plug and play but RX-only
    - Maybe some full TX-RX capable SDR could work, sounds expensive

EXAMPLE PROJECTS - ECHOLINK V1 & V2



EXAMPLE PROJECTS - DIGIPEATER



INTERFACING A SDR

- USB – just plug it in
- May require drivers

INTERFACING A RADIO

- Connections similar to TNC
    - Audio input/output
    - PTT and/or COS signals
- Solutions
    - Commercial all-in-one (e.g. Signalink)
    - Homebrew interface

INTERFACING A RADIO

- Commercial all-in-one solution
  - Boring
  - Expensive
  - May need to find drivers

INTERFACING A RADIO

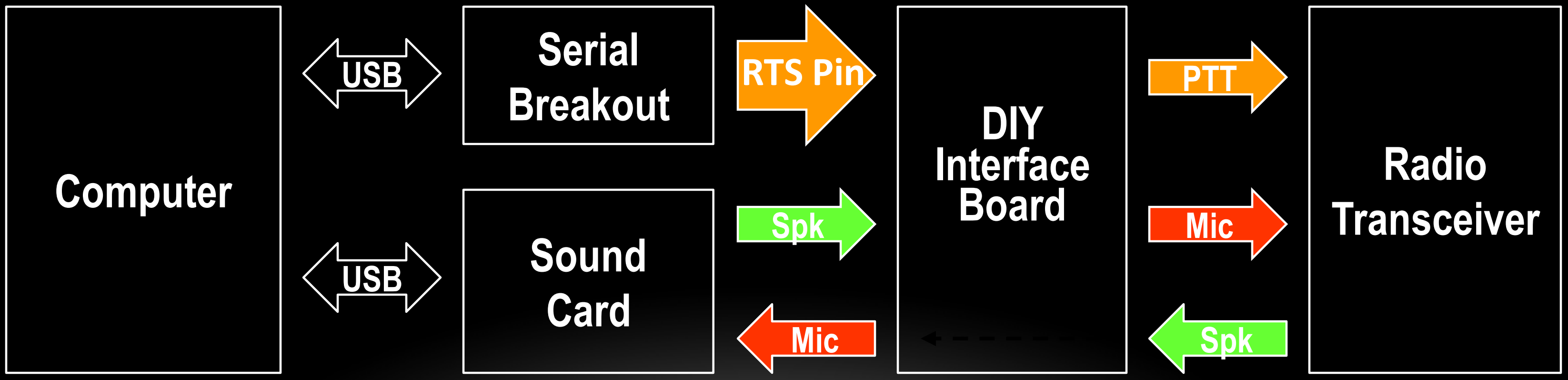
- Audio input/output
  - No built in mic/line-input
  - Recommend USB sound-card
  - Should have some sort of simple isolation between soundcard and radio
    - 1:1 audio transformer
    - Capacitors
    - Resistors

INTERFACING A RADIO

- PTT/COS signals
  - GPIO-pins directly controlling
    - Depends on application support
  - Built-in serial on GPIO – configure CTS/RTS capability
    - Annoying OS configuration settings to worry about
    - Problem, at bootup or crash settings reset, may default to key up radio
  - External serial adapter with CTS/RTS capability
    - Can be hard to find serial adapters with CTS/RTS (note RTS ≠ DTR)
    - Usually chip defaults to proper “idle”, default to un-key radio

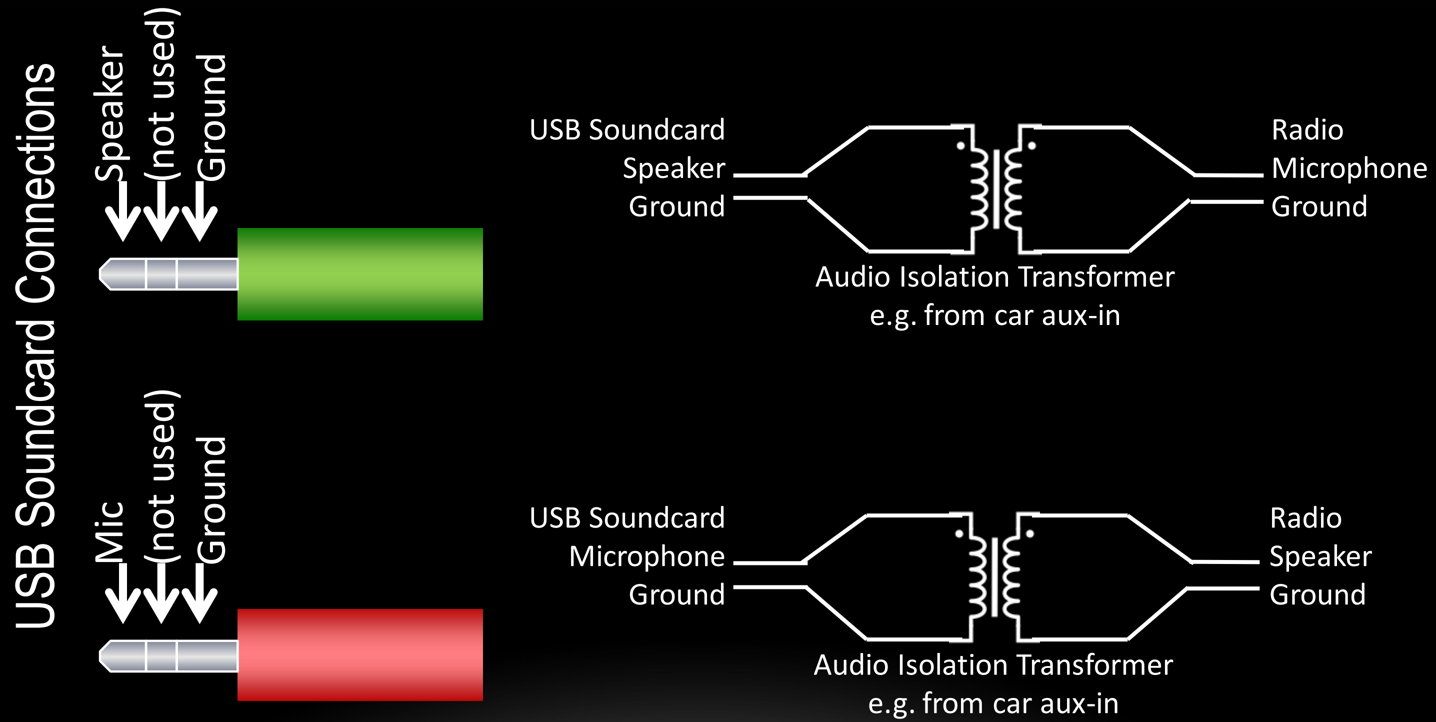
INTERFACING A RADIO

- For flexibility, I am trying to make this modular and compatible with most software
- USB soundcard + DIY isolation (1:1 transformers)
- USB serial with CTS/RTS to opto-isolator for PTT
- Have not played with COS return on serial breakout yet, the radio I have with output does not provide sufficient power to switch an opto-isolator (maybe a transistor would work)



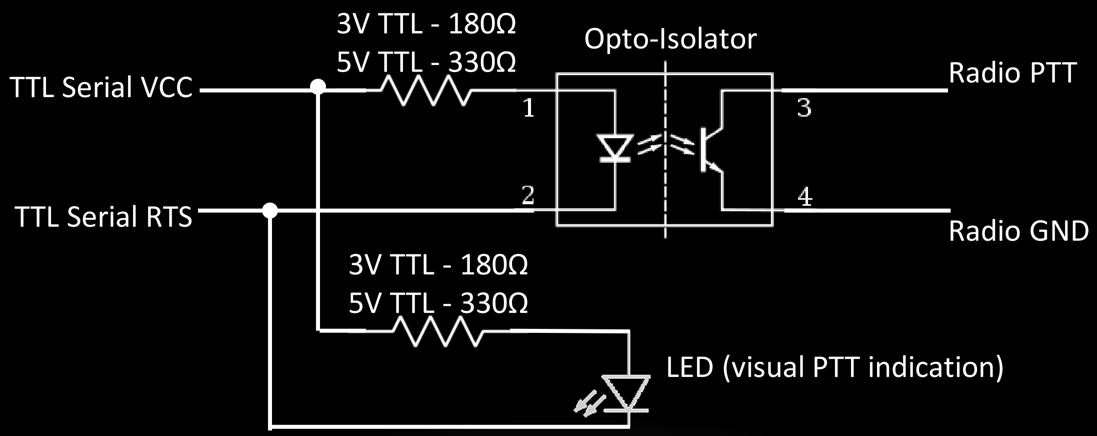
INTERFACING A RADIO

- USB soundcard + DIY isolation (1:1 transformers)



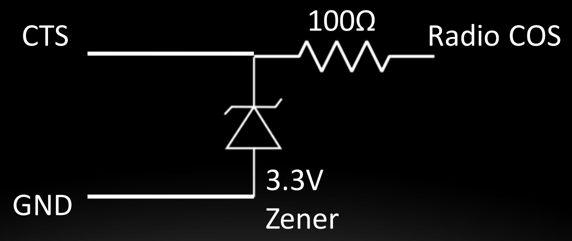
INTERFACING A RADIO

- USB serial with CTS/RTS to opto-isolator for PTT
- Opto-isolator makes it easy to "invert" the serial chip which goes to 0V when it's "triggered" and also has to be protected from higher voltages (my radio uses +8V for PTT pin and the serial chip can only stand up to 3.3V)



INTERFACING A RADIO

- Have not played with COS return on serial breakout yet, my radio's output won't power an opto-isolator
- I think I would use either a transistor or resistor w/ 3.3V or 5V zener-limiter depending if it needs inverting and what voltage the serial adapter uses.
- This is an example of the interface I used for 3.3V GPIO connections, It should work the same way for the serial CTS connection if the radio is +V idle and 0V on squelch open.
- Important thing is to limit the maximum voltage going into the serial chip to no more than its VCC (typically 3.3V or 5V depending on the chip) – this is easy to with a zener diode.



HARDWARE

- USB serial adapter
  - Tested: SparkFun Serial Basic Breakout - CH340G (needs chip-soldering to get RTS)
    https://www.sparkfun.com/products/14050
  - Not Tested: SparkFun USB to Serial Breakout - FT232RL (all pins on thru-hole pads)
    https://www.sparkfun.com/products/12731
- USB soundcard
  - Tested: SYBA external USB Stereo Sound Adapter with Microphone
    (uses C-Media chipset)
    https://www.amazon.com/gp/product/B001MSS6CS/

SOFTWARE

- Raspbian Linux (Debian based for Raspberry Pi on ARM)
  - USB serial adapter is typically “/dev/ttyUSB0”
  - USB soundcard is typically "plughw:1,0”
- APRS Software
  - APRX and Soundmodem (radio + soundcard/PTT)
  - pymultimonaprs (USB RTL-SDR)
- Repeater & Echolink Software
  - svxlink – highly configurable repeater, simplex-repeater, echolink, etc. controller
  - OpenRepeater project (haven’t used this myself but it’s another repeater controller)
- Digital modes
  - FlDigi (haven’t tried this myself but it sounds promising)