The Internet of Musical Things

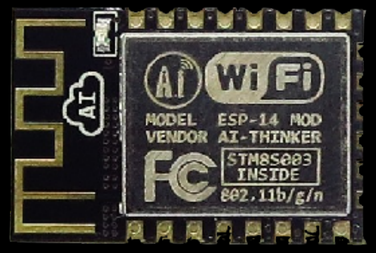
Exploring Wireless Connectivity in Musical Networks

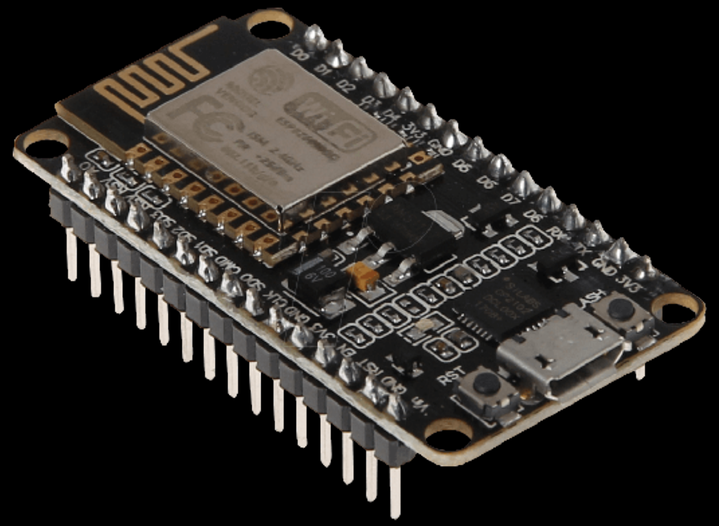
Jeff Gregorio
jgregorio@drexel.edu

METlab

DREXEL UNIVERSITY
ExCITe Center
Expressive and Creative Interaction Technologies

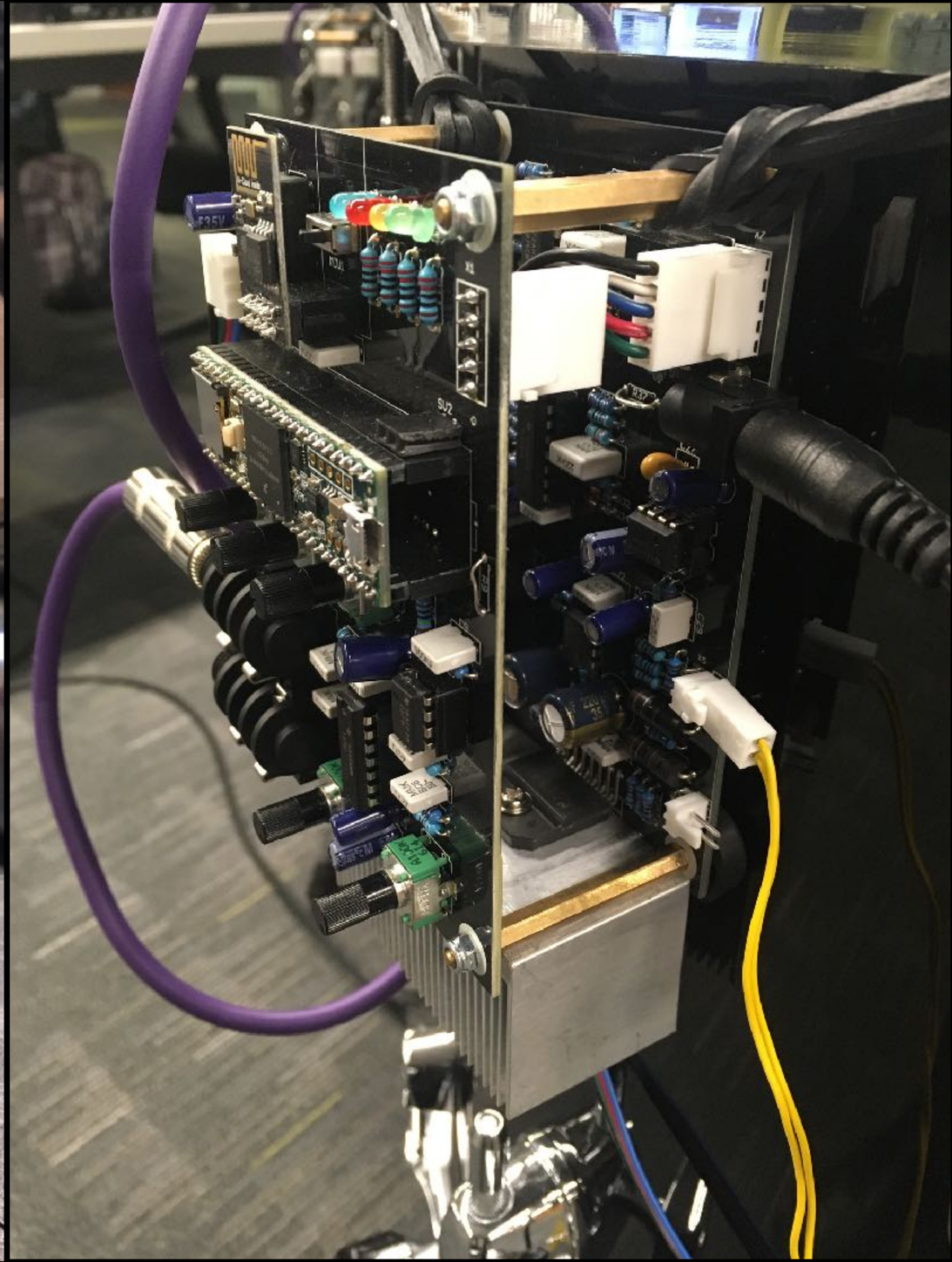
The Internet of Things





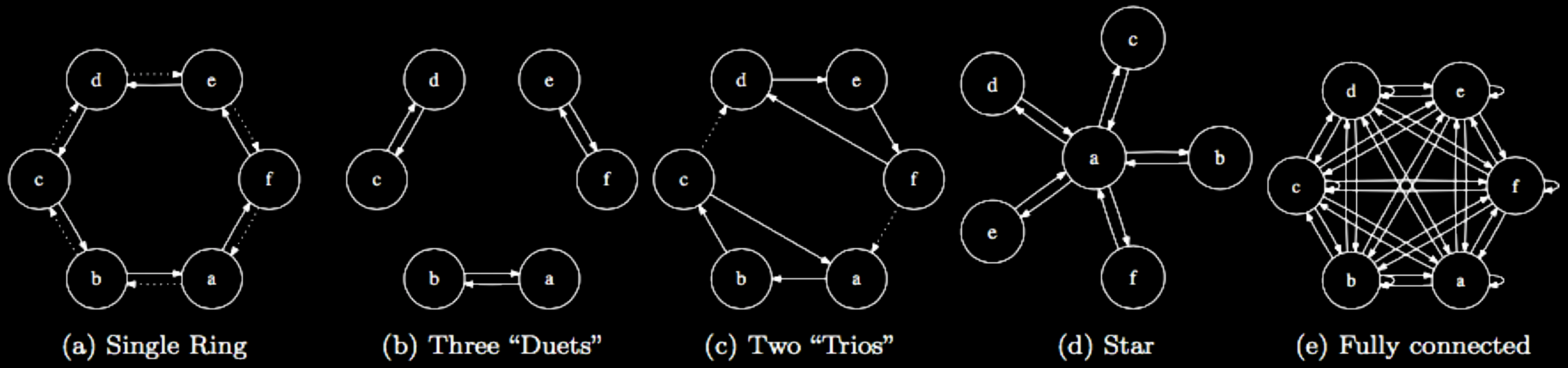
A Henge of Drums


Pearl




REMO

Inspiration: Ensemble Feedback



Muhammad Haz Wan Rosli, Karl Yerkes, Matthew Wright, Timothy Wood, Hannah Wolfe, Charlie Roberts, Anis Haron, and Fernando Rincon Estrada. 2015. Ensemble Feedback Instruments. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Edgar Berdahl and Jesse Allison (Eds.). Louisiana State University, Baton Rouge, Louisiana, USA, 144–149.

Networking Protocols

Transmission Control Protocol (TCP):

- “Greetings 192.168.1.27:11245, I would like to make a connection”
- “Acknowledged, 192.168.1.4:24545. Yes, let’s connect”
- “I am streaming you data now”
- “I accept your data”
- “Are we still connected?”
- “Yes, we are still connected”
- “I am sending you more data”
- “I accept your data”
- “Are we still connected?
- …

Networking Protocols

User Datagram Protocol (UDP):

- “Hey 192.168.1.27:11245, here’s some data. Think fast!”
- “Hey 192.168.1.27:11245, here’s some data. Think fast!”
- “Hey 192.168.1.27:11245, here’s some data. Think fast!”
- “Hey 192.168.1.27:11245, here’s some data. Think fast!”
- “Oh, I have new data from 192.168.1.4:24545”
- “Oh, I have new data from 192.168.1.4:24545”

UDP Broadcast:

- “Hey 192.168.1.(whatever):11245, here’s some data”

Networking Protocols

TCP:

- Reliable delivery
- Higher latency

UDP:

- Probable delivery
- Lower latency

Networking Protocols

Open Sound Control (OSC):

- Can be built on top of TCP or UDP
- UDP is more common and useful for musical instruments
- Format: /<path> [arg1] [arg2] … [argN]

Message examples:

/dco/waveform 2
/dco/waveform “tri”
/lfo/rate 3.229
/adsr/attack 300.0
/adsr/gate true
/settings “sawtooth” 3.229 300.0 true

Arduino and Microcontrollers

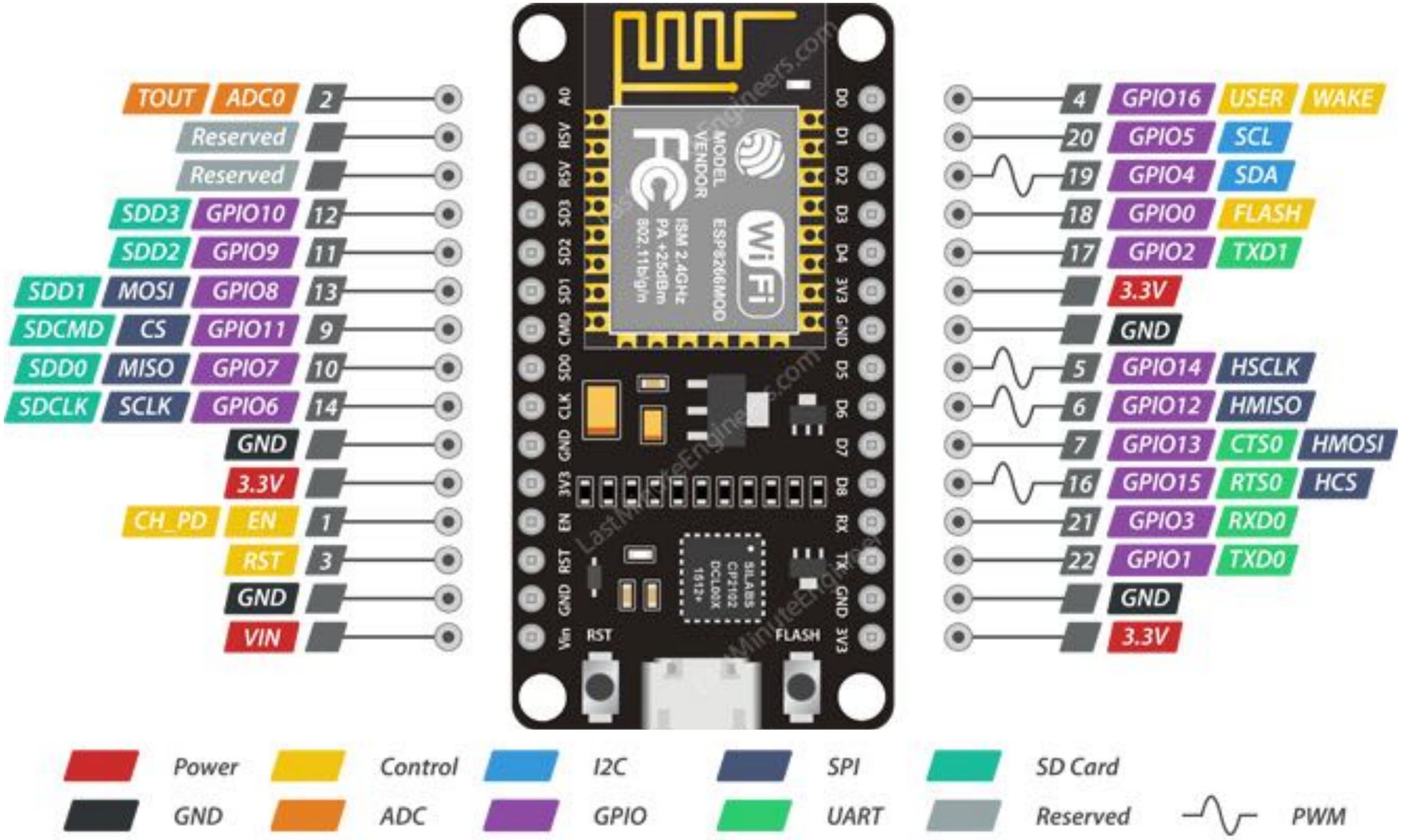
Microcontroller:

- A compact, embedded computer
- Interfaces with sensors and other computers by reading and writing digital/analog signals to specific pins
- Typically has no operating system
- Not great for multi-tasking

Arduino:

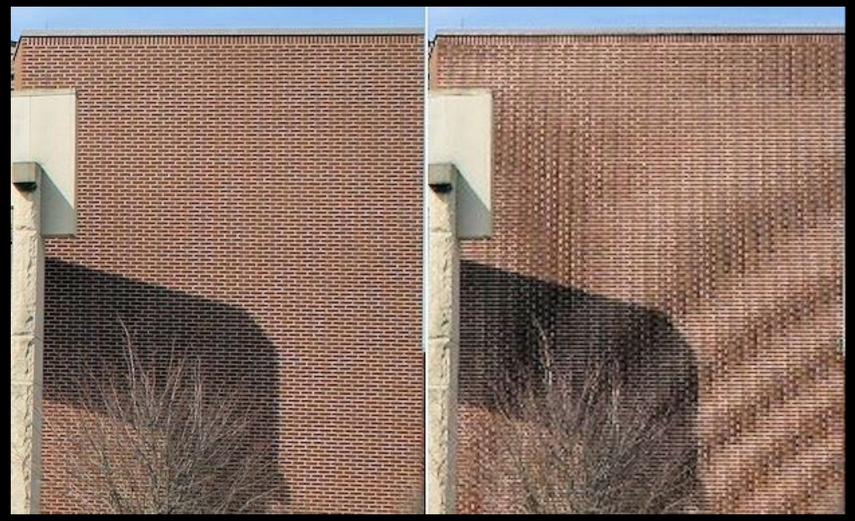
- An Italian company that manufactures open source hardware and provides an integrated development environment (the Arduino IDE)

NodeMCU


TOUT ADC0 2
Reserved
Reserved
SDD3 GPIO10 12
SDD2 GPIO9 11
SDD1 MOSI GPIO8 13
SDCMD CS GPIO11 9
SDD0 MISO GPIO7 10
SDCLK SCLK GPIO6 14
GND
3.3V
CH_PD EN 1
RST 3
GND
VIN
A0 RSV RSV SD3 SD2 SD1 CMD SD0 CLK GND 3V3 EN RST GND Vin
D0 D1 D2 D3 D4 3V3 GND D5 D6 D7 D8 RX TX GND 3V3
MODEL
VENDOR
ESP8266MOD
ISM 2.4GHz
PA +25dBm
802.11b/g/n
Wi Fi
SILABS
CP2102
RST
FLASH
4 GPIO16 USER WAKE
20 GPIO5 SCL
19 GPIO4 SDA
18 GPIO0 FLASH
17 GPIO2 TXD1
3.3V
GND
5 GPIO14 HSCLK
6 GPIO12 HMISO
7 GPIO13 CTS0 HMOSI
16 GPIO15 RTS0 HCS
21 GPIO3 RXD0
22 GPIO1 TXD0
GND
3.3V
Power
Control
I2C
SPI
SD Card
GND
ADC
GPIO
UART
Reserved
PWM

Digital to Analog Conversion

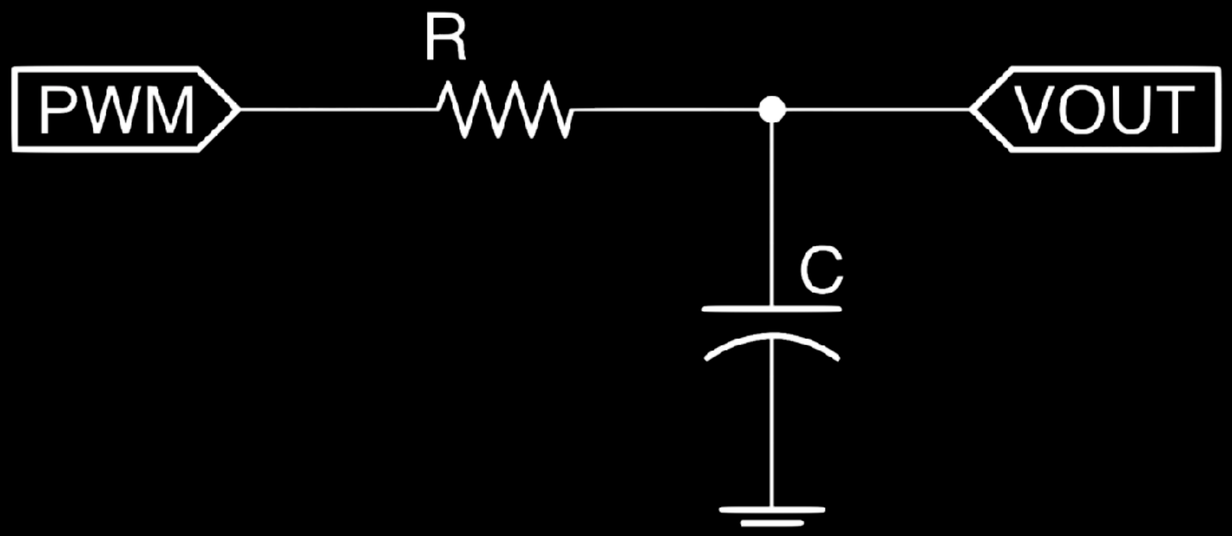
Aliasing:



Digital to Analog Conversion

Digital to Analog Conversion

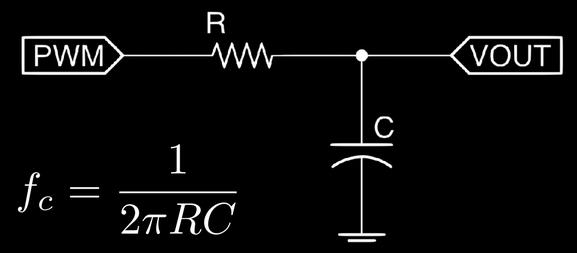
Reconstruction Filter (First Order Low-Pass):



$$f_c = \frac{1}{2\pi RC}$$

Attenuation	Frequency
-3dB	fc
-9dB	2fc
-15dB	4fc
-21dB	8fc
-27dB	16fc
-33dB	32fc
-39dB	64fc

Digital to Analog Conversion



$$f_c = \frac{1}{2\pi RC}$$

Design for sampling rate of 16kHz and -33dB attenuation at 8kHz:

$$f_c = \frac{8000Hz}{32} = 250Hz$$ **Acceptable?**

Pick common R or C value, calculate the other:

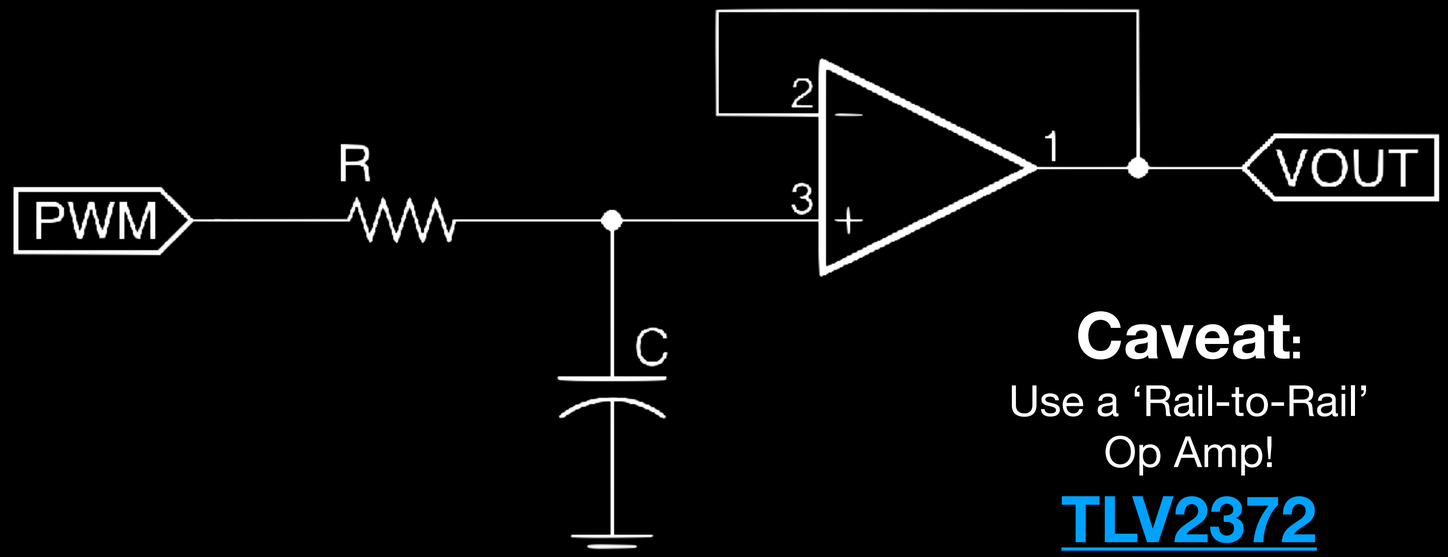
$$250Hz = \frac{1}{2\pi R \cdot 0.1\mu F}$$

$$R = 6336\Omega \approx 6.8k\Omega$$ **Close enough!**

Attenuation	Frequency
-3dB	fc
-9dB	2fc
-15dB	4fc
-21dB	8fc
-27dB	16fc
-33dB	32fc
-39dB	64fc

Digital to Analog Conversion

Reconstruction Filter (First Order Low-Pass, Buffered):



Caveat:

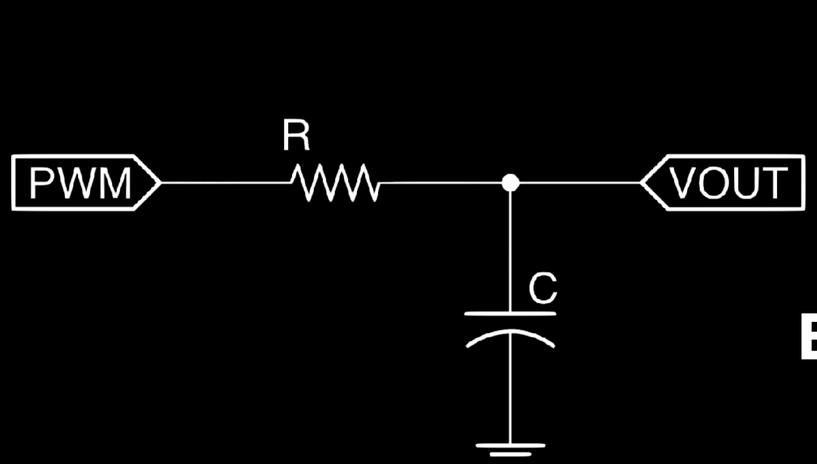
Use a ‘Rail-to-Rail’ Op Amp!

TLV2372

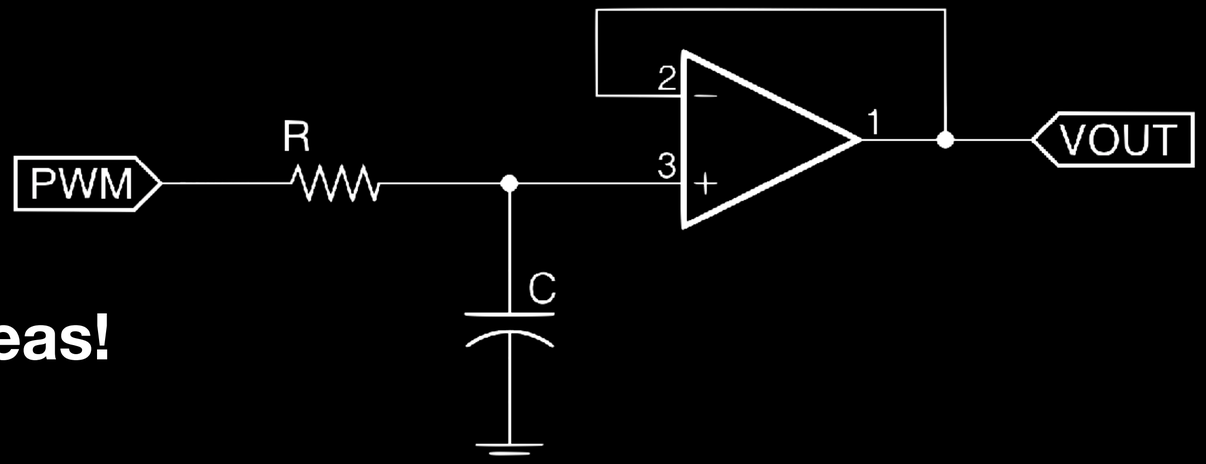
Digital to Analog Conversion

Takeaways:

- Don’t try to generate frequencies greater than half the sampling rate
- Play it safe; don’t go anywhere near half the sampling rate
- A wide gap between your highest frequency and this rate means you can use cheap and simple reconstruction filters



Easy Peas!



Digital to Analog Conversion

One last thing:

Sample rate ≠ PWM rate

- **Most Arduinos default to PWM rate ≈ 490 - 980Hz**
- **That's no good, since our reconstruction wouldn't filter enough of it out. Some Arduinos allow you to change the PWM rate so it's above the sample rate (https://www.arduino.cc/en/Tutorial/SecretsOfArduinoPWM)**
- **Our NodeMCU examples use the "sigma delta" library for PWM instead of the standard analogWrite, which allows us to use a PWM rate of 240,000Hz**

GO!

https://github.com/JeffGregorio/IoTMusicWorkshop