Power II

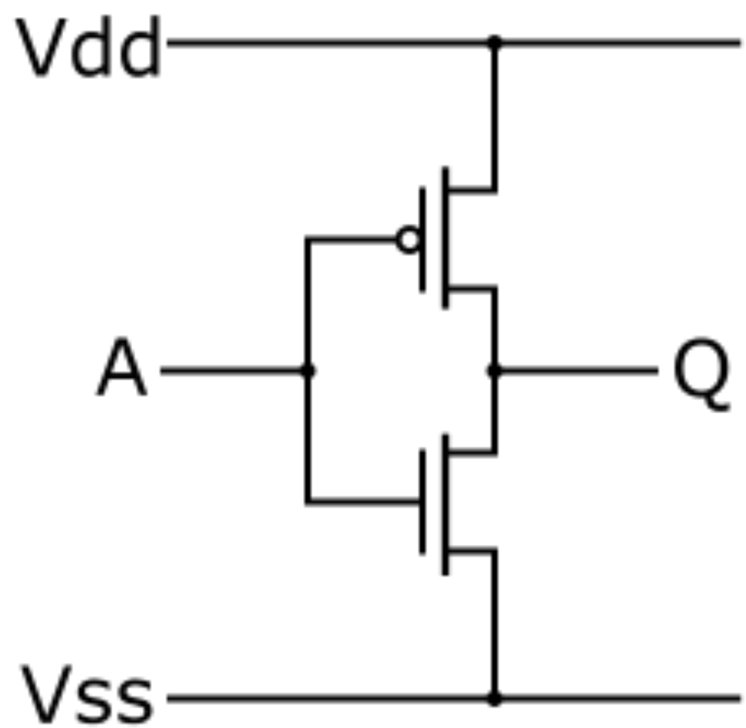
6.9000

Spring 2024

What actually uses the power in our Modern Digital Circuits?

All modern digital electronics mostly use CMOS architectures:
Complementary **M**etal **O**xide **S**emiconductor (Field Effect Transistors)

- Complementary nature means almost no current flowing (no power) at rest
- Power only* expended on the switch since we need to charge up any MOSFETs that we're driving with our output



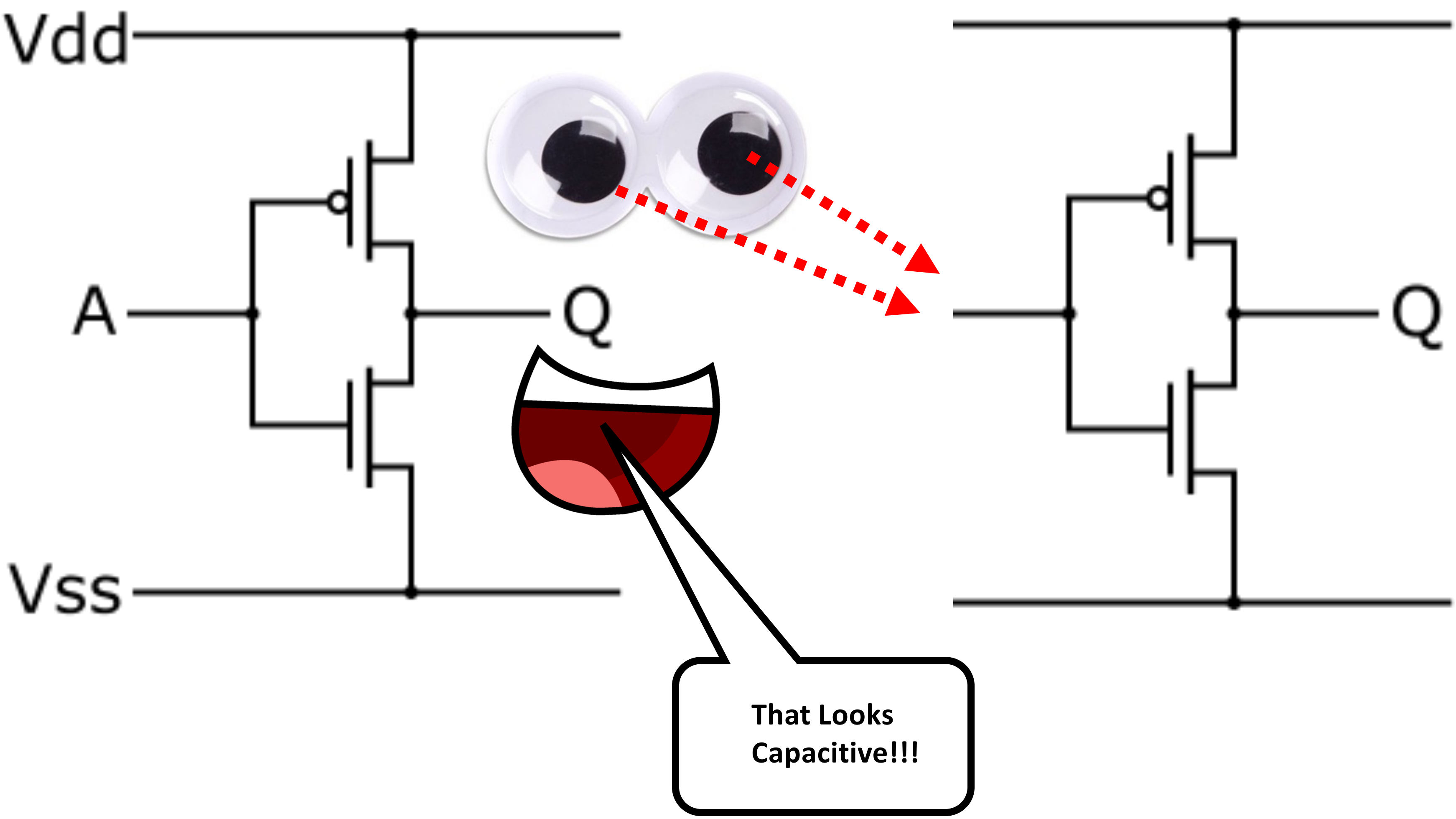
*ideally

CMOS drives other CMOS

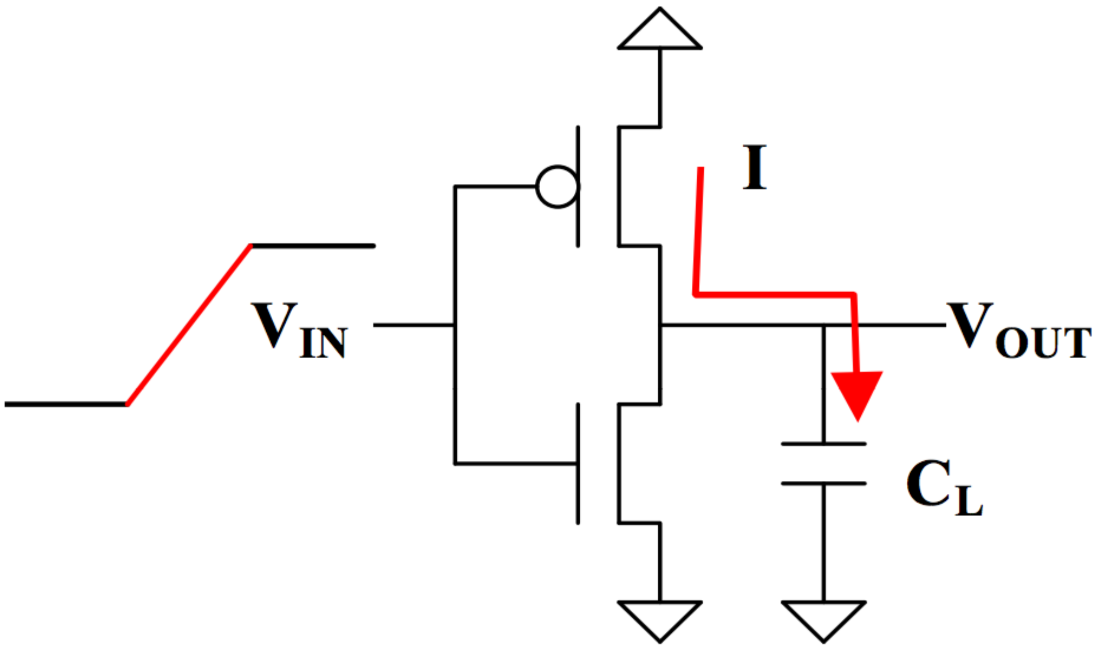
Metal
Oxide
Semiconductor

Forms capacitor

It is CMOS all the way down



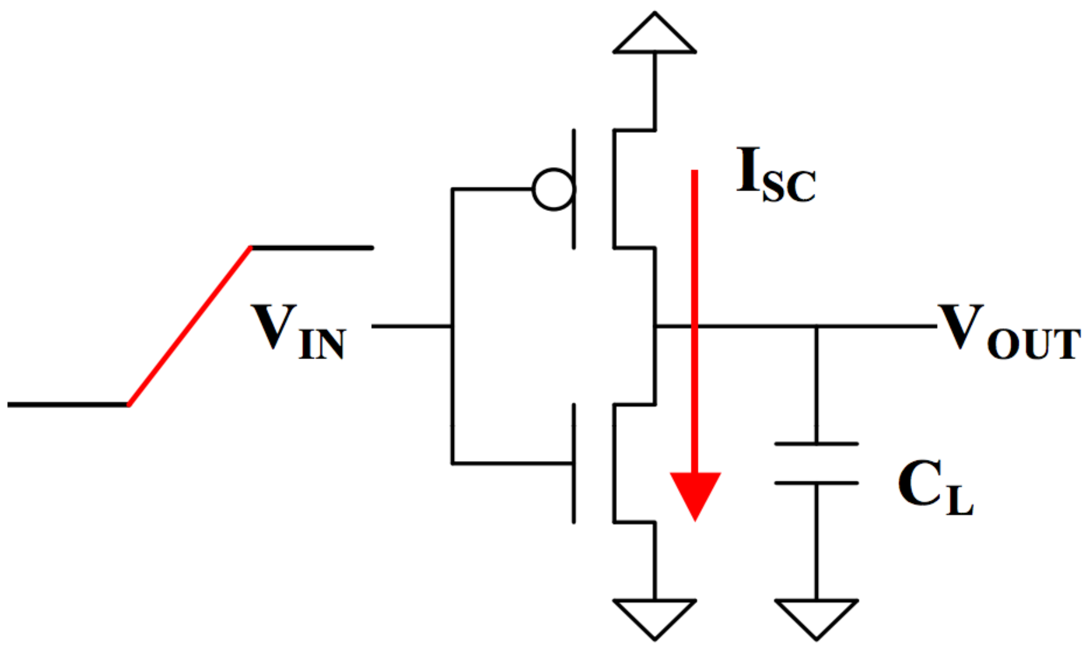
Takes energy to charge up capacitors (Dynamic Power Consumption)

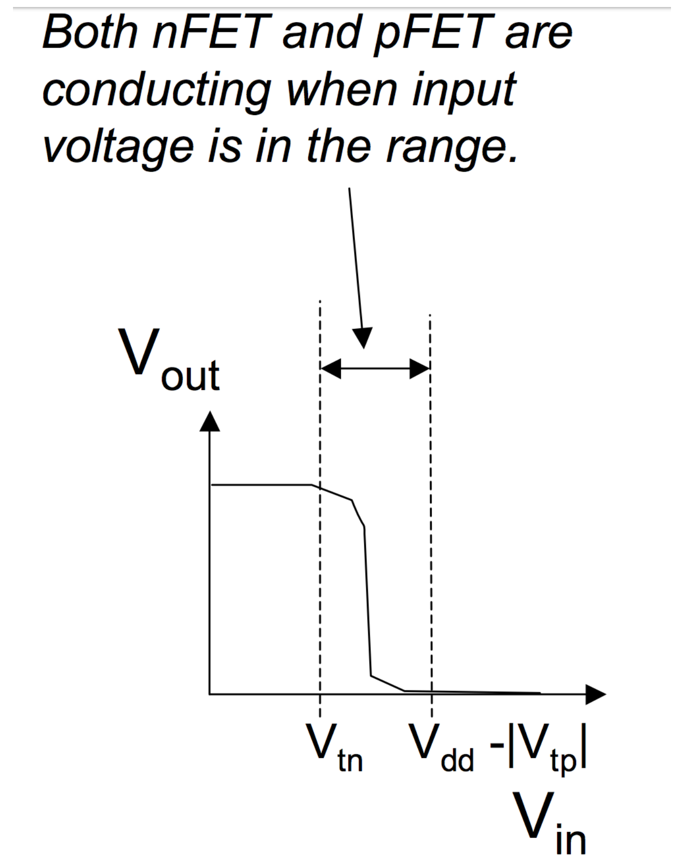


C_L represents the next stage's input capacitance

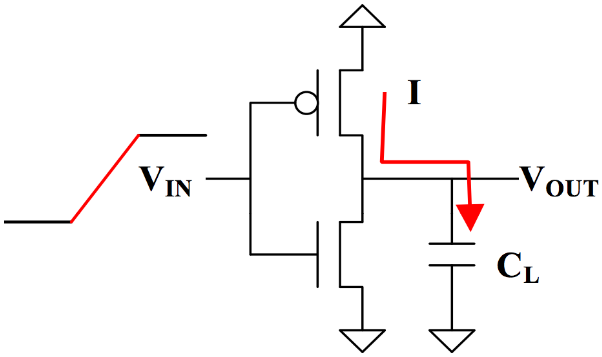
http://web.eecs.umich.edu/~twenisch/470_F07/lectures/21.pdf *GO BLUE*

During Transition short circuit current also appears (Dynamic Power Consumption)



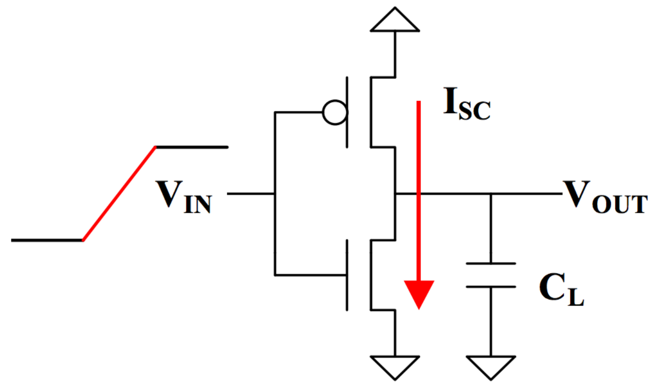


Dynamic Power Consumption



Capacitive Charging

- Caused by need to store up finite charge
- $P \propto CV^2 f$
- C: capacitance of gate
- V: V_{dd} of system
- f: frequency of switching

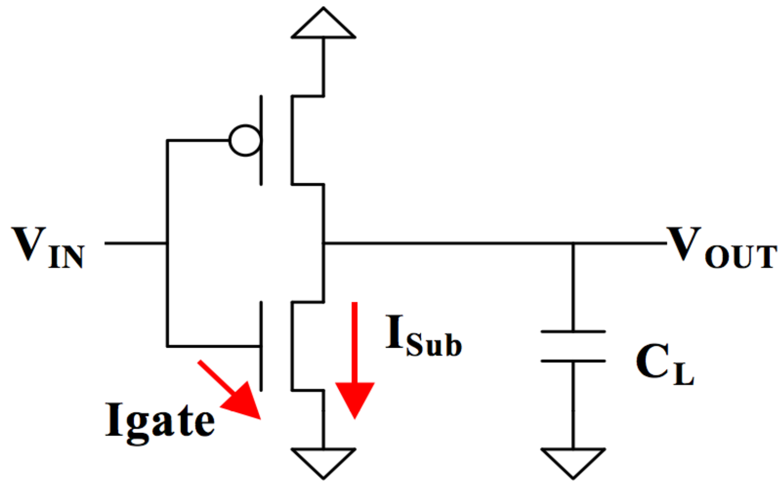


Short Circuit

- $P \propto t_{sc} V I_{peak}$
- t_{sc}: in crossover
- V: V_{dd} of system
- I_{peak}: Max current at crossover
- Good news this is usually rather small compared to **capacitive**

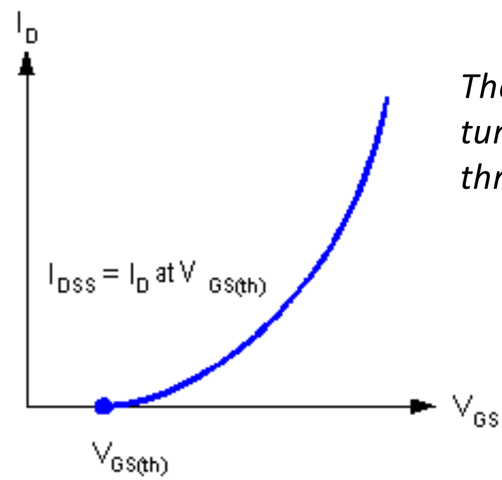
Static Power Consumption

- Burn power even when not switching :/
- As we scale operating voltage and size this becomes more and more of a problem



Sub-Threshold Leakage:

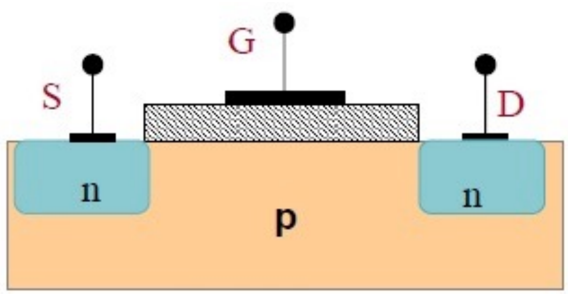
Doesn't start right at zero



The fact that no transistor turns off below its threshold voltage

Gate Leakage:

Where an electron or hole just tunnels through the FET gate



MOSFET Structure

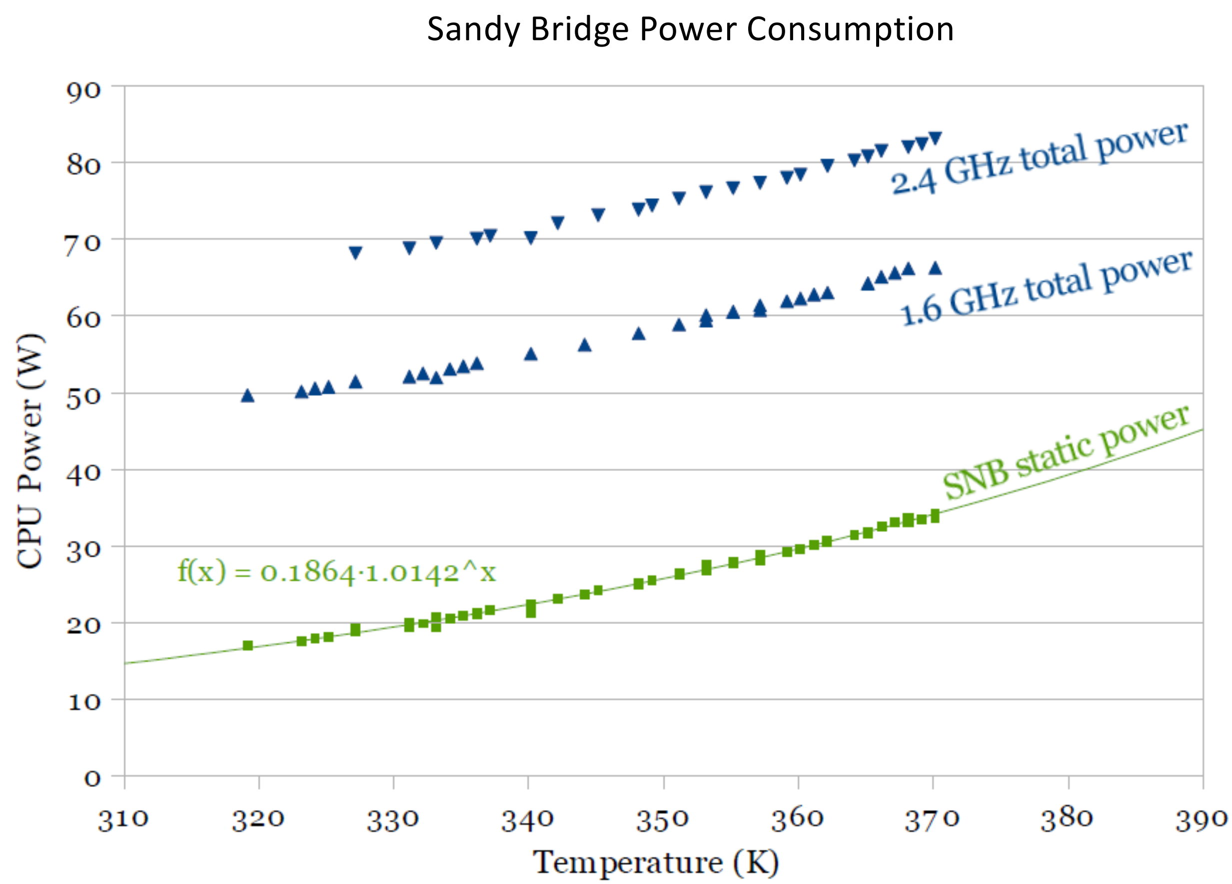
http://www-inst.eecs.berkeley.edu/~cs150/fa11/agenda/lec/lec22-power.pdf

http://web.eecs.umich.edu/~twenisch/470_F07/lectures/21.pdf GO BLUE

Power Consumed $P_{total} = P_{dynamic} + P_{static}$

- In the CMOS era, historically static power has been smaller compared to dynamic power

- This has changed in recent years as things have gotten smaller!

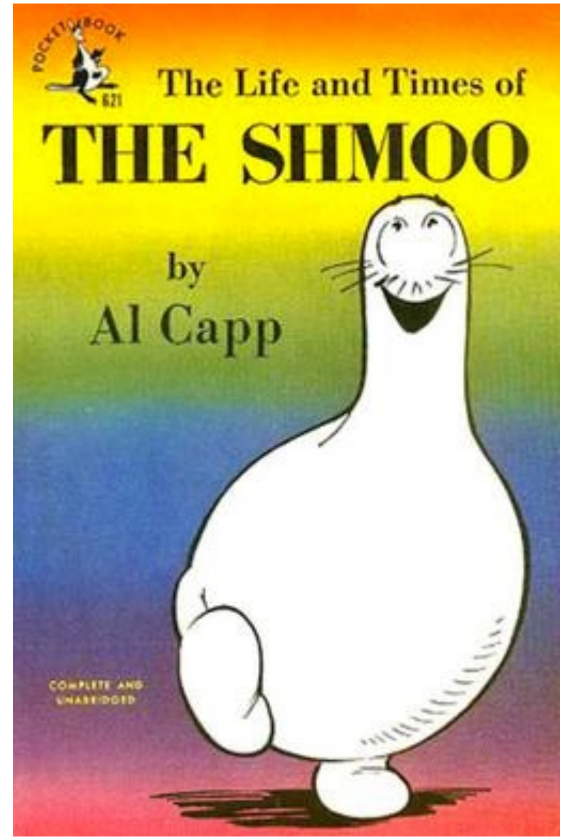


Shmoo plot

https://blog.stuffedcow.net/2012/10/intel32nm-22nm-core-i5-comparison/

Aside: Shmoo Plot

- Sometimes hear plots of various performance specs on semiconductors called "Shmoo" plots
- Called that because they plots look like Shmoos, weird bowling-pin like creatures from Lil Abner,
- Anyways sometimes these comparison plots are called Shmoos

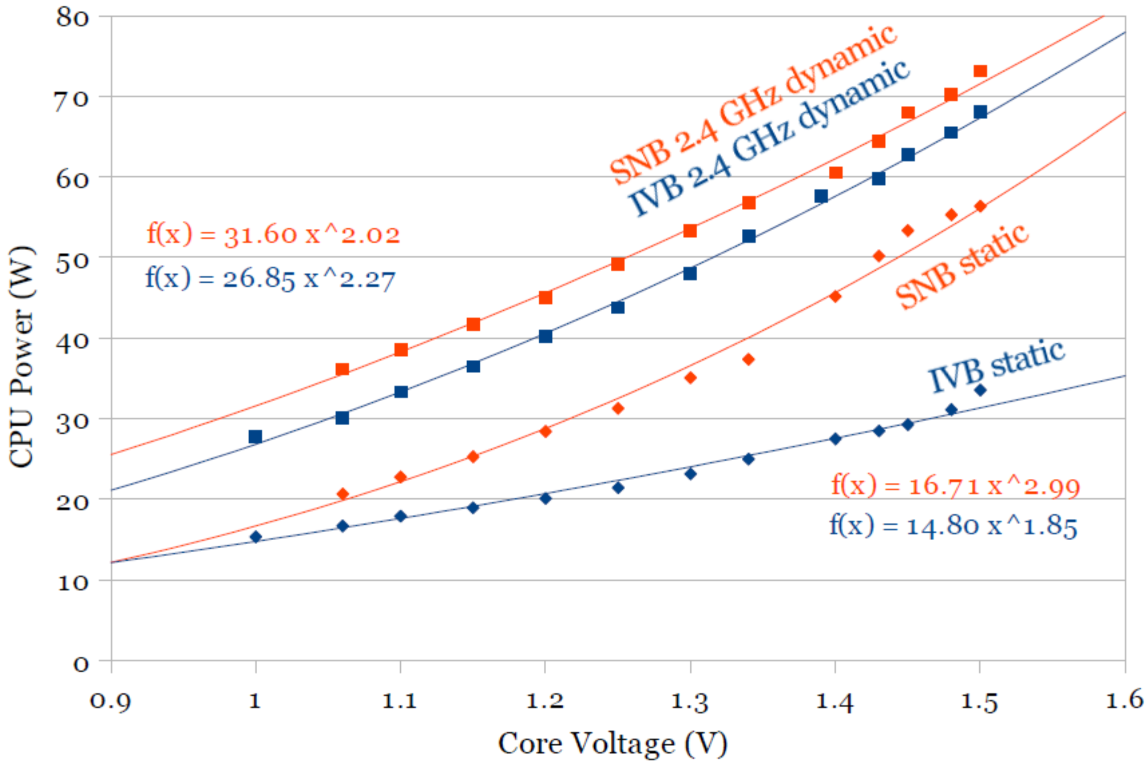


**Wikipedia finally explained this to me…pre-semiconductor, Shmoo plots for magnetic devices looked like Schmoos*

Sandy Bridge vs. Ivy Bridge (32nm vs. 22 nm core i5)

- Sandy Bridge was older model transistor

- Ivy Bridge was 3D transistor which greatly improved static loss (less leakage)



Shmoo plot

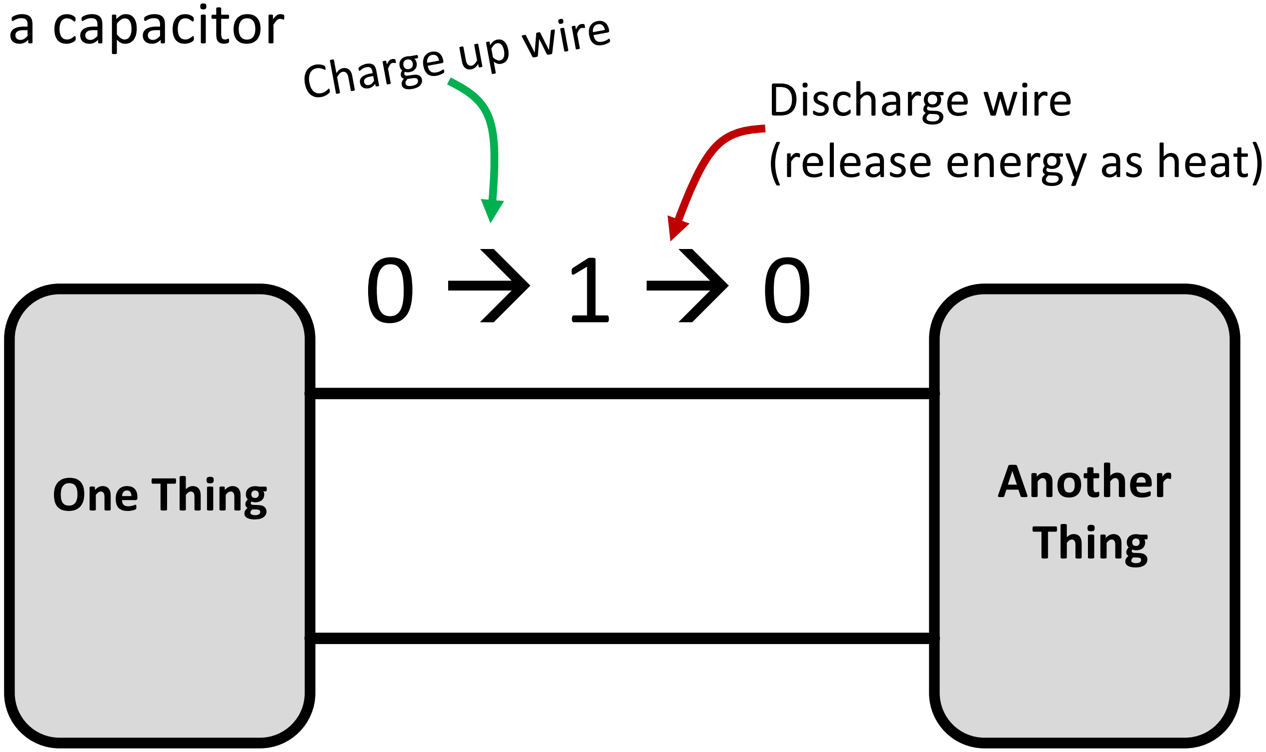
Two Major Ways Power is Consumed on a Device

- Dynamic Power Consumption: By flipping bits in the process of calculations, you will be burning energy. The more bits flipped, the more energy burned.

- Static Power Consumption: Just by “being on” you’re eating power…even if you’re not doing anything.

Dynamic Power Consumption on Our Device Comes From Charge/Discharge Cycles

- All the wires and all the transistors and other parts in our device can be modeled as piles of resistors and capacitors (8.02, 6.002).
- It takes energy to make 1's and 0's (the voltages) appear and change on the wires
- You can think of sending digital information as charging/discharging a capacitor



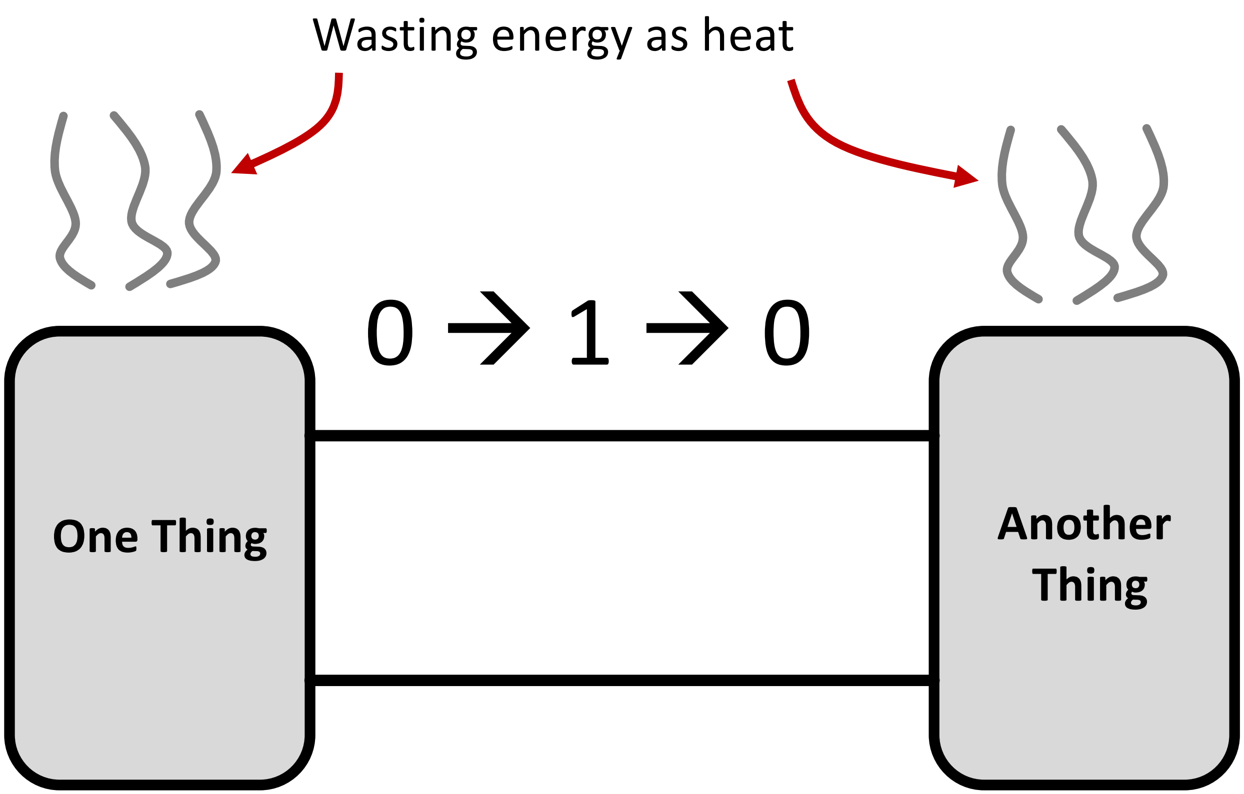
Energy Expended:

$$\propto \frac{1}{2} C V^2$$

C is capacitance of devices (transistors)

Static Power Consumption is power loss from our Devices Just being on in the first place

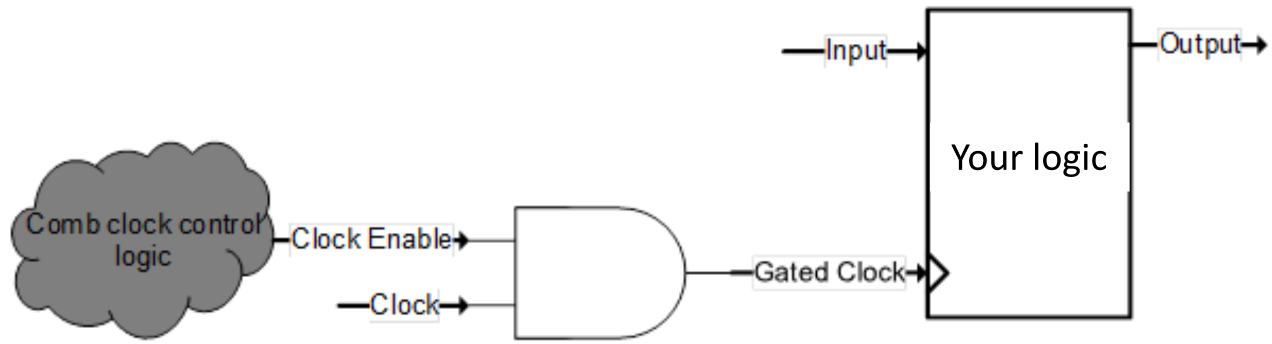
- Just by being on, the devices burn power (even if they're doing nothing)



How to Manage?

- If you’re designing chips from scratch you have some options:
- Dynamic Power Consumption:
  - Clock Gating: Stop the clock that is going to your logic…this will essentially “freeze” the state of your module and prevent if from running in place
- Static Power Consumption:
  - Power Gating: Actually turn off your logic…this will basically remove it from your system and not bleed power by just being on.

Clock-Gating



- Since most modern digital logic is synchronous and only evolves on the edges of clocks (sequential logic), simply stopping the clock (making its period go to infinity) can “freeze” the system
- The system will not flip bits for no reason
- Relatively easy to implement into chip design

https://anysilicon.com/the-ultimate-guide-to-clock-gating/

Power-Gating

- More involved than clock gating! You actually deactivate whole portions of the circuit (have large high side or low side enable transistors)
- Takes up a lot more real estate on chip
- Requires a lot more devices to make work…Have to:
  - Isolate the “dead” portion from ”live portions”
  - Save and reload state of system before after turnoff
  - Worry about pipelining and propagation of removal/reappearance
  - Longer shutdown and startup times
- But does minimize power wasted!!!!

https://anysilicon.com/power-gating/

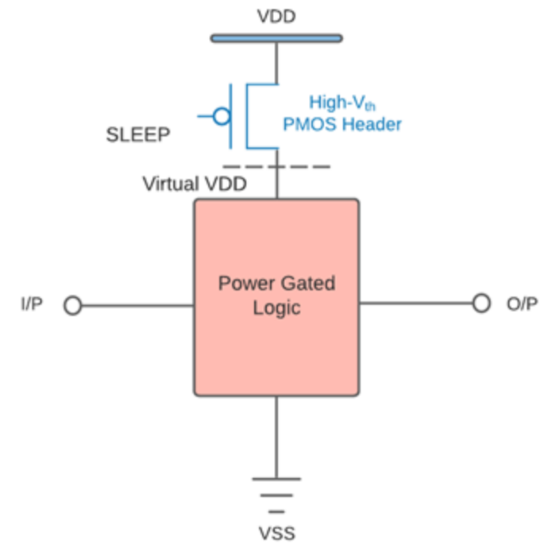


Fig. 1.3: Header Switch Cell

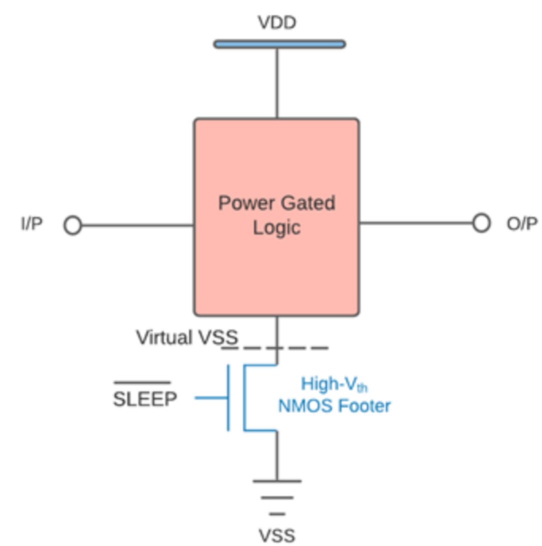


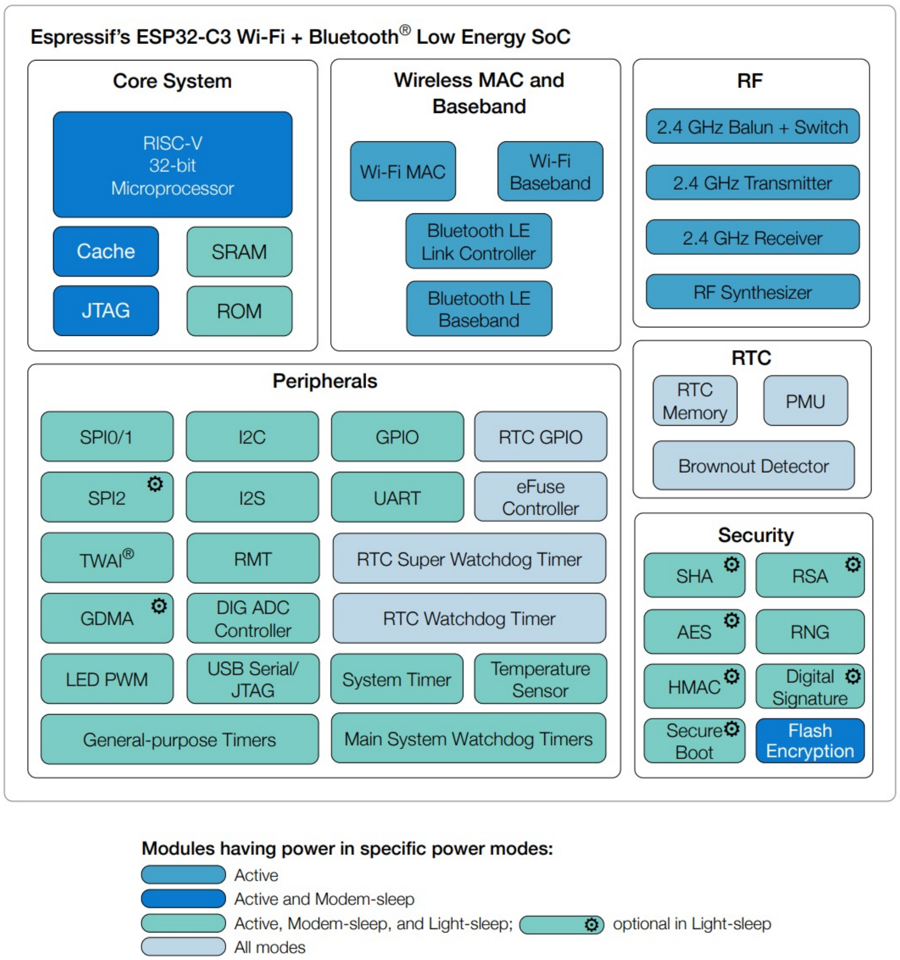
Fig. 1.4: Footer Switch cell

What Can We Do

- We can't redesign our chips in 6.9000. There's not enough time
- We need to instead use them more intelligently
- Smarter programming/better designs in software
- But also many of the features we just talked about can be accessed through in-built API-calls on the devices we have
- Minimize:
  - Time it takes to do something
  - Time we're "ON"
  - If we're "ON" make sure we're doing as much as possible
  - No lollygagging around.

ESP32 C3 Power Modes

- There's about four of them power modes depending on who you talk to.



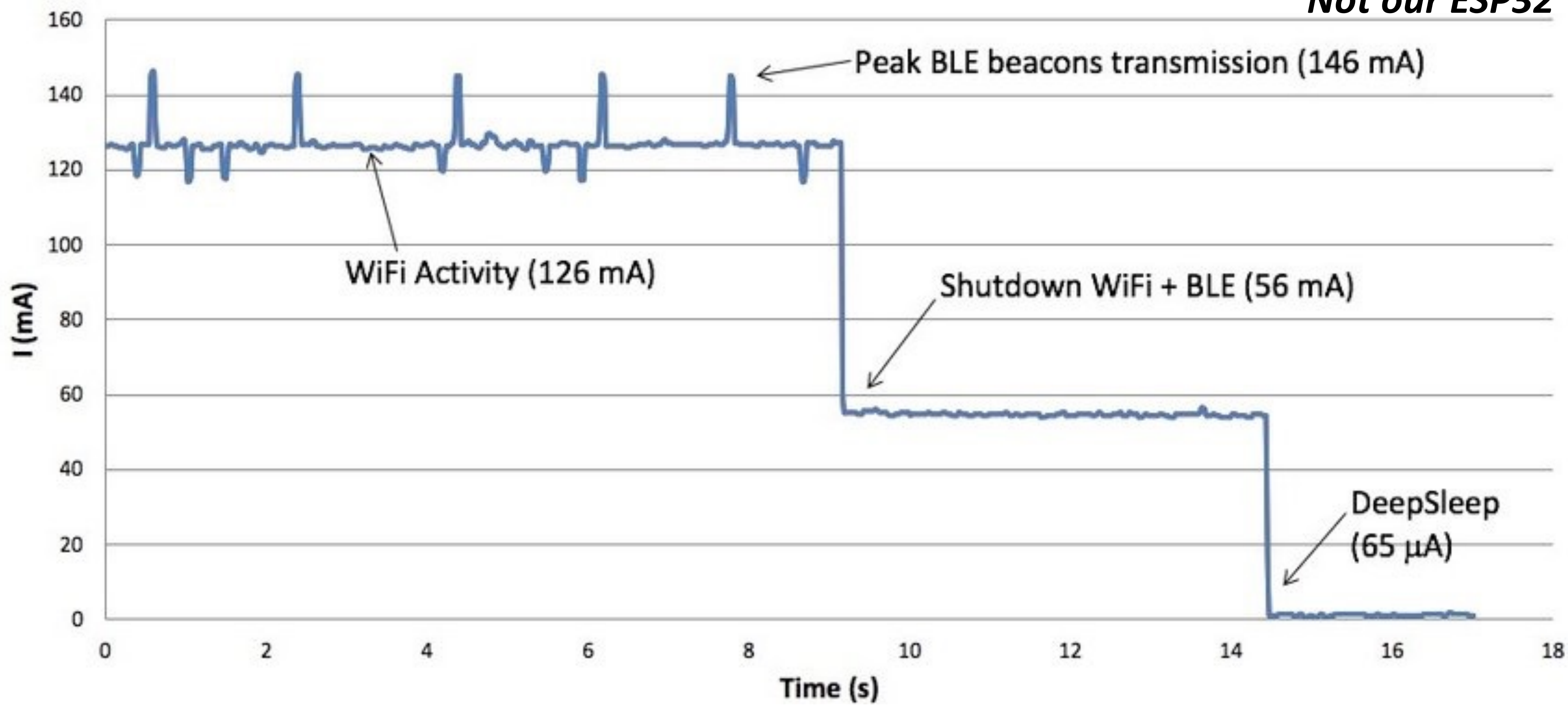
Work mode	Description		Peak (mA)
Active (RF working)	TX	802.11b, 1 Mbps, @20.5 dBm	345
		802.11g, 54 Mbps, @18 dBm	285
		802.11n, HT20, MCS7, @17.5 dBm	280
		802.11n, HT40, MCS7, @17 dBm	280
	RX	802.11b/g/n, HT20	82
		802.11n, HT40	84

Mode	CPU Frequency (MHz)	Description	Typ All Peripherals Clocks Disabled (mA)	Typ All Peripherals Clocks Enabled (mA)[1]
Modem-sleep[2,3]	160	CPU is idle	16	21
		CPU is running	23	28
	80	CPU is idle	13	18
		CPU is running	17	22

Mode	Description	Typ (μA)
Light-sleep	VDD_SPI and Wi-Fi are powered down, and all GPIOs are high-impedance	130
Deep-sleep	RTC timer + RTC memory	5
Power off	CHIP_EN is set to low level, the chip is powered off	1

ESP32 Power Consumption is Very Complicated

Not our ESP32



https://www.researchgate.net/figure/Energy-consumption-of-Sparkfun-ESP32-Things_fig4_332407808

ESP32 C3 Measurements

During regular running no wifi:

Mode	CPU Frequency (MHz)	Description	Typ All Peripherals Clocks Disabled (mA)	Typ All Peripherals Clocks Enabled (mA)[1]
Modem-sleep[2,3]	160	CPU is idle	16	21
		CPU is running	23	28
	80	CPU is idle	13	18
		CPU is running	17	22

Probably right???

During running with Wifi at some point during the request/contact cycle

Work mode		Description	Peak (mA)
Active (RF working)	TX	802.11b, 1 Mbps, @20.5 dBm	345
		802.11g, 54 Mbps, @18 dBm	285
		802.11n, HT20, MCS7, @17.5 dBm	280
		802.11n, HT40, MCS7, @17 dBm	280
	RX	802.11b/g/n, HT20	82
		802.11n, HT40	84

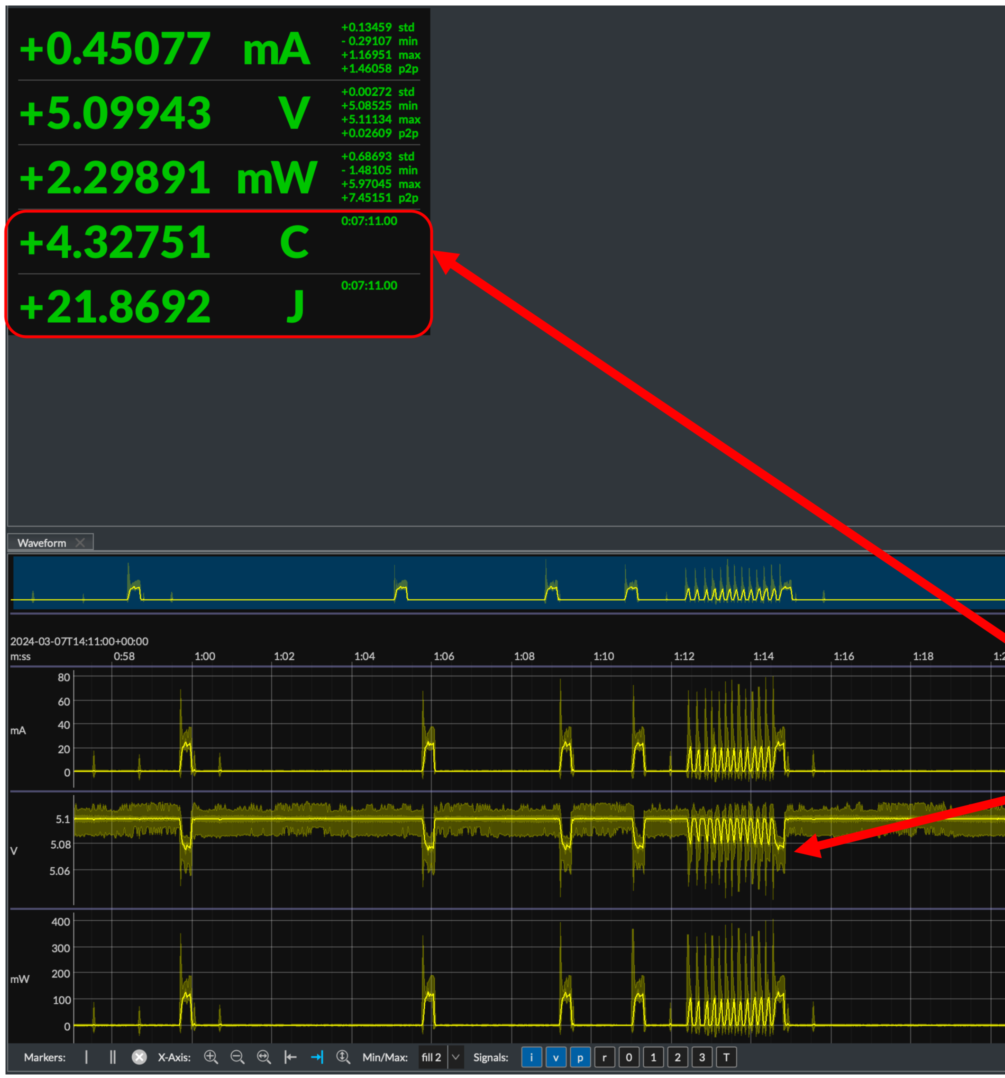
Probably right???

The power consumption is complicated

Trying and failing to connect to web resource (DNS fail iunno)

Light Sleeps and Wakes

Worry about Power but also Energy



- The power pattern that appears during complicated tasks such as a WiFi request may not be best encompassed by an instantaneous power
- May need to worry more about energy per message
- Or something else related to integrating power over time!!!!

Light Sleep: `esp_light_sleep_start();`

Seems to go to sleep on a timer…

Mode	Description	Typ (μA)
Light-sleep	VDD_SPI and Wi-Fi are powered down, and all GPIOs are high-impedance	130
Deep-sleep	RTC timer + RTC memory	5
Power off	CHIP_EN is set to low level, the chip is powered off	1

Definitely Lower Power but is it good enough?

Why is it 4 times as large?

Maybe all pins aren't high-impedance?

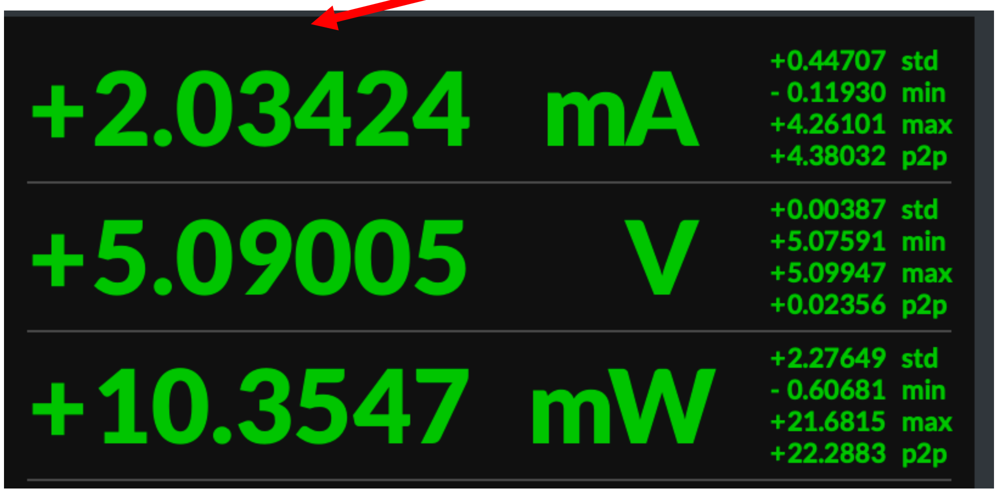
Iunno why is this so off

Light Sleep?

```
esp_light_sleep_start();
```

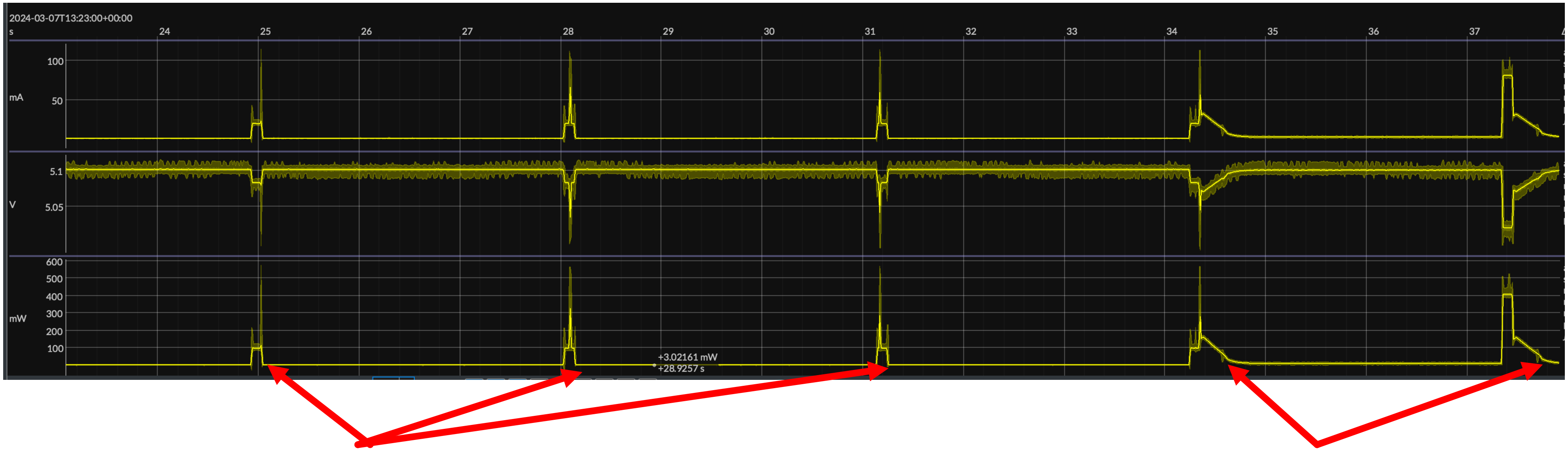
But sometimes…

Mode	Description	Typ (μA)
Light-sleep	VDD_SPI and Wi-Fi are powered down, and all GPIOs are high-impedance	130
Deep-sleep	RTC timer + RTC memory	5
Power off	CHIP_EN is set to low level, the chip is powered off	1



WTF?

During the Light-Sleep-Wake Cycle in Lab:



Goes back to
0.5 mA

Doesn't go back to
0.5mA…now 2mA?

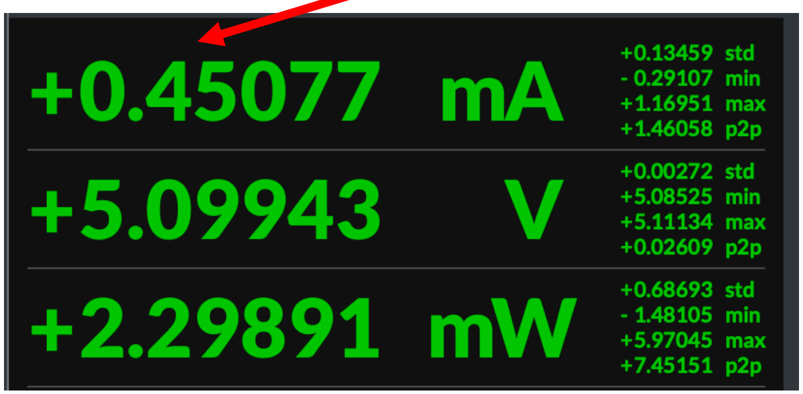
- This needs to be figured out. An inconsistent out-of-spec performance makes more me worried than a consistent out-of-spec performance

Same ESP32 but without Arduino overlay…just raw C and the Espressif IDF toolchain:

```c
#include<stdio.h>
#include "esp_sleep.h"

void app_main() {
esp_sleep_enable_timer_wakeup(6000000); // 6 sec
while (1){
  printf("hi there\n");
  esp_deep_sleep_start();
  }
}
```

Goes to this number consistently during light sleep



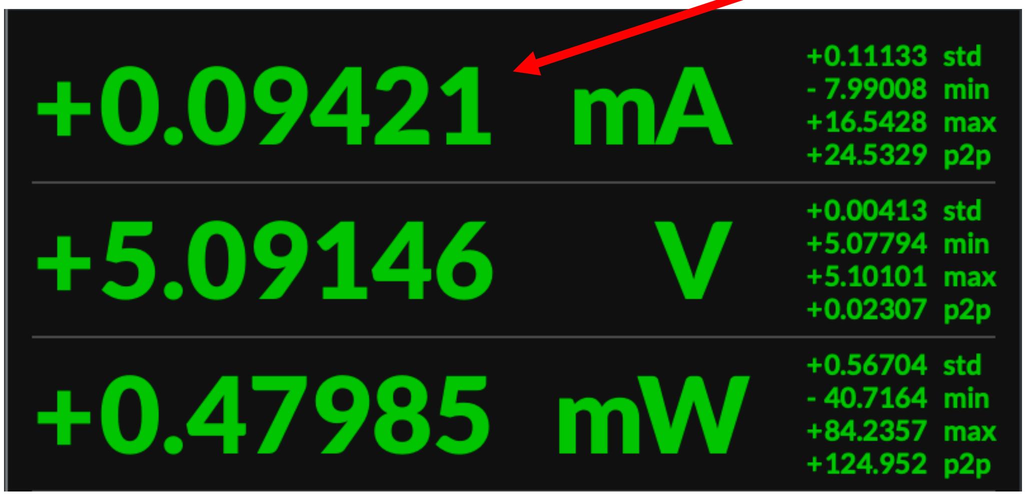
So maybe Arduino.h is doing some stuff we don’t want

Oooh goodness we (as in you, the student) will probably have to dig into this…

And then there’s Deep Sleep

```
esp_deep_sleep_start();
```

Mode	Description	Typ (μA)
Light-sleep	VDD_SPI and Wi-Fi are powered down, and all GPIOs are high-impedance	130
Deep-sleep	RTC timer + RTC memory	5
Power off	CHIP_EN is set to low level, the chip is powered off	1



Definitely Lower Power but is it good enough?

Why is it 18 ish times larger?

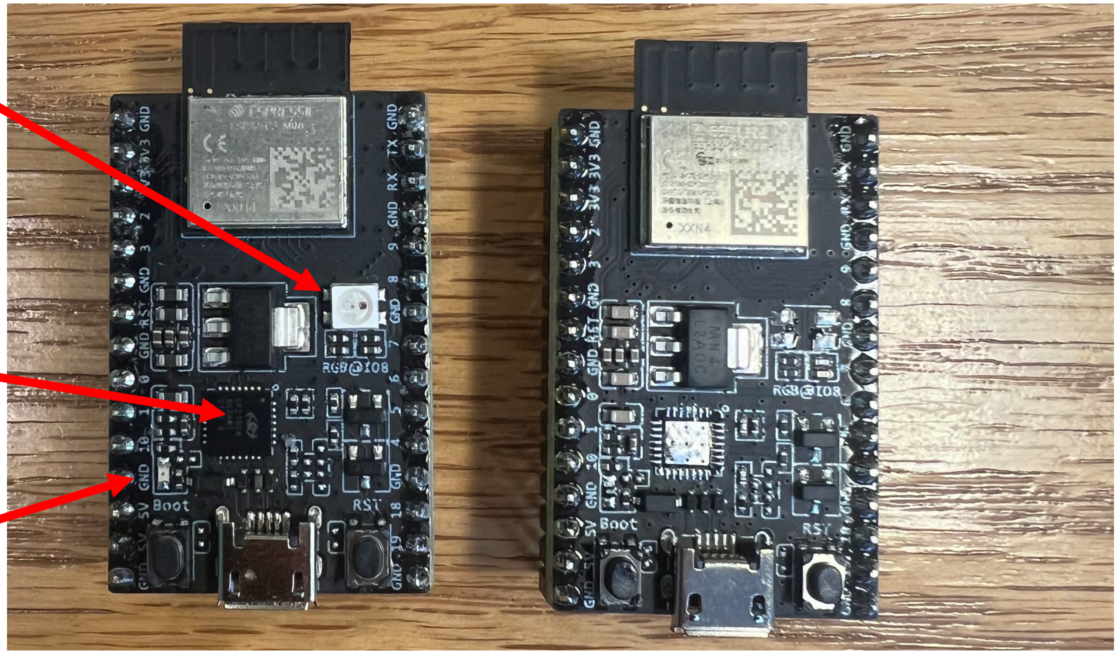
Should we worry?

Remember Board

LED that Comes on Sometimes

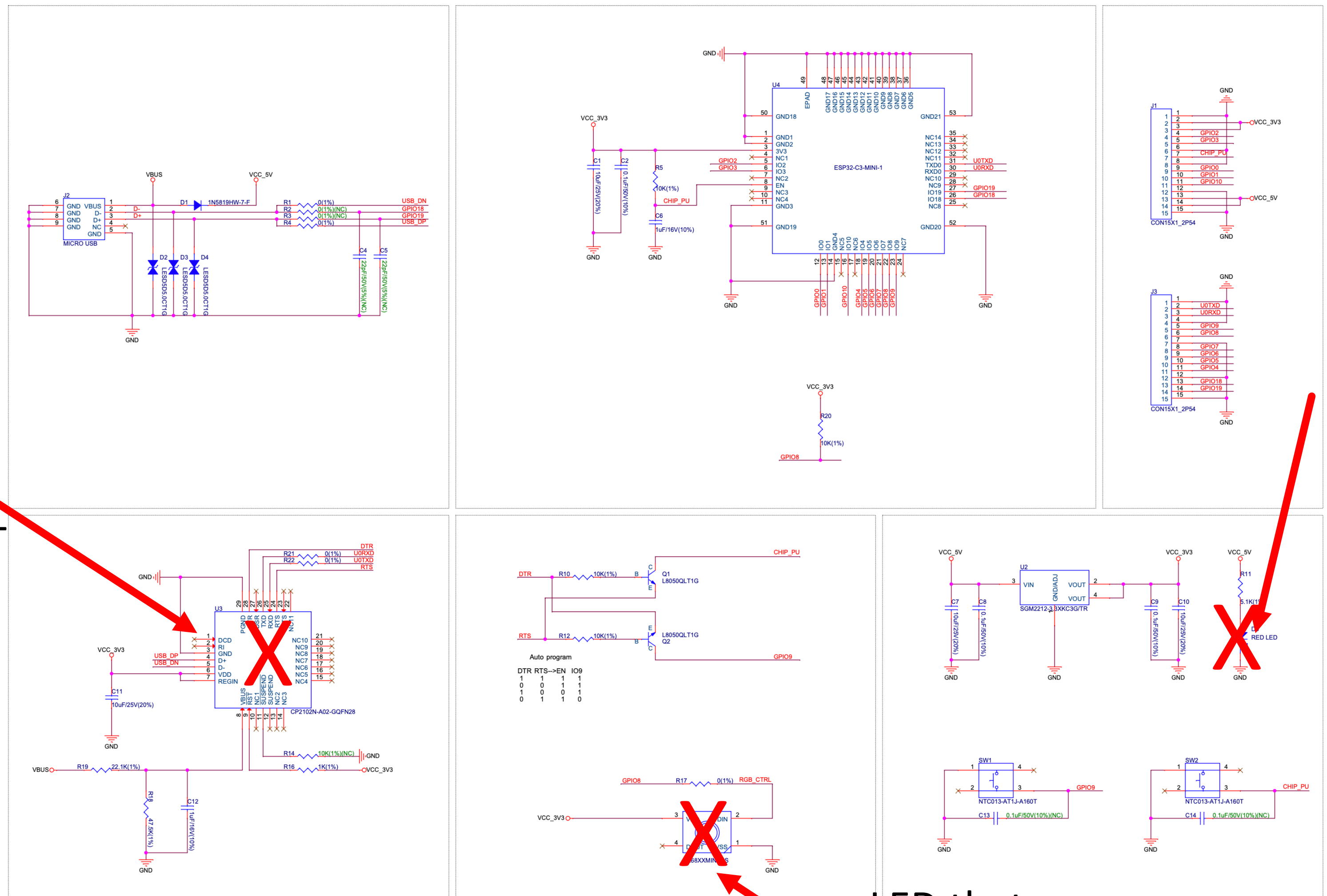
CP2102 USB-UART Chip

Always-On LED



Remember the board? We did remove the non-important stuff from it for those previous numbers…so I would expect we'd be close.

ESP32 C3 Devkit M1 Schematic

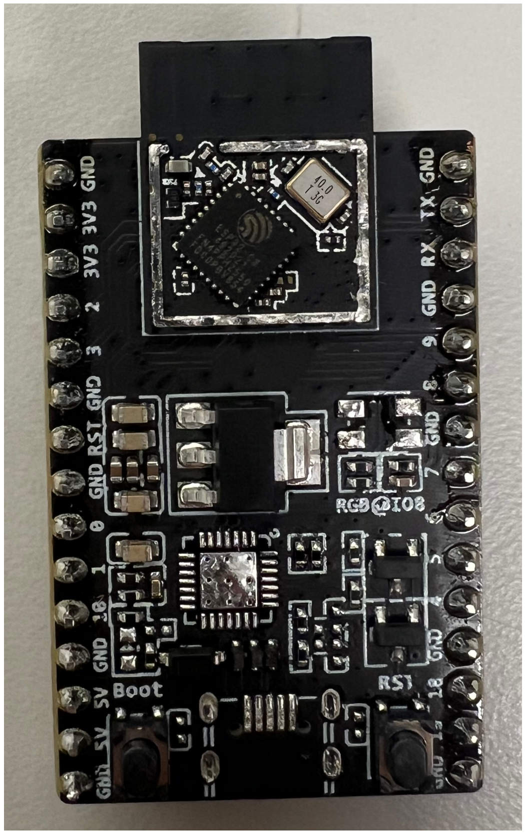


CP2102 USB-UART Chip

Always-On LED

LED that Comes on Sometimes

Is there anything else in the Mini Module?



This is the reference design of the module.

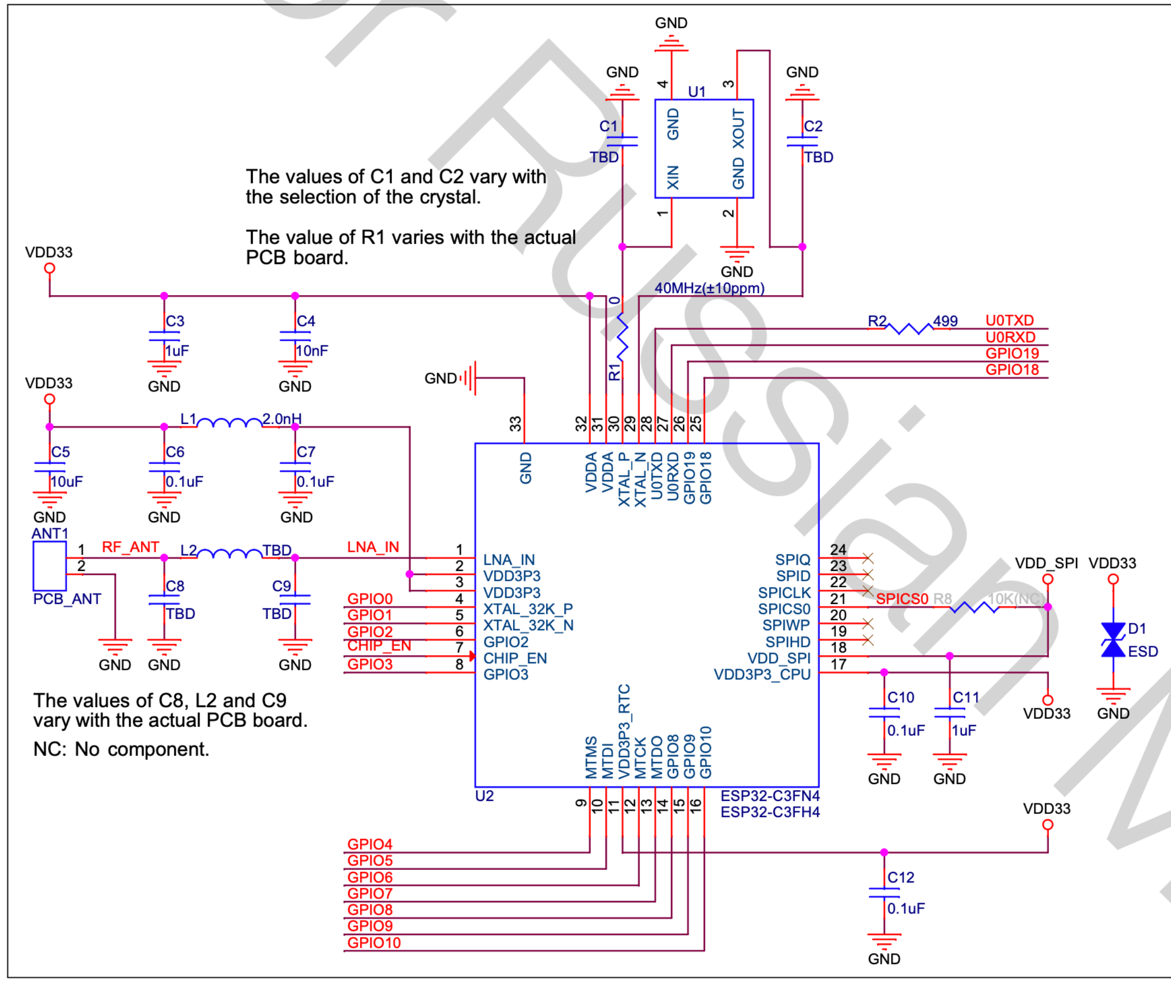


Figure 5: ESP32-C3-MINI-1 Schematics

https://www.espressif.com/sites/default/files/russianDocumentation/esp32-c3-mini-1_datasheet_en.pdf

Is there anything else in the Mini Module?

Nothing…

Ugh….

This is the reference design of the module.

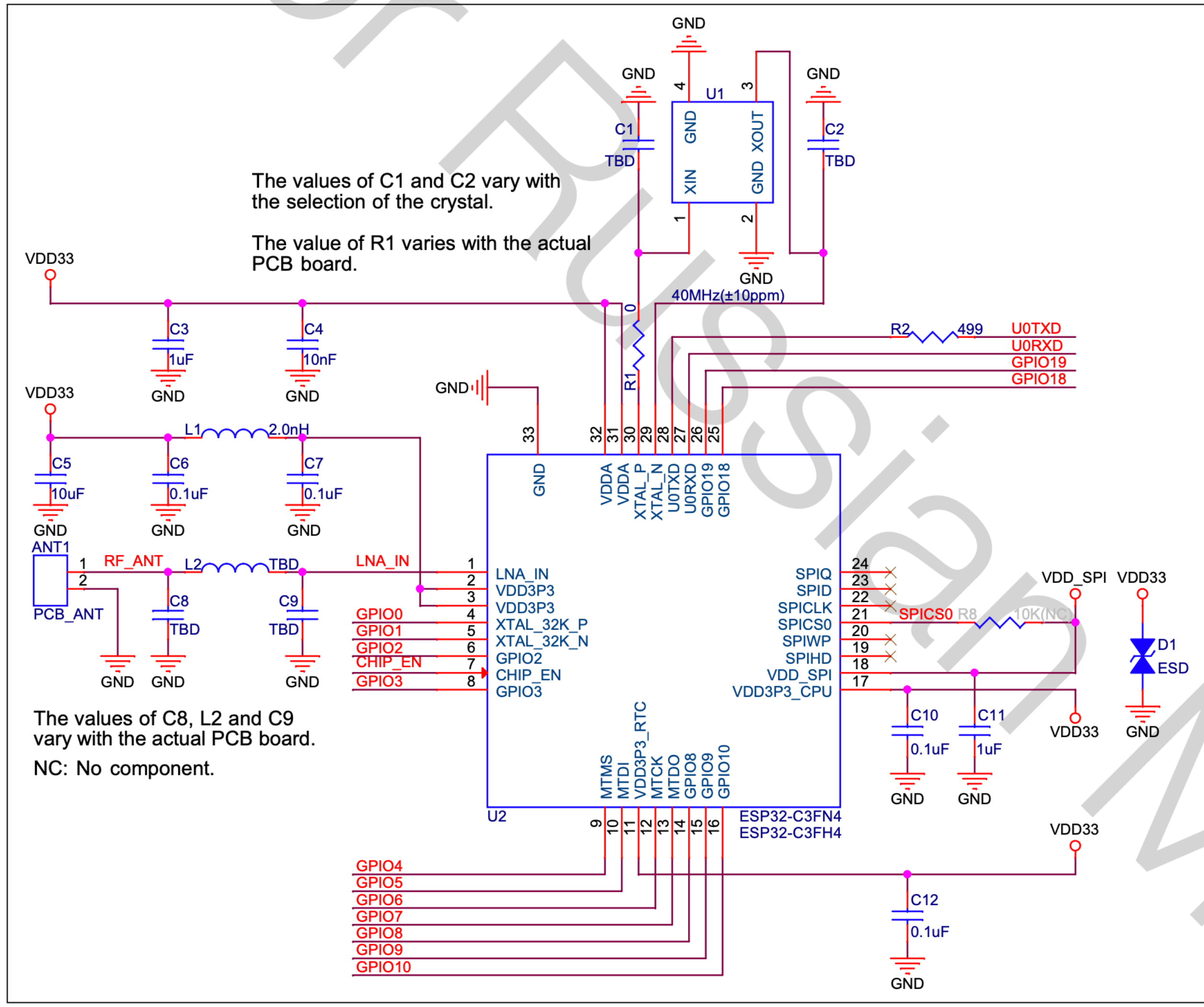
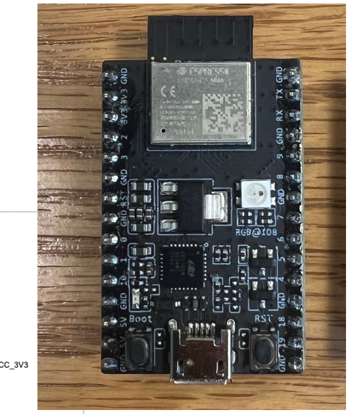
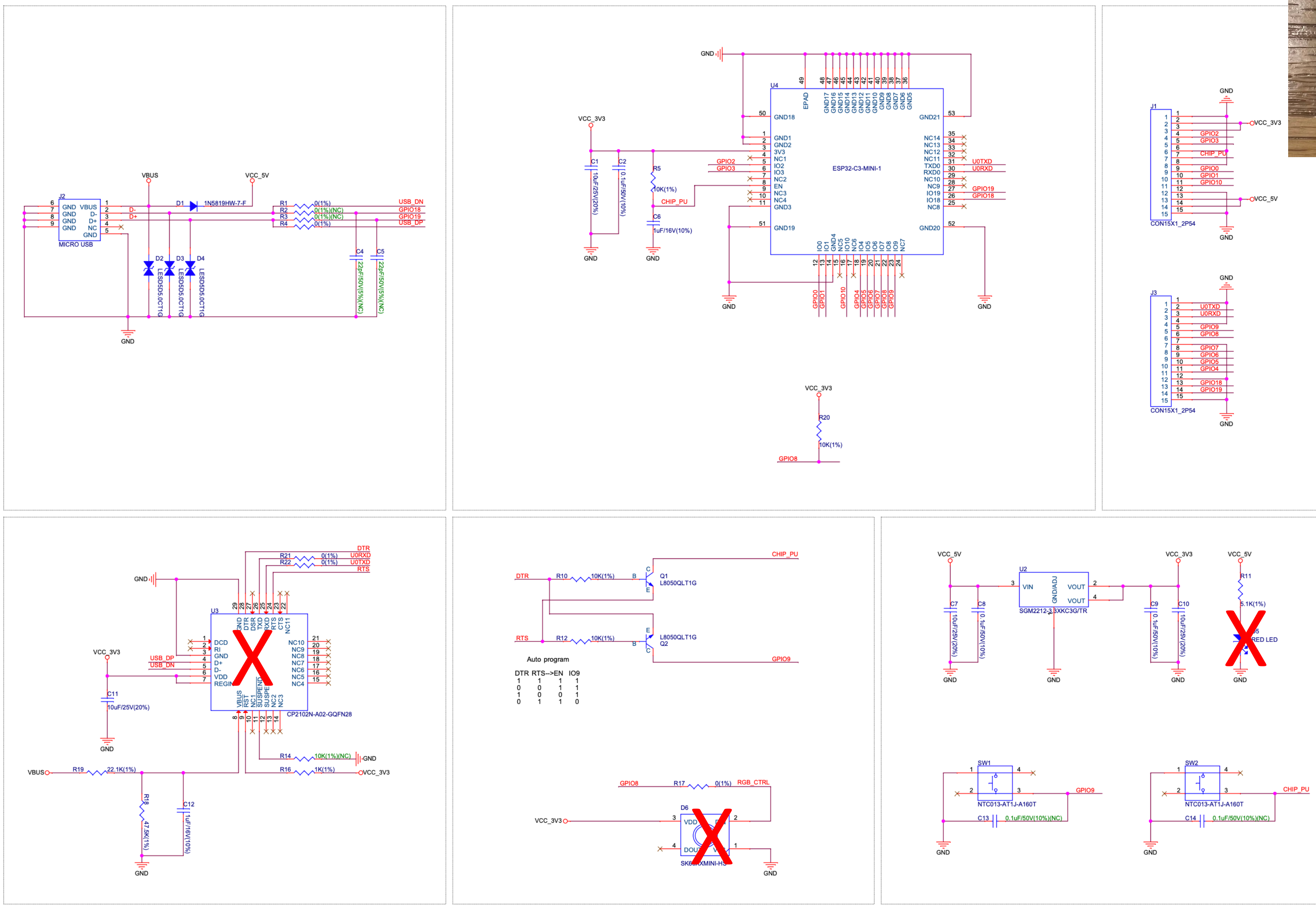


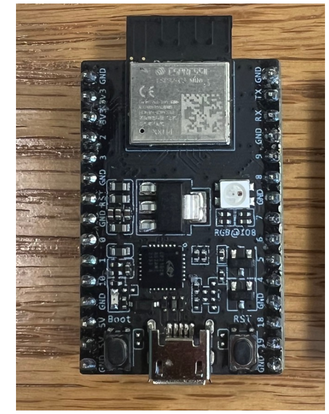
Figure 5: ESP32-C3-MINI-1 Schematics

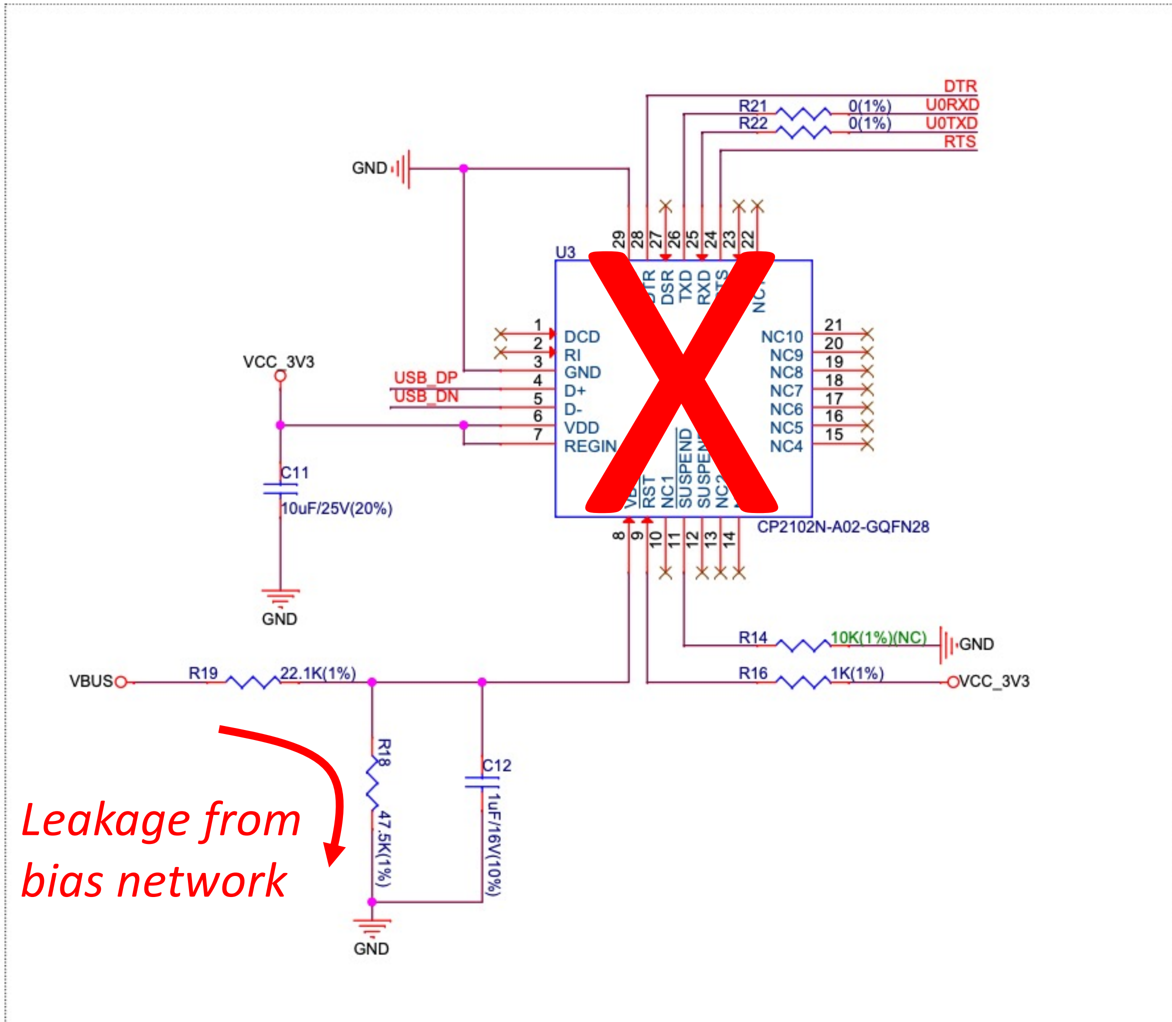
Back to the Whole Board





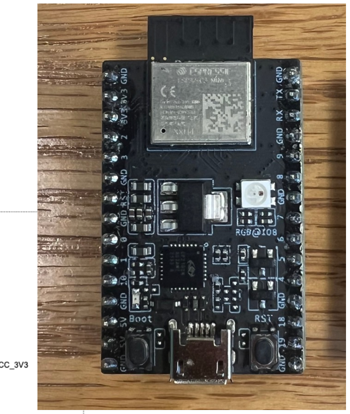
Any particular spots?

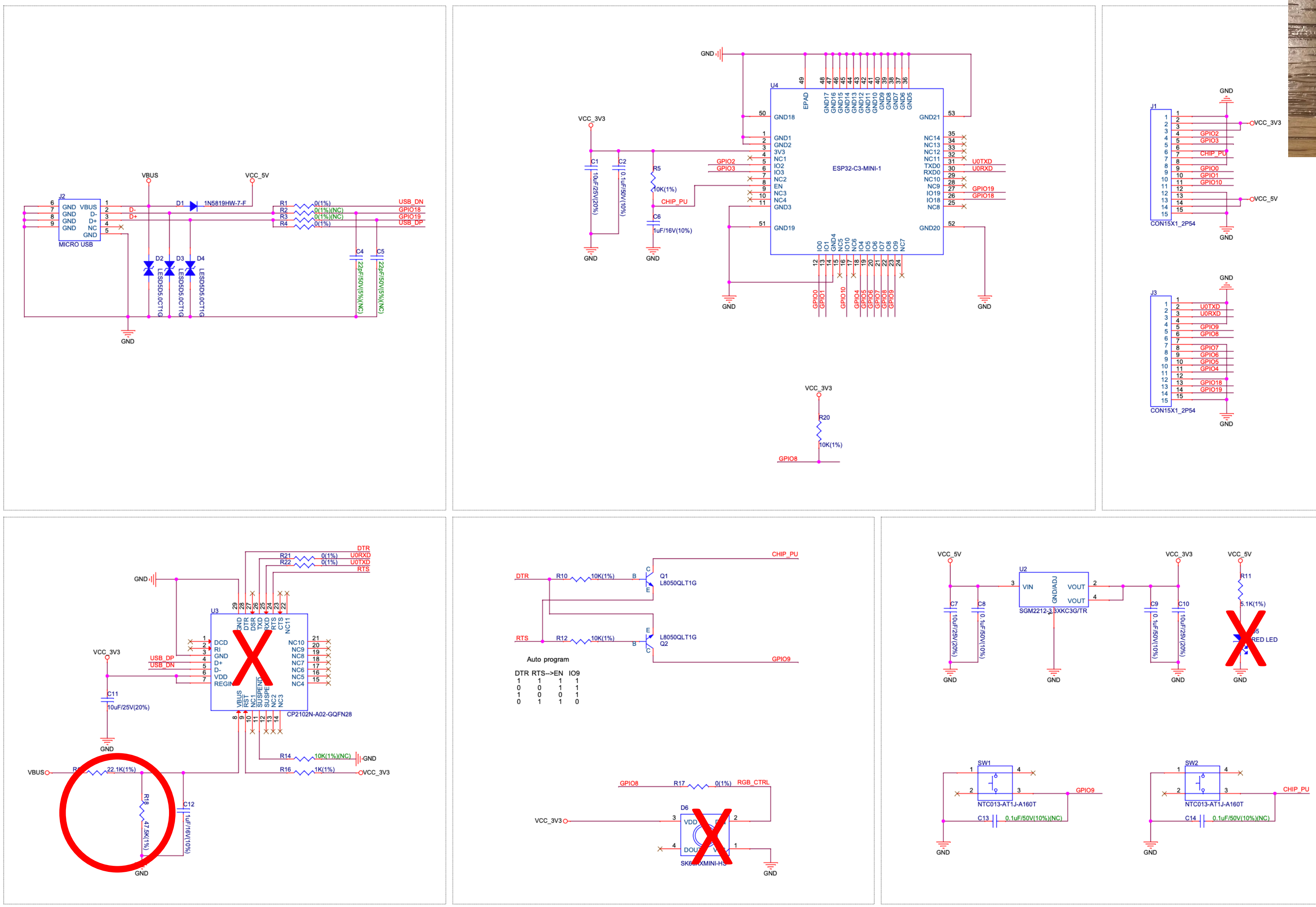




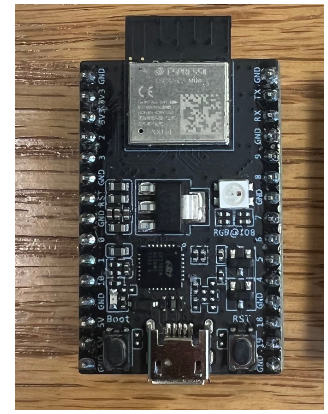
So maybe about 70 uA for that bias network…completely uneeded

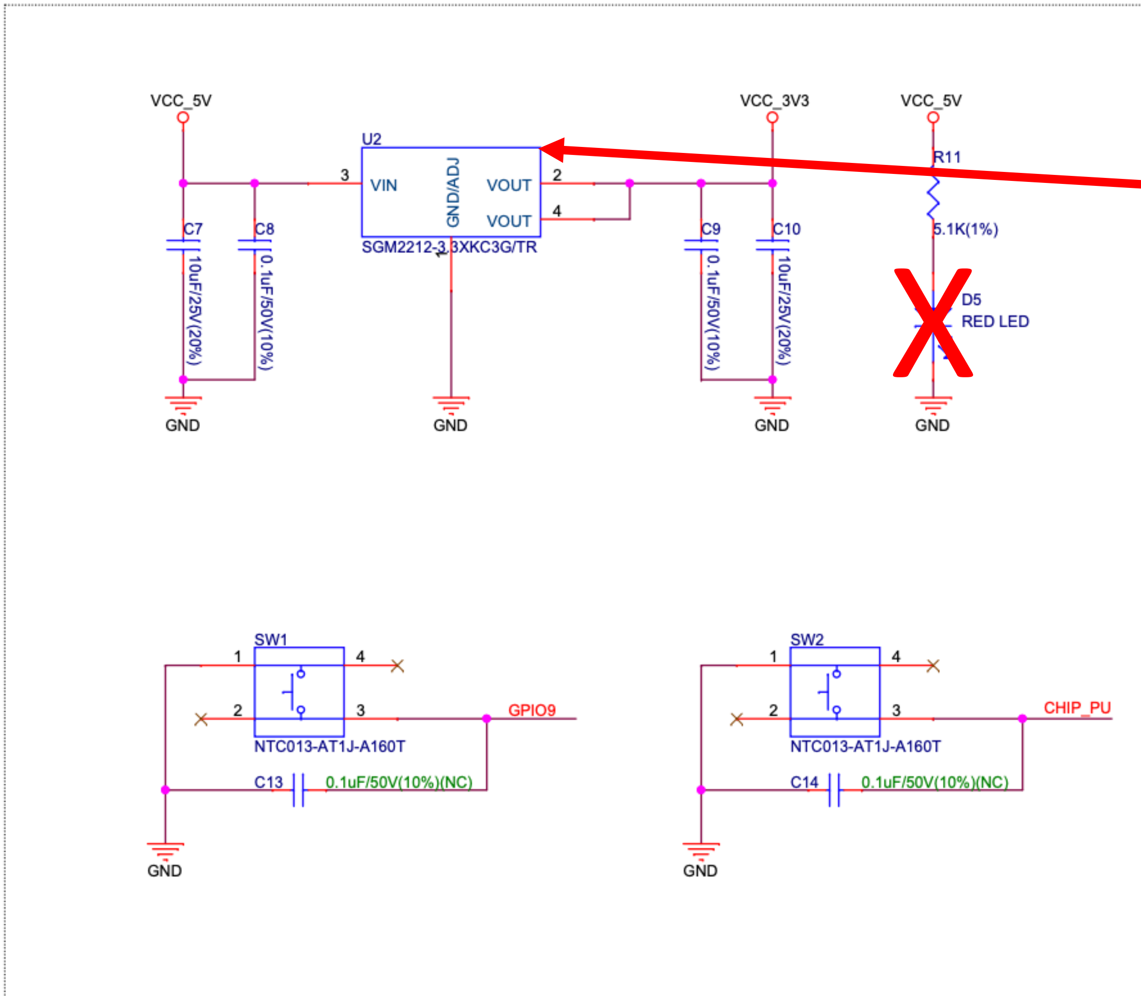
Back to the Whole Board





Any particular spots?

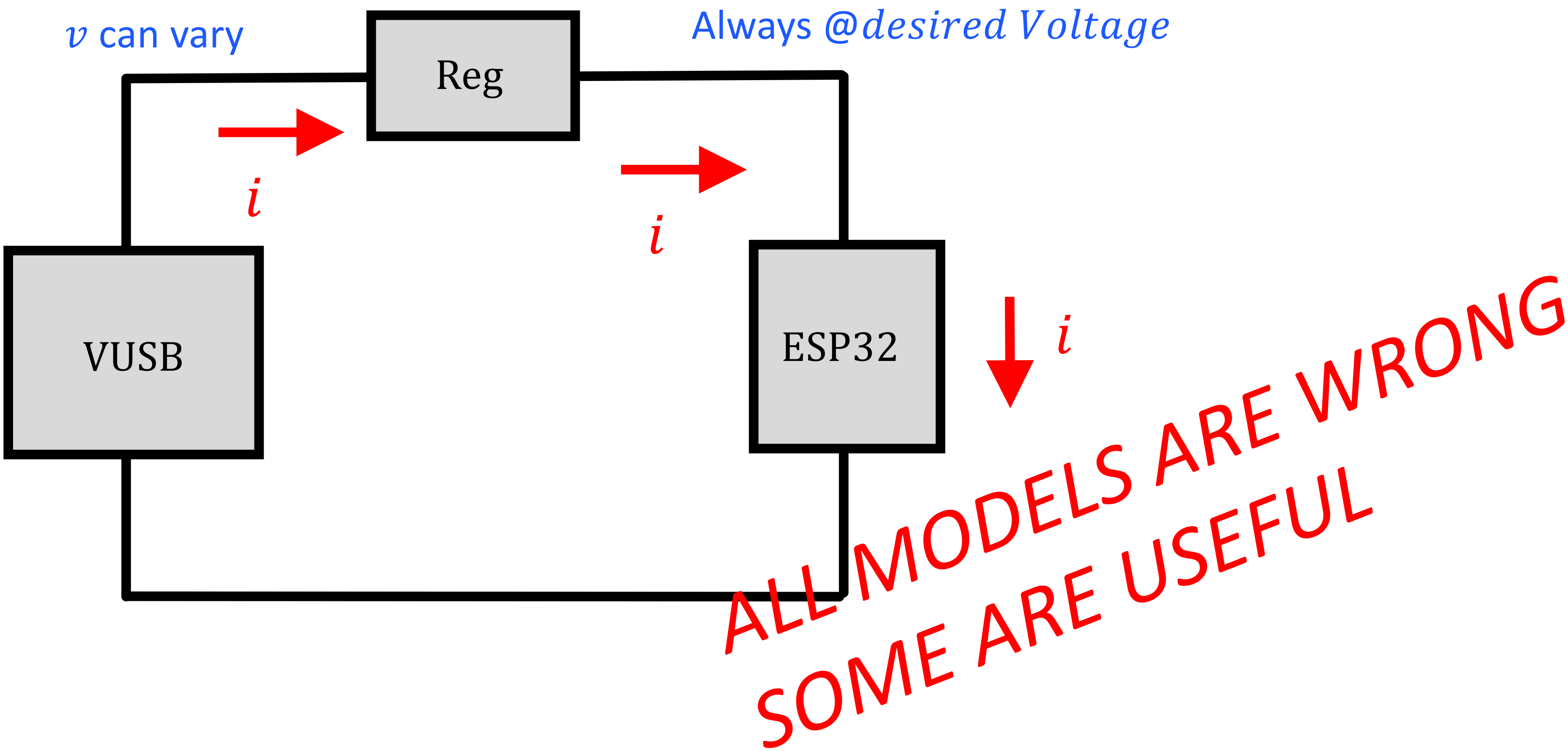




The linear regulator will have an internal path to ground

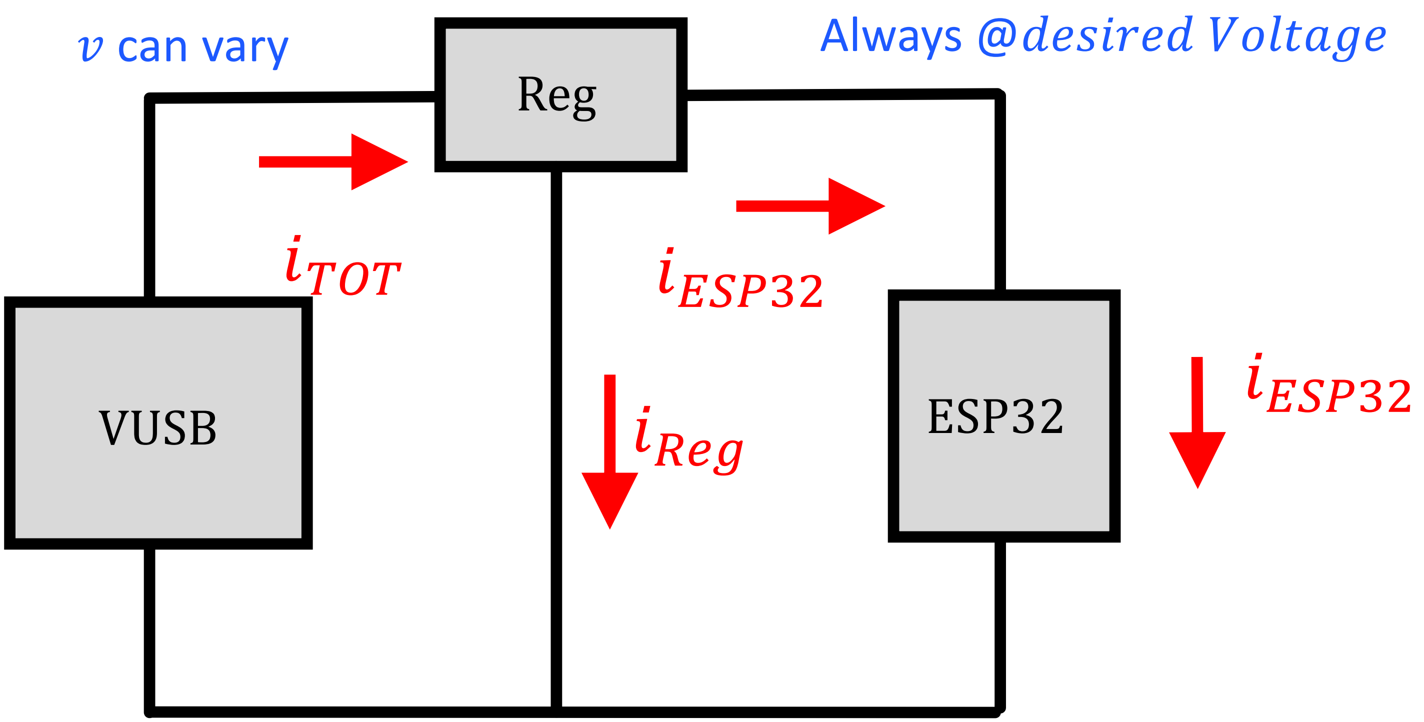
We saw a Linear Regulator on Tuesday

- Constant Current Device (KCL maintained)
- Can only regulate down in voltage

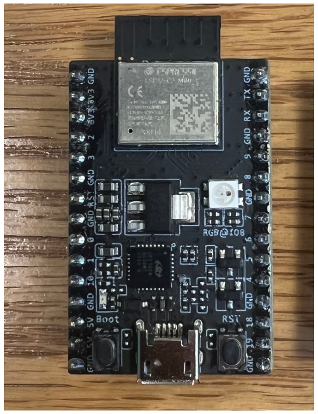


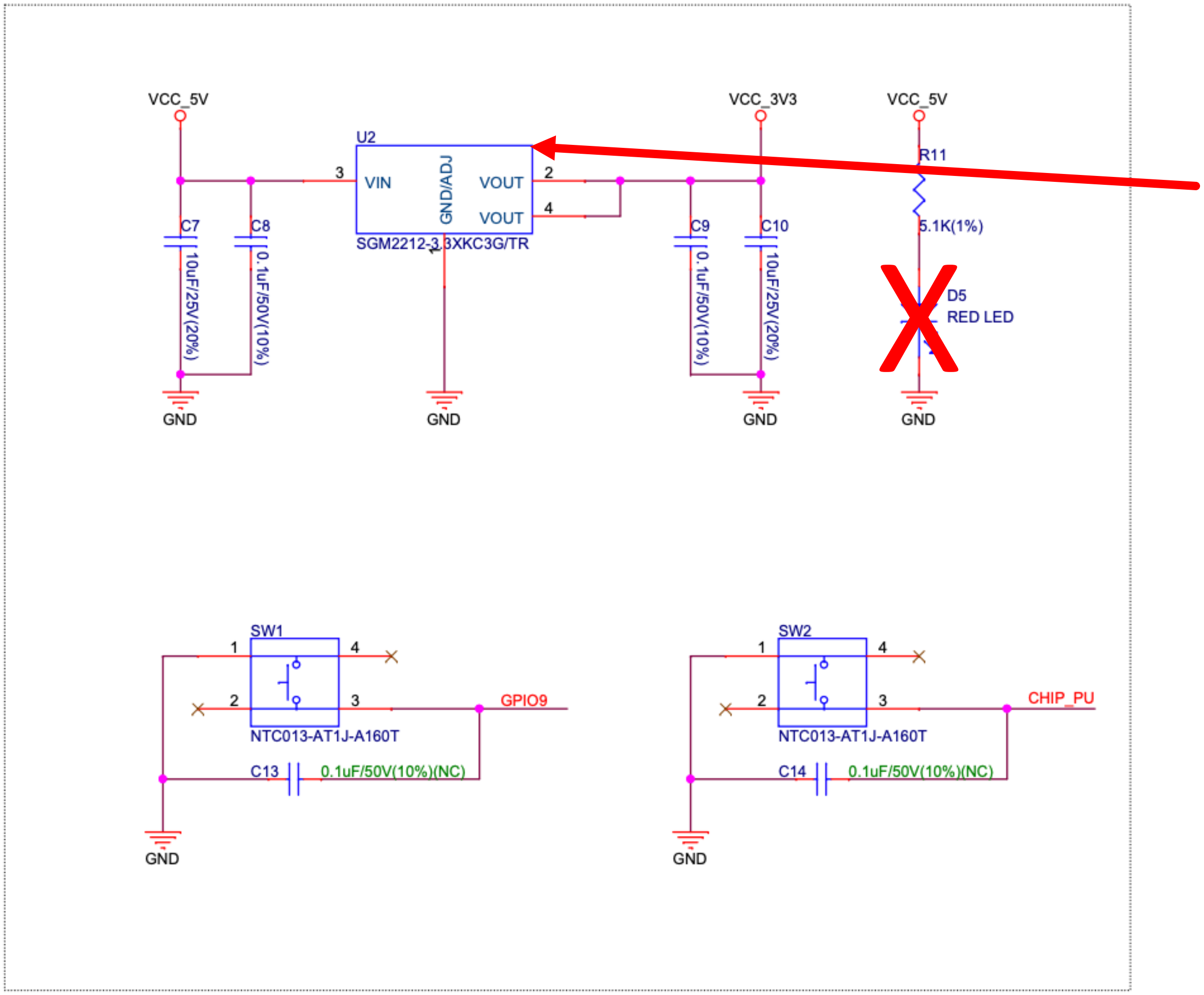
We saw a Linear Regulator on Tuesday

- Constant Current Device (KCL maintained)
- Can only regulate down in voltage



Any particular spots?





The linear regulator will have an internal path to ground

SGM2212

SGMICRO

SGM2212
800mA, Low Noise, Low Quiescent Current, High PSRR, Low Dropout Linear Regulator

GENERAL DESCRIPTION

The SGM2212 is a low noise, low quiescent current, high PSRR, fast transient response and low dropout voltage linear regulator which is designed using CMOS technology. It provides 800mA output current capability. The operating input voltage range is from 2.7V to 20V. The fixed output voltages are 1.8V, 2.5V, 2.8V, 3.3V, 5V and adjustable output voltage range is from 1.25V to 15V.

Other features include short-circuit current limit and thermal shutdown protection.

The SGM2212 is available in Green TO-252-2, TO-263-3, SOT-223-3 and TDFN-3×3-8L packages. It operates over an operating temperature range of -40℃ to +125℃.

FEATURES

- **Input Voltage Range: 2.7V to 20V**
- **Output Current: 800mA**
- **Fixed Output Voltages:1.8V, 2.5V, 2.8V, 3.3V, 5V**
- **Adjustable Output Voltage Range: 1.25V to 15V**
- **Output Voltage Accuracy: ±1% at +25℃**
- **Line Regulation: 0.012% (MAX)**
- **Load Regulation: 0.4% (MAX)**
- **Stable with Small Case Size Ceramic Capacitors**
- **Output Current Limit**
- **Thermal Shutdown Protection**
- **-40℃ to +125℃ Operating Temperature Ra**
- **Available in Green TO-252-2, TO-263-3, SOT-223-3 and TDFN-3×3-8L Packages**

APPLICATIONS

Portable Electronic Device

Awww yeahhhhhh

APPLICATION INFORMATION

The SGM2212 is a low noise, fast transient response high performance LDO, it consumes only 80μA (TYP) quiescent current and provides 800mA output current. The SGM2212 provides the protection function for output overload, output short-circuit condition and overheating.

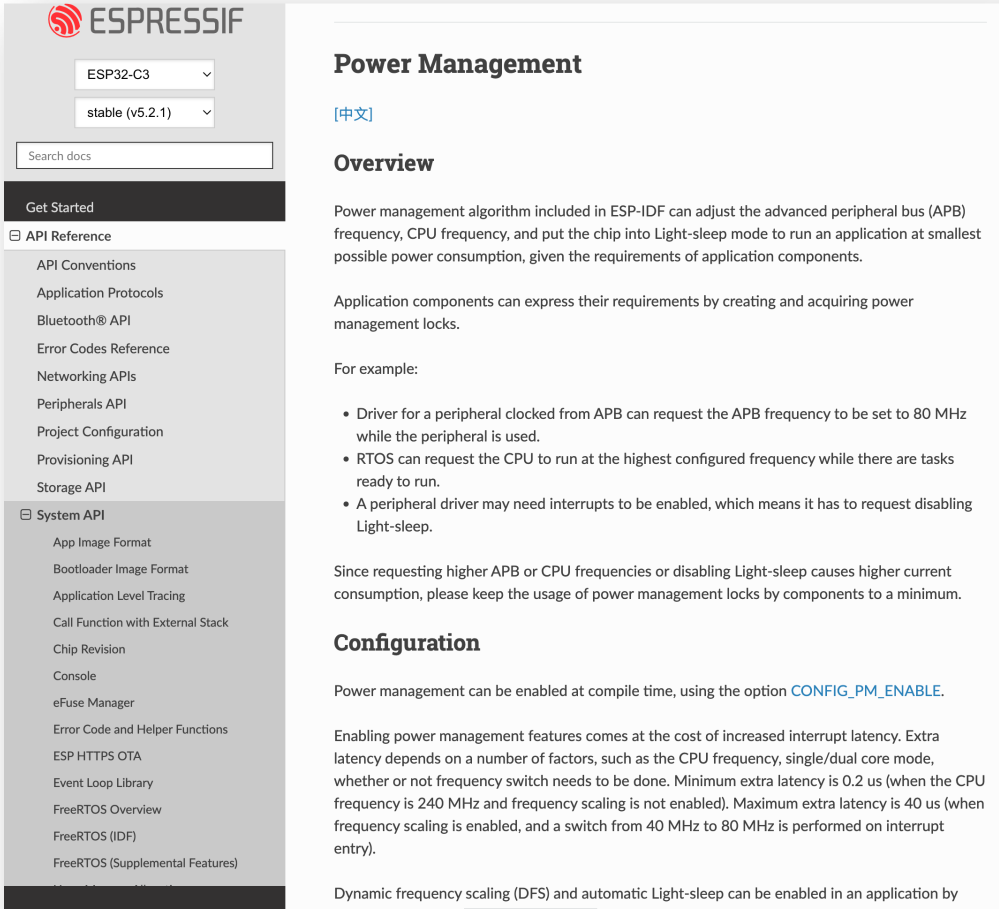
The SGM2212 is suitable for application which has noise sensitive circuit such as battery-powered equipment and smartphones.

https://www.sg-micro.com/uploads/soft/20220506/1651829970.pdf

So probably (haven’t tested yet)

- Between that bias network and the quiescent draw of that linear regulator, there’s going to be another ~150 uA of current in different modes.
- That can more than explain the deep sleep discrepancy
- Get us closer to ideal light sleep…in conjunction with Arduino finding… (only off by factor of 2)
- BUT ALL OF THESE ARE GUESS AND SHOULD BE TESTED

Must dig through docs and app notes!



Power Management

[中文]

Overview

Power management algorithm included in ESP-IDF can adjust the advanced peripheral bus (APB) frequency, CPU frequency, and put the chip into Light-sleep mode to run an application at smallest possible power consumption, given the requirements of application components.

Application components can express their requirements by creating and acquiring power management locks.

For example:

- Driver for a peripheral clocked from APB can request the APB frequency to be set to 80 MHz while the peripheral is used.
- RTOS can request the CPU to run at the highest configured frequency while there are tasks ready to run.
- A peripheral driver may need interrupts to be enabled, which means it has to request disabling Light-sleep.

Since requesting higher APB or CPU frequencies or disabling Light-sleep causes higher current consumption, please keep the usage of power management locks by components to a minimum.

Configuration

Power management can be enabled at compile time, using the option CONFIG_PM_ENABLE.

Enabling power management features comes at the cost of increased interrupt latency. Extra latency depends on a number of factors, such as the CPU frequency, single/dual core mode, whether or not frequency switch needs to be done. Minimum extra latency is 0.2 us (when the CPU frequency is 240 MHz and frequency scaling is not enabled). Maximum extra latency is 40 us (when frequency scaling is enabled, and a switch from 40 MHz to 80 MHz is performed on interrupt entry).

Dynamic frequency scaling (DFS) and automatic Light-sleep can be enabled in an application by

https://www.espressif.com/sites/default/files/documentation/ESP32-C3%20Wireless%20Adventure.pdf

CHAPTER 12

https://docs.espressif.com/projects/esp-idf/en/stable/esp32c3/api-reference/system/power_management.html

Some other sites about ESP family sleep

MIGHT HAVE SOME JUICY TIDBITS HERE

https://blog.voltaicsystems.com/how-to-put-an-esp32-into-deep-sleep/

https://randomnerdtutorials.com/esp32-timer-wake-up-deep-sleep/

Other Hungry Parts?

- I think this is going to be another part you'll have to wrestle with
- Anything that transmits (ESP WiFi, Cellular) does this by throwing away huge amounts of energy into the ether and expecting the receive parties to be picking up but a small fraction of it

Some preliminary research I did this morning

M5STACK SOLUTION | STORE | SOFTWARE | ABOUT US | DOCU

Module type	SIM7080G
Support CAT-M band	B1/B2/B3/B4/B5/B8/B12/B13/B14/B18/B19/B20/B25/B26/B27/B28/B66/B85
Support CAT-NB band	B1/B2/B3/B4/B5/B8/B12/B13/B18/B19/B20/B25/B26/B28/B66/B71/B85
Cat-M upstream and downstream speed	Uplink: 1119Kbps Downlink:589Kbps
NB-IoT upstream and downstream speed	Uplink: 150Kbps Downlink:136Kbps
RF Power Class	Class 5 (Typ. 21dbm)
SIM Card Slot Specifications	MicroSIM
Standby Operating Current	DC5V/46mA
Network current	DC5V/71mA
Communication Interface	UART: baud 115200 8N1
Net Weight	4.5g
Gross Weight	17.8g
Product Size	30.1*20.1*\5.5mm
Package Size	91.1*135.5mm

:/

https://shop.m5stack.com/products/m5stamp-cat-m-module-sim7080g

SIM7080G Module

- The internet and various semi-legitimate sources seem to suggest there is a sleep mode for this module…
- At that point, it consumes 1.2 mA (not great not terrible)

- Operating Frequency
  - GNSS L1: 1575.42±1.023MHz
  - GLONASS: 1597.5~1605.8 MHz
  - BeiDou: 1559.05~1563.14 MHz
  - Galileo L1: 1575.42±1.023MHz
- Refreshing frequency: 1 Hz (by default)
- GNSS data format: NMEA-0183
- GNSS antenna: active antenna

Other Parameters

- Power supply: 5V
- Logic level: 5V / 3.3V (Switchable via jumper caps)
- Overall current (idle mode): 39mA
- Module sole current (VBAT=3.8V)
- Idle mode: 10mA
- Sleep mode: 1.2mA
- PSM mode: 3.2uA
- eDRX mode: 0.59mA (eDRX=81.92s)
- Operating temperature: -40°C ~ 85°C
- Storage temperature: -45°C ~ 90°C
- Dimension: 30.5mm x 65mm

https://www.waveshare.com/wiki/SIM7080G_Cat-M/NB-IoT_HAT

Maker of Chip explains how to get there

Program it

Might need to use DTR pin??

3.3.2 RI and DTR Behavior

The RI pin description:

The RI pin can be used to interrupt output signal to inform the host controller such as application CPU. Before that, users must use AT command "AT+CFGRI=1" to enable this function.

Normally RI will keep high level until certain conditions such as receiving SMS, or a URC report coming, then it will output a low level pulse 120ms, in the end, it will become high level.

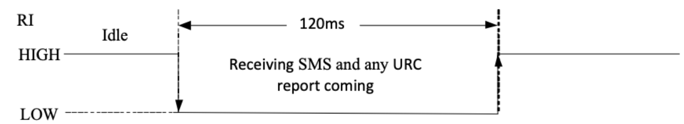


Figure 14: RI behaviour（SMS and URC report）

The DTR pin description:

After setting the AT command "AT+CSCLK=1", and then pulling up the DTR pin, SIM7080G will enter sleep mode when module is in idle mode. In sleep mode, the UART is unavailable. When SIM7080G enters sleep mode, pulling down DTR can wake up module.

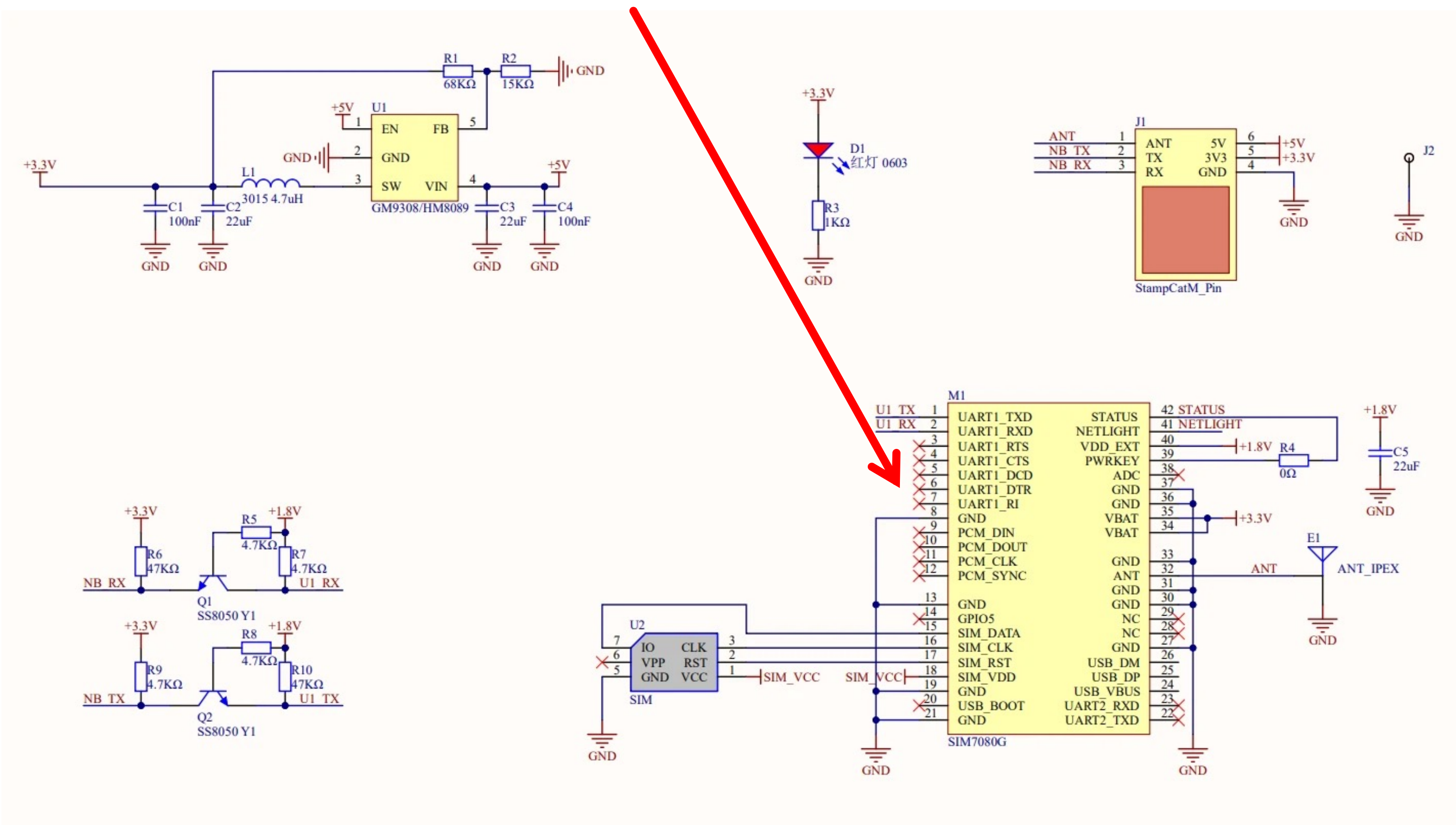
After setting the AT command "AT+CSCLK=0", SIM7080G will do nothing when the DTR pin is pulling up.

Note: For more details of AT commands about UART, please refer to document [1] and [20].

https://www.waveshare.com/w/upload/e/e7/SIM7080G_Hardware_Design_V1.03.pdf

DTR Pin?

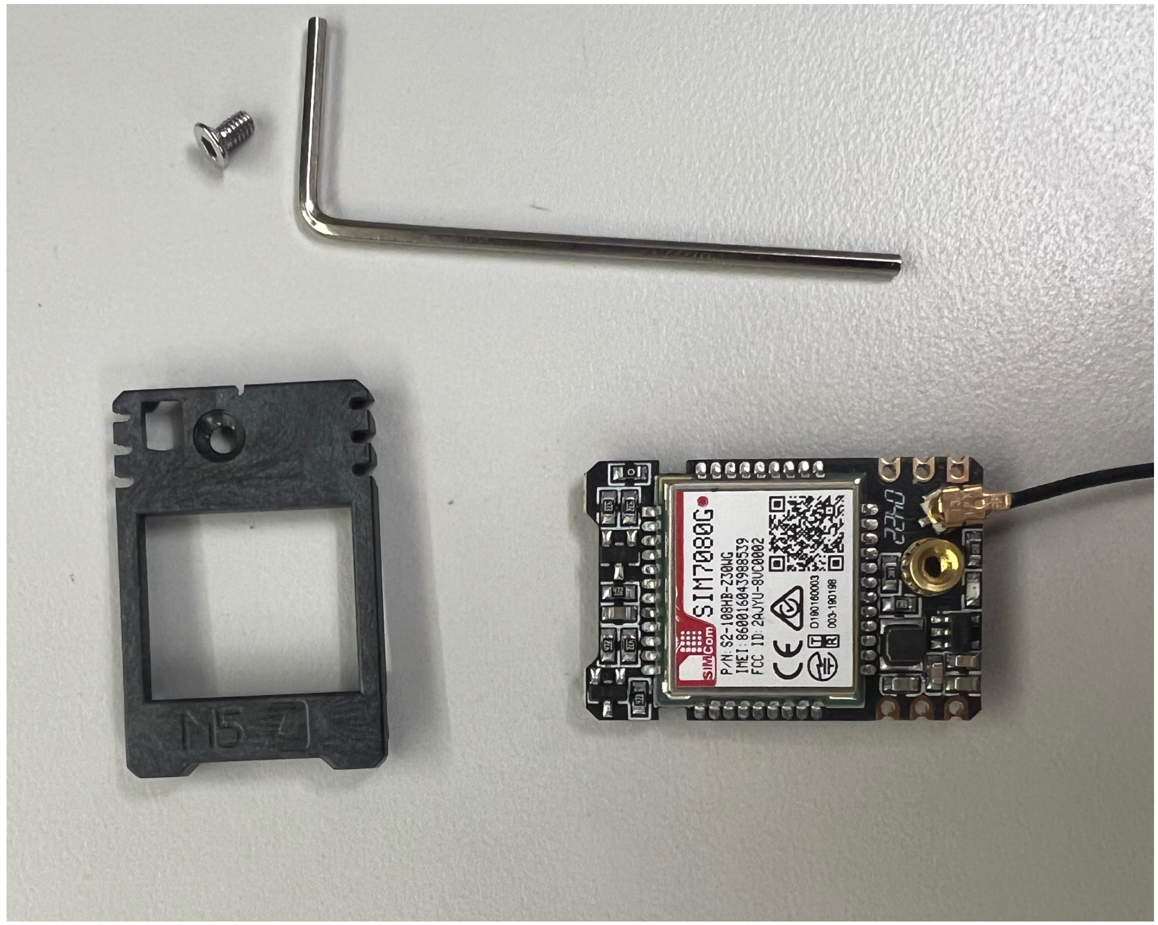
DTR



https://docs.m5stack.com/en/stamp/stamp_catm

Will probably need to solder a wire onto this thing

- That's ok,
- The pins are pretty accessible
- But this needs to be figured out!



Cell Module

- Assuming you can get to ~1.2 mA there's no guarantee that this will be good enough
- If ~1.2 mA proves to be too much for a resting current/power you may need to look into power-gating this component
- Can be done, but will take additional engineering as mentione earlier.

What about the other modules?

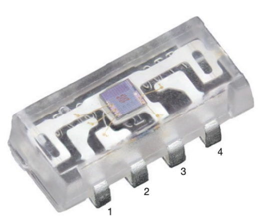
- I think, most of the other sensors are taking tiny sips of power (could be mistaken)
- All of these *SHOULD BE VERIFIED THOUGH!!!!!*

VISHAY. www.vishay.com

VEML7700

Vishay Semiconductors

High Accuracy Ambient Light Sensor With I²C Interface



Pinning

1: SCL

2: V_{DD}

3: GND

4: SDA

DESCRIPTION

VEML7700 is a high accuracy ambient light digital 16-bit resolution sensor in a miniature transparent 6.8 mm x 2.35 mm x 3.0 mm package. It includes a high sensitive photo diode, a low noise amplifier, a 16-bit A/D converter and supports an easy to use I²C bus communication interface.

The ambient light result is as digital value available.

FEATURES

- Package type: surface-mount
- Package: side view
- Dimensions (L x W x H in mm): 6.8 x 2.35 x 3.0
- Integrated modules: ambient light sensor (ALS)
- Supply voltage range V_{DD}: 2.5 V to 3.6 V
- Communication via I²C interface
- Floor life: 72 h, MSL 4, according to J-STD-020
- Low shut down current consumption: typ. 0.5 µA
- Material categorization: for definitions of compliance please see www.vishay.com/doc?99912

Pb Pb-free e4 RoHS COMPLIANT HALOGEN FREE GREEN (5-2008)

AMBIENT LIGHT FUNCTION

- 16-bit dynamic range for ambient light detection from 0 lx to about 140 klx with resolution down to 0.0042 lx/ct, supports low transmittance (dark) lens design
- 100 Hz and 120 Hz flicker noise rejection
- Excellent temperature compensation
- High dynamic detection resolution
- Software shutdown mode control

APPLICATIONS

- Ambient light sensor for backlight dimming of e.g. TV displays, smart phones, touch phones, PDA, GPS
- Ambient light sensor for industrial on- / off-lighting operation
- Optical switch for consumer, computing, and industrial devices and displays

VISHAY. www.vishay.com

VEML7700

Vishay Semiconductors

ABSOLUTE MAXIMUM RATINGS (T_{amb} = 25 °C, unless otherwise specified)

PARAMETER	TEST CONDITION	SYMBOL	MIN.	MAX.	UNIT
Supply voltage		V_{DD}	0	4	V
Operation temperature range		T_{amb}	-25	+85	°C
Storage temperature range		T_{stg}	-25	+85	°C
Total power dissipation	$T_{amb} \le 25$ °C	P_{tot}	-	50	mW
Junction temperature		T_j	-	100	°C

BASIC CHARACTERISTICS (T_{amb} = 25 °C, unless otherwise specified)

PARAMETER	TEST CONDITION	SYMBOL	MIN.	TYP.	MAX.	UNIT
Supply voltage		V_{DD}	2.5	3.3	3.6	V
Shut down current (2)	V_{DD} is 3.3 V	I_{sd}	-	0.5	-	µA
Operation mode current (1)	V_{DD} is 3.3 V, PSM = 11, refresh time 4100 ms	I_{DD}	-	2	-	µA
	V_{DD} is 3.3 V, PSM = 00, refresh time 600 ms	I_{DD}	-	8	-	µA
	V_{DD} is 3.3 V, PSM_EN = 0, refresh time 100 ms	I_{DD}		45	-	µA
I²C clock rate range		f_{SCL}	10	-	400	kHz
I²C bus input H-level range	V_{DD} is 3.3 V	V_{ih}	1.3	-	3.6	V
I²C bus input L-level range	V_{DD} is 3.3 V	V_{il}	-0.3	-	0.4	V
Digital current out (low, current sink)		I_{ol}	3	-	-	mA
Digital resolution (LSB count) (3)	With ALS_GAIN = x 2, ALS_IT = 800 ms		-	0.0042	-	lx/step
Detectable maximum illuminance (3)	With ALS_GAIN = x 1/8, ALS_IT = 25 ms	$E_{V\,max.}$	-	140 000	-	lx
Dark offset (2)	With ALS_GAIN = x 2, ALS_IT = 800 ms		-	3	-	step

Notes

(1) Light source: white LED

(2) Light conditions: dark

(3) Refer to the application note for resolution table at different settings

Power Budget!

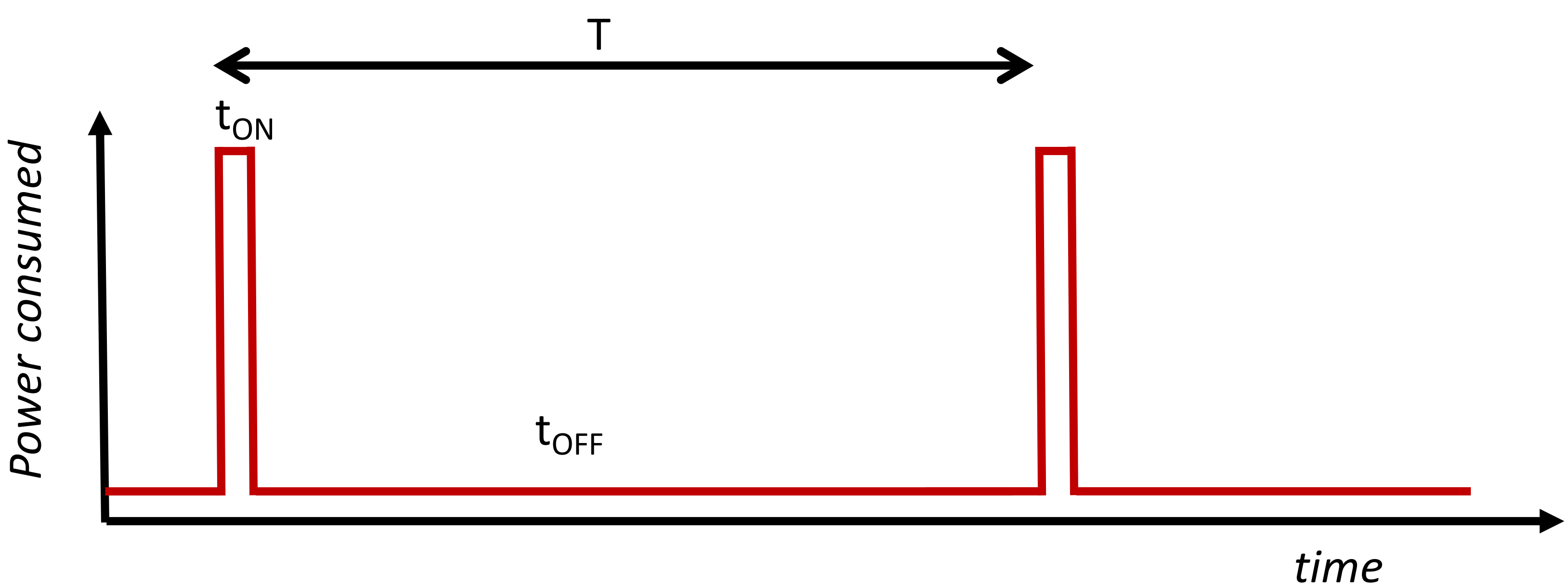
Power Budget

- Now that you're getting an idea of the power of your components you need to engineer with that.
- We'll be relying on placing devices into different power modes in loops whenever possible to make them utilize less energy
- These patterns of power consumption can then be roughly time-averaged and help us determine how much our average consumption will be.
- This can then help us figure out how much we need to be putting into the system (from solar, etc…)
- And how much we need to store up for when we don't have power input (at night for example)

Duty Cycling

- The core of this budgeting will be duty-cycing the components when possible
- Turn on in bursts, do what is needed, and turn off otherwise.
- The ratio of the "ON" to overall is called the duty cycle

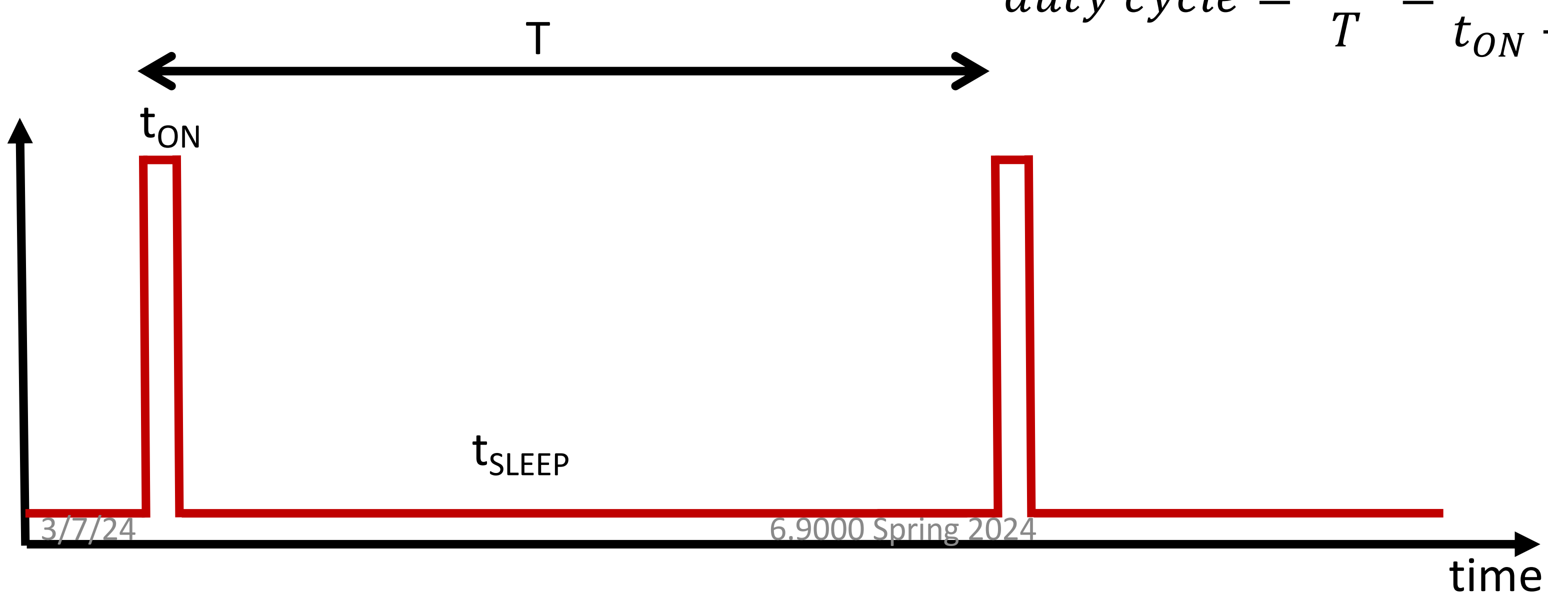
$$duty\ cycle = \frac{t_{ON}}{T} = \frac{t_{ON}}{t_{ON} + t_{OFF}} \cdot 100\%$$



Duty Cycling

- Example:
  - We have an ESP32 powered with a 200 mAh coin-cell battery. Convert to 3.3V at 100% efficiency
  - We want it to run for one month
  - We want to take measurements and send them over WiFi, but **to measure and connect to WiFi requires a t_{ON}= 10 sec.**
  - How often can we make measurements/send them to the server if we want the coin cell to last for one month?

$$duty\ cycle = \frac{t_{ON}}{T} = \frac{t_{ON}}{t_{ON} + t_{SLEEP}} \cdot 100\%$$



Duty Cycling

- Adjust Duty Cycle to achieve Average Current:

$$= \frac{t_{on} \cdot i_{on} + t_{sleep} \cdot i_{sleep}}{T} = 0.278 \text{ mA}$$

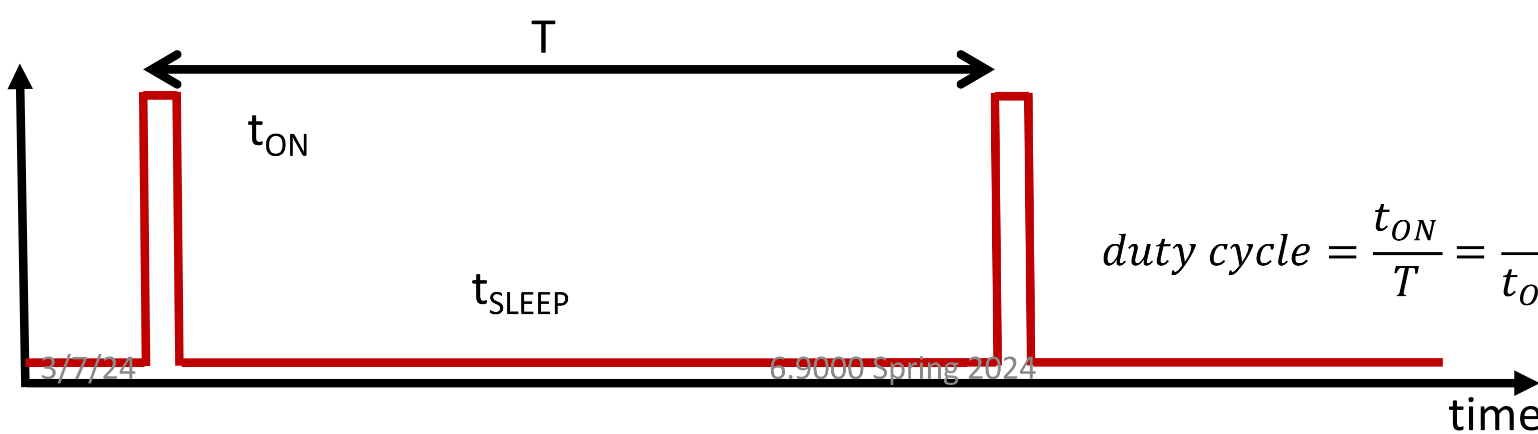
Plug in known values and rewrite in terms of things we're solving for:

$$= \frac{10\text{s} \cdot 150\text{mA} + t_{sleep} \cdot 0.005\text{mA}}{10\text{s} + t_{sleep}} = 0.278 \text{ mA}$$

Solve for t_{sleep}:

$$t_{sleep} \approx 5395\text{s}$$

So we'd need to:
Come on for 10 seconds
Sleep for 5400 seconds
Duty Cycle: 0.185%



$$duty\ cycle = \frac{t_{ON}}{T} = \frac{t_{ON}}{t_{ON} + t_{OFF}} \cdot 100\%$$

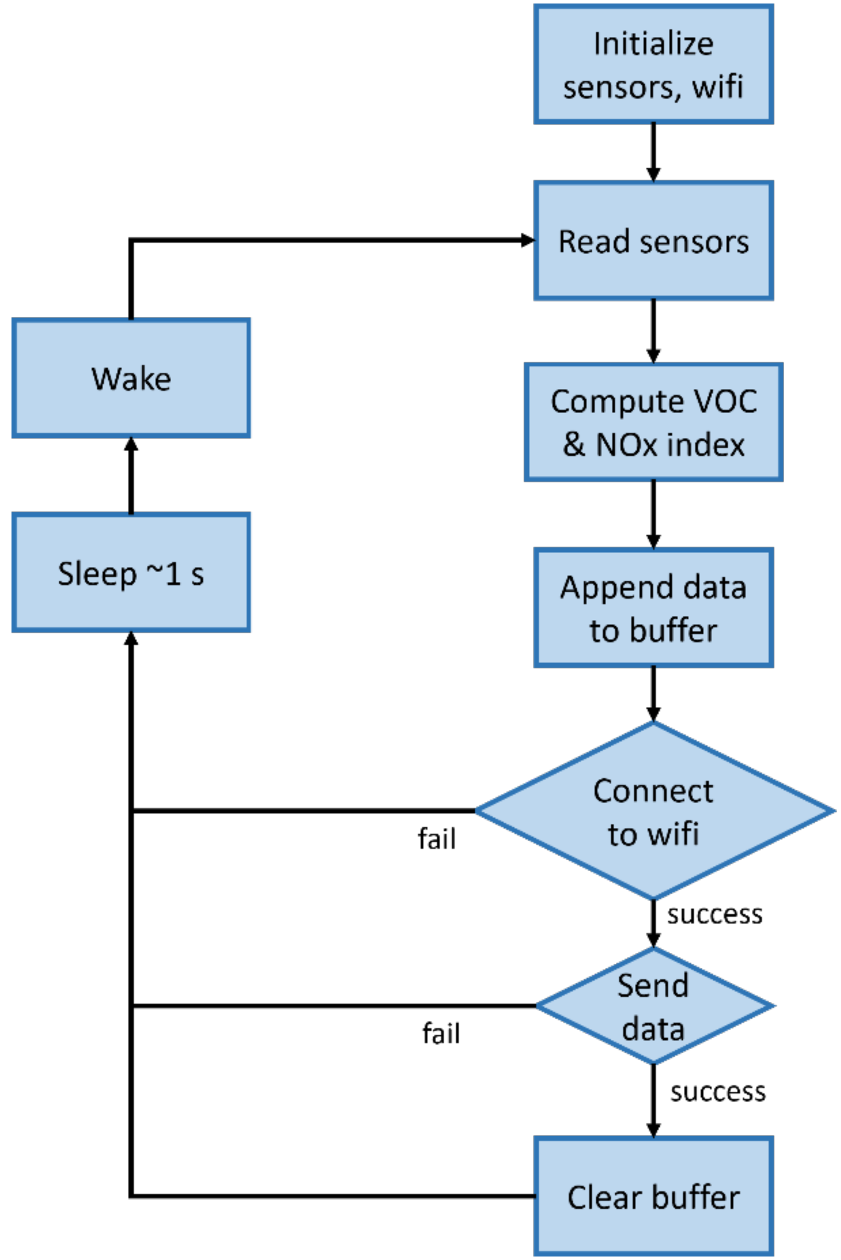
Power Budget

- Now that you're getting an idea of the power of your components you need to engineer with that.

- The first step is to identify your entire system's flowchart of operation.
- Functionality must come first since we need this system to do a thing so you should start with your "ideal" flowchart of operations

Power Budget

- An example from last year using the dev boards
- Very similar to some functionality you'll need this year.
- For each state go through and determine what is "on" what is "off" and all the shades of grey in between



Power Budget

- Can do things in either mW or mA…often see mA if you can assume a constant system voltage
- Determine the draw (measured ideal or estimated less-ideal) of each component in the mode it should be in in that state

A	B
Power budget	
	Read sensors
MCU subsystem	
ESP32C3	28.00
LED	5.00
	33.00
Sensor subsystem	
SGP41	3.00
SHTC3	0.90
LED	5.00
I2C	0.66
	9.56
Power subsystem	
MCP73871	0.03
AP7361C	0.06
LEDs	5.00
Therm	0.05
PROG1	3.70
	8.84
cycle time (ms)	**51.40**
10000	70
	3.6
	0.4%

- Fill out the budget as you go along

Power budget					
	Read sensors	Math	Xmit data	Sleep	
MCU subsystem					
ESP32C3	28.00	28.00	345.00	0.13	
LED	5.00	5.00	5.00	5.00	
	33.00	33.00	350.00	5.13	
Sensor subsystem					
SGP41	3.00	0.03	0.03	0.03	
SHTC3	0.90	0.07	0.07	0.07	
LED	5.00	5.00	5.00	5.00	
I2C	0.66	0.33	0.33	0.00	
	9.56	5.43	5.43	5.10	
Power subsystem					
MCP73871	0.03	0.03	0.03	0.03	
AP7361C	0.06	0.06	0.06	0.06	
LEDs	5.00	5.00	5.00	5.00	
Therm	0.05	0.05	0.05	0.05	
PROG1	3.70	3.70	3.70	3.70	
	8.84	8.84	8.84	8.84	
cycle time (ms)	51.40	47.27	364.27	19.07	total current (mA)
10000	70	100	200	9630	duration (ms)
	3.6	4.7	72.9	183.7	charge (mA-sec)
	1.4%	1.8%	27.5%	69.4%	% energy
			Average current/cycle	26.49	mA
			Battery capacity	2200	mA-h
			Lifetime	83.1	h

- Identify problem regions and adjust
- LEDs were always on on these dev boards…

Power budget					
	Read sensors	Math	Xmit data	Sleep	
MCU subsystem					
ESP32C3	28.00	28.00	345.00	0.13	
LED	5.00	0.00	0.00	0.00	
	33.00	28.00	345.00	0.13	
Sensor subsystem					
SGP41	3.00	0.03	0.03	0.03	
SHTC3	0.90	0.07	0.07	0.07	
LED	5.00	0.00	0.00	0.00	
I2C	0.66	0.33	0.33	0.00	
	9.56	0.43	0.43	0.10	
Power subsystem					
MCP73871	0.03	0.03	0.03	0.03	
AP7361C	0.06	0.06	0.06	0.06	
LEDs	5.00	5.00	5.00	5.00	
Therm	0.05	0.05	0.05	0.05	
PROG1	3.70	3.70	3.70	3.70	
	8.84	8.84	8.84	8.84	
cycle time (ms)	51.40	37.27	354.27	9.07	total current (mA)
10000	70	100	200	9630	duration (ms)
	3.6	3.7	70.9	87.4	charge (mA-sec)
	2.2%	2.3%	42.8%	52.8%	% energy
			Average current/cycle	16.56	mA
			Battery capacity	2200	mA-h
			Lifetime	132.9	h

Focus

- You need to figure out what your average current draw is going to be!
- That number must be less than the average that your source can provide
- And ideally at least several factors smaller than what the source can provide to account for life sucking and never working out how you want

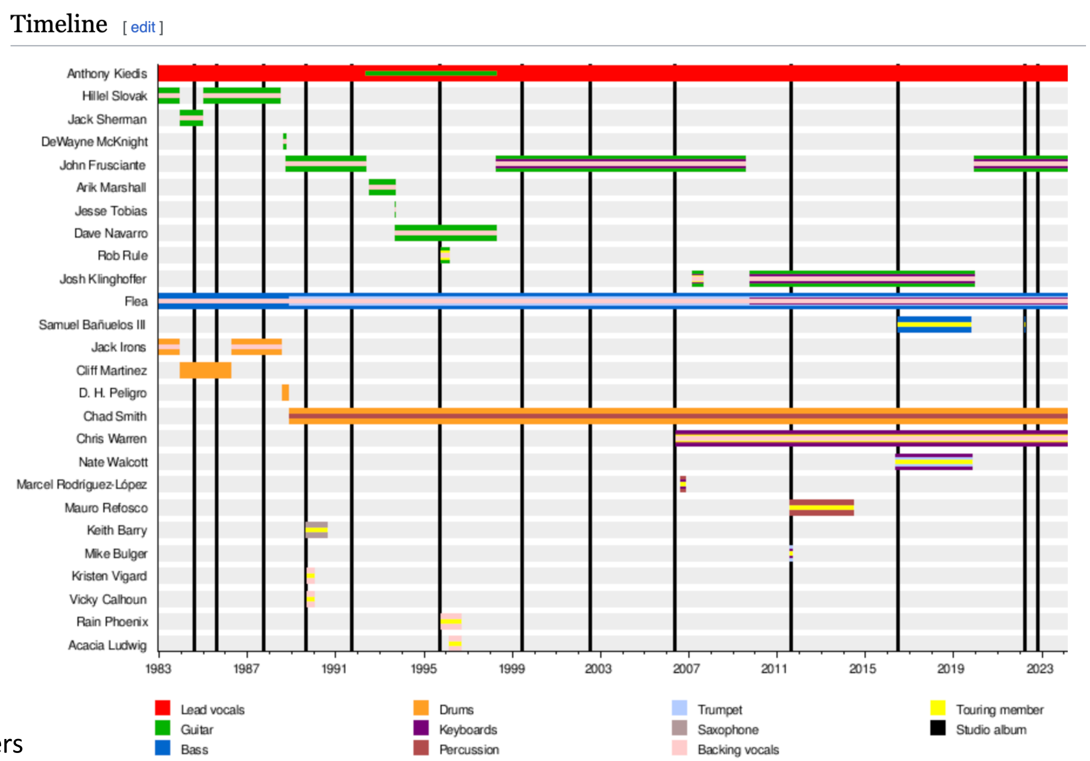
Average current/cycle	16.56	mA
Battery capacity	2200	mA-h
Lifetime	132.9	h

- Lifetime can also give you a sense of how long you can go when it is dark out or cloudy or whatever

Timeline

- When you have many components coming on in various states another way to draw this out might be like what they do to show membership of bands on Wikipedia
- Sort of like a gant chart

- There is free software



https://en.wikipedia.org/wiki/List_of_Red_Hot_Chili_Peppers_band_members

Timeline

- But we also want to see actual numbers calculated
- And a spreadsheet will likely be the best choice for this.

A	B	C
Power budget		
	Read sensors	Math
MCU subsystem		
ESP32C3	28.00	28.00
LED	5.00	5.00
	33.00	33.00
Sensor subsystem		
SGP41	3.00	0.03
SHTC3	0.90	0.07
LED	5.00	5.00
I2C	0.66	0.33
	9.56	5.43
Power subsystem		
MCP73871	0.03	0.03
AP7361C	0.06	0.06
LEDs	5.00	5.00
Therm	0.05	0.05
PROG1	3.70	3.70
	8.84	8.84
cycle time (ms)	51.40	47.27
10000	70	100
	3.6	4.7
	0.4%	0.5%

https://en.wikipedia.org/wiki/List_of_Red_Hot_Chili_Peppers_band_members

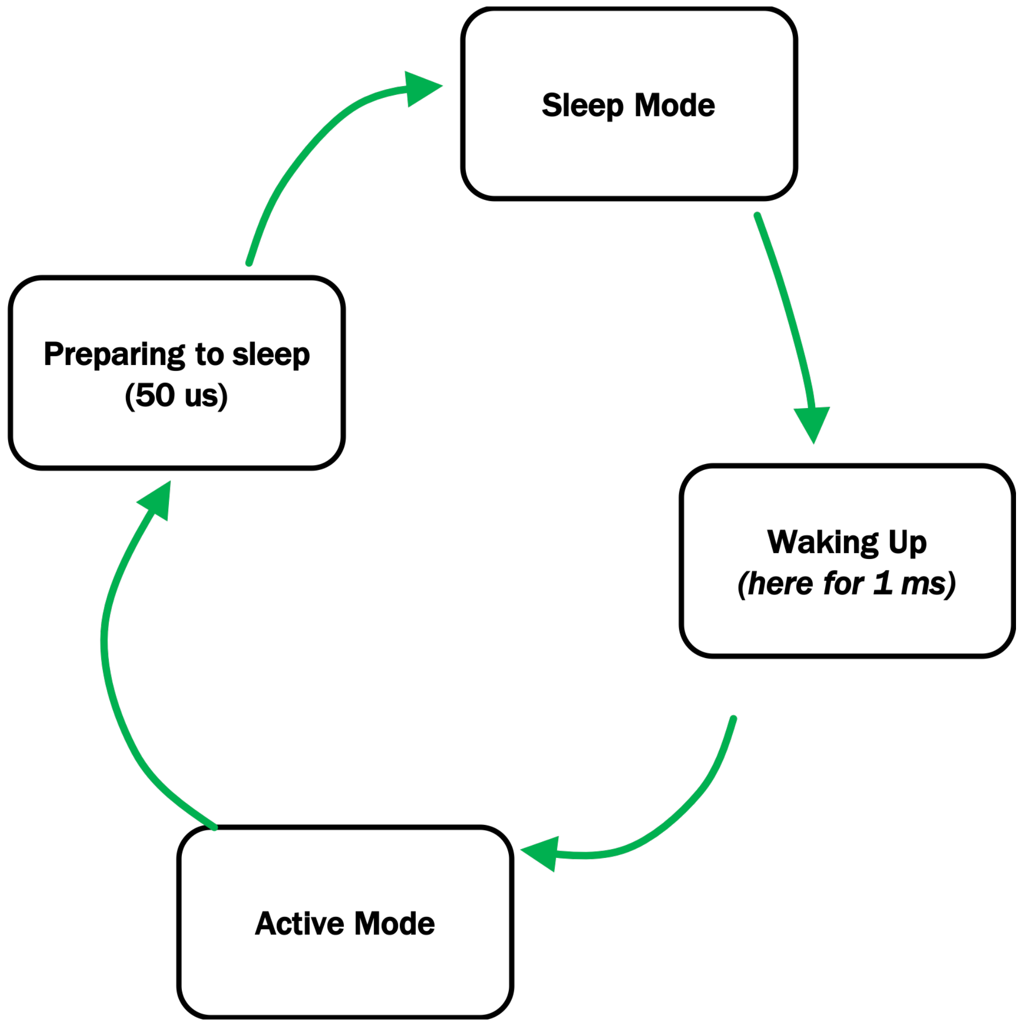
Your System

- Your system is going to have a much more complicated flow chart

- Likely inner loops…outer loops.
- So you'll be duty-cycling the duty-cyles (and that's fine)

Also Keep in Mind Start-Up Costs

- If you send a device to sleep, it cannot just immediately go to sleep and then wake up and start doing stuff again. There will be shutdown and startup costs!
- In modern processors, this is usually 10,000's of clock cycles (meaning potentially milliseconds lost). So if you're going to sleep you better make sure it is worth it

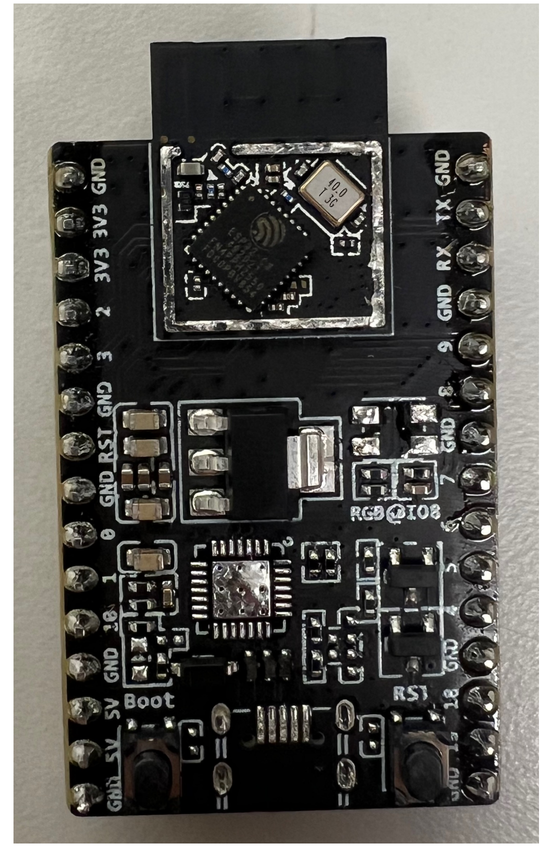


Start-Up Costs

- The ESP can come back pretty quick (microseconds from light sleep…100’s of microseconds from deep I believe)
- Certain functionalities on the ESP may be much more problematic:
  - Turning on WiFi/getting connections is costly in terms of time (and power)
  - Sending a given packet is relatively cheap
  - May motivate building up data if transferring it over WiFi, though that’s limited via other things.

Start-Up Costs

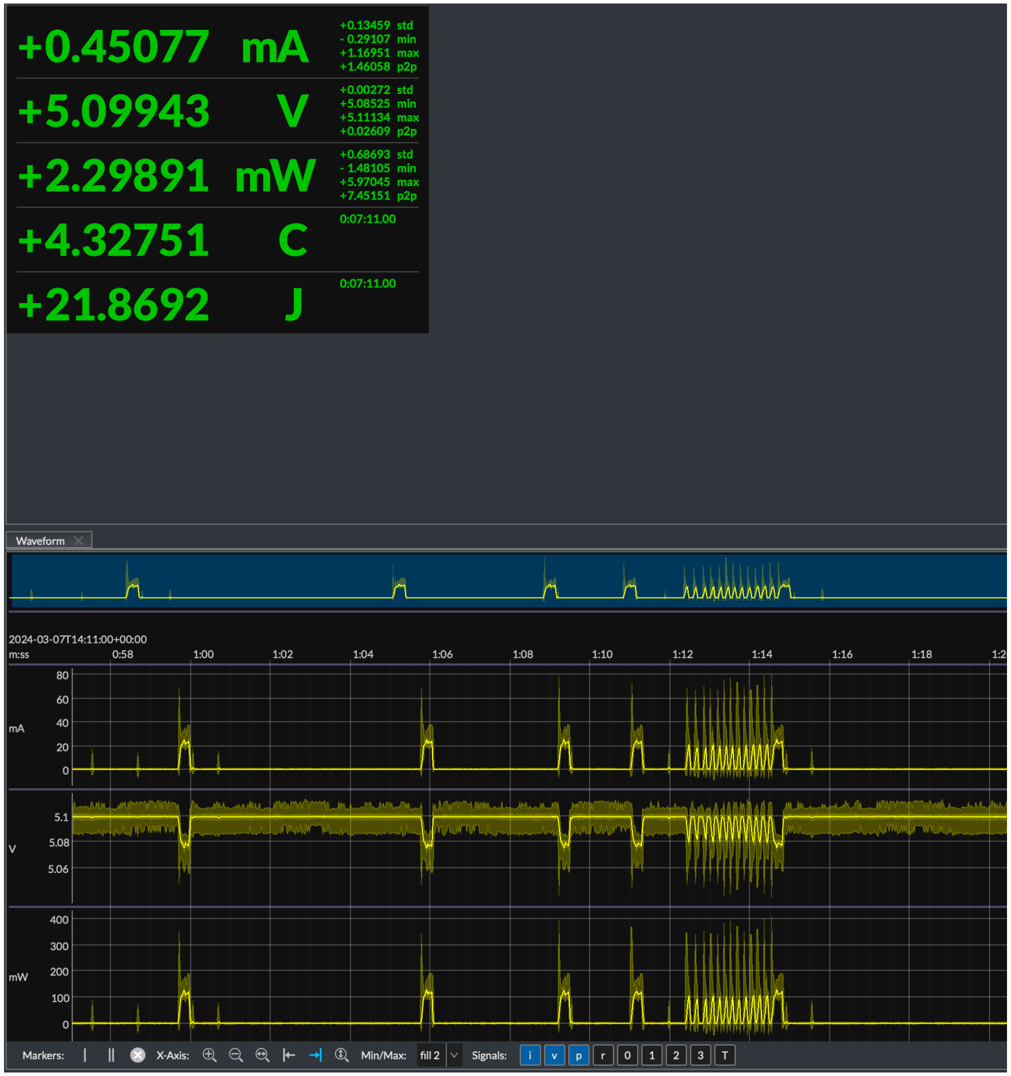
- The ESP can come back pretty quick (microseconds from light sleep…100’s of microseconds from deep I believe)
- Certain functionalities on the ESP may be much more problematic:
  - Turning on WiFi/getting connections is costly in terms of time (and power)
  - Sending a given packet is relatively cheap
  - May motivate building up data if transferring it over WiFi, though that’s limited via other things.



Start-Up Costs: Cell Module

- It takes quite a few minutes for this thing to wake up and find a network successfully…that will need to be figured into everything
- Less well-documented.
- Will need to be studied and measured

No matter what you will then need to get actual data

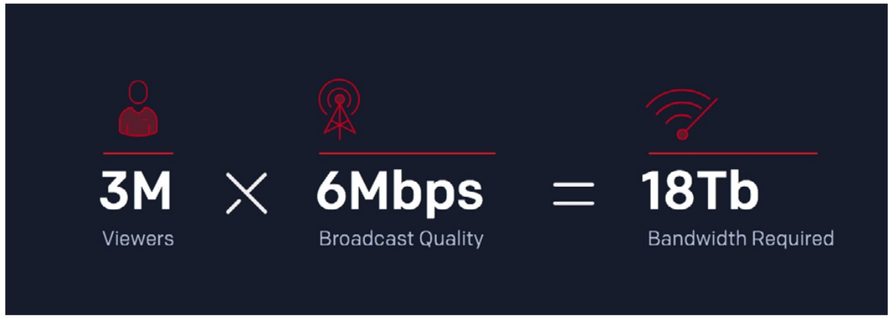


- Compare your estimated power budget records to actual corrected data.
- Compare and contrast and find where the issues are!

Making Code More Efficient

- The faster your code runs, the better it is:
  - Can turn off quicker, or can go to the next task quicker…
  - Either way bleeding less power just by being on
- This applies to on the server as well.
  - Just because the power to run your python comes from a power plant in Canada or New Jersey doesn't mean we should ignore it
  - Data centers consume an ever-increasing portion of the US energy consumption in the US (~2.0% currently)

Twitch



2018 numbers

- Let’s assume Twitch pays 0.1 cents per GB transfer at all levels (conservative…probably more)
- Bandwidth costs are therefore about $2.25 per second or $8,100 per hour to stream to all its users.
- Average Twitch user watches 20 + hours/week…(I know…)
- Streaming costs for Twitch are $162,000 per week just to pay for bandwidth
- ~$8.42 Million a year in bandwidth costs (conservative)

*only bandwidth cost…nothing else

https://www.xilinx.com/publications/powered-by-xilinx/Twitch-Case-Study.pdf

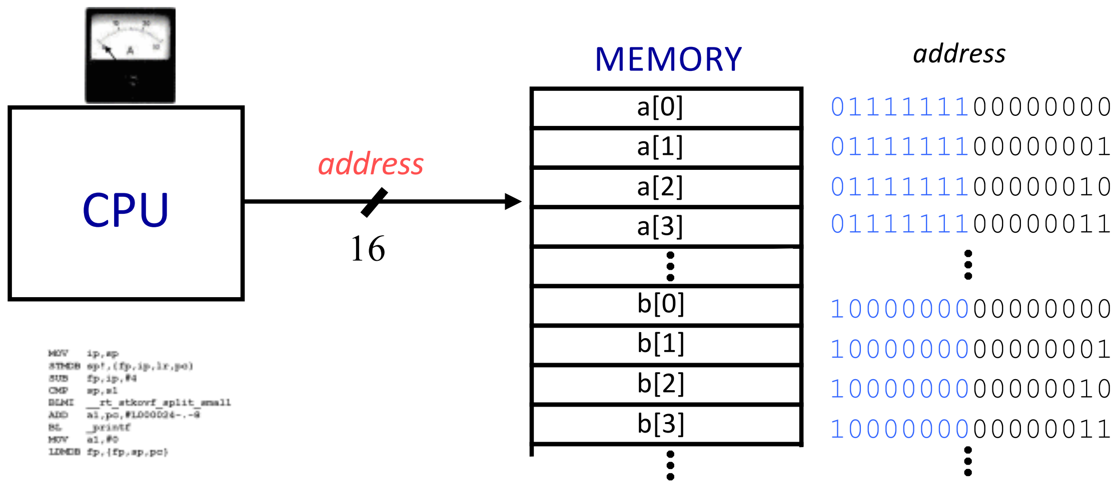
Twitch

- In 2016-2018, Twitch spent millions migrating its hardware and algorithms over to a new platform for a 25% speed-up
- Might not seem like much, but that'll immediately save ~$2million a year just on streaming
- Ignoring support costs, etc…

https://www.xilinx.com/publications/powered-by-xilinx/Twitch-Case-Study.pdf

Even in C/C++ you can do smart/dumb things

- Even how we write our C/C++ can matter!
- Though imperceptibly with the tools we are using!



Here's some C:

```
float a [256], b[256];
float pi= 3.14;

for (i = 0; i < 255; i++) {
   a[i] = sin(pi * i /256);
   b[i] = cos(pi * i /256);
}
```

Address bus will undergo:
512(8)+2+4+8+16+32+64+128+256
= 4607 bit transitions

```
float a [256], b[256];
float pi= 3.14;

for (i = 0; i < 255; i++) {a[i] = sin(pi * i /256);}
for (i = 0; i < 255; i++) {b[i] = cos(pi * i /256);}
```

Address bus will undergo:
2(8)+2(2+4+8+16+32+64+128+256)
= 1030 transitions

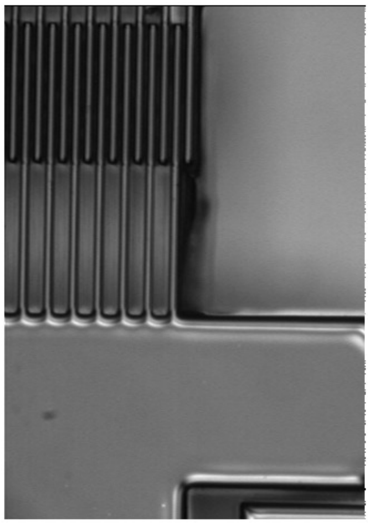
Energy Harvesting?

- I think solar is the consensus for how to harvest energy, but the fact that we can get the ESP32 down into the uW of power does open up a lot of possibilities in the longer term.

- Energy Scavenging

Energy Scavenging: Mechanical

MEMS Generator



Jose Mur Miranda/
Jeff Lang

Vibration-to-Electric Conversion

~ 10mW

Power Harvesting Shoes

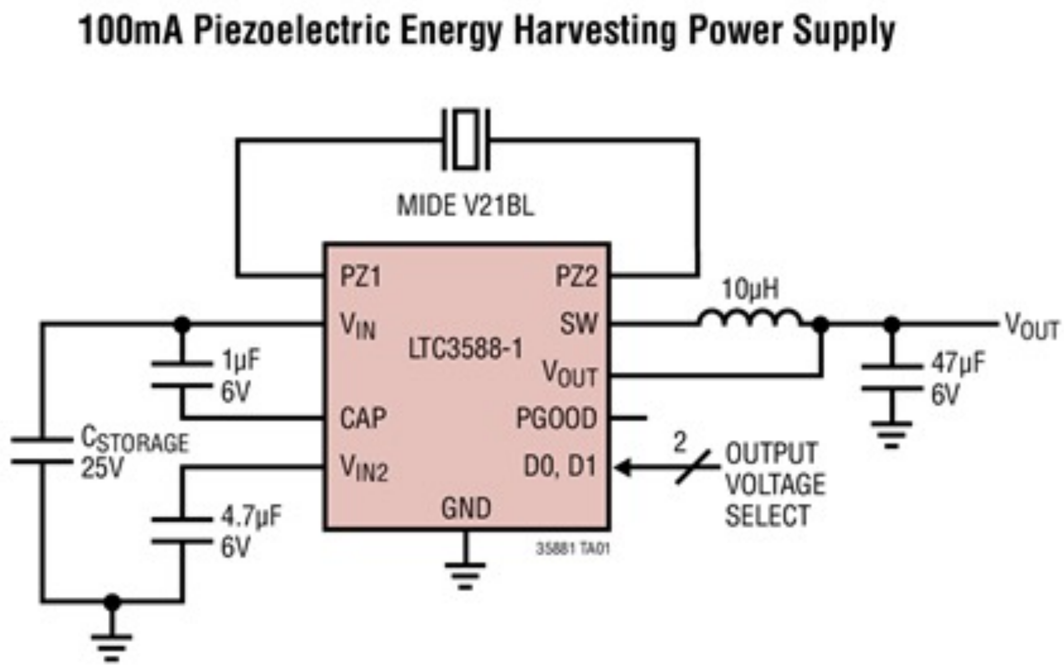
Joe Paradiso
(Media Lab)

After 3-6 steps, it provides 3 mA for 0.5 sec

~10mW

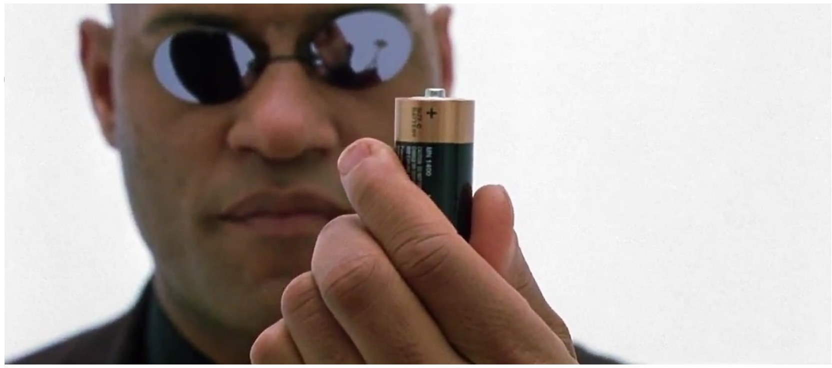
Energy Scavenging: Mechanical

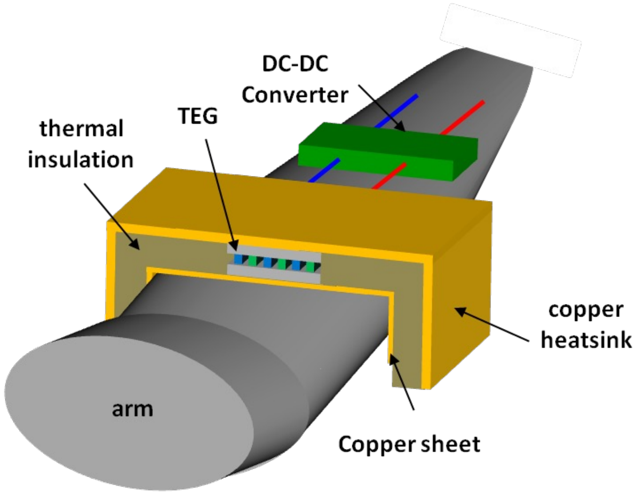
Optimized for :
collecting 60 Hz vibrations at low sub-g accelerations!

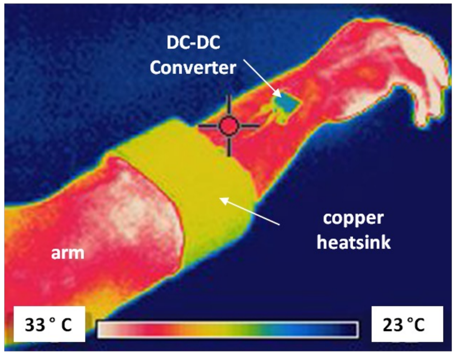


LTC3588

Low-Profile Wearable Body-Powere Thermoelectric Generator





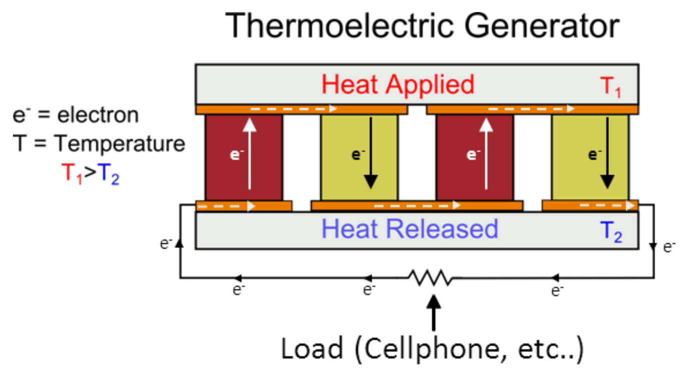


- Low profile, lightweight, conformal.
- Utilization of small temperature difference
- Utilization of natural convection for cooling

Credit: Krishna Settaluri MIT ‘2010

Energy harvesting

- Thermo-electric generator
  - Thermoelectric material converts temperature difference into voltage

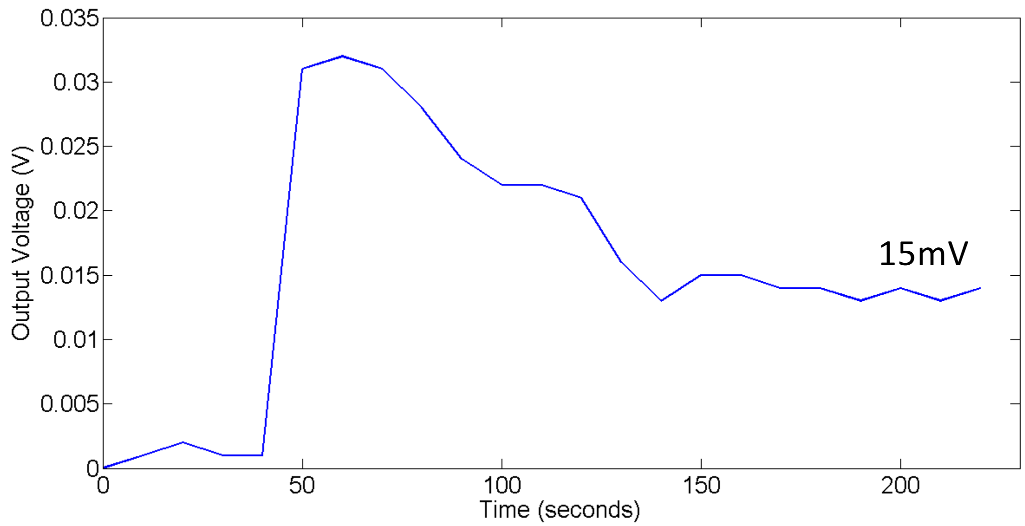


40 K temp difference
1.8 V @ 368 mA

https://www.adafruit.com/products/700
http://electronicdesign.com/content/content/73937/73937-fig2.gif

Experimental Results

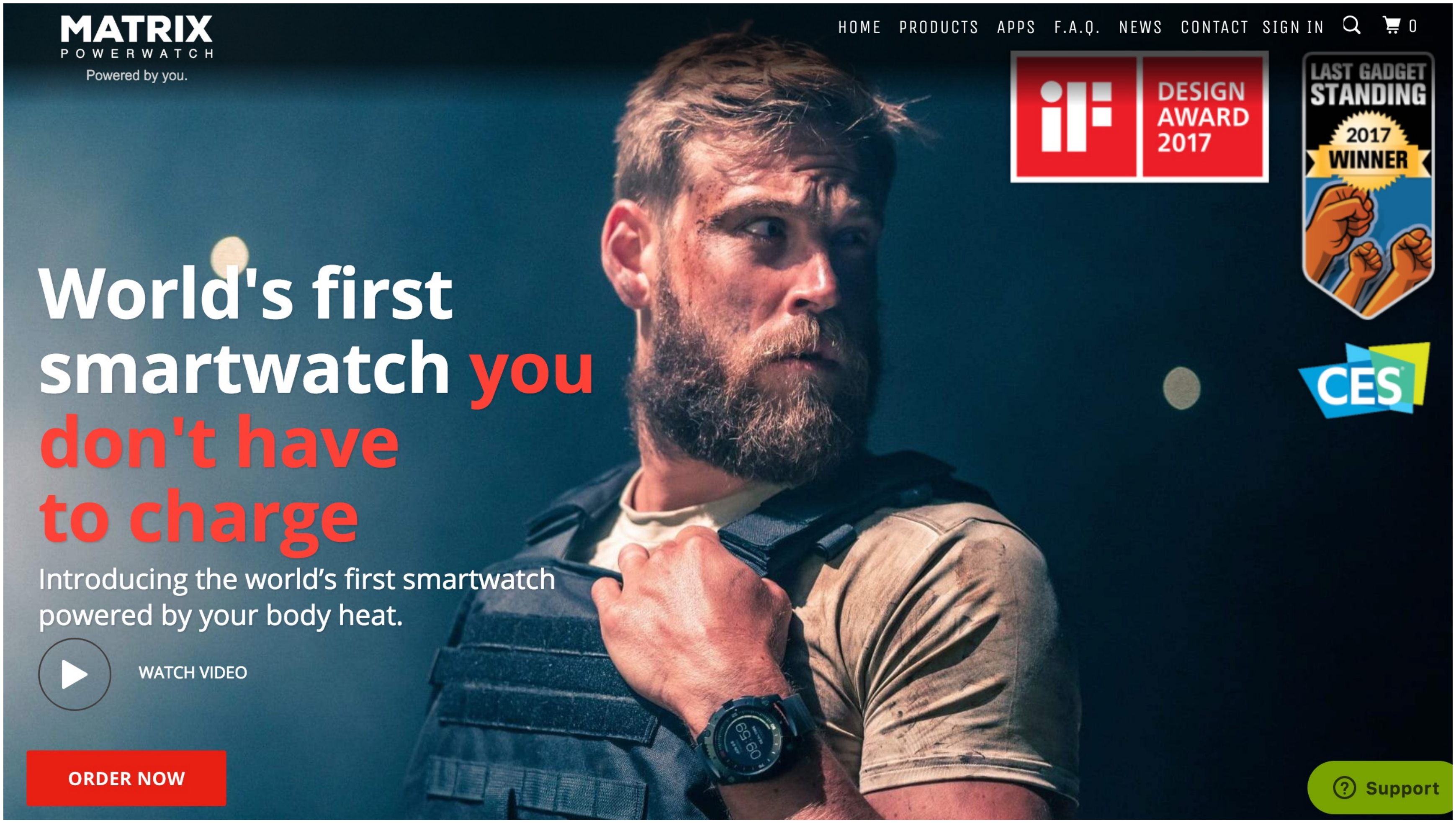
16 TEG Islands (2 TEG modules)



Optimal Electrical Load Resistance	33Ω (20Ω theoretical)
Optimized Power	11µW

Credit: Krishna Settaluri MIT '2010

Apparently Powered by Body Heat



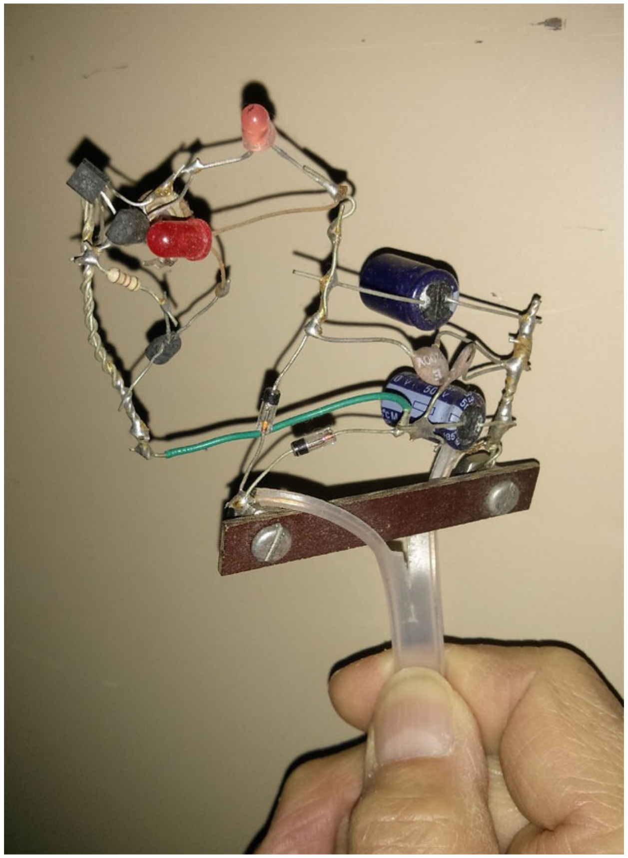
https://www.powerwatch.com/

Ambient RF

Prudential Center
FM Stations:
WZLX 100.7, WBMX 104.1, WMJX 106.7, and WXKS-FM 107.9, WBOS 92.9, WBQT 96.9, and WROR-FM 105.7.

Power output:
22,000 watts

Recovered:
~ 200 uW



Ambient RF

TABLE V
HARVESTERS CHARGE AND DISCHARGE TIMES (t_c, t_d, RESPECTIVELY) FOR A SPECIFIED LOAD

Band	Wire					Tape				
	t_c (s) load independent	t_d (s) load dependant	t_{cycle} (s) load dependant	$P_{dc}(t_d)$ (μW)	$P_{dc}(t_{cycle})$ (μW)	t_c (s) load independent	t_d (s) load dependant	t_{cycle} (s) load dependant	$P_{dc}(t_d)$ (μW)	$P_{dc}(t_{cycle})$ (μW)
DTV	26	12	38	9.6	3	14	18	32	8.2	3.6
GSM900	14	10	24	11.5	4.8	8	13	21	14.4	5.5
GSM1800	43	15	58	7.7	2	22	27	49	5.2	2.4
3G v2	167	3	170	38.4	0.7	96	5	101	1.2	1.1
Multiband ΣV	43	7	50	66	2.3	-	-	-	-	-
Multiband ΣI	55	5	60	92.2	2	-	-	-	-	-

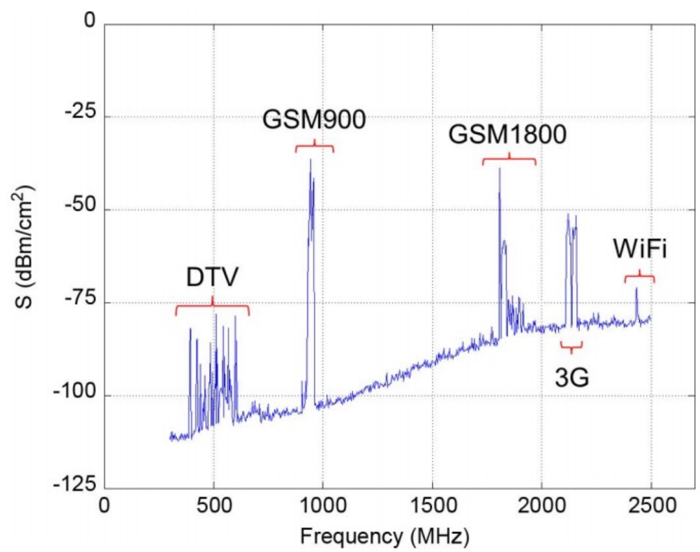


Fig. 1. Input RF power density measurements outside the Northfields London Underground station.

Energy to be had in the signals that are all around us

Previously not practical since even simple circuits used lots of power, but as transistors have scaled…gotten reasonable….couple with ASICs and you could be in business

Ambient RF Energy Harvesting in Urban and Semi-Urban Environments, Manuel Piñuela, Student Member, IEEE, Paul D. Mitcheson, Senior Member, IEEE, and Stepan Lucyszyn, Senior Member, IEEE 2013

WiFi Harvesting

Nature, 2019

LETTER

https://doi.org/10.1038/s41586-019-0892-1

Two-dimensional MoS_2-enabled flexible rectenna for Wi-Fi-band wireless energy harvesting

Xu Zhang[1], Jesús Grajal[2], Jose Luis Vazquez-Roy[3], Ujwal Radhakrishna[1], Xiaoxue Wang[4], Winston Chern[1], Lin Zhou[1], Yuxuan Lin[1], Pin-Chun Shen[1], Xiang Ji[1], Xi Ling[5], Ahmad Zubair[1], Yuhao Zhang[1], Han Wang[6], Madan Dubey[7], Jing Kong[1], Mildred Dresselhaus[1,8] & Tomás Palacios[1]*

The mechanical and electronic properties of two-dimensional materials make them promising for use in flexible electronics[1–3]. Their atomic thickness and large-scale synthesis capability could enable the development of 'smart skin'[1,3–5], which could transform ordinary objects into an intelligent distributed sensor network[6]. However, although many important components of such a distributed electronic system have already been demonstrated (for example, transistors, sensors and memory devices based on two-dimensional materials[1,2,4,7]), an efficient, flexible and always-on energy-harvesting solution, which is indispensable for self-powered systems, is still missing. Electromagnetic radiation from Wi-Fi systems operating at 2.4 and 5.9 gigahertz[8] is becoming increasingly ubiquitous and would be ideal to harvest for powering future distributed electronics. However, the high frequencies used for Wi-Fi communications have remained elusive to radiofrequency harvesters (that is, rectennas) made of flexible semiconductors owing to their limited transport properties[9–12]. Here we demonstrate an atomically thin and flexible rectenna based on a MoS_2 semiconducting–metallic-phase heterojunction with a cutoff frequency of 10 gigahertz, which represents an improvement in speed of roughly one order of magnitude compared with current state-of-the-art flexible rectifiers[9–12]. This flexible MoS_2-based rectifier operates up to the X-band[8] (8 to 12 gigahertz) and covers most of the unlicensed industrial, scientific and medical radio band, including the Wi-Fi channels. By integrating the ultrafast MoS_2 rectifier with a flexible Wi-Fi-band antenna, we fabricate a fully flexible and integrated rectenna that achieves wireless energy harvesting of electromagnetic radiation in the Wi-Fi band with zero external bias (battery-free). Moreover, our MoS_2 rectifier acts as a flexible mixer, realizing frequency conversion beyond 10 gigahertz. This work provides a universal energy-harvesting building block that can be integrated with various flexible electronic systems.

that exhibit a cutoff frequency of 1.6 GHz[11]. However, the random distribution of particle sizes and separation distances results in a low on/off current ratio and unreliable turn-on voltage, which deteriorates their rectification performance and reliability for large-scale production. In addition, almost all the above methods use a vertical structure to increase the effective device area and thereby to reach a sufficiently high on-current, I_{on}. However, in such a structure, the top and bottom electrodes of the diode inevitably form a parallel-plate capacitor with large parasitic capacitance, which considerably hinders its high-speed applications. Lateral p–intrinsic–n (PIN) diodes made from single-crystal silicon[18] and germanium[19] nanomembranes can be fabricated on flexible substrates for operation at 10 GHz. However, the use of PIN diodes is usually limited to RF switches and power attenuators, and such diodes are not applicable to energy harvesting[8]. Besides, the high cost of single-crystal silicon and germanium nanomembranes, as well as the complexity of their materials and processing, render them unfavourable for practical applications.

Nowadays, Wi-Fi is becoming increasingly ubiquitous in both indoor and outdoor environments and provides an abundant source of always-on RF energy. It would be highly desirable if wearable electronics could directly harvest the radiation in the Wi-Fi band (2.4 GHz and 5.9 GHz) for wireless charging. However, owing to the aforementioned challenges, a flexible RF rectifier that is fast enough to achieve Wi-Fi-band wireless energy harvesting has not been demonstrated. In this work, we present an atomically thin and fully flexible MoS_2-based rectifier with a cutoff frequency of 10 GHz at zero external bias, using a self-aligned fabrication technique. MoS_2 is an emerging two-dimensional (2D) semiconductor with high mechanical robustness and low-cost large-scale synthesis technology[2,20,21]. By patterning MoS_2 into a metallic–semiconducting (1T/1T′–2H) phase heterostructure[22] (Fig. 1a), we demonstrate a lateral Schottky diode with junction capacitance lower than 10 fF. In combination with a reduc-