Physical Computing

WEEK 9 - Spark and IoT

Photon(WIFI) & Electron(3G)

A wifi and 3G enabled IoT Arduino like platform

Uses it’s own IDEs

* CLI
* Atom Skinned & Called Particle Dev
* Cloud one called Particle Build
* tinker app
* iOS & android support
* ParticleJS
* Uses Node

Photon specific specs

Fits in a standard breadboard (with headers)

Surface mountable for machine assembly (without headers)

Broadcom BCM43362 Wi-Fi chip

STM32F205 120Mhz ARM Cortex M3

1MB flash, 128KB RAM

802.11b/g/n

FCC/CE/IC certified

Open source hardware

vs Arduino

Fits in a standard breadboard (with headers)

Surface mountable for machine assembly (without headers)

Broadcom BCM43362 Wi-Fi chip

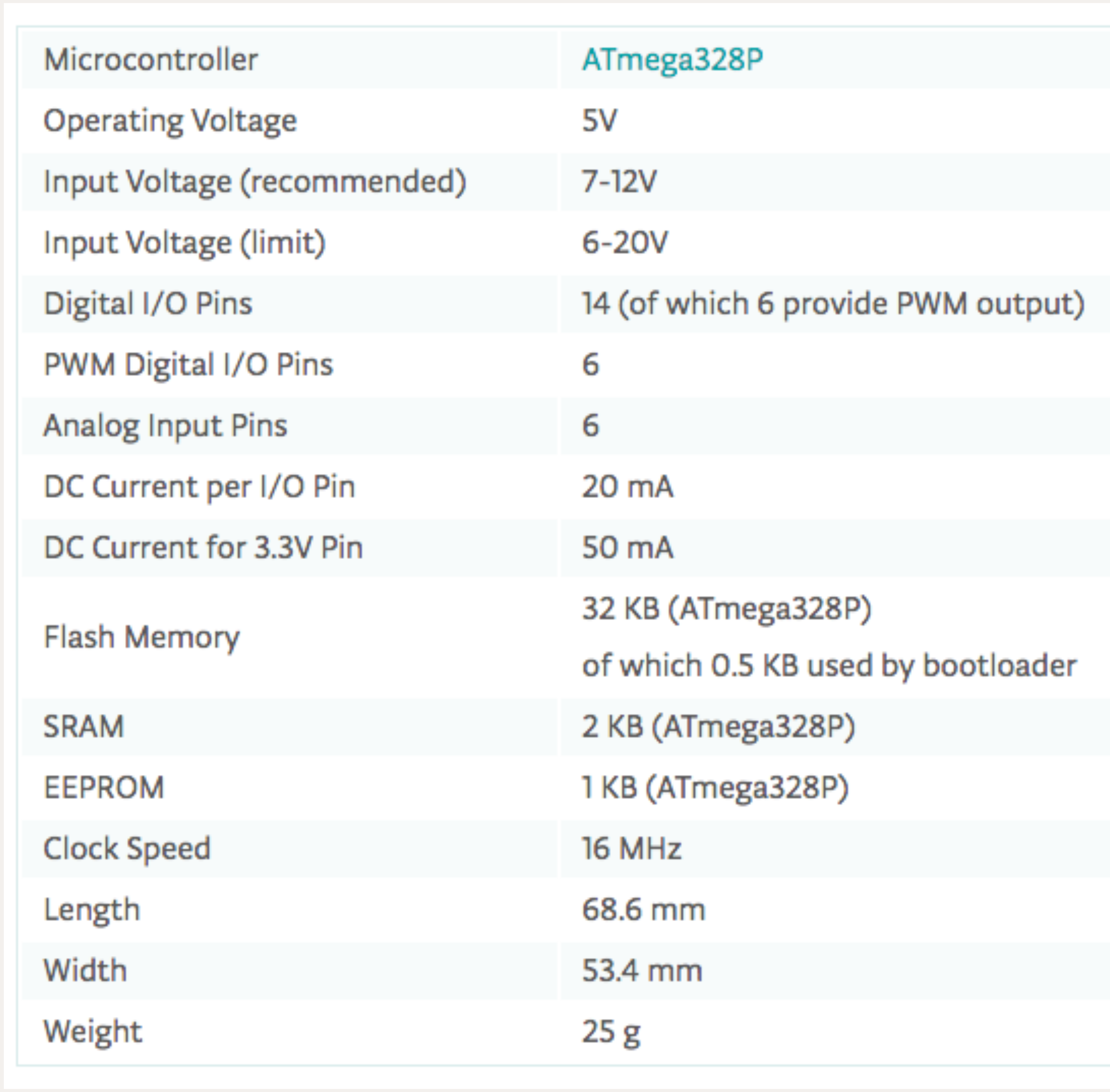
STM32F205 120Mhz ARM Cortex M3

1MB flash, 128KB RAM

802.11b/g/n

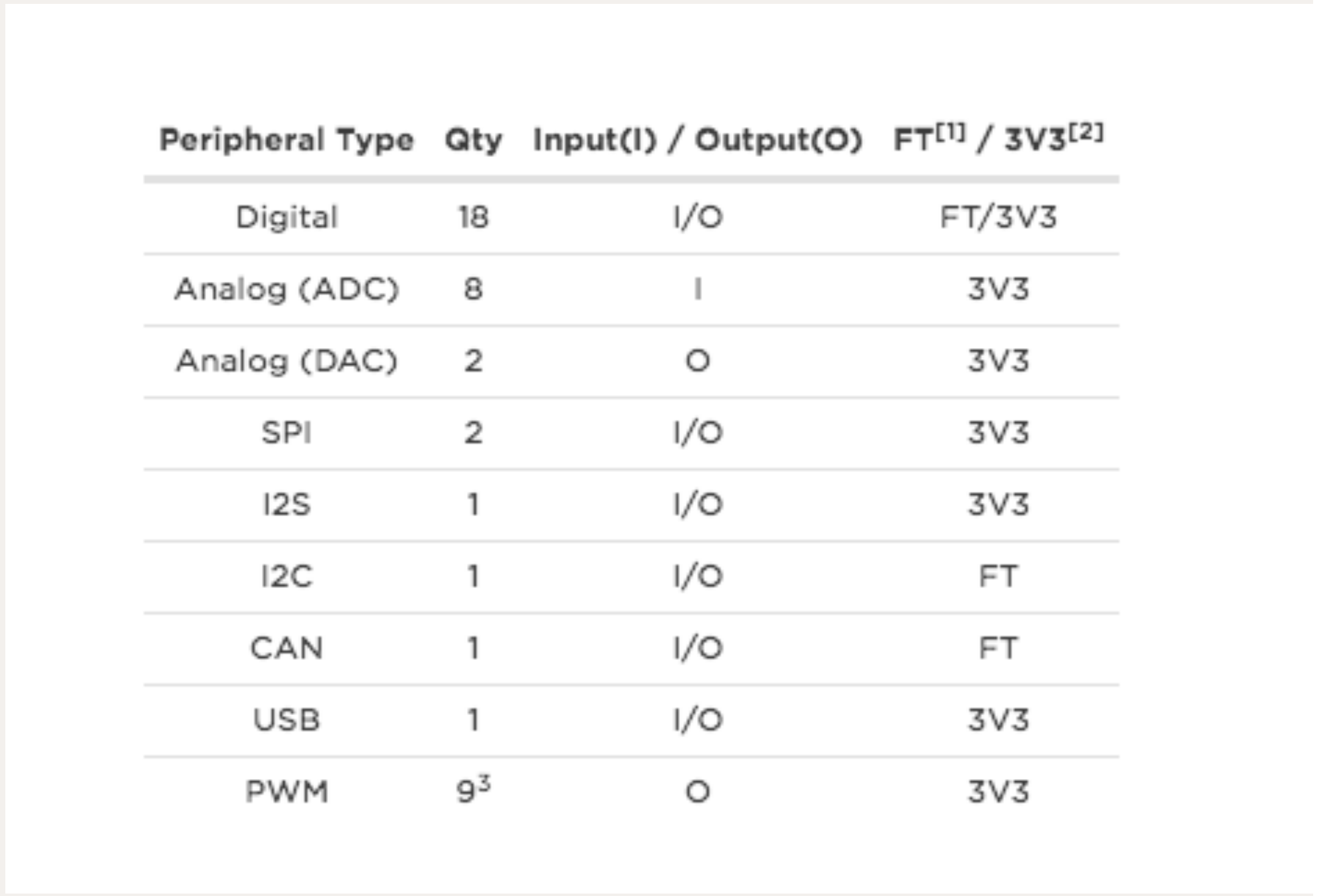
FCC/CE/IC certified

Open source hardware

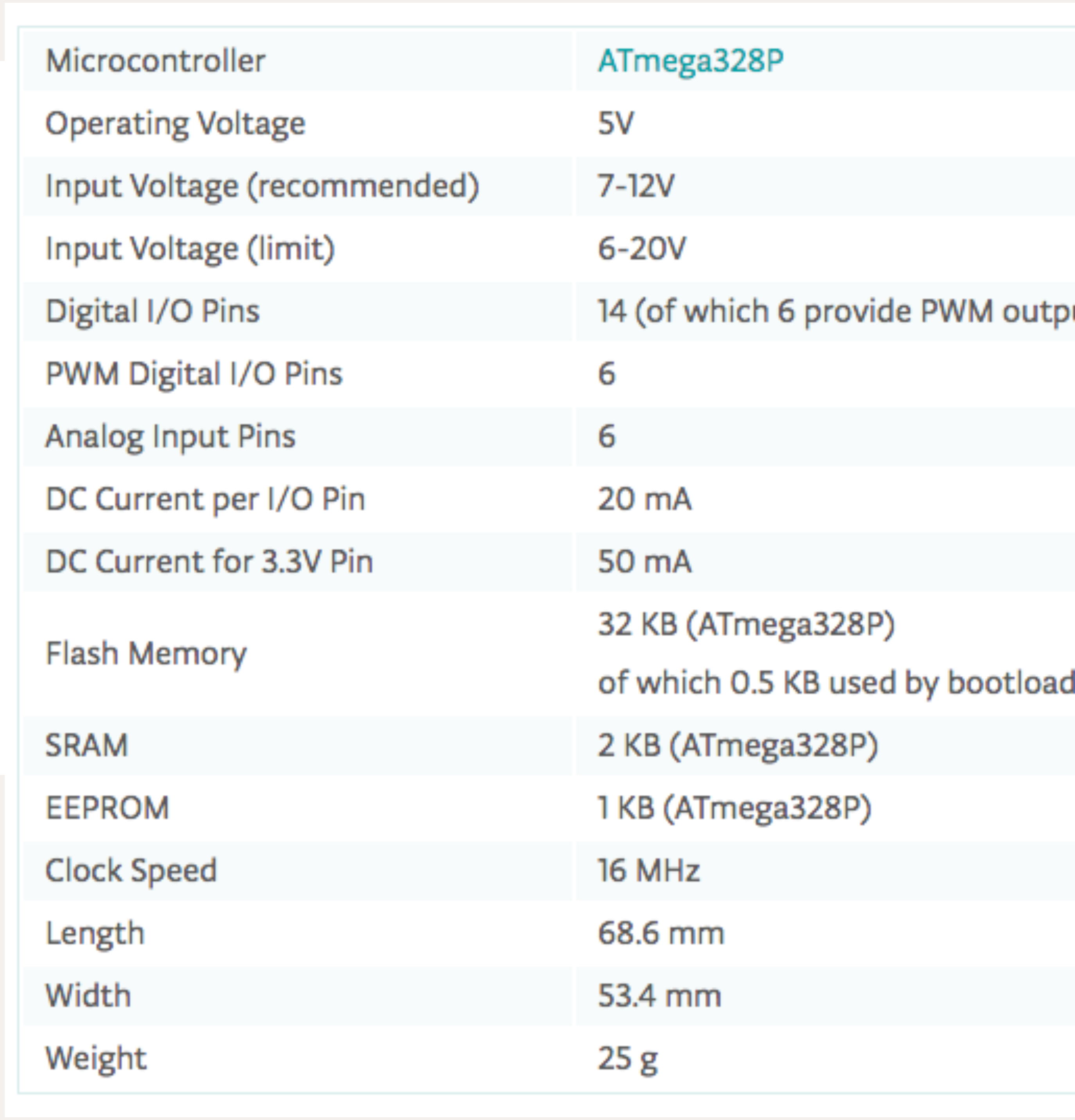


Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

vs Arduino



Peripheral Type	Qty	Input(I) / Output(O)	FT[1] / 3V3[2]
Digital	18	I/O	FT/3V3
Analog (ADC)	8	I	3V3
Analog (DAC)	2	O	3V3
SPI	2	I/O	3V3
I2S	1	I/O	3V3
I2C	1	I/O	FT
CAN	1	I/O	FT
USB	1	I/O	3V3
PWM	9[3]	O	3V3

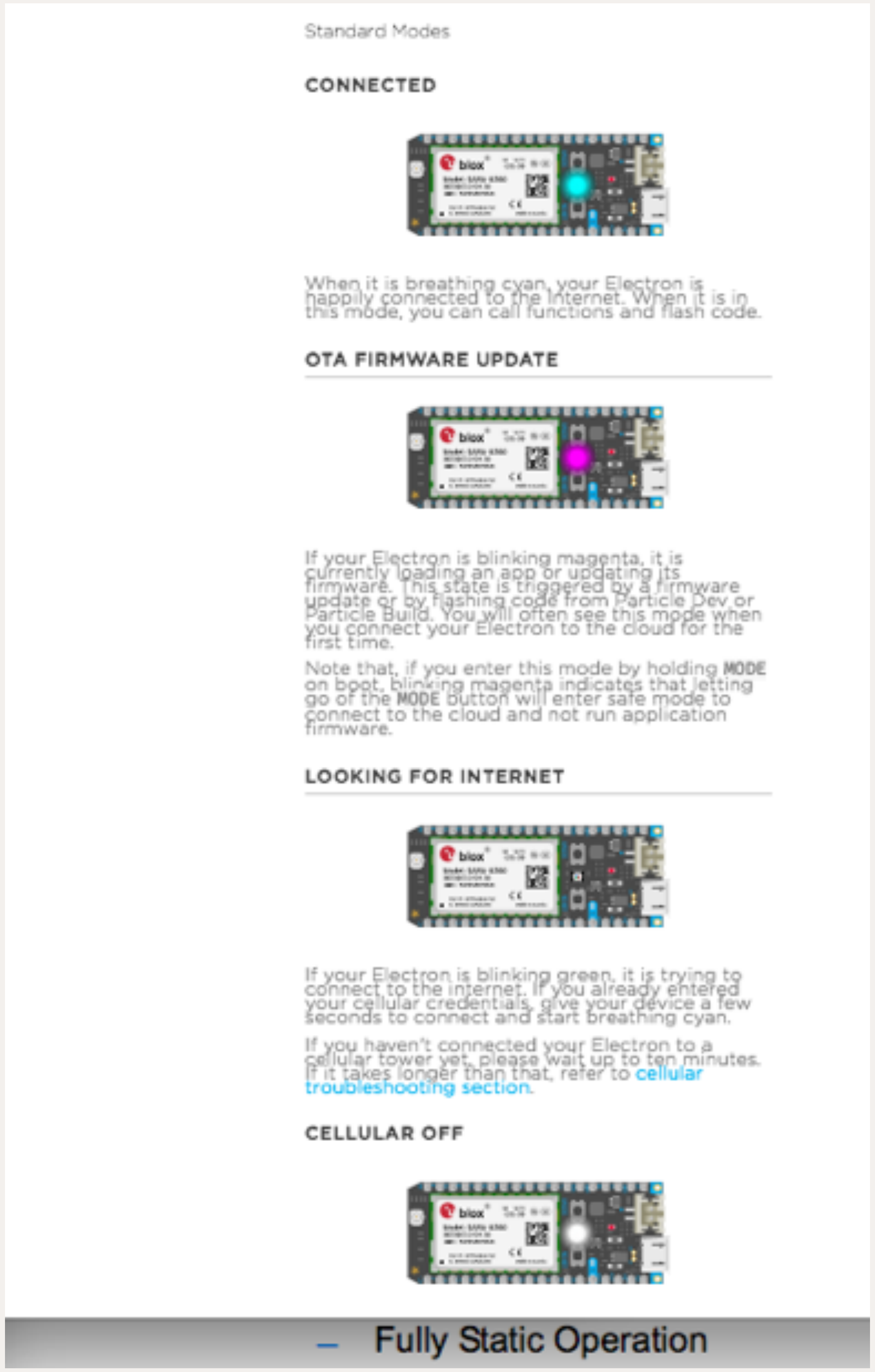


Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM outp
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootload
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Getting online

MODES: https://docs.particle.io/guide/getting-started/modes/photon/

While you are at it, make sure to set up an account



Standard Modes

CONNECTED

When it is breathing cyan, your Electron is happily connected to the Internet. When it is in this mode, you can call functions and flash code.

OTA FIRMWARE UPDATE

If your Electron is blinking magenta, it is currently loading an app or updating its firmware. This state is triggered by a firmware update or by flashing code from Particle Dev or Particle Build. You will often see this mode when you connect your Electron to the cloud for the first time.

Note that, if you enter this mode by holding `MODE` on boot, blinking magenta indicates that letting go of the `MODE` button will enter safe mode to connect to the cloud and not run application firmware.

LOOKING FOR INTERNET

If your Electron is blinking green, it is trying to connect to the internet. If you already entered your cellular credentials, give your device a few seconds to connect and start breathing cyan.

If you haven't connected your Electron to a cellular tower yet, please wait up to ten minutes. If it takes longer than that, refer to cellular troubleshooting section.

CELLULAR OFF

Fully Static Operation

CLI

This is the smoothest way I have found to claim a core. You have to claim and name it to use it. It's hard to share them. I had to email to get one released as of late. There is functionality for this but it seems to not work well.

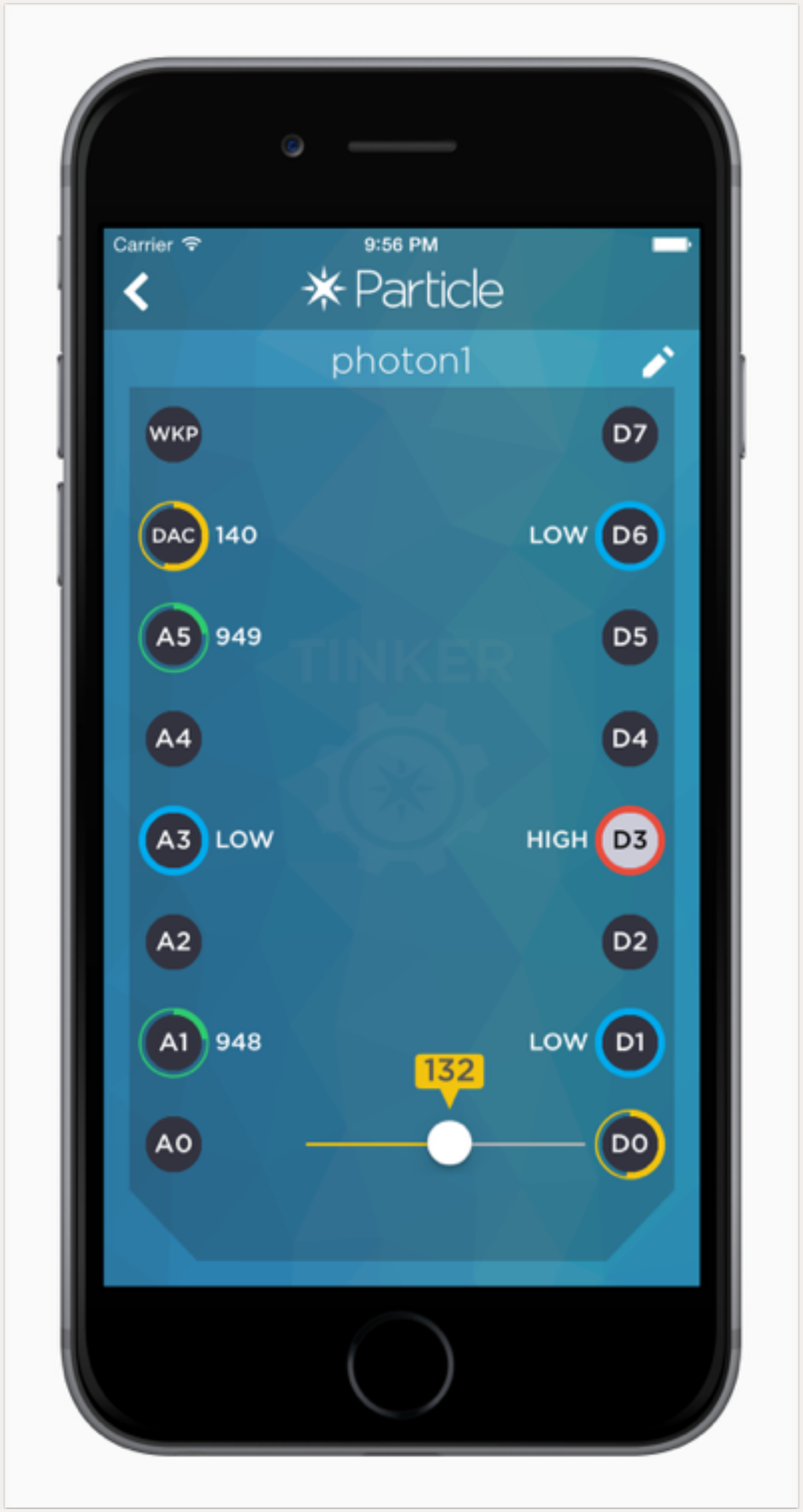
> **particle help**

> particle setup

others? > particle
token, binary, cloud, config, function, keys, serial, udp
update, variable, webhook, wireless

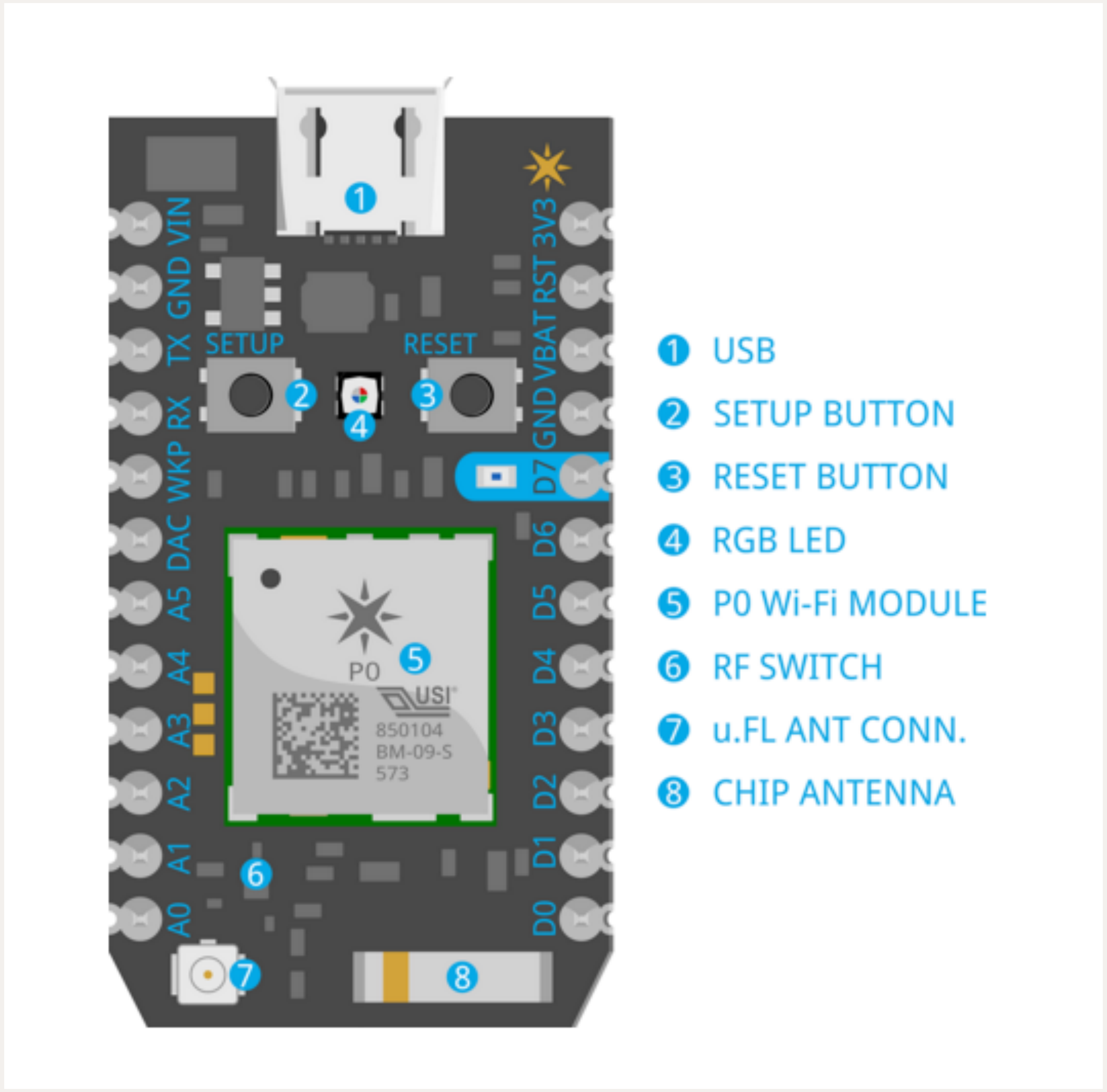
Tinker

Fun toy for rapid testing



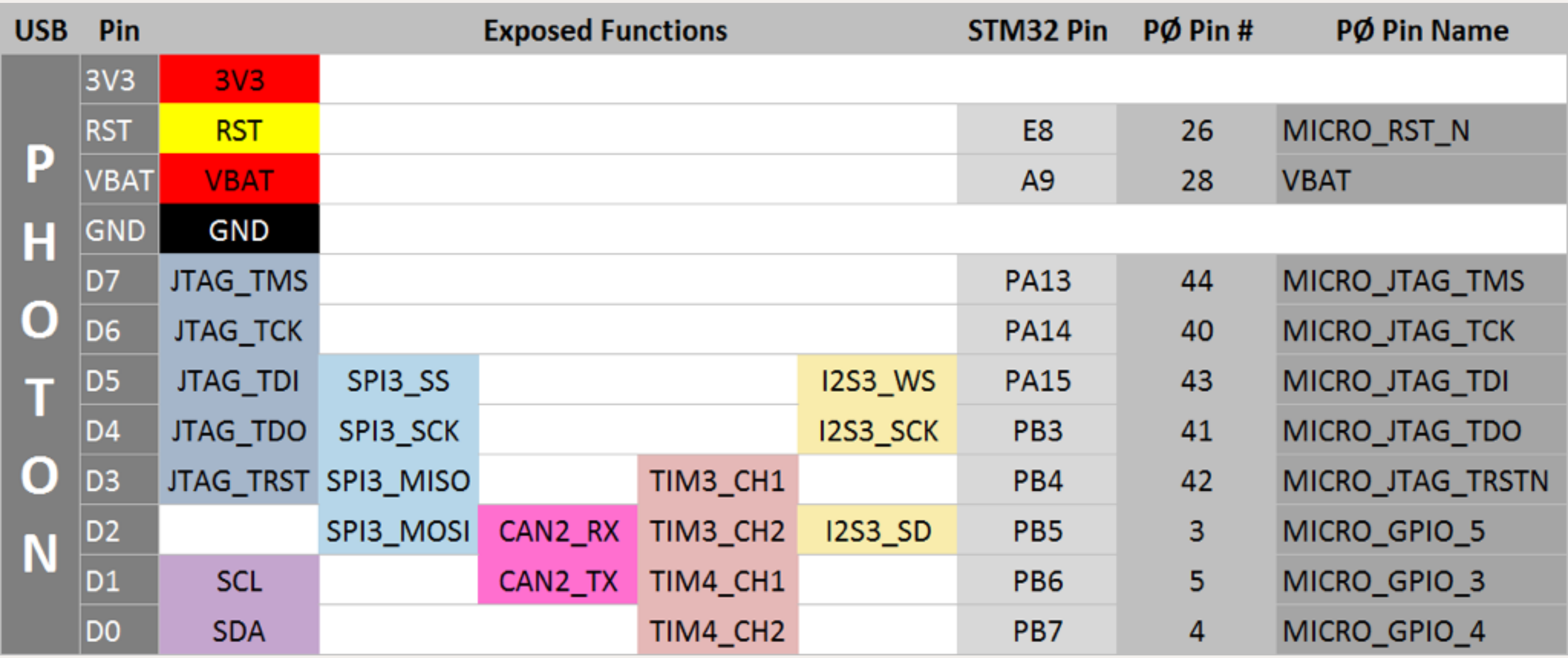
Pin definitions

All the pins are slightly different : https://docs.particle.io/datasheets/photon-datasheet/



Pin definitions

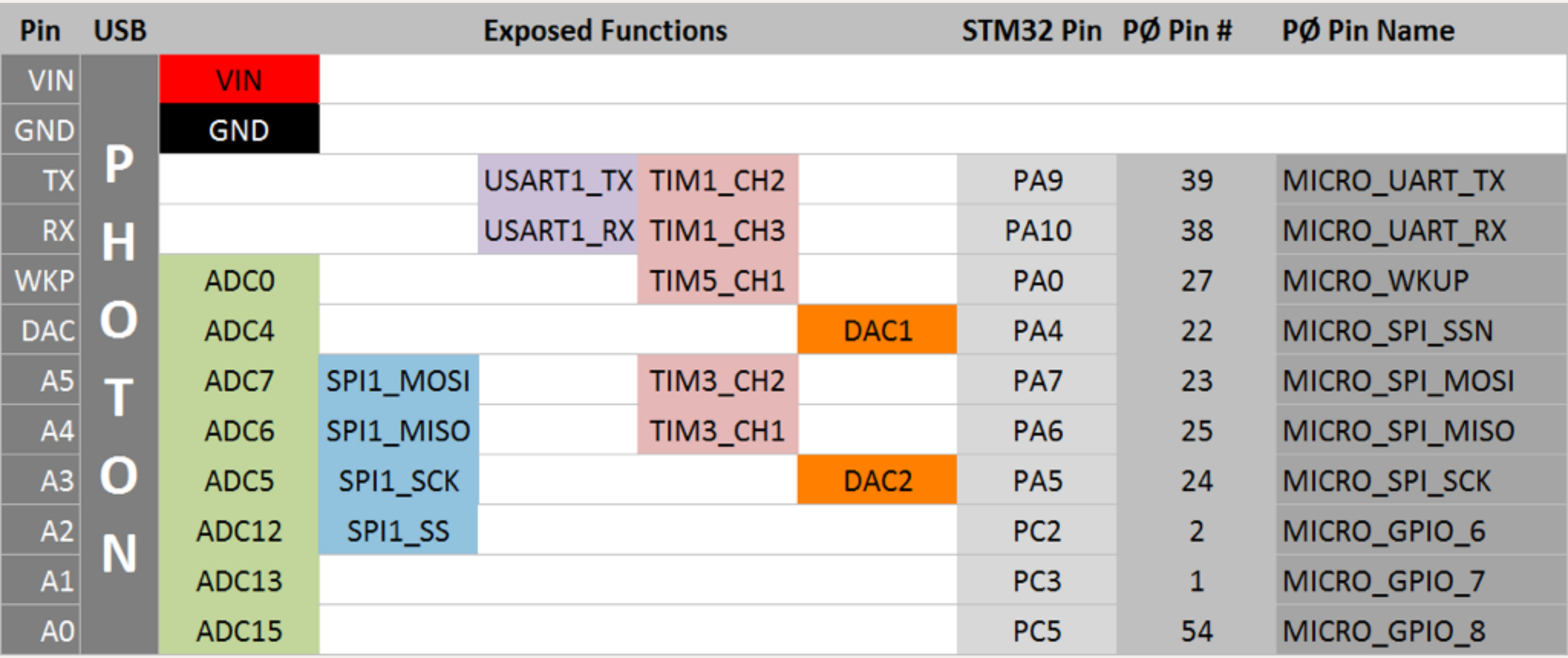
All the pins are slightly different : https://docs.particle.io/datasheets/photon-datasheet/



USB	Pin	Exposed Functions					STM32 Pin	PØ Pin #	PØ Pin Name
PHOTON	3V3	3V3							
	RST	RST					E8	26	MICRO_RST_N
	VBAT	VBAT					A9	28	VBAT
	GND	GND							
	D7	JTAG_TMS					PA13	44	MICRO_JTAG_TMS
	D6	JTAG_TCK					PA14	40	MICRO_JTAG_TCK
	D5	JTAG_TDI	SPI3_SS			I2S3_WS	PA15	43	MICRO_JTAG_TDI
	D4	JTAG_TDO	SPI3_SCK			I2S3_SCK	PB3	41	MICRO_JTAG_TDO
	D3	JTAG_TRST	SPI3_MISO		TIM3_CH1		PB4	42	MICRO_JTAG_TRSTN
	D2		SPI3_MOSI	CAN2_RX	TIM3_CH2	I2S3_SD	PB5	3	MICRO_GPIO_5
	D1	SCL		CAN2_TX	TIM4_CH1		PB6	5	MICRO_GPIO_3
	D0	SDA			TIM4_CH2		PB7	4	MICRO_GPIO_4

Pin definitions

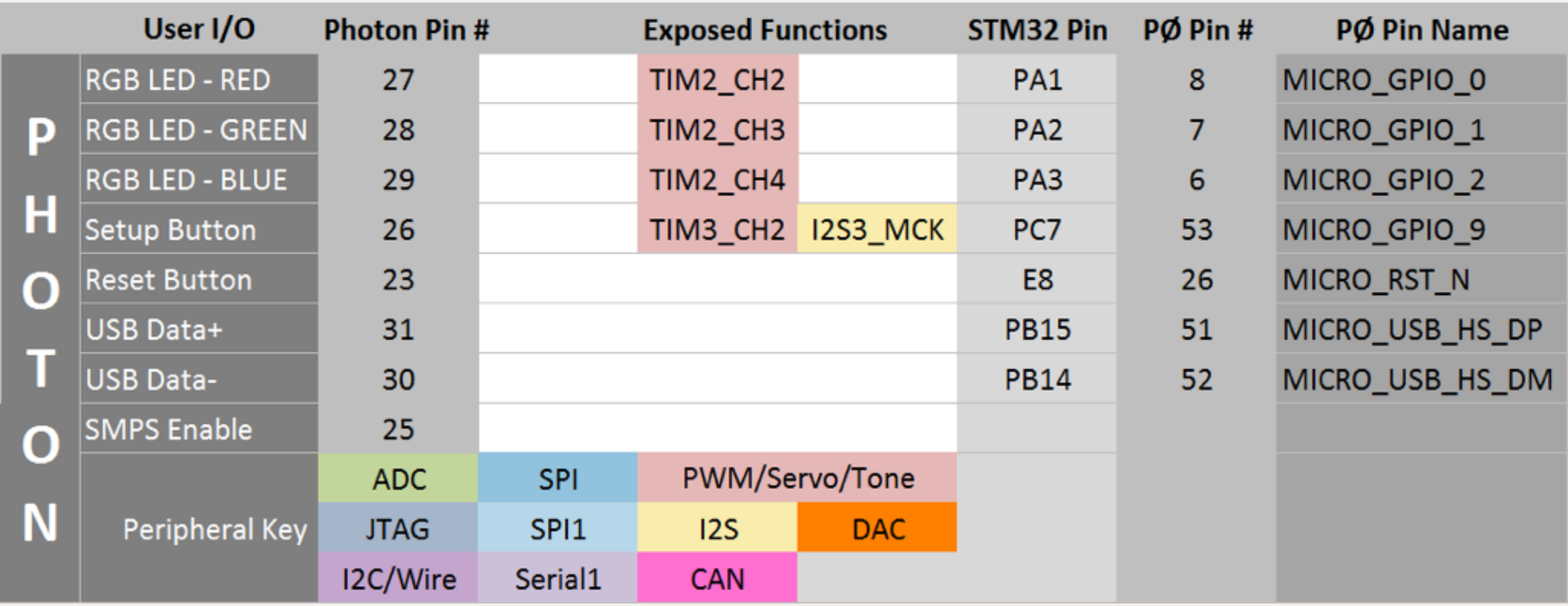
All the pins are slightly different : https://docs.particle.io/datasheets/photon-datasheet/



Pin	USB	Exposed Functions					STM32 Pin	PØ Pin #	PØ Pin Name
VIN	PHOTON	VIN							
GND		GND							
TX				USART1_TX	TIM1_CH2		PA9	39	MICRO_UART_TX
RX				USART1_RX	TIM1_CH3		PA10	38	MICRO_UART_RX
WKP		ADC0			TIM5_CH1		PA0	27	MICRO_WKUP
DAC		ADC4				DAC1	PA4	22	MICRO_SPI_SSN
A5		ADC7	SPI1_MOSI		TIM3_CH2		PA7	23	MICRO_SPI_MOSI
A4		ADC6	SPI1_MISO		TIM3_CH1		PA6	25	MICRO_SPI_MISO
A3		ADC5	SPI1_SCK			DAC2	PA5	24	MICRO_SPI_SCK
A2		ADC12	SPI1_SS				PC2	2	MICRO_GPIO_6
A1		ADC13					PC3	1	MICRO_GPIO_7
A0		ADC15					PC5	54	MICRO_GPIO_8

Pin definitions

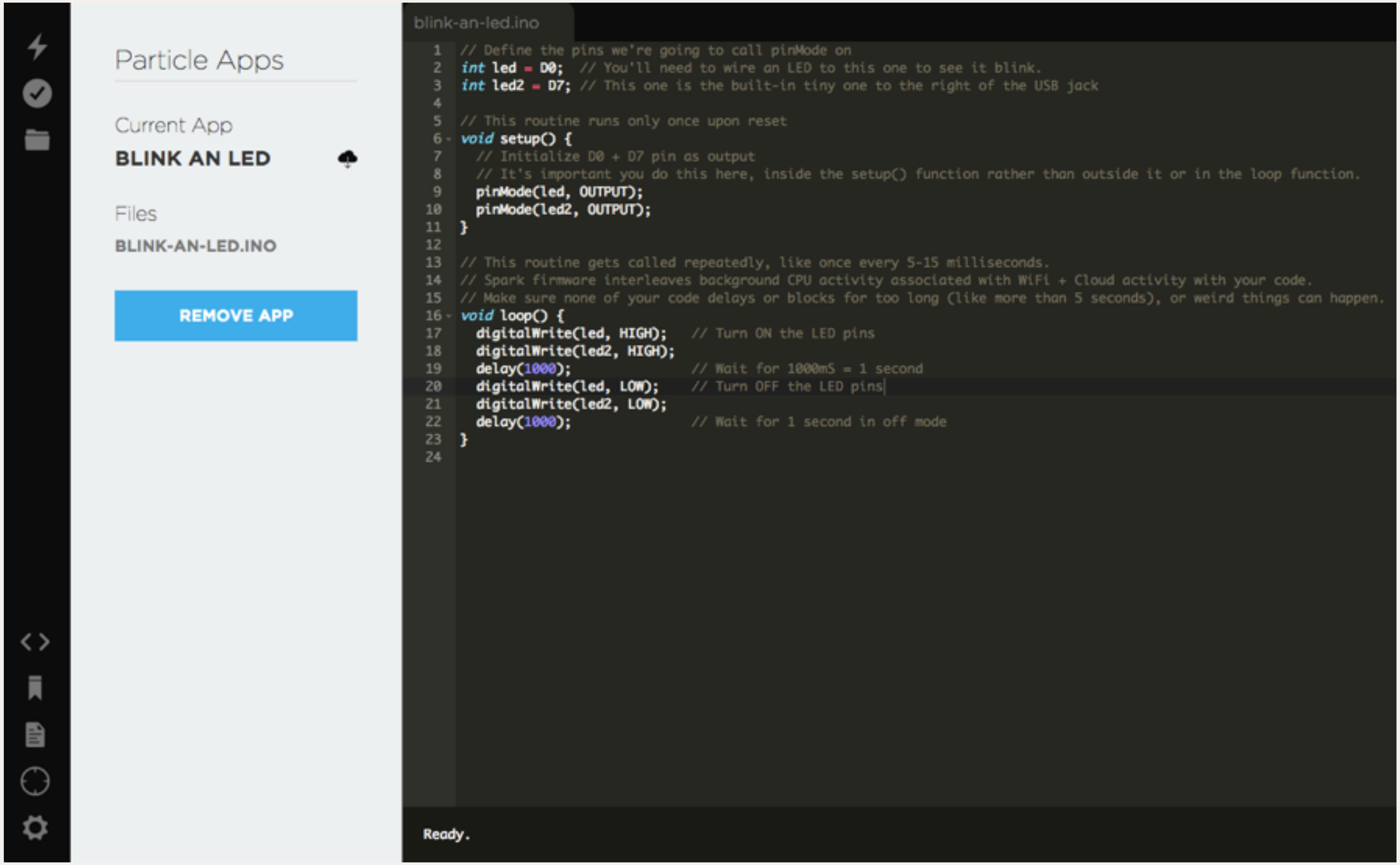
All the pins are slightly different : https://docs.particle.io/datasheets/photon-datasheet/



	User I/O	Photon Pin #	Exposed Functions			STM32 Pin	PØ Pin #	PØ Pin Name
PHOTON	RGB LED - RED	27		TIM2_CH2		PA1	8	MICRO_GPIO_0
	RGB LED - GREEN	28		TIM2_CH3		PA2	7	MICRO_GPIO_1
	RGB LED - BLUE	29		TIM2_CH4		PA3	6	MICRO_GPIO_2
	Setup Button	26		TIM3_CH2	I2S3_MCK	PC7	53	MICRO_GPIO_9
	Reset Button	23				E8	26	MICRO_RST_N
	USB Data+	31				PB15	51	MICRO_USB_HS_DP
	USB Data-	30				PB14	52	MICRO_USB_HS_DM
	SMPS Enable	25						
	Peripheral Key	ADC	SPI	PWM/Servo/Tone				
		JTAG	SPI1	I2S	DAC			
		I2C/Wire	Serial1	CAN				

Web IDE

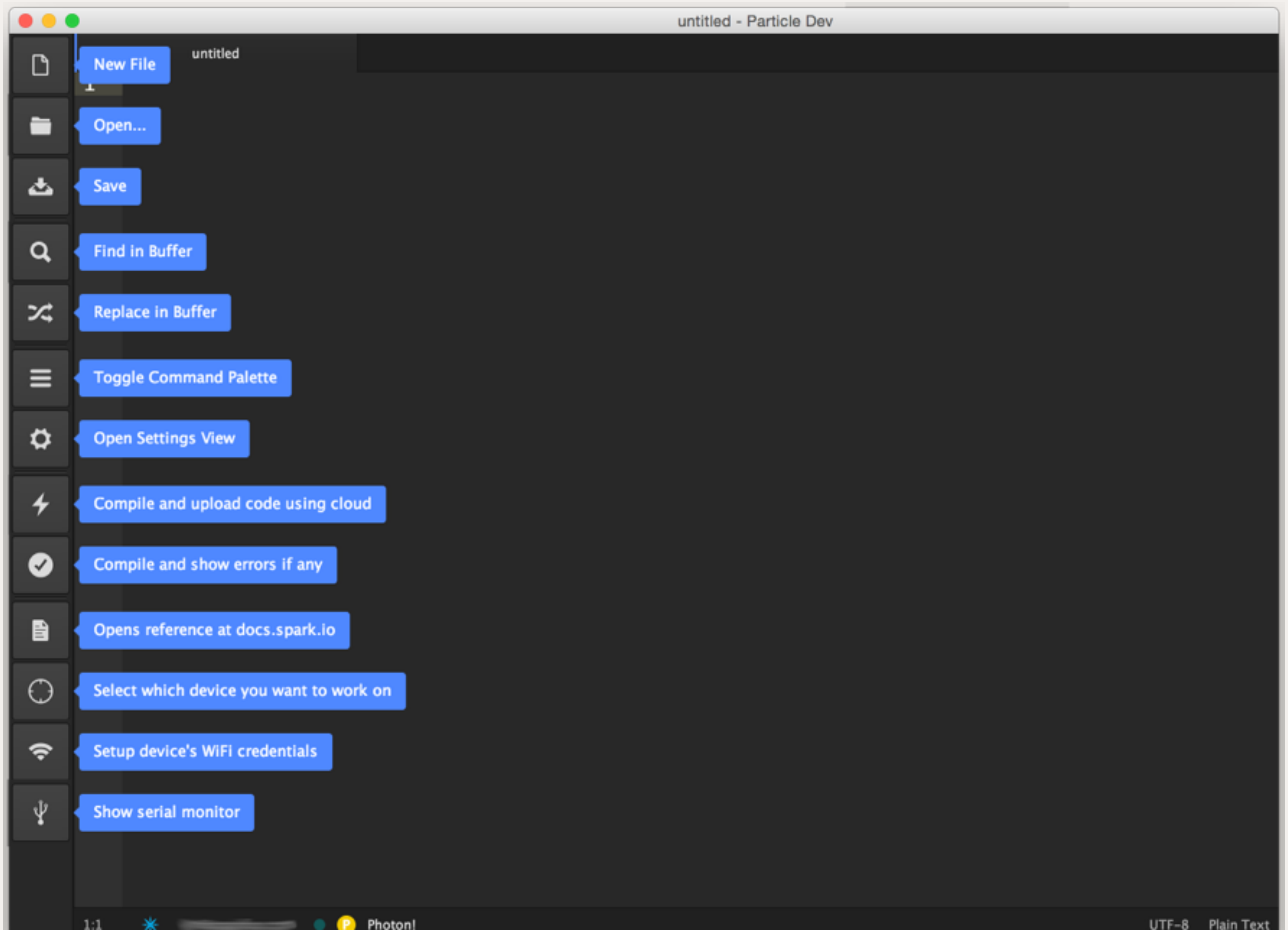
Powerful cloud platform, easy to use with libraries, good for getting your token and device IDs



```
// Define the pins we're going to call pinMode on
int led = D0;  // You'll need to wire an LED to this one to see it blink.
int led2 = D7; // This one is the built-in tiny one to the right of the USB jack

// This routine runs only once upon reset
void setup() {
  // Initialize D0 + D7 pin as output
  // It's important you do this here, inside the setup() function rather than outside it or in the loop function.
  pinMode(led, OUTPUT);
  pinMode(led2, OUTPUT);
}

// This routine gets called repeatedly, like once every 5-15 milliseconds.
// Spark firmware interleaves background CPU activity associated with WiFi + Cloud activity with your code.
// Make sure none of your code delays or blocks for too long (like more than 5 seconds), or weird things can happen.
void loop() {
  digitalWrite(led, HIGH);   // Turn ON the LED pins
  digitalWrite(led2, HIGH);
  delay(1000);               // Wait for 1000mS = 1 second
  digitalWrite(led, LOW);    // Turn OFF the LED pins
  digitalWrite(led2, LOW);
  delay(1000);               // Wait for 1 second in off mode
}

```

```
Ready.
```

Particle DEV Atom for desktop control



untitled - Particle Dev

untitled

New File

Open...

Save

Find in Buffer

Replace in Buffer

Toggle Command Palette

Open Settings View

Compile and upload code using cloud

Compile and show errors if any

Opens reference at docs.spark.io

Select which device you want to work on

Setup device's WiFi credentials

Show serial monitor

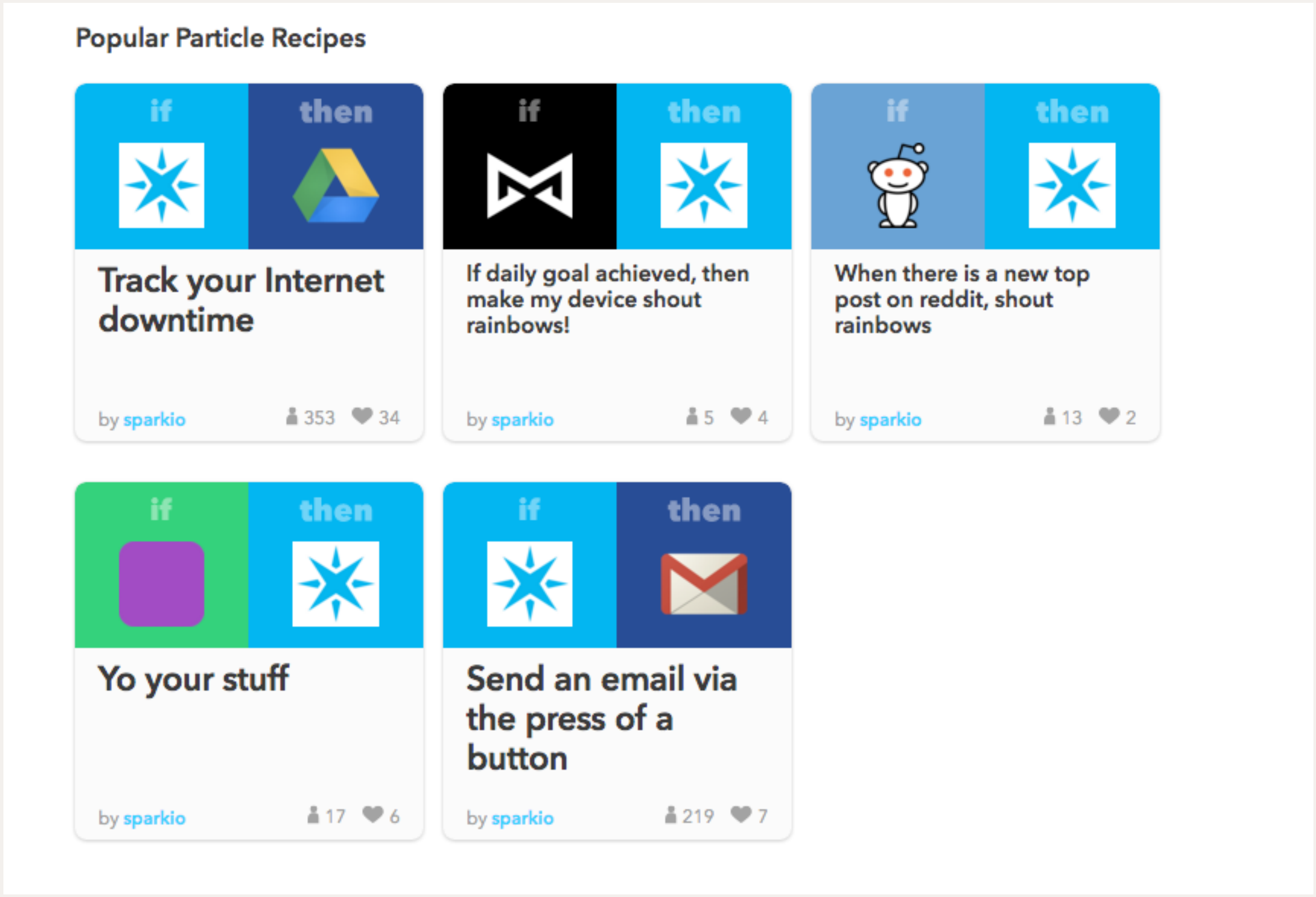
1:1 [redacted] Photon! UTF-8 Plain Text

Particle DEV

Good resource for setting up step by step: https://learn.sparkfun.com/tutorials/photon-development-guide/particle-dev-half-online-half-offline

Things you can do :
* publish variables so other programs can get the data
* call functions in the particle .ino file from another language like java, python, node or c++ (processing, oF)
* subscribe to open data streams online
* IFTTT

IFTTT



Popular Particle Recipes

Track your Internet downtime

by sparkio 353 34

If daily goal achieved, then make my device shout rainbows!

by sparkio 5 4

When there is a new top post on reddit, shout rainbows

by sparkio 13 2

Yo your stuff

by sparkio 17 6

Send an email via the press of a button

by sparkio 219 7

Particle DEV

What is a call back?

A callback is basically a first order function where you are passing a function to some other piece of code that has an opportunity to call it at a later point in time. They are a common way events call functions.

What's a curl request? curl is an open source command line tool and library for transferring data with URL syntax

Can be used from terminal in unix

Particle DEV

get variable value

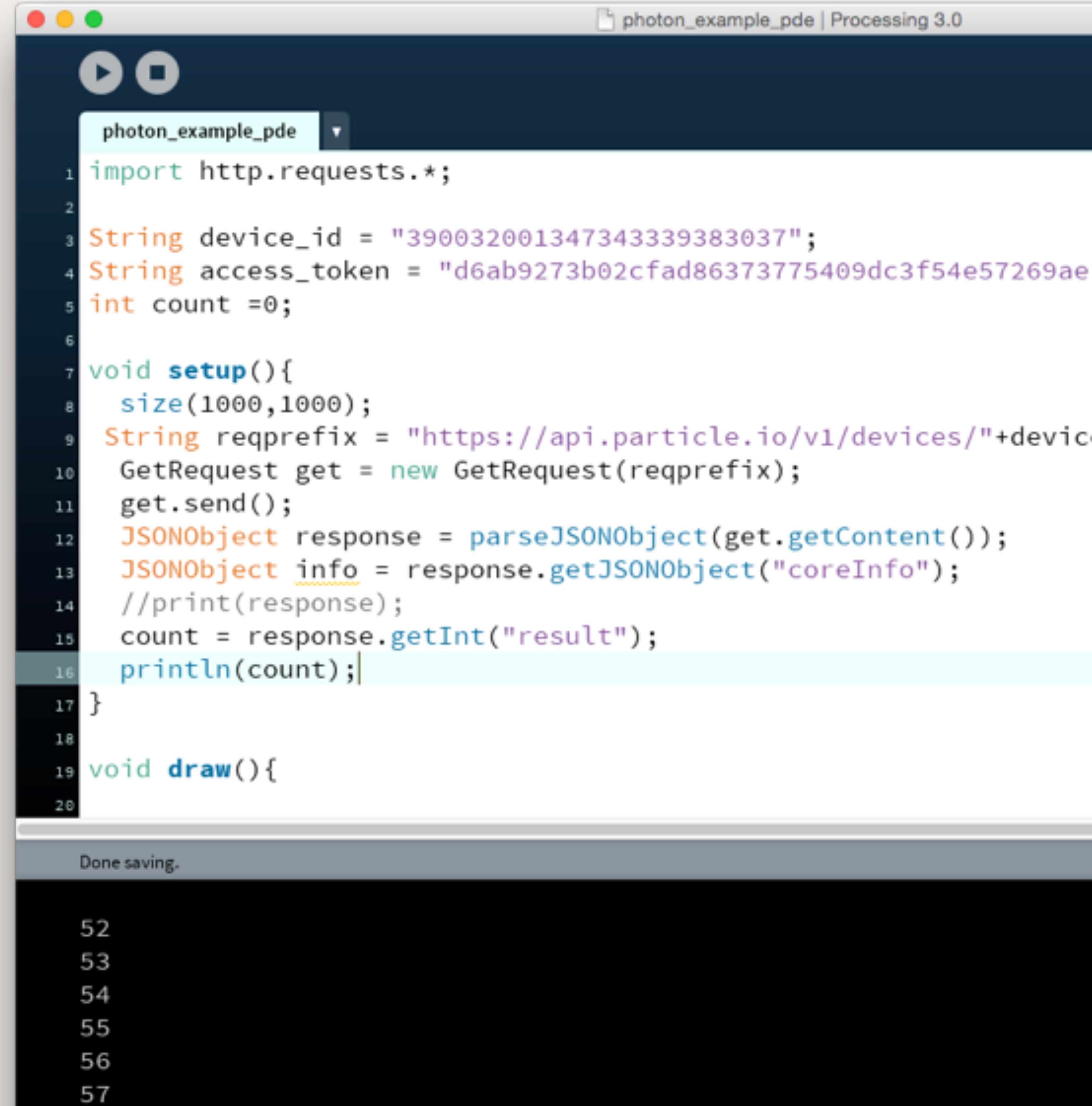
```
curl https://api.particle.io/v1/devices/
390032001347343339038037/counter\?
access_token=d6ab9273b02cfad86373775409dc3f54e57269ae
```

list of exposed functions& vars

```
curl https://api.particle.io/v1/devices/
0123456789abcdef01234567\?access_token\=1234
```

Works w/processing using http.requests

get = go get a url for me
post = send data to a url for me



photon_example_pde | Processing 3.0

```
import http.requests.*;

String device_id = "390032001347343339383037";
String access_token = "d6ab9273b02cfad86373775409dc3f54e57269ae
int count =0;

void setup(){
  size(1000,1000);
 String reqprefix = "https://api.particle.io/v1/devices/"+devic
  GetRequest get = new GetRequest(reqprefix);
  get.send();
  JSONObject response = parseJSONObject(get.getContent());
  JSONObject info = response.getJSONObject("coreInfo");
  //print(response);
  count = response.getInt("result");
  println(count);
}

void draw(){
```

Done saving.

```
52
53
54
55
56
57
```