Building an Azure IoT Controlled Device in Less than 60 Minutes and Costs Less than 10 USD

Introduction to the Internet of Things

OzCode
Your Road to Magical Debugging

CodeValue
Inspiring Code, Creating Value.

Alon Fliess

Chief Software Architect

alonf@codevalue.net

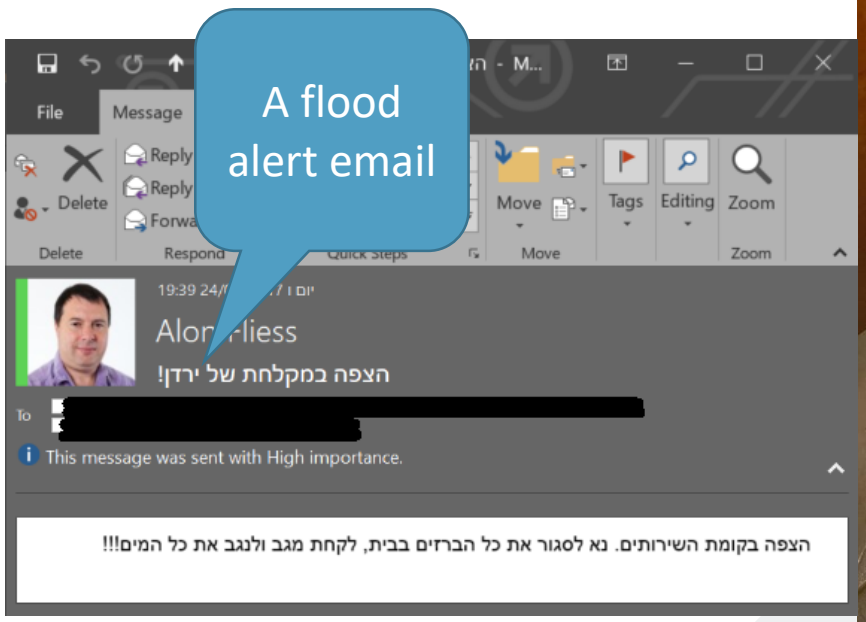
http://blogs.microsoft.co.il/blogs/alon

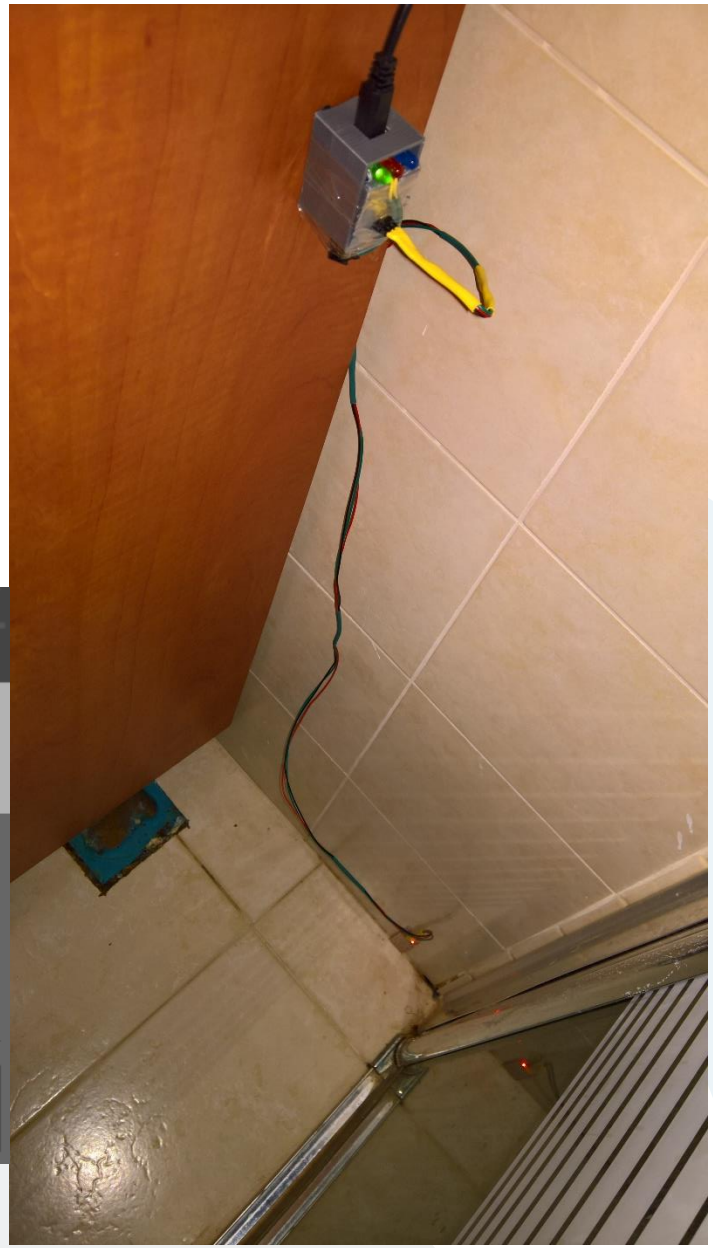
MVP Microsoft Most Valuable Professional

Microsoft Regional Director

A True War Story

- My wife is shouting:
  - ALON!!! There's water everywhere!!!
- By the time the plumber came to fix it, in 4 hours I had this:
- A water flood detector
  - Using small device + Azure IoT Hub + Service Bus Queue + Azure Logic App + Office 365 Email + Twilio SMS





About Me

- Alon Fliess:
  - Chief Software Architect & Co-Founder at CodeValue Ltd.
  - Over 25 years of hands-on experience
  - Microsoft Regional Director & Microsoft MVP
  - Active member of several Patterns & Practices councils
  - Renowned speaker at both international and domestic events

About CodeValue

- Cloud Computing
- Advanced Mobile Technologies
- UI/UX & Graphic Design
- Advanced Web Technologies
- Cross Platform Development
- ALM & DevOps
- Software Architecture
- IOT & Embedded Software
- Training & Mentoring
- Development Management & Methodology

Google Developer Expert

Microsoft Regional Director

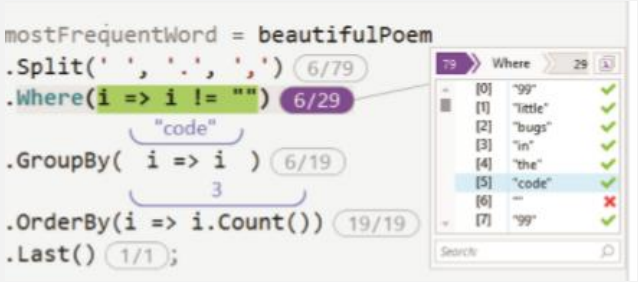
MVP Microsoft Most Valuable Professional

amazon web services | Certified Solutions Architect - Associate

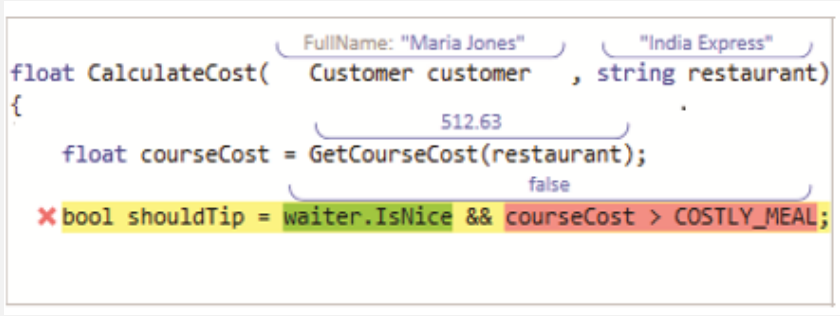
Debug like a wizard

Quit debugging, spend more time writing brilliant software

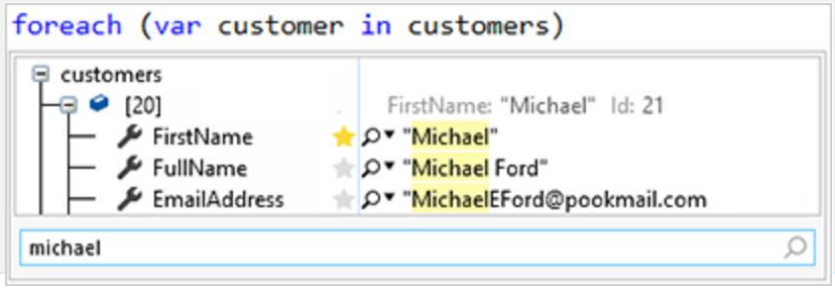
LINQ Debugging / Know the flow of your LINQ queries



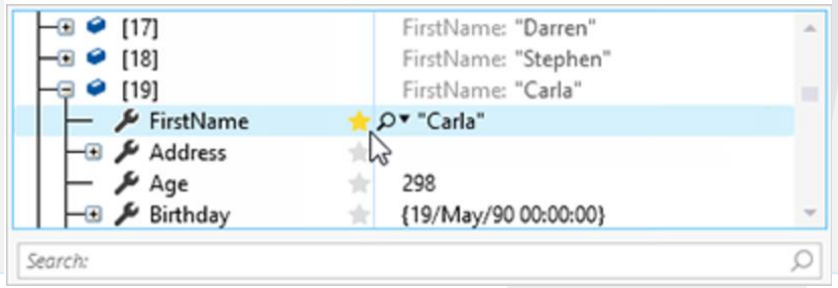
Magic Glance / Figure out complex expressions



Search/ Find that needle in a haystack of data



Reveal / Focus on data that actually matter



With our Visual Studio extension for C#, follow the road to a bug-free world

oz-code.com | @oz_code

OzCode
Magical Debugging

Agenda

- About
- Introduction
- The Device
- The Cloud - Azure IoT
- An affordable IoT controlled device
- The real world scenario
- Summary

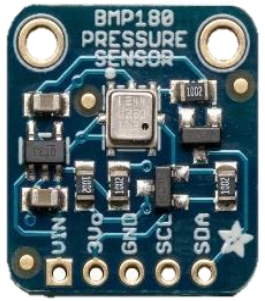
Technology advances in “Buzzwords” steps

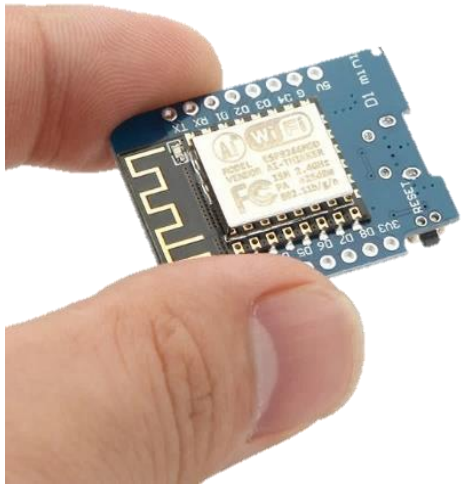
- It starts with the basic technology and slowly evolves
- After several years the world is ready to embrace the technology
- And here is where the big buzz begins
- For the Internet of Things:
  - The basic technology is already here for almost a decade by now
    - Amazon Web Services cloud has started in 2006
    - Azure was announced in October 2008 and released on February 1, 2010
    - Devices such as the Amtel AVR are more than 20 years old
    - The Arduino family that many IoT hobbyists are using it as a cheap IoT end device is 10 years old

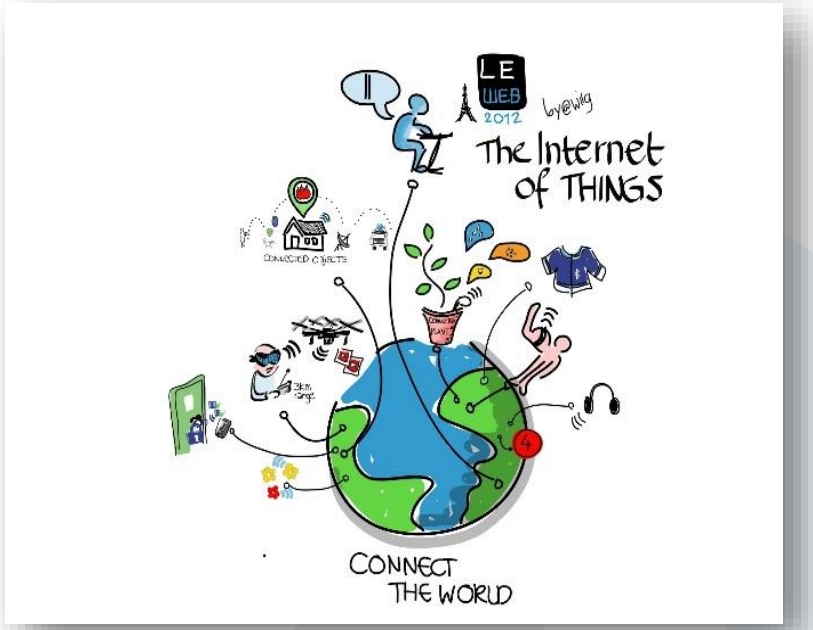
IoT - More Than The Core Technology

- It is not (just) the technology that makes IoT what it is
- It is the
  - Concepts, perception, commitment and the challenges
  - Facts that the entire industry is dealing with it nowadays
- The IoT Challenge:
  - ***vast** amount of **devices** using different hardware and software technologies, are **connected** between them and to the **cloud** which in turn provides many **services**, which handle a **huge stream of data** and **analyze** it and extract vital **information** about the current state of the system and via extended processing it can even **predict** future state*

IoT System Basic Components



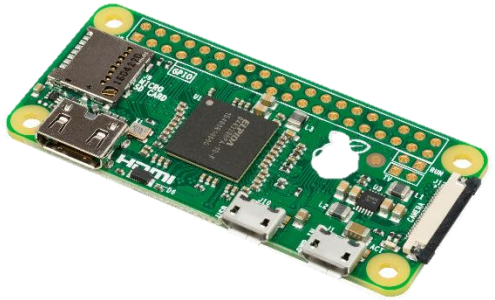


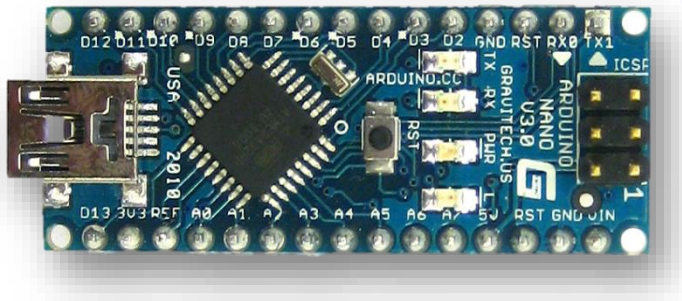



```
var sensorData = await _bmp180.GetSensorDataAsync(Bmp180.UltraHighResolution);
var messageString = JsonConvert.SerializeObject(sensorData);
var message = new
    Microsoft.Azure.Devices.Client.Message(Encoding.ASCII.GetBytes(messageString));
await deviceClient.SendEventAsync(message);
```

The Device

- There are many System on a Chip (SoC) devices to choose from
- Raspberry Pi family
- Arduino Compatible Family
  - ESP 8266 based devices
- Intel devices
- …





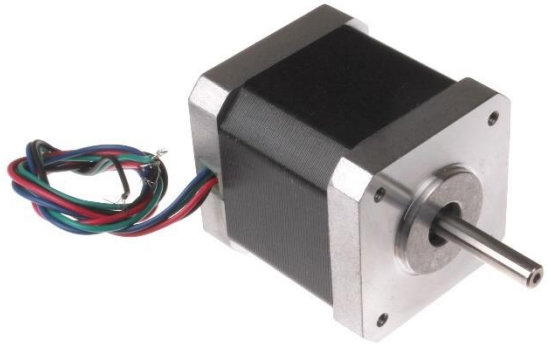
Raspberry Pi Kit
Windows 10 and Raspbian
Samples in C and C#

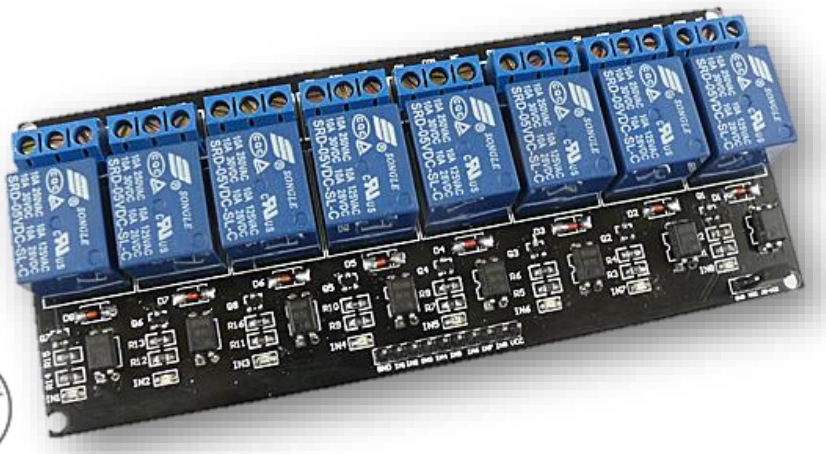
Intel Edison Kit
Linux Yocto
Samples in JavaScript (Node.js)

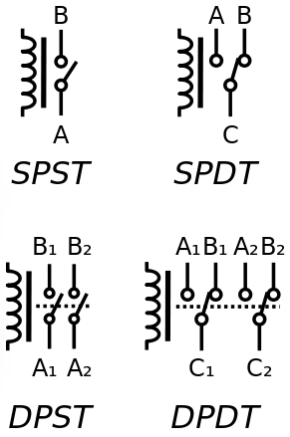
Feather Huzzah ESP8266 Kit
RTOS
Samples in Arduino IDE and C

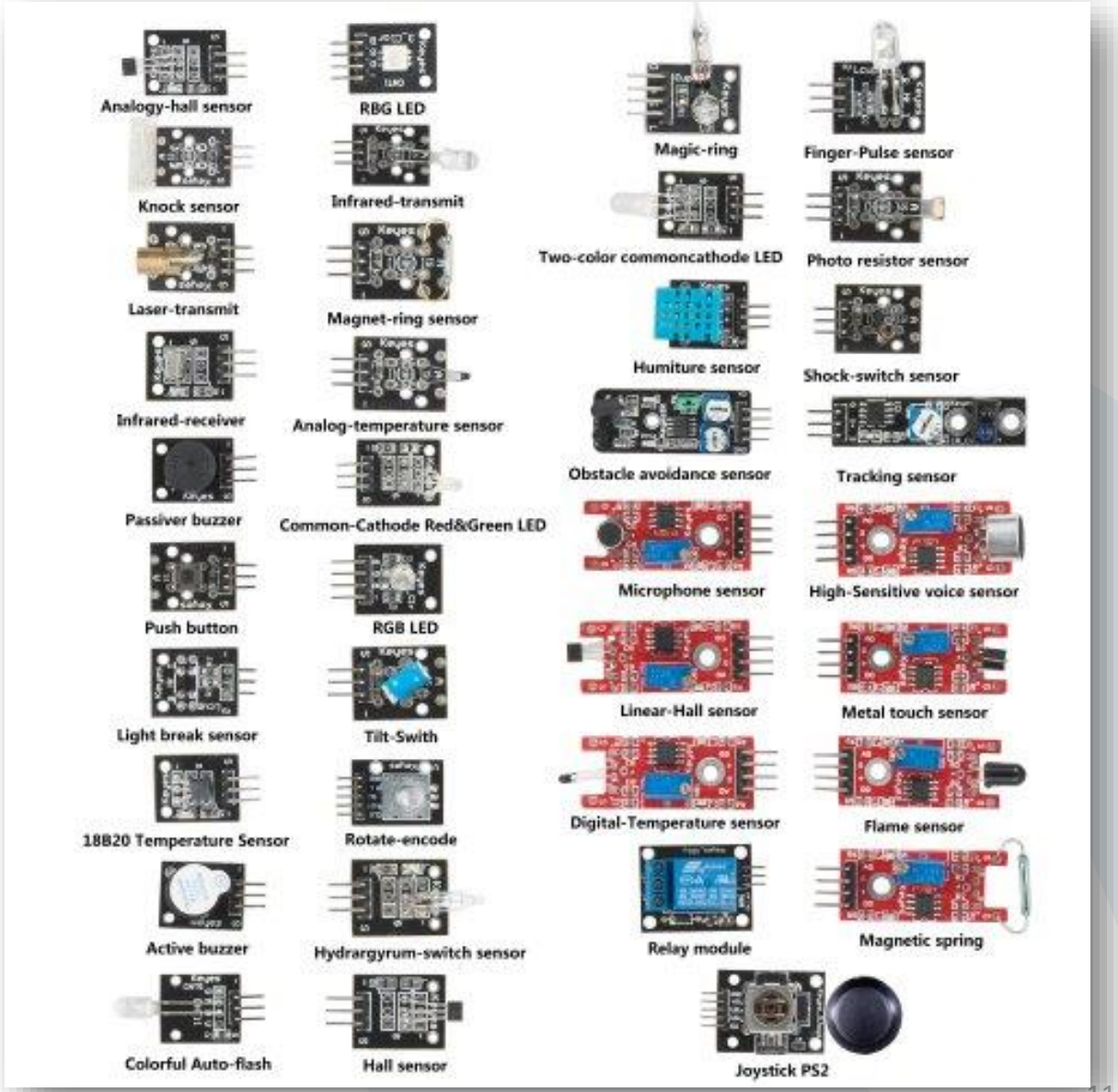
Sensors, Actuators, Motors

- There are plenty of them







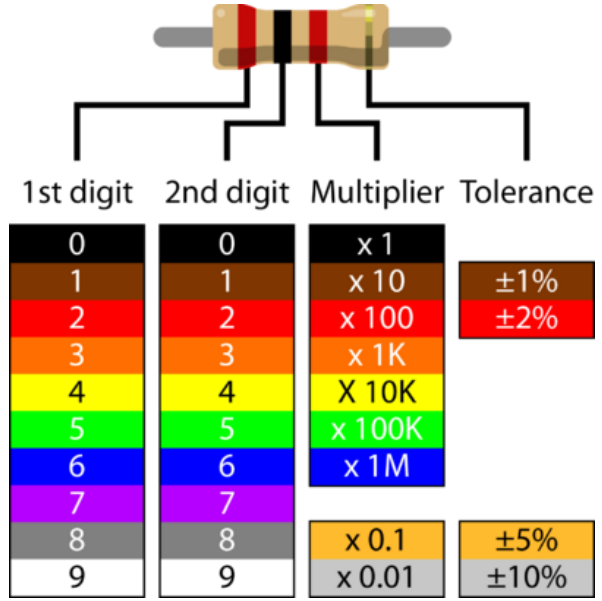


How do I play with it?

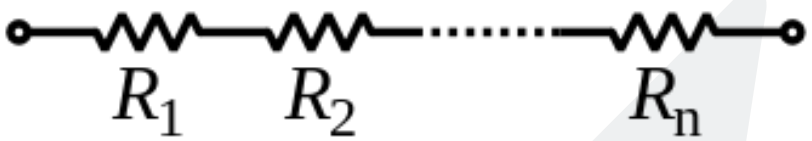
- Pick your weapon
  - A prototype board that has networking capabilities
    - Raspberry Pi, WeMos, Intel IoT, Tessel, NetDuino
    - The complete list @ Azure IoT hardware catalog
- Some Electronics & Hardware Programming
  - Understand how to connect sensors and communicate with them
- Pick your Cloud Services and technologies
  - Microsoft Azure, AWS
  - Do something with the (Big) data

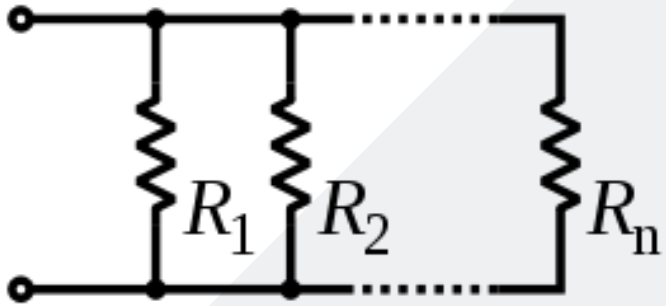
Electronics 101

- Ohm's law: V = R*I
  - V ➔ Volt (V, mV)
  - I ➔ Ampere (A, mA)
  - R ➔ Ohm (Ω)



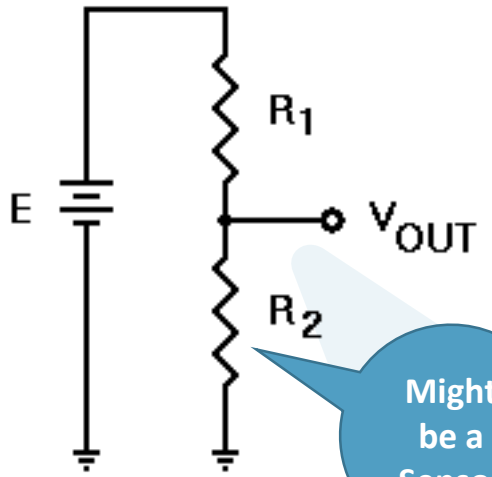
- Series and Parallel Resistors
  - Series: $R_{EQ} = R_1 + R_2 + \ldots + R_n$
  - Parallel: $1/R_{EQ} = 1/R_1 + 1/R_2 + \ldots + 1/R_n$

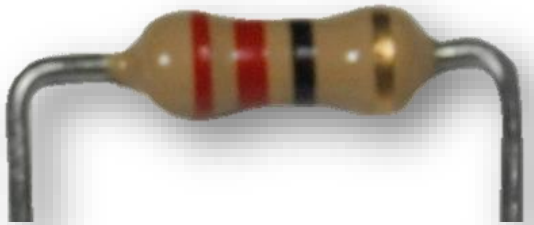


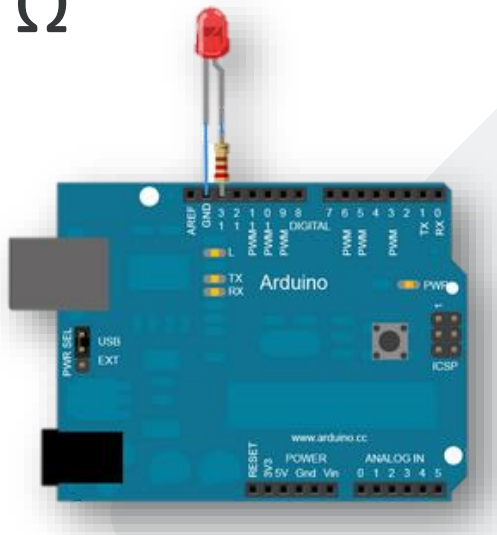


Resistive divider, LED current limiter

- Voltage divider: $V_{OUT} = V_{IN}*(R_2/(R_1 + R_2))$
  - 3.3V supply, R_1 = 5kΩ, R_2 = 10kΩ
    V_{OUT} = 3.3V*(10kΩ/(10kΩ+5kΩ)) = 2.2V
- LED resistor: $R = (V_{SUPPLY} - V_{LED})/I_{LED}$
  - 5V supply, 0.7V 20mA LED: R = (5V – 0.7V)/20mA = 215Ω
    ➔ Nearest higher rated resistor 220 Ω



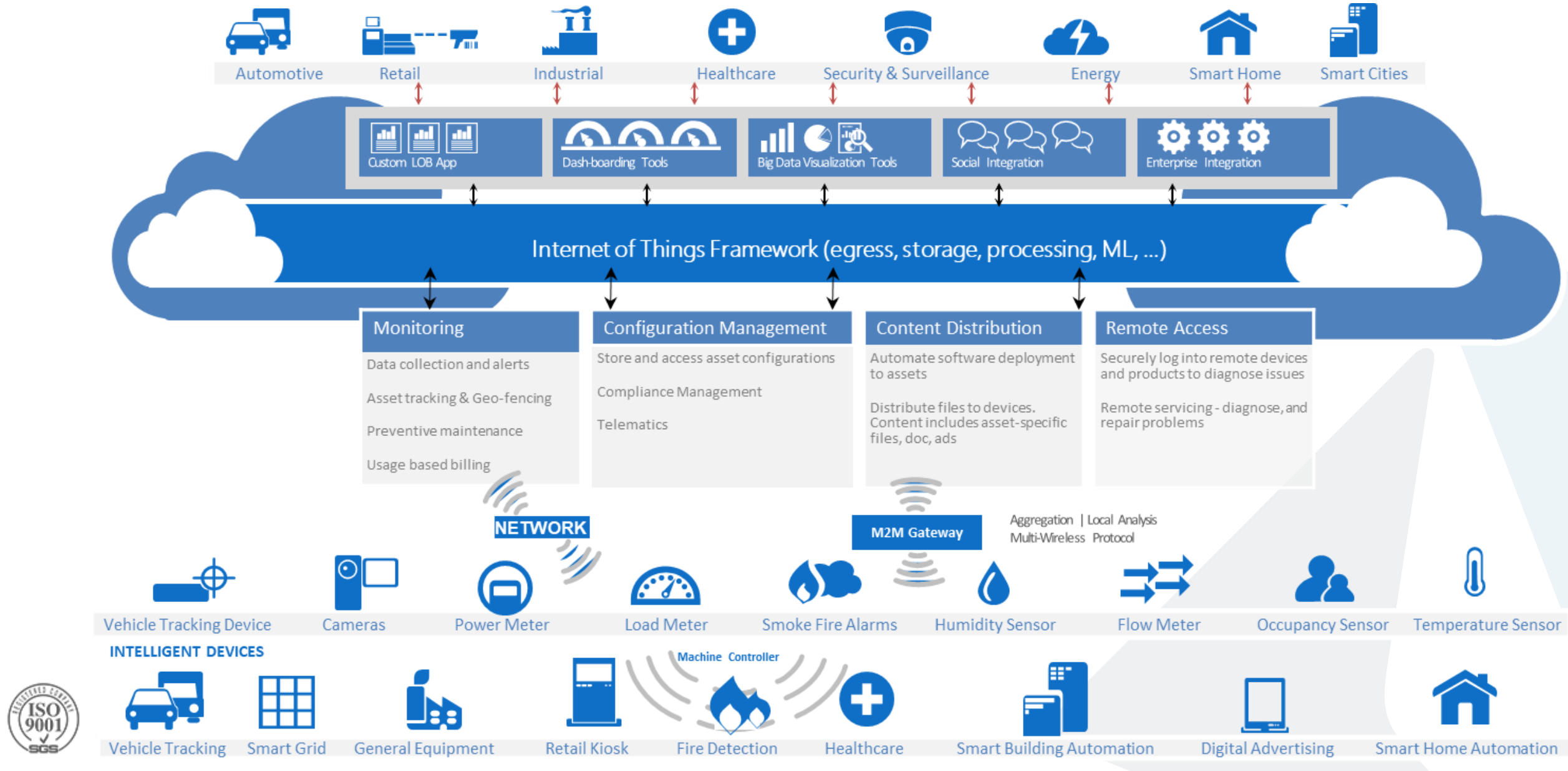




The Modern IoT System

- Most large IoT systems include one or more of the following:
  - Many different end **devices** with **sensors** and **actuators**
  - Local **gateways**
  - A collection of **cloud services** that enables:
    - **Registration** of end devices
    - **Management** of end devices
    - **Controlling** of end devices
    - Different **communication protocols** that provide reliability and security
    - The ability to **collect a vast amount** of data in a very **high rate**
    - The ability to **analyze** the **stream** of information in **close to real-time** manner
    - The ability to **analyze** the **current** and **historical** collected information
    - The ability to **show** the resulted **conclusion** and the **collected data**

High Level Architecture

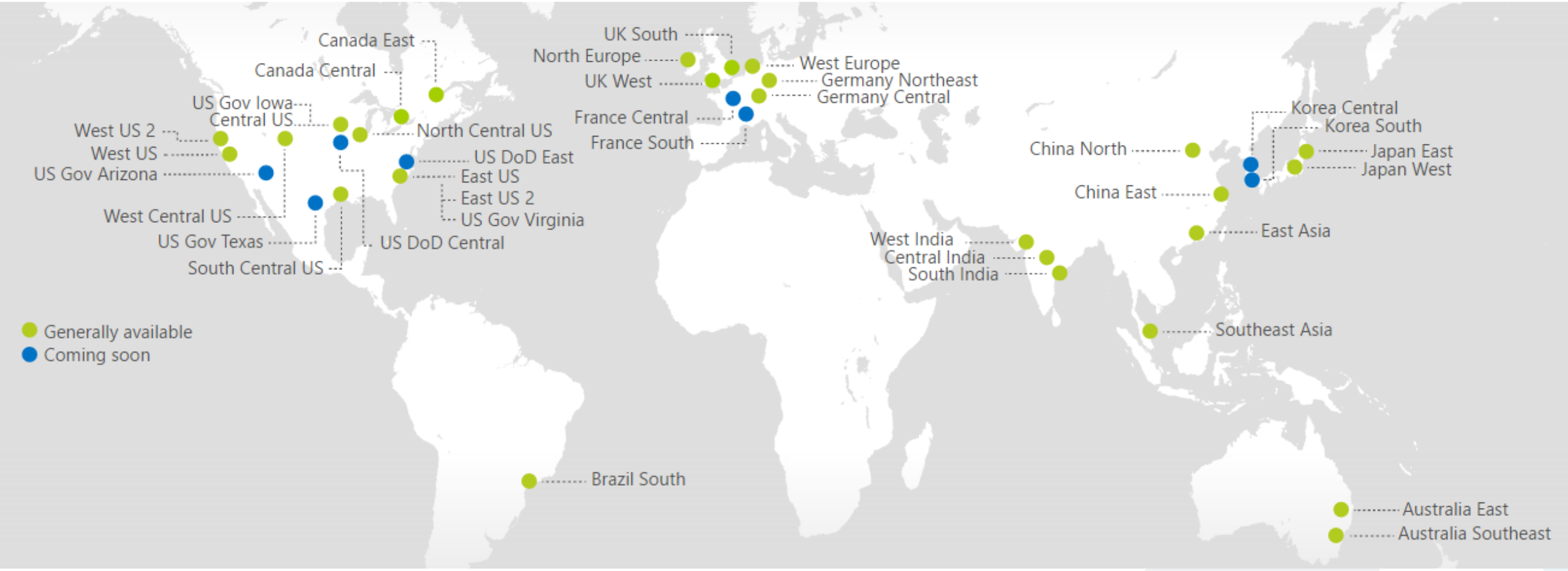


Azure: the color of the sky on a clear summer's day

Wikipedia

Azure Regions

38 Regions Worldwide, 30 ONLINE…huge capacity around the world…growing every year



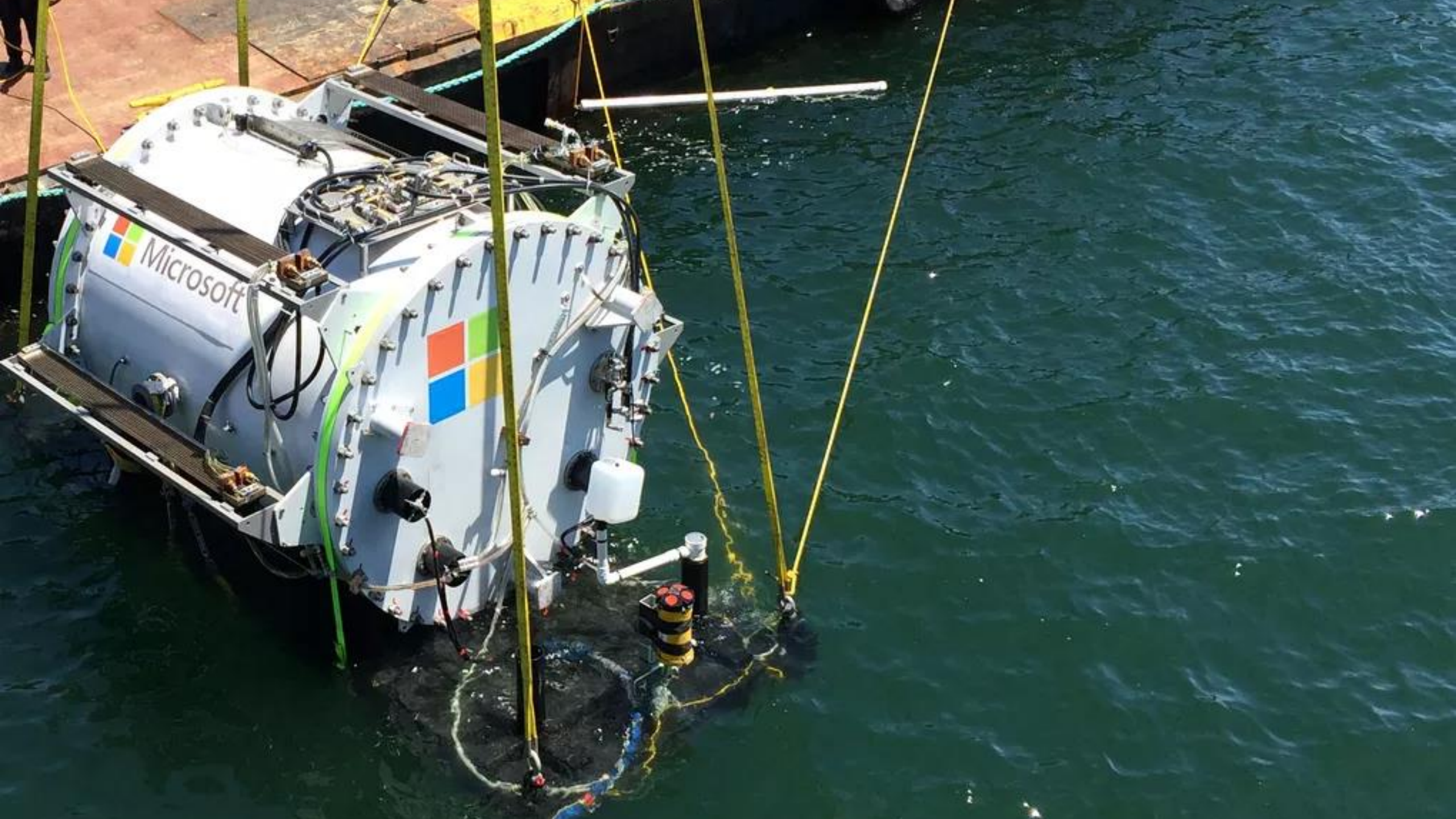
- 100+ datacenters
- Top 3 networks in the world
- Second Largest Dark Fiber Network

- China Operated by 21Vianet
- Germany Operated by Deutsche Telekom
- 2.5x AWS, 7x Google DC Regions

Operational

Announced/Not Operational

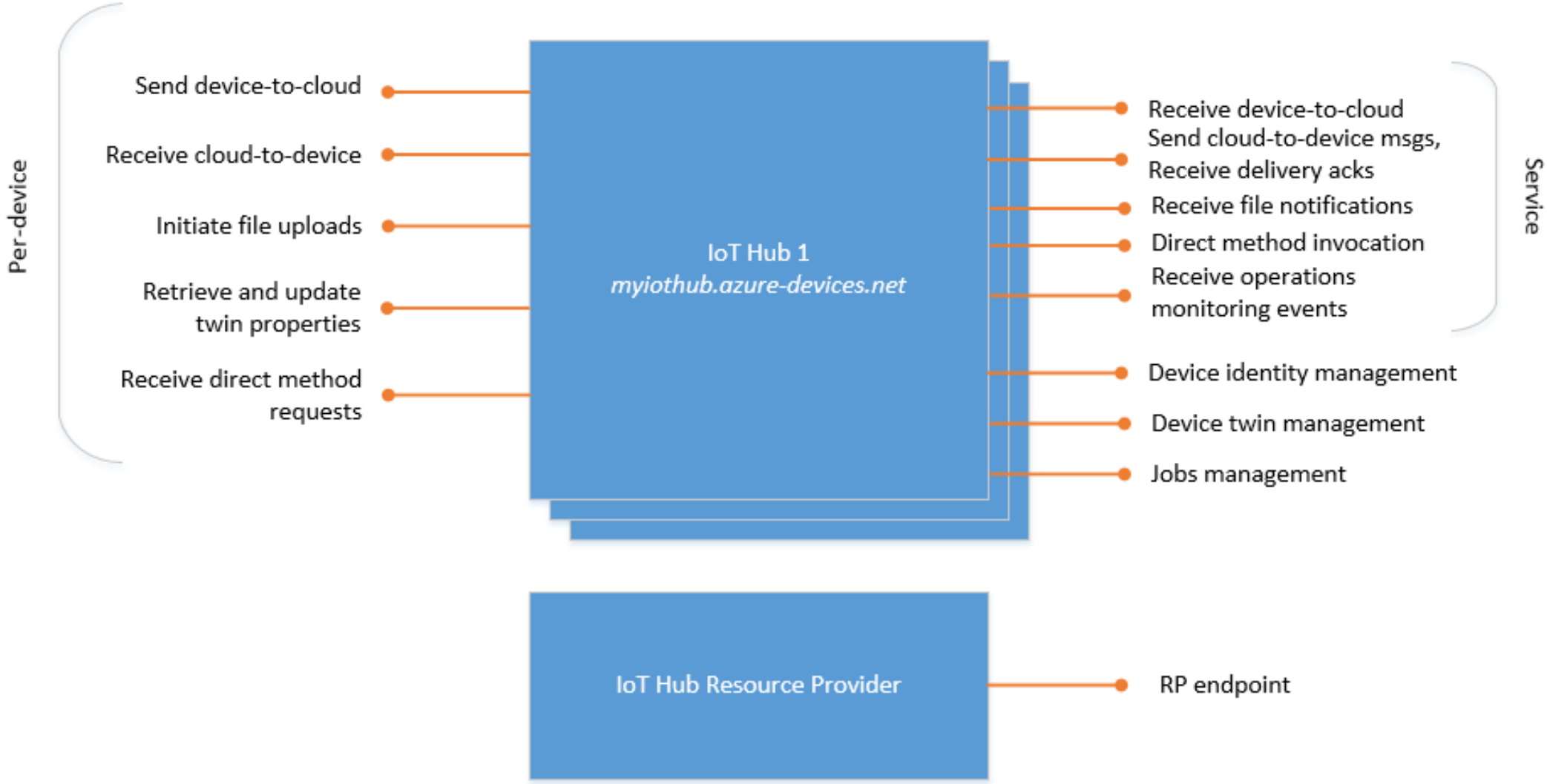
https://azure.microsoft.com/en-us/regions


Microsoft

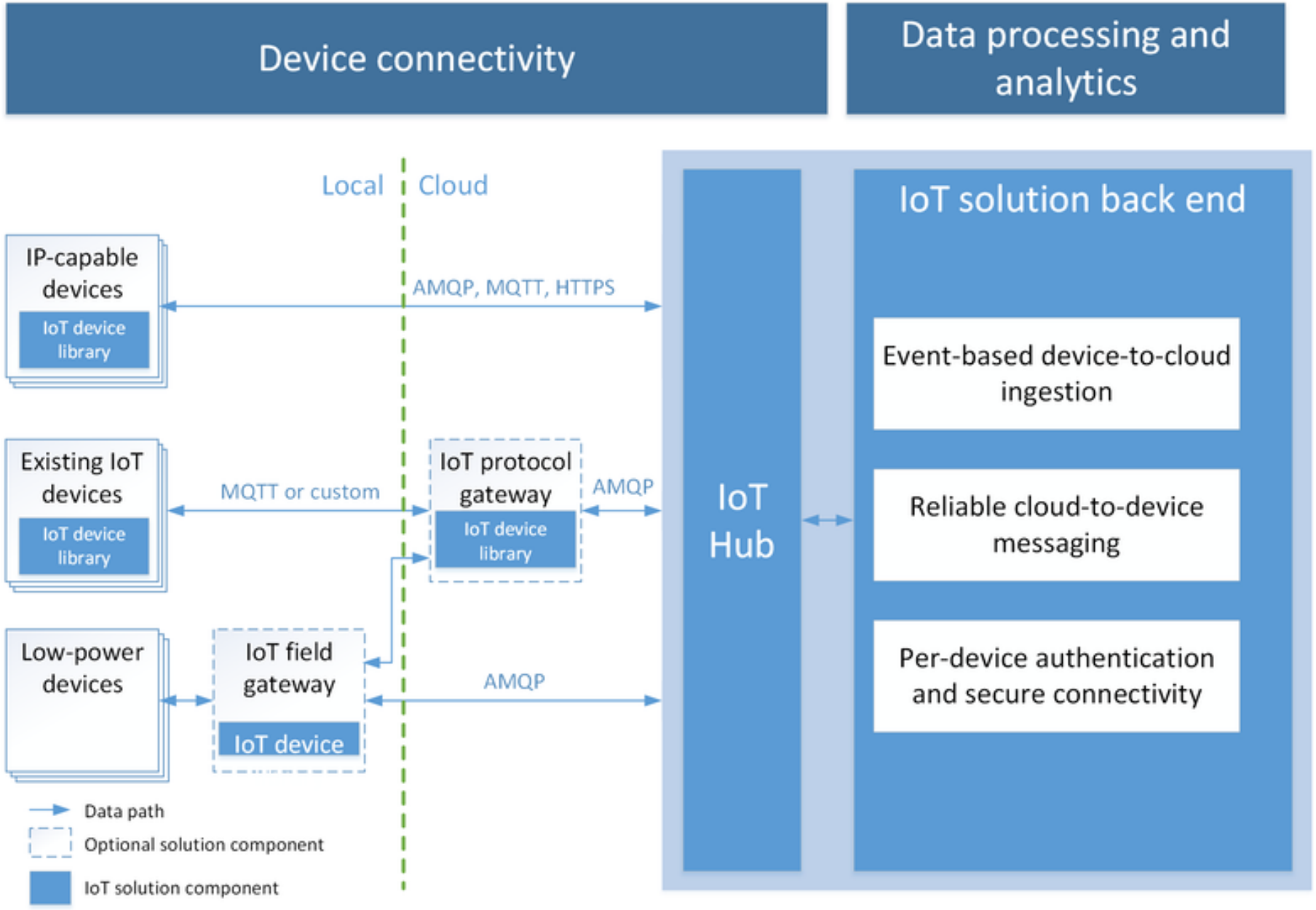
Introducing Microsoft Azure IoT Hub

- IoT Hub is available as a stand-alone service or as one of the services used in the new Azure IoT Suite
- Azure IoT Hub is designed to connect your devices to Azure. It supports:
  - Millions of simultaneously connected devices
  - Per-device authentication
  - High throughput data ingestion
  - Scale device management
  - Reliable command and control

Azure IoT Hub



Azure IoT Hub



Azure IoT Hub

Designed for IoT

- Connect up to 10 million devices

Service assisted communications

- Secure bi-directional communication
- Command and control

Cloud-scale messaging

- Device-to-cloud and Cloud-to-device
- Durable messages ("at least once" semantics)

Cloud-facing telemetry ingestion

- Delivery receipts, expired messages
- Device communication errors

Per-device authentication

- Individual device identities and credentials

Connection multiplexing

- Single device-cloud connection for all communications (C2D, D2C)

Multi-protocol support

- Natively supports AMQP, MQTT, HTTP
- Designed for extensibility to custom protocols

Multi-platform

- Device SDKs available for multiple platforms (e.g. RTOS, Linux, Windows, Arduino)
- Multi-platform Service SDKs

Azure IoT Hub SDKs

- **Device-facing**
  - For devices and field gateways
- **Platforms**
  - Many devices
  - RTOS (FreeRTOS)
  - Linux
    (Ubuntu, Debian, Fedora, Raspbian, Angstrom)
  - Windows 7/8/10
  - ARM mbed
  - Android
  - iOS
- **Device SKD by programming language**
  - For device side development
  - Azure IoT device SDK for C
  - Azure IoT device SDK for .NET
  - Azure IoT device SDK for Java
  - Azure IoT device SDK for Node.js
  - Azure IoT device SDK for Python

- **Service-facing SDK by programming language**
  - For back-ends and cloud gateways
  - Azure IoT service SDK for .NET
  - Azure IoT service SDK for Node.js
  - Azure IoT service SDK for Java
  - Azure IoT service SDK for Python
- **Azure IoT Gateway SDK**
  - Infrastructure and modules to create IoT gateway solutions
- **Azure IoT Hub REST API**
  - For all the rest…
- **Advance IoT Hub topics**
  - IoT Hub endpoints
  - IoT Hub query language for device twins and jobs
  - Quotas and throttling
  - IoT Hub MQTT support

C Language Device SDK

- Many low price, low energy, SoC can be developed only by using the C language
- The IoT team has built a full-blown C SDK to connect and communicate with the IoT Hub
  - It supports all IoT Hub Device capabilities, including:
    - Secure connection and communication using three protocols (HTTP, AMQP, MQTT)
    - Sending telemetry messages using JSON serialization and set of macros to provide message serialization
    - Receiving messages from the cloud
    - Handling device twin synchronization
    - Invoke a function with request-reply message exchange pattern when the IoT Hub calls
    - Upload files
- There are two levels of functions:
  - With *_LL_* - low level API – for device that has no threading capabilities
  - With no *_LL_* - support background message processing using threads
- Follow this intro to understand the various functions

Connecting and Defining A Model

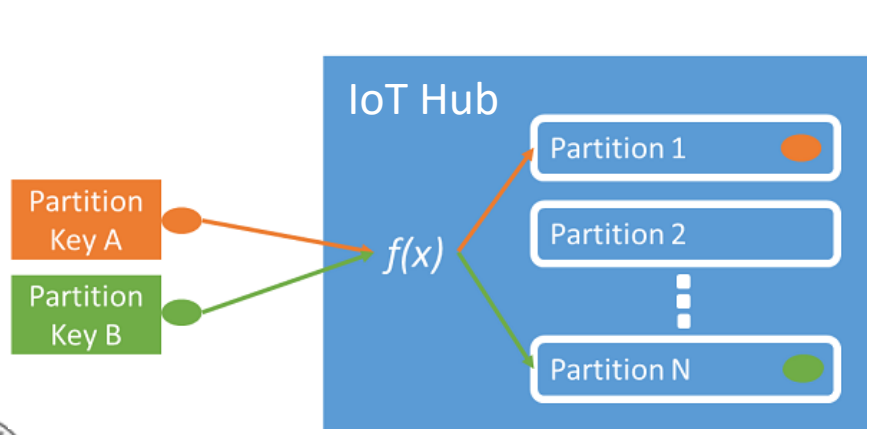
```
if ((iotHubClientHandle = IoTHubClient_LL_CreateFromConnectionString(connectionString, MQTT_Protocol)) ==
{
    (void)printf("ERROR: iotHubClientHandle is NULL!\r\n");
}
else
{
```

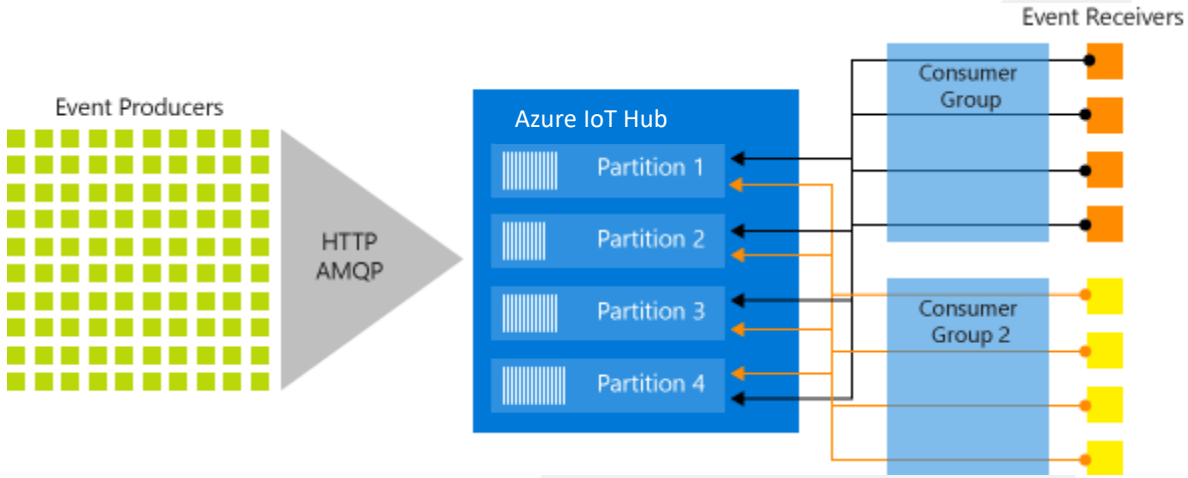
```
BEGIN_NAMESPACE(WeatherStation);

DECLARE_MODEL(ContosoAnemometer,
WITH_DATA(ascii_char_ptr, DeviceId),
WITH_DATA(int, WindSpeed),
WITH_ACTION(TurnFanOn),
WITH_ACTION(TurnFanOff),
WITH_ACTION(SetAirResistance, int, Position)
);

END_NAMESPACE(WeatherStation);
```

```
ContosoAnemometer* myWeather = CREATE_MODEL_INSTANCE(WeatherStation, ContosoAnemometer);
if (myWeather == NULL)
{
    (void)printf("Failed on CREATE_MODEL_INSTANCE\r\n");
}
else
{
```

Processing IoT Hub Messages – Event Hub

- You can process IoT Hub device to cloud messages using either:
  - The built-in Event-Hub compatible endpoint
  - Rout the events to an Azure Service Bus queue
- Azure Event Hub is a very powerful telemetry ingestion service that was created by the Service Bus team
  - The key to scale for Event Hubs is the idea of **partitioned consumers**
    - **Partitioned consumers** enables very high scale by removing the contention bottleneck and facilitating end to end parallelism





Automatic Routing an IoT Message

- You can route device-to-cloud messages to a custom endpoint based on the message properties
  - Create a routing rule to send messages to an instance of an Event Hub, a Service Bus Queue and a Service Bus Topic
  - Message can be routed to multiple custom endpoint if it match multiple conditions
  - Routing can be done by querying the message header, or message body, using SQL like syntax

```
$body.message.Weather.Location.State = 'WA'
$body.Weather.HistoricalData[0].Month = 'Feb'
$body.Weather.Temperature = 50 AND
$body.message.Weather.IsEnabled
length($body.Weather.Location.State) = 2
$body.Weather.Temperature = 50 AND Status = 'Active'
```

IoT Hub and IoT Device Communication Protocols

- IoT Hub supports three protocols:
  - HTTP – Use for devices that cannot support other protocols or that are rarely connected
  - AMQP – Use on field and cloud gateways to take advantage of connection multiplexing across devices
  - MQTT – Extremely lightweight, Use on all devices that do not require to connect multiple devices
- You can choose to use any protocol, however you need to take some protocol characteristics into considerations:
  - HTTP does not have an efficient way to implement server push
    - As such, when you are using HTTP, devices poll IoT Hub for cloud-to-device messages
  - AMQP returns errors for many conditions, while MQTT terminates the connection
    - As a result your exception handling logic might require some changes
  - MQTT does not support the *reject* operations when receiving cloud-to-device messages
    - If your back-end app needs to receive a response from the device app, consider using direct methods
  - The MQTT and HTTP libraries have a smaller footprint than the AMQP libraries

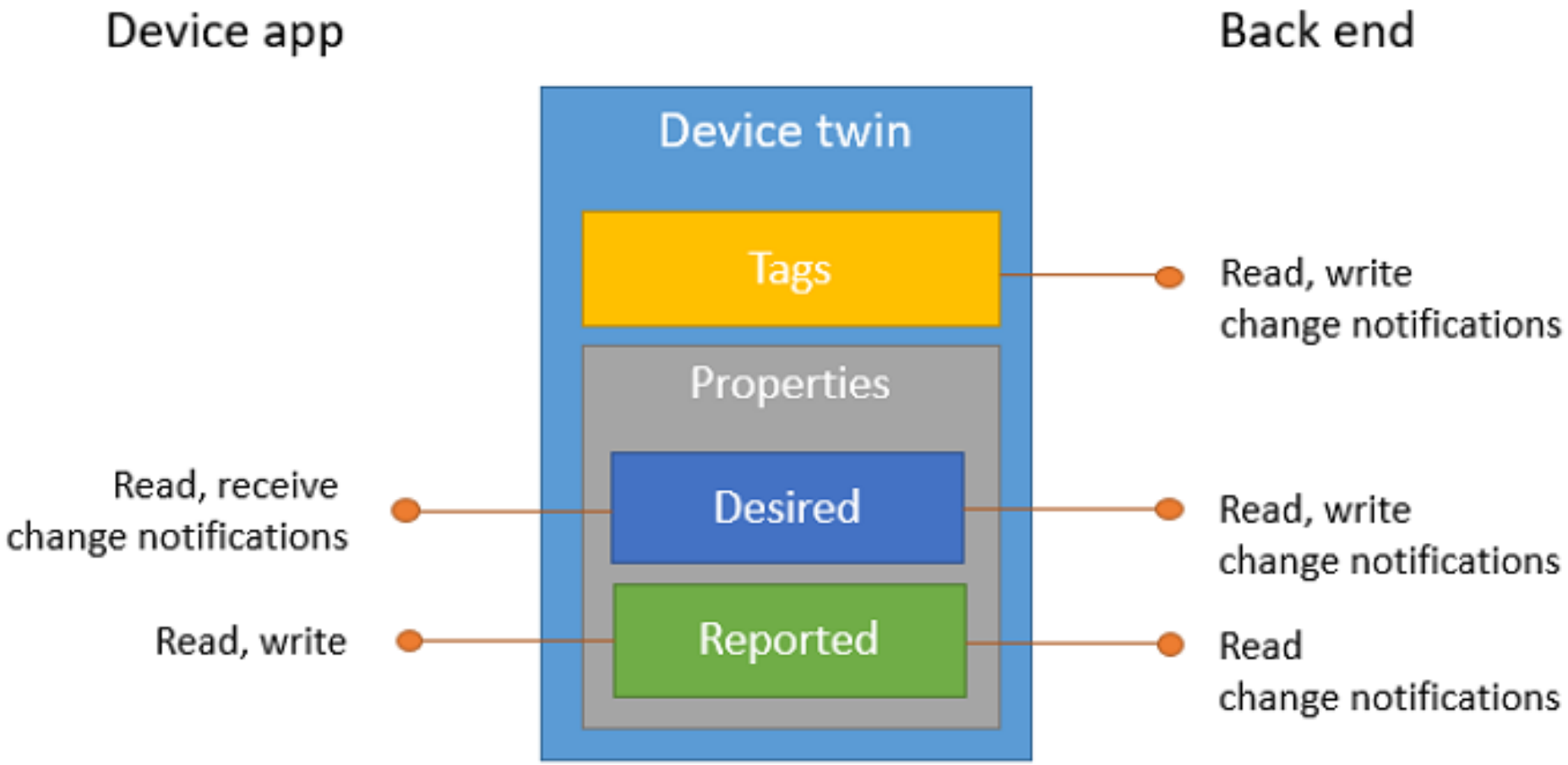
Port Numbers

Protocol	Port
MQTT	8883
MQTT over WebSockets	443
AMQP	5671
AMQP over WebSockets	443
HTTP	443

Device Twins

- Device twins are JSON documents that store device state information:
  - metadata, configurations, and conditions
- The IoT Hub persists a device twin for each registered device
- Use device twins to:
  - Store device-specific metadata in the cloud
  - Report current state information such as available capabilities and conditions from your device app
  - Synchronize the state of long-running workflows between device app and cloud app
  - Query your device metadata, configuration, or state
  - Get notified when a twin is modified

Device Twins



Direct Method - Calling a function in the device

- As opposed to other IoT Hub message exchange patterns that are one-way, a method call is a request-reply interaction
  - Other cloud to device communication are based on sending messages to the device, or setting desired properties
- Each device method targets a single device
  - Jobs provide a way to invoke direct methods on multiple devices, and schedule method invocation for disconnected devices
- Direct methods are synchronous and either succeed or fail
  - Failure occurs after a timeout period (default: 30 secs, settable up to 1 Hour)
- Great for interactive scenarios such as turning on a light from a phone
- Direct method are HTTP-only from the cloud side, and MQTT-only from the device side
- The payload for method requests and responses is a JSON document up to 8KB

Direct Method

```
private static async Task InvokeMethod()
{
    var methodInvocation = new CloudToDeviceMethod("writeLine") { ResponseTimeout = TimeSpan.FromSeconds(30) };
    methodInvocation.SetPayloadJson("'a line to be written'");

    var response = await serviceClient.InvokeDeviceMethodAsync("myDeviceId", methodInvocation);

    Console.WriteLine("Response status: {0}, payload:", response.Status);
    Console.WriteLine(response.GetPayloadAsJson());
}
```

```
serviceClient = ServiceClient.CreateFromConnectionString(connectionString);
InvokeMethod().Wait();
Console.WriteLine("Press Enter to exit.");
Console.ReadLine();
```

Device Side C SDK – Handling direct method

```
else if (IoTHubClient_LL_SetDeviceMethodCallback(iotHubClientHandle, DeviceMethodCallback, myWeather) != IOTHUB_CLIENT_OK)
{
    (void)printf("Failed on IoTHubClient_SetDeviceMethodCallback\r\n");
}
```

Device Jobs

- To handle massive amount of devices and to communicate with offline devices, use Jobs:
  - Jobs encapsulate the execution of device twin updates and direct methods against a set of devices at a schedule time
  - The job is described as a JSON document
- Jobs are initiated by the cloud app and maintained by IoT Hub
  - Once a job is initiated, querying for jobs enables the cloud app to refresh the status of running jobs
- More information

Upload Files

- Use file upload to send media files and large telemetry batches
- You must first link an Azure Storage account to the IoT Hub
  - You can do that using the portal
- The device initiates an upload
- When the upload completes, the device notifies the IoT hub
- See file upload notifications
- The SDK makes it easy: (C#)

```
private static async void SendToBlobAsync()
{
    string fileName = "image.jpg";
    Console.WriteLine("Uploading file: {0}", fileName);
    var watch = System.Diagnostics.Stopwatch.StartNew();

    using (var sourceData = new FileStream(@"image.jpg", FileMode.Open))
    {
        await deviceClient.UploadToBlobAsync(fileName, sourceData);
    }

    watch.Stop();
    Console.WriteLine("Time to upload file: {0}ms\n", watch.ElapsedMilliseconds);
}
```

Azure IoT Edge

- Azure IoT Edge is:
  - The evolution of the Azure IoT Gateway SDK
  - A service that extends cloud capabilities to the edge and distributes intelligence across IoT devices
  - Open-source and cross-platform support for building custom logic at the edge
- Azure IoT Edge provides offline intelligent scenarios
  - Analytics, Machine Learning
- Azure IoT Edge is under private preview and will be available later this year

Azure IoT Hub Summary

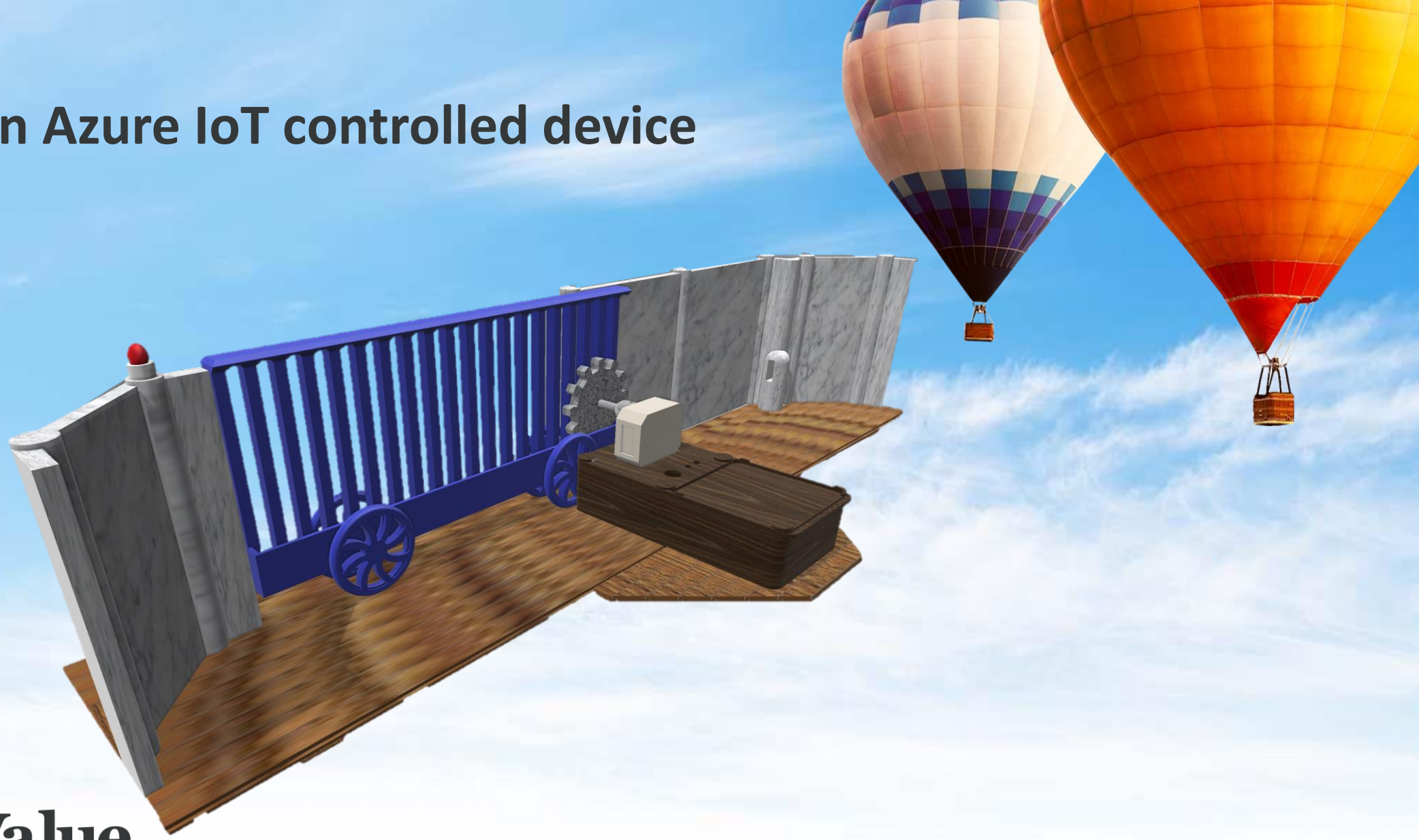
- Azure IoT Hub is designed to connect your devices to Azure. It supports:
  - Millions of simultaneously connected devices
  - Per-device authentication
  - High throughput data ingestion
  - Scale device management
  - Reliable command and control
  - HTTP, MQTT, AMQP communication protocols
  - Cloud to Device and Device to Cloud messaging
  - State transfer with device twins
  - Query language
  - Job Management
  - File Upload
  - Many platforms using many programming language SDKs

The Citizen App

- Make any citizen a sensor(s)
  - Text, picture, video, audio, location
- Large city scale – 10s Millions
- Server-less Architecture:
  - IoT Hub per city
  - Stream Analytics
  - CosmosDB
  - Azure Functions
  - File uploads
- Small team: 2-3 developers, 6-8 months

Amazing!

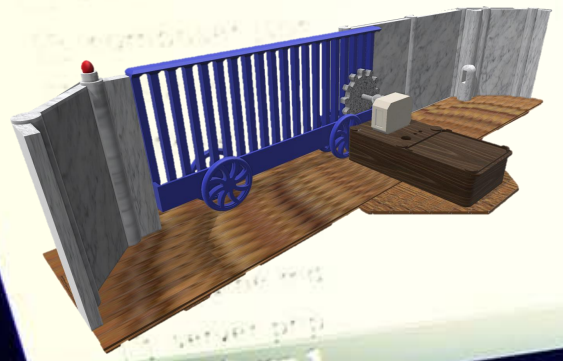
Building an Azure IoT controlled device



CodeValue

Inspiring Code, Creating Value.

Demo



```
68      $mockQueryBuilder->shouldReceive('newQuery')->once()->andReturn($query);
69      $relation->expects($this->once())->method('touchIfTouching');
70
71      $this->assertTrue($relation->detach());
72  }
73
74
75  public function getRelation()
76  {
77      list($builder, $parent) = $this->getRelationArguments();
78
79      return new MorphToMany($builder, $parent, 'taggable', 'taggables', 'taggable_id', 'tag_id');
80  }
81
82  public function getRelationArguments()
83  {
84      $parent = m::mock('Illuminate\Database\Eloquent\Model');
85      $parent->shouldReceive('getMorphClass')->andReturn(get_class($parent));
86      $parent->shouldReceive('getKey')->andReturn(1);
87
```

Line 81, Column 1

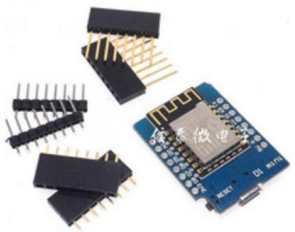
ESP8266 – The Right Device for the Job

Free shipping 1pcs/lot **ESP8266** ESP-01 remote serial Port WIFI wireless module through walls Wang

BEST CHOICE ELECTRONICS CO.,LTD

Offline

US $1.68 / piece
Free Shipping
Feedback(220) | Orders (244)
Add to Wish List

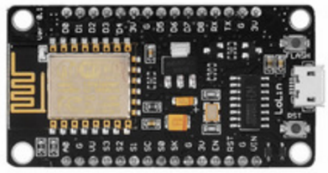


D1 mini - Mini NodeMcu 4M bytes Lua WIFI Internet of Things development board based ESP8266 by **WeMos**

BEST CHOICE ELECTRONICS CO.,LTD

Offline

US $2.85 / piece
Free Shipping
Feedback(31) | Orders (66)
Add to Wish List



CH340G **NodeMcu** V3 Lua Wireless Internet Of Things Development Board 3.3V Network WIFI Connector Module Based ESP8266 ESP-12E

Worldchips

Chat now!

US $2.98 / piece
Free Shipping
Orders (68)
Add to Wish List

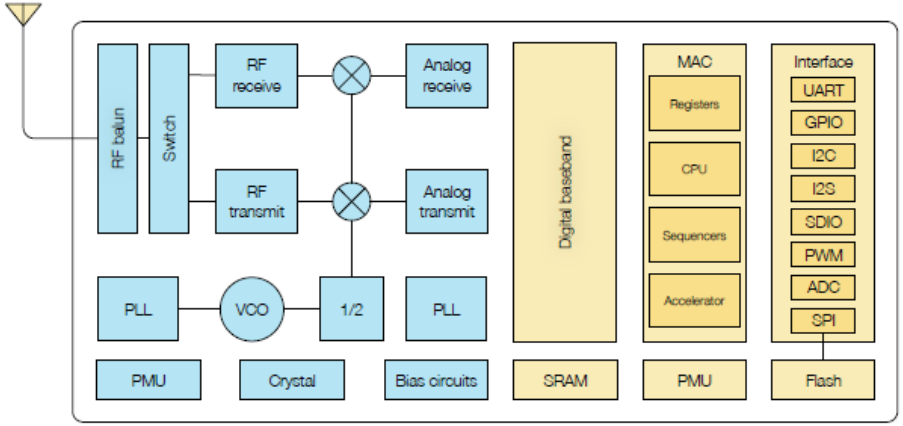
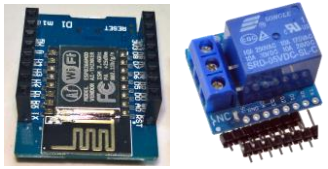
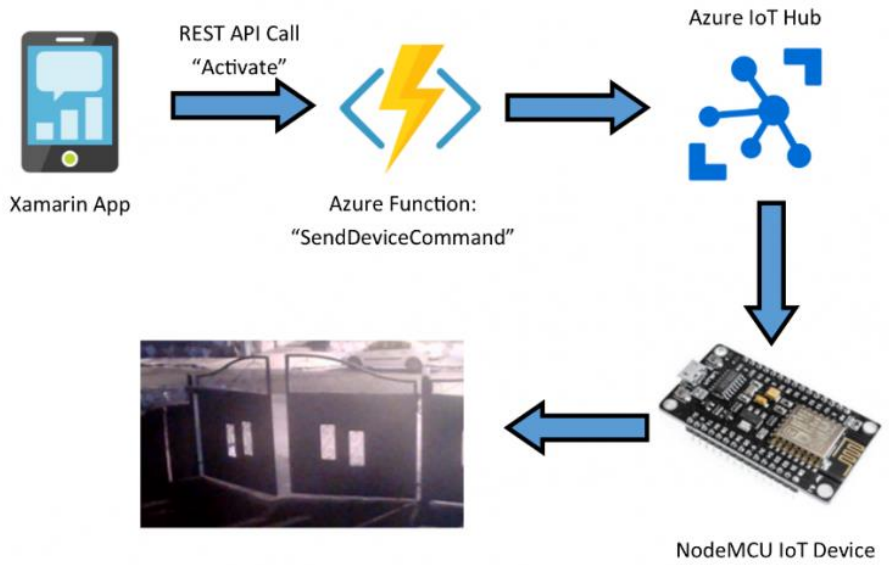


Figure 1-1: ESP8266EX Block Diagram

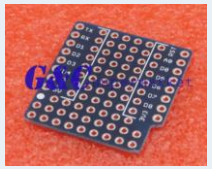
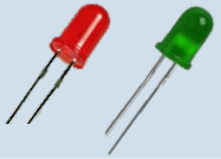
The Basic System



- WeMos D1 Mini Pro + Relay Shilled
- Azure IoT Hub
- Xamarin App with One Button
- Azure IoT Hub Device SDK C App
- Building it
  - Create an IoT Hub using Azure Portal
  - Connect The Relay Shield to the WeMos D1
  - Connect the WeMos D1 to the computer
  - Write simple Arduino app that triggers the relay on Azure IoT Hub Command
    - Create and use the device connection string
  - Compile and download the Arduino code
  - Test using Device Explorer
  - Create Azure Function that sends commands to devices
  - Write a simple Xamarin App that sends the command using the Azure Function



Our Cloud Gate Controller BOM < $10

Device	Picture	Price
WeMos D1 Mini		$2.85
WeMos ProtoBoard Shield		$0.68
WeMos Relay Shield		$1.00
Button		0.0143
5 mm Led X 2 (Red & Green) + 330Ω Resistor X 2	 	$0.0089 X 2 $0.0059 X 2
Power Supply Module + AC 110C-240V Converter	 	$0.58 $2.08

The V1 System Problem

- The Gate model system uses a stepper motor
  - The stepper motor requires a smooth sequence
- To have a smooth movement, the sequence has to keep the same frequency
  - Azure IoT Client SDK, does polling from time to time, and delays this sequence!

Clockwise Rotation ➡

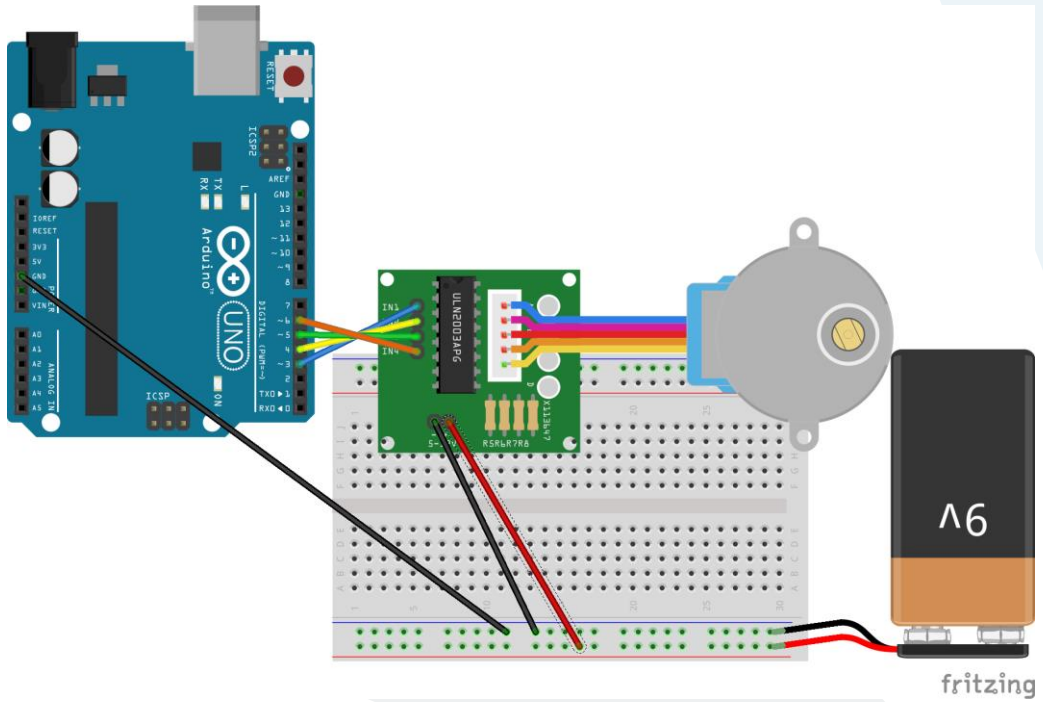
Index	1a	1b	2a	2b
1	1	0	0	1
2	1	1	0	0
3	0	1	1	0
4	0	0	1	1
5	1	0	0	1
6	1	1	0	0
7	0	1	1	0
8	0	0	1	1

Alternate Full Step Sequence
(Provides more torque)

Clockwise Rotation ➡

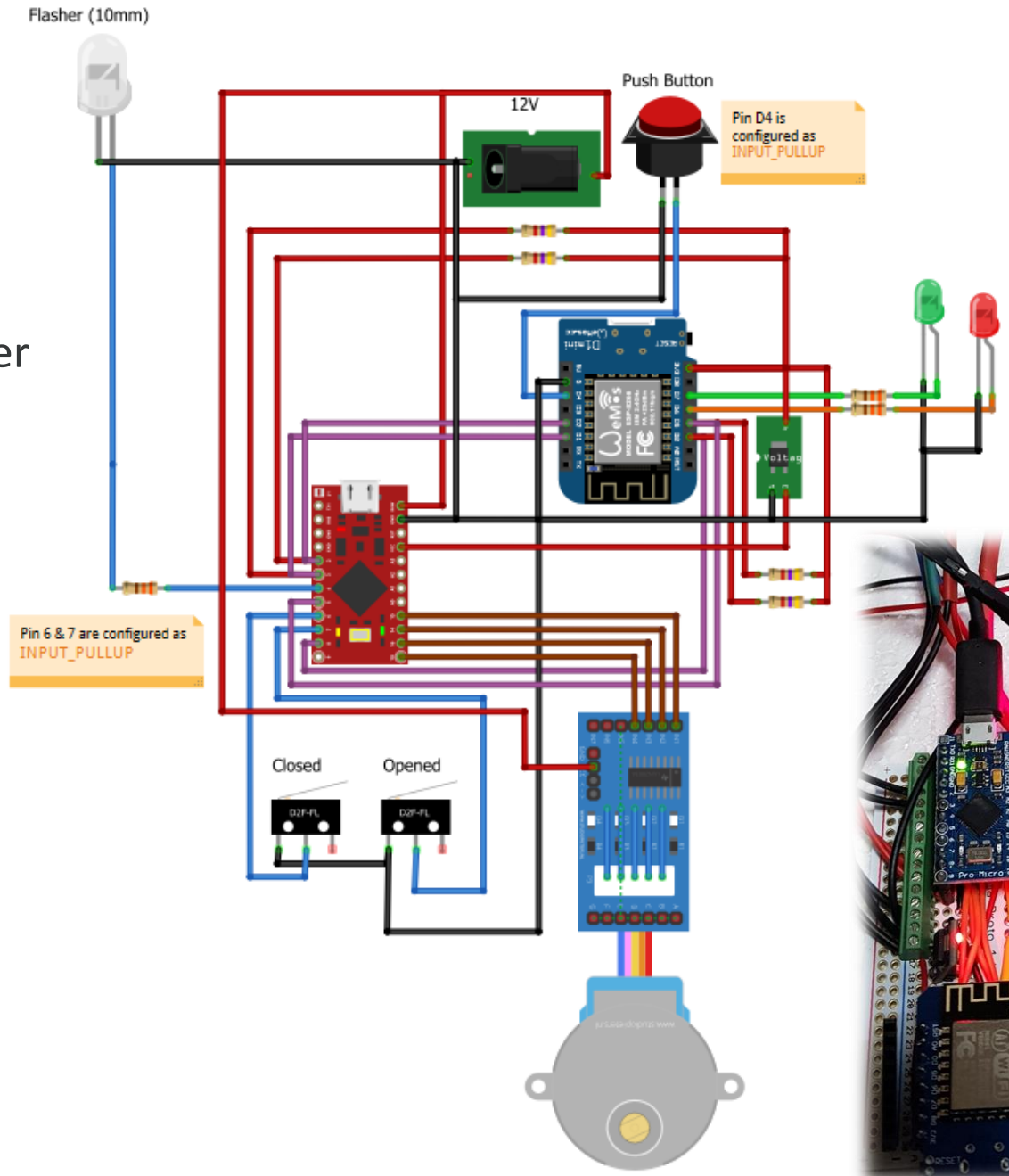
Index	1a	1b	2a	2b
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1
9	1	0	0	0
10	1	1	0	0
11	0	1	0	0
12	0	1	1	0
13	0	0	1	0
14	0	0	1	1
15	0	0	0	1
16	1	0	0	1

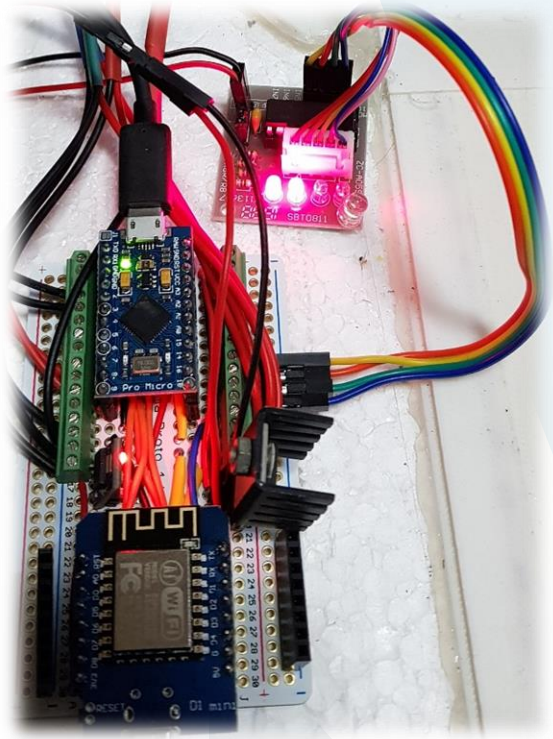
Half Step Sequence



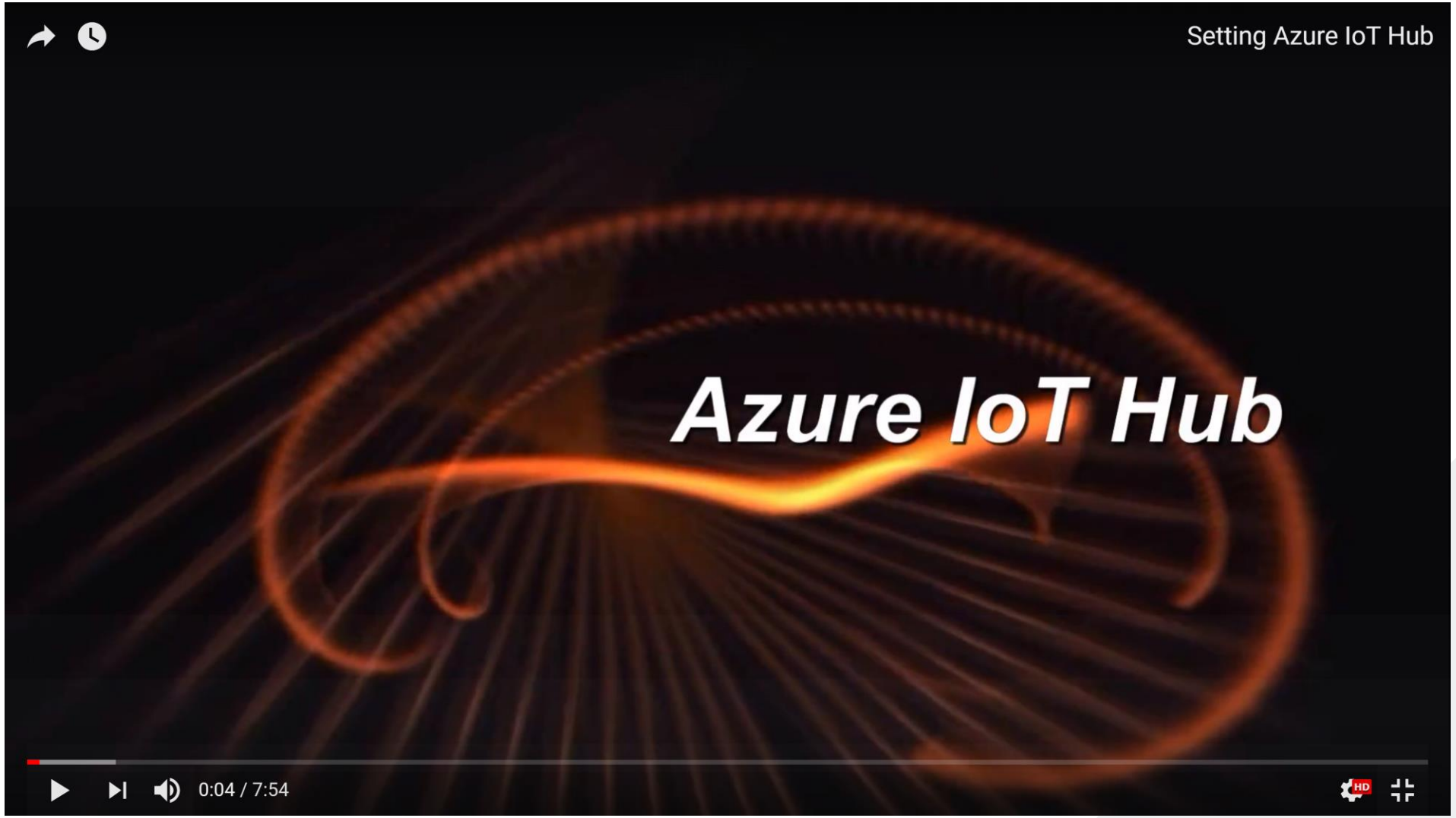
The V2 Solution: Have Two Arduinos

- The WeMod D1 – The Cloud Controller Smooth connectivity
- The Pro Micro – The Gate Controller
  - Smooth gate operation
- Communication, can be:
  - I^2C, Serial, SPI, Custom
- Since there is a conflict of voltage 3.3V to 5V, I chose to use pullup resistors, and implement a custom communication protocol



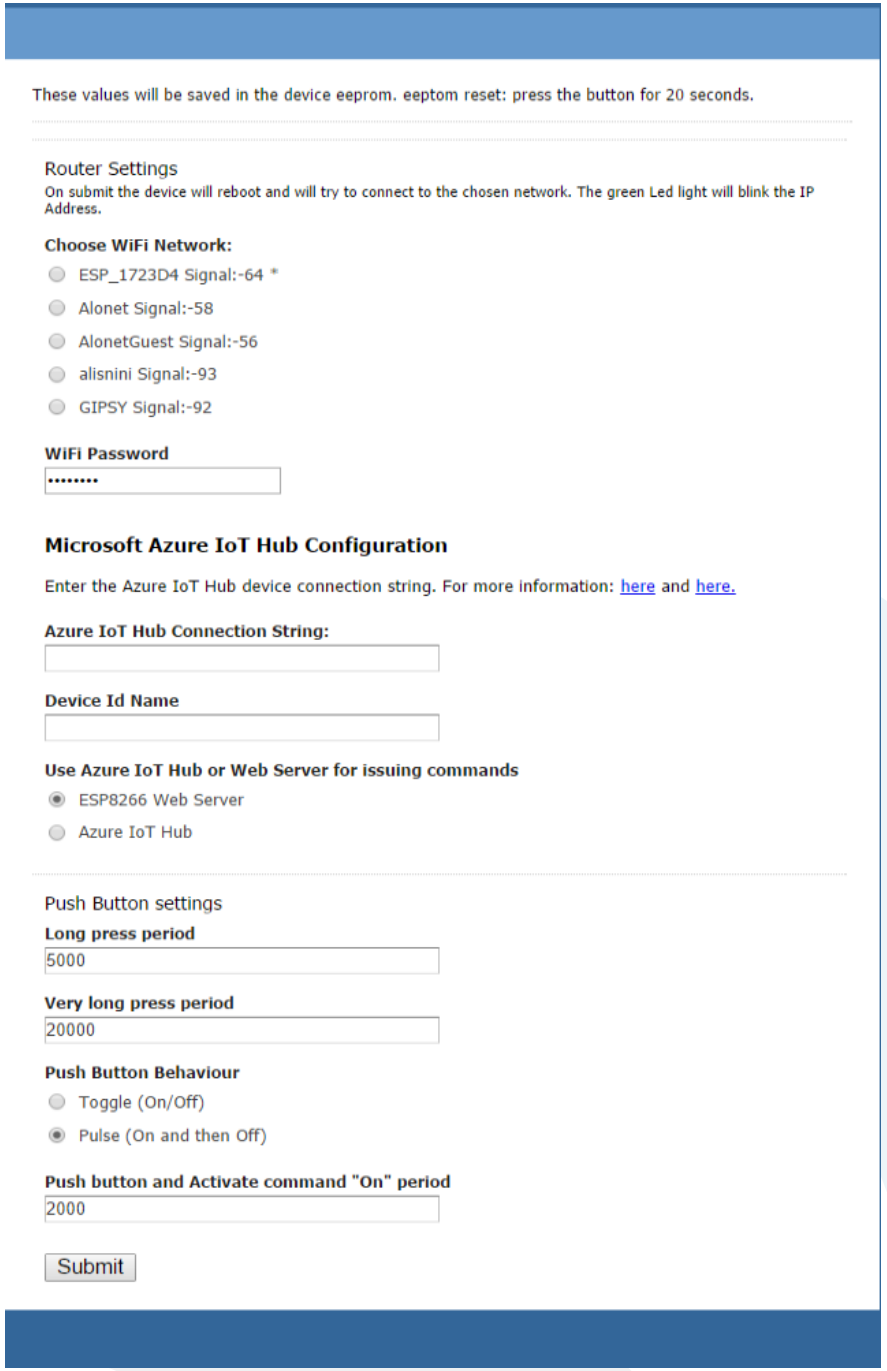


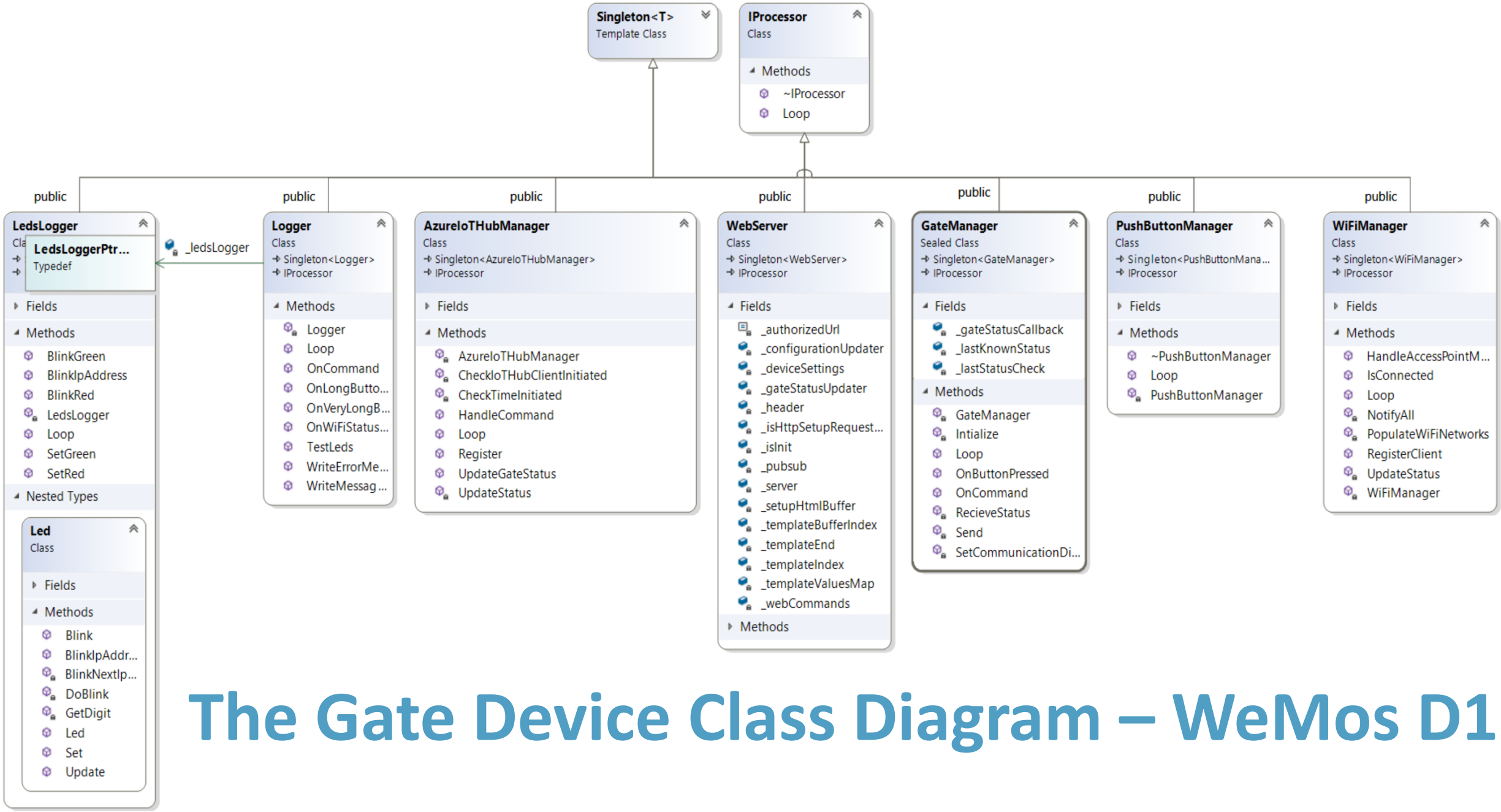
Setting up the IoT Hub and Azure Services



The Gate System

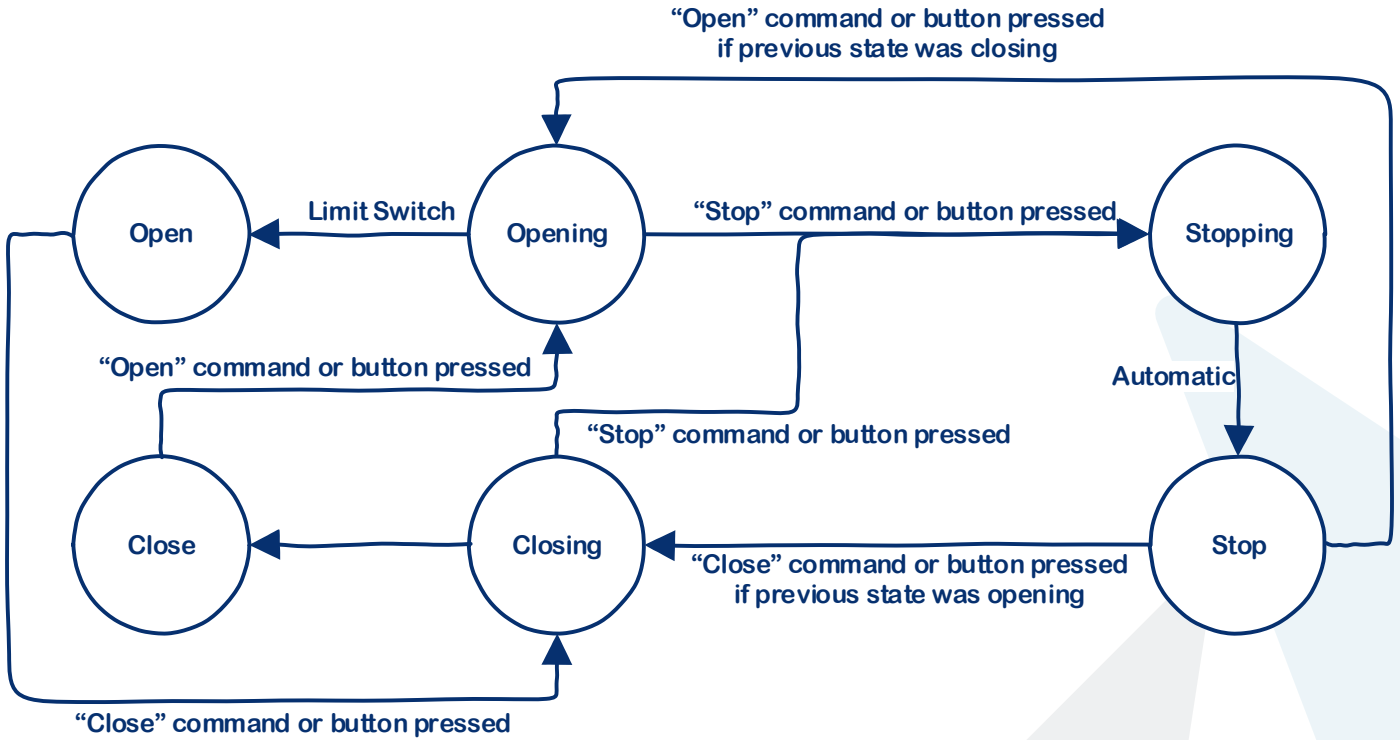
- A complete solution:
  - Provides HTTP web interface for setup and operation
  - Access point mode to enables device settings and secure registration via a dedicated web page
  - Once registered to an IoT hub, all communication is secured and done only with the cloud
    - Send the command “web” or reset the device to get back the web interface
- Communication:
  - Accepts the commands Open, Close, Stop, Web
  - Provides Status: Open, Opening, Stopped, Closing & Close





The Gate Device Class Diagram – WeMos D1

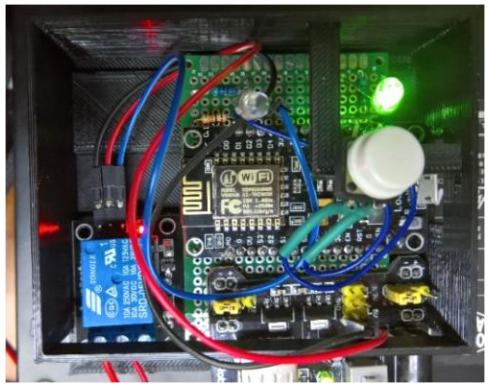
Gate Sliding State Machine – Pro Micro



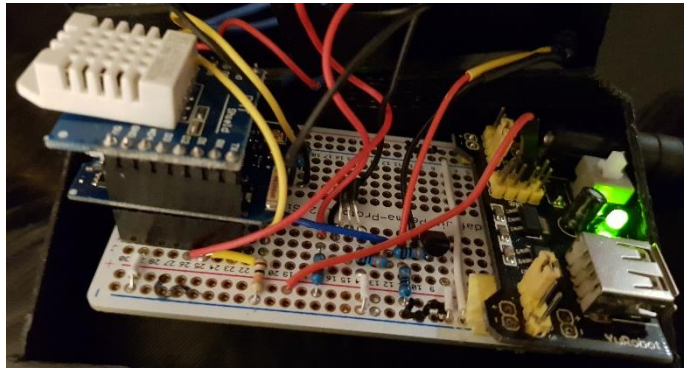
Activating a Real Gate

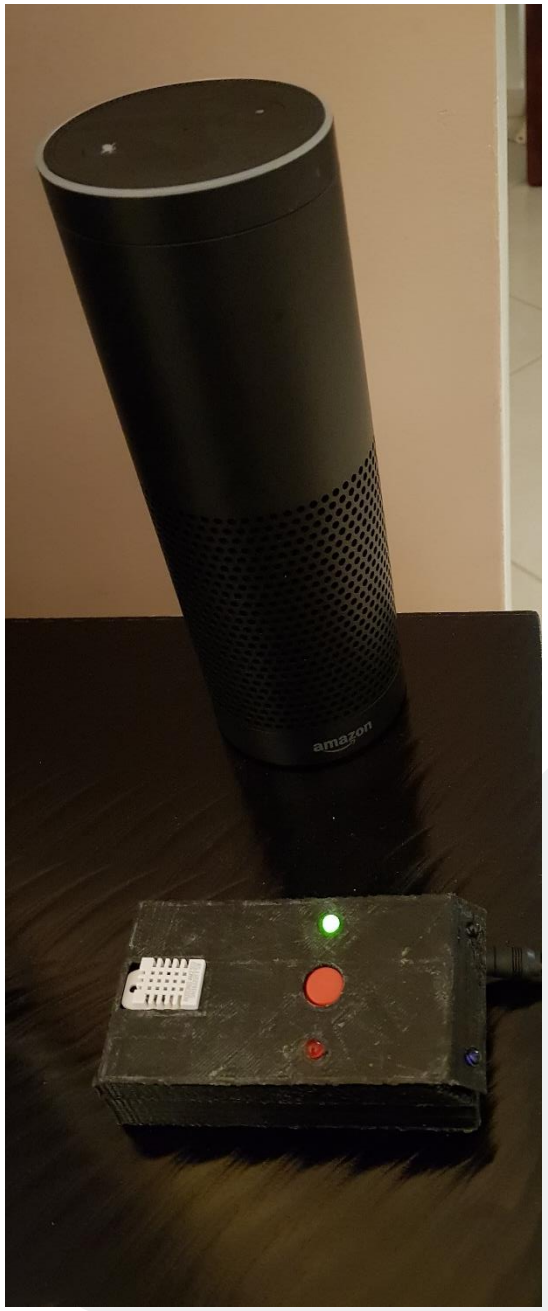
Other Usage

- My 3D Printer LED light



- Alexa: Turn on ground level AC
- Alexa set ground level AC to 20 degrees





Summary

- We have seen:
  - How easy it is to build and develop an IoT system using Arduino + ESP8266 + Azure + Visual Studi with Visual Micro

Resources

- Demo code:
  - https://github.com/alonf/BasicGateController
  - Setup IoT Hub video: https://youtu.be/vq5AeLlsWx4
- My MSDN articles:
  - Introduction to the Internet of Things – From the Device to Microsoft Azure Cloud
    - https://blogs.msdn.microsoft.com/microsoft_press/2015/04/27/from-the-mvps-introduction-to-the-internet-of-things-from-the-device-to-microsoft-azure-cloud/
  - Efficient IoT With Azure
    - https://blogs.msdn.microsoft.com/mvpawardprogram/2016/11/15/efficient-iot-with-azure/
  - Secure Provisioning of IoT device using Azure IoT Hub device SDK
    - https://blogs.msdn.microsoft.com/mvpawardprogram/2017/03/14/provisioning-of-iot-device/
- Thingiverse
  - http://www.thingiverse.com/thing:2253418
- Azure IoT
  - IoT SDKs - https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-devguide-sdks
  - GitHub - https://github.com/Azure/azure-iot-sdks
  - Azure IoT Suite - https://azure.microsoft.com/en-us/suites/iot-suite/
  - Azure IoT Hub - https://azure.microsoft.com/en-us/services/iot-hub/


Q
A

Thank you!