SMART PARKING SPOTS

Tom Tan@2019

CONTENTS

- Requirements
- Anatomy
- Live Application Demo
- Summary

TASK SUMMARY

Design a smart parking solution that helps motorists find parking in downtown areas.

	Requirements
1.	**Hardware/Sensor technology** to identify if a parking spot is occupied
2.	**Connectivity technology** to transmit current state of the parking spot
3.	A **scalable data processing architecture** in AWS to maintain parking data
4.	**Parking data is made available** to 3rd party app developers to build interesting parking applications
5.	Platform **security** and data **protection**

SENSORS - CONSTRAINTS

- installment environment
  - indoors, outdoors etc
  - all weather
  - crowded, accidental trespassing
- maintenance:
  - ideally no maintenance
  - low power
  - easy damage change

SENSORS - OPTIONS

- Passive Infrared Detection motion sensor
- Ultrasonic sensor
- Camera
- **Terrestrial magnetism sensors**
  - Non-obstrusive for all environments, esp. outdoors
  - low power consumption
  - high precision
  - embeddable

CONTROLLERS - CONSTRAINTS

- I2C port for sensor connection
- WIFI support for internet connection
- AWS Device SDK support for MQTT
- Embeddable, battery-powered

CONTROLLERS - OPTIONS

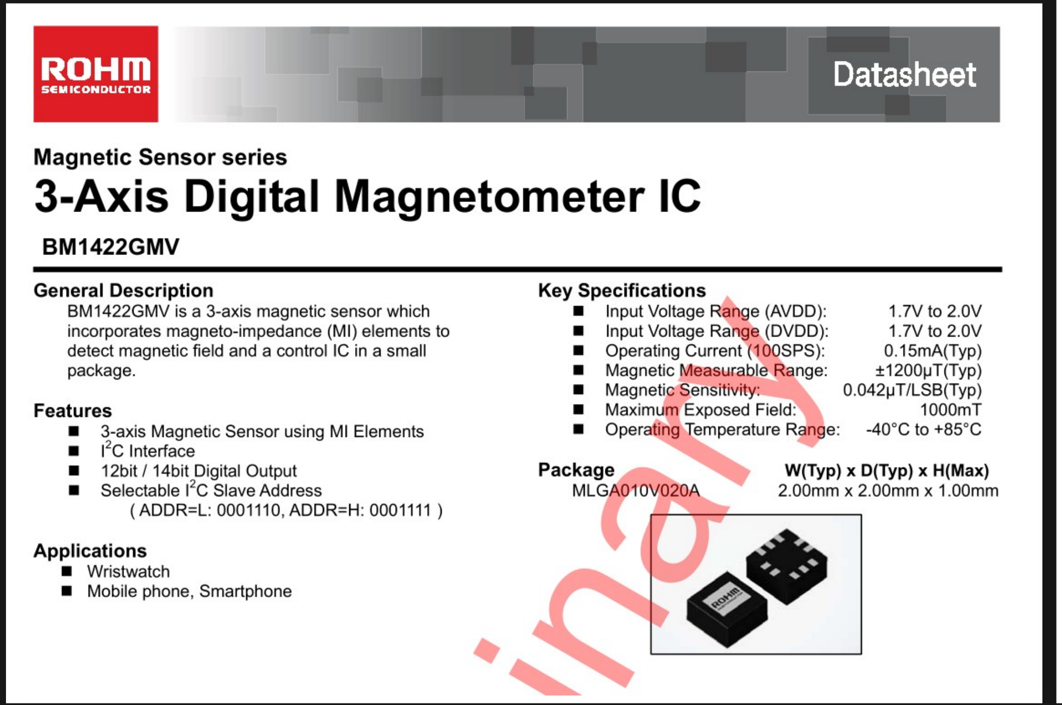
- ESP32
- Raspberry Pi 3
- BeagleBone
- Embeded PC/PLC
- Arduino
  - well supported by AWS SDK
  - economic in terms of cost and power consumption.

CONNECTIVITY CONSTRAINTS& OPTIONS

- Constraints
  - Connection to internet/Cloud service
  - No thing-thing talks
  - Wireless
- Technologies
  - BLE, Zigbee, RFID, 5G
  - NBIoT, Lora, **WIFI**

DEVICE SUMMARY

- Arduino & Sensor IC



ROHM SEMICONDUCTOR

Datasheet

Magnetic Sensor series

3-Axis Digital Magnetometer IC

BM1422GMV

General Description

BM1422GMV is a 3-axis magnetic sensor which incorporates magneto-impedance (MI) elements to detect magnetic field and a control IC in a small package.

Features

- 3-axis Magnetic Sensor using MI Elements
- I^2C Interface
- 12bit / 14bit Digital Output
- Selectable I^2C Slave Address
  (ADDR=L: 0001110, ADDR=H: 0001111)

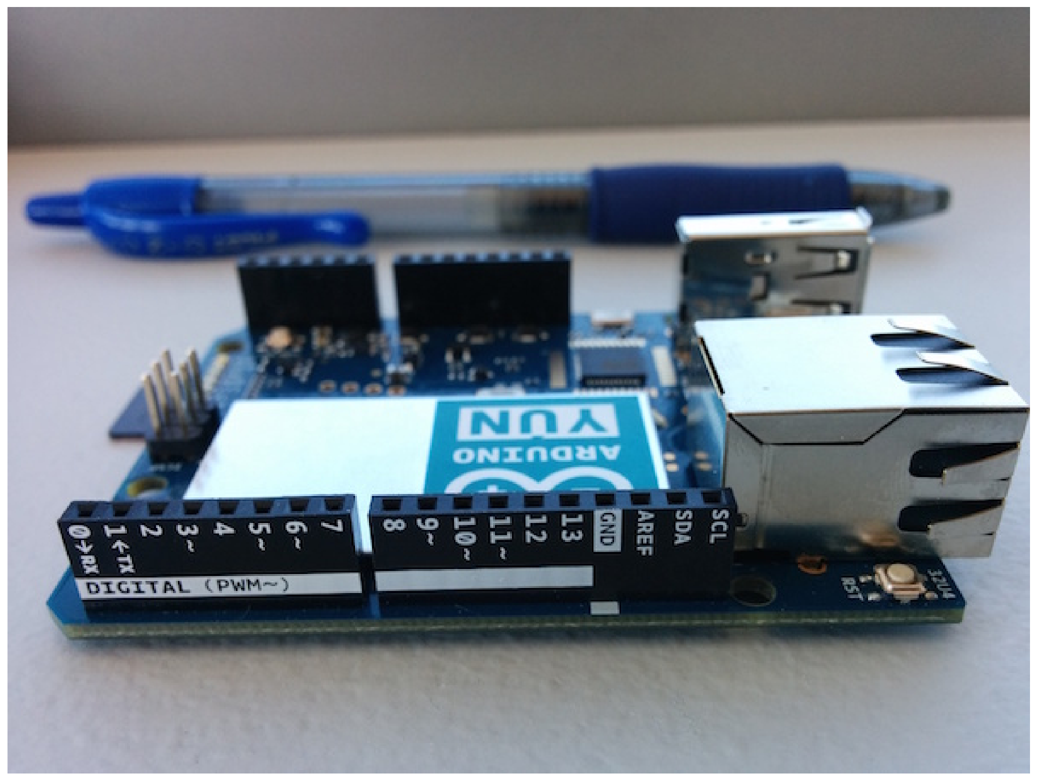
Applications

- Wristwatch
- Mobile phone, Smartphone

Key Specifications

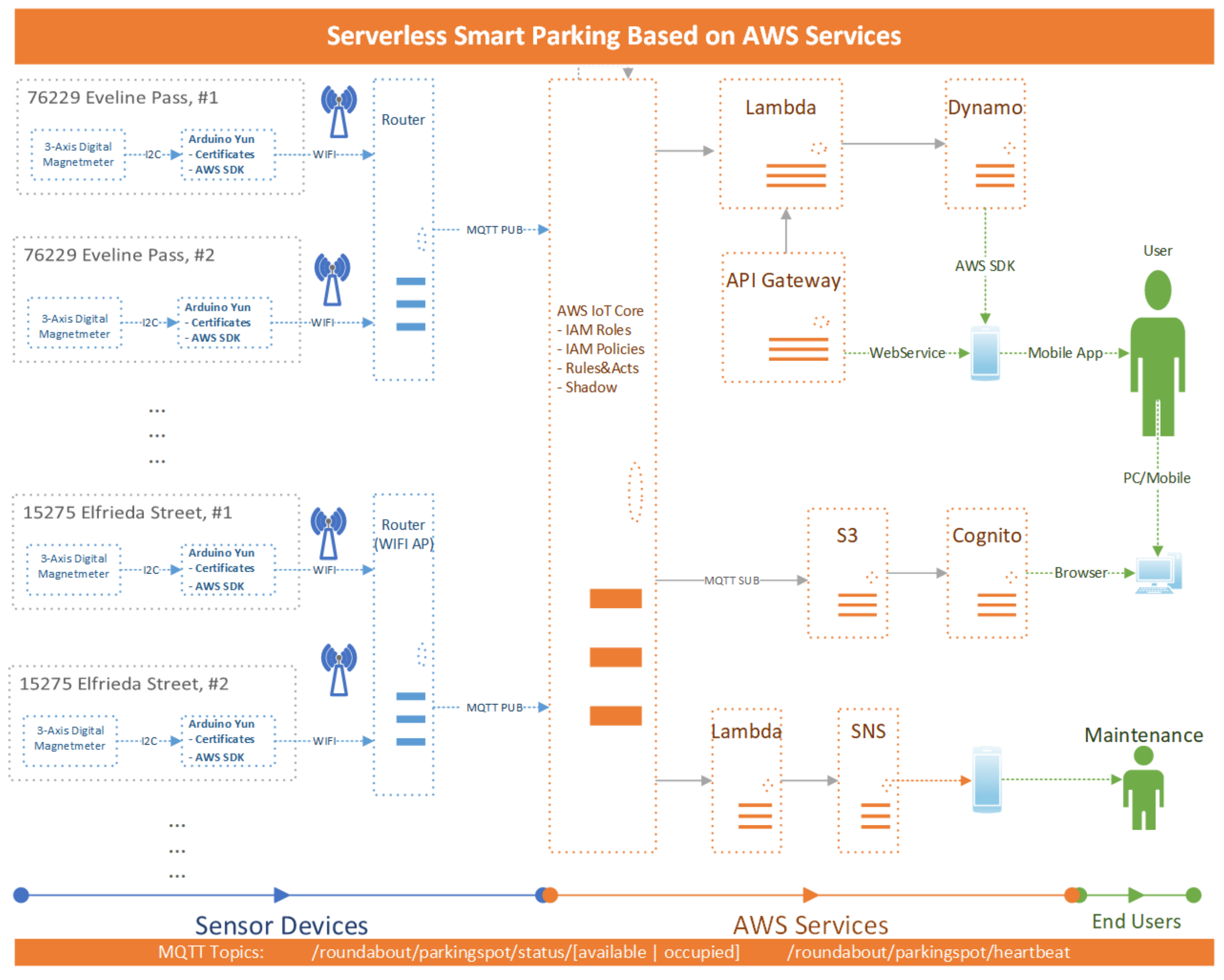
- Input Voltage Range (AVDD): 1.7V to 2.0V
- Input Voltage Range (DVDD): 1.7V to 2.0V
- Operating Current (100SPS): 0.15mA(Typ)
- Magnetic Measurable Range: ±1200μT(Typ)
- Magnetic Sensitivity: 0.042μT/LSB(Typ)
- Maximum Exposed Field: 1000mT
- Operating Temperature Range: -40°C to +85°C

Package	**W(Typ) x D(Typ) x H(Max)**
MLGA010V020A	2.00mm x 2.00mm x 1.00mm



THE REST IS LEFT TO AWS

ARCHITECTURE DIAGRAM



HOW DOES IT WORK?

A battery-powered sensor module, one for each parking spot, sends WIFI signals to a router nearby, which, in turn, relays it to the AWS IoT Core endpoint over internet.

The sensor module is an Arduino Yun embbeded controller, with AWS IoT device SDK.

It publishes to MQTT topic `/roundabout/parkingspot/status/available` when the spot is empty `/roundabout/parkingspot/status/occupied` when occupied.

It also publishes a heartbeats to `/roundabout/parkingspot/heartbeat` periodically, e.g. 20min.

Each sensor should have a corresponding `IoT shadow` in IoT Core service for device health management, e.g. firmware versions, but not implemented in the demo for simplicity.

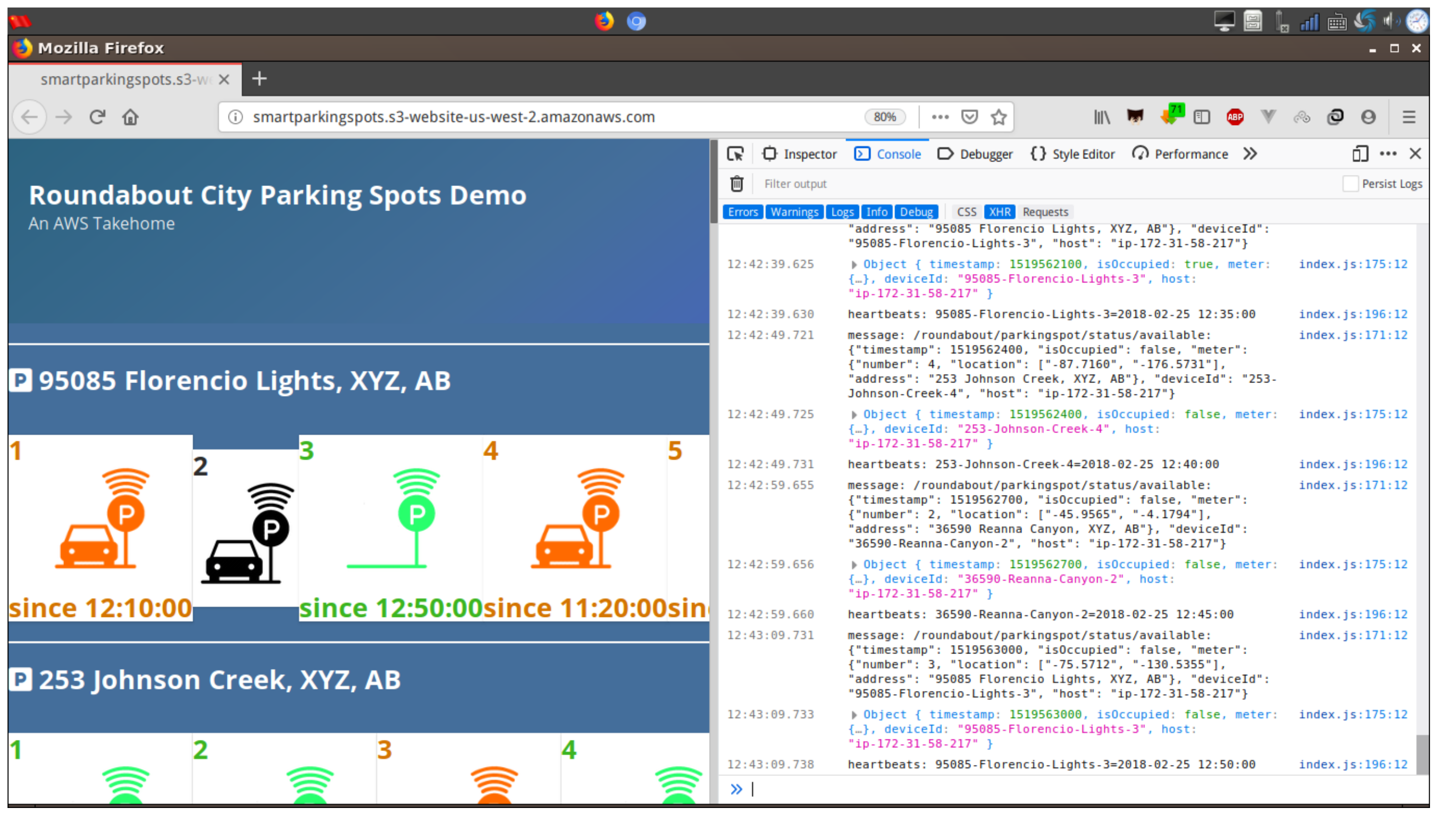
It's also possible to have a nearby `greengrass` device in production, but not implemented in the demo for simplicity.

AWS SERVICES

- Scalable data processing architecture · Parking data is made available · Security&protection

Greengrass	optionally for upgrade. Not in the demo due to lack of hardware.	scalable acquisition & dispatching
IoT Core	for device registration, management, communication, MQTT message relay/dispatch, device s/w update etc.	scalable acquisition & dispatching
DynamoDB	for historic lookup, analysis etc.	scalable persistence
API Gateway	REST interface of lookup the state of a parking meter at a given point in time	scalable provisioning
S3	static website hosting	application demo
EC2	simulating IoT sensor devices publishing messages	application demo
SNS	notification of device damages	scalable provisioning
Cognito	enabling anonymous pub/sub IoT messages from static website	security&protection
IAM	Roles, policy management	security&protection
Lambda	process iot messages(e.g. add uuid message id, filter&dispatch messages per subtopic) and persists to DynamoDB	glue

APPLICATION DEMO

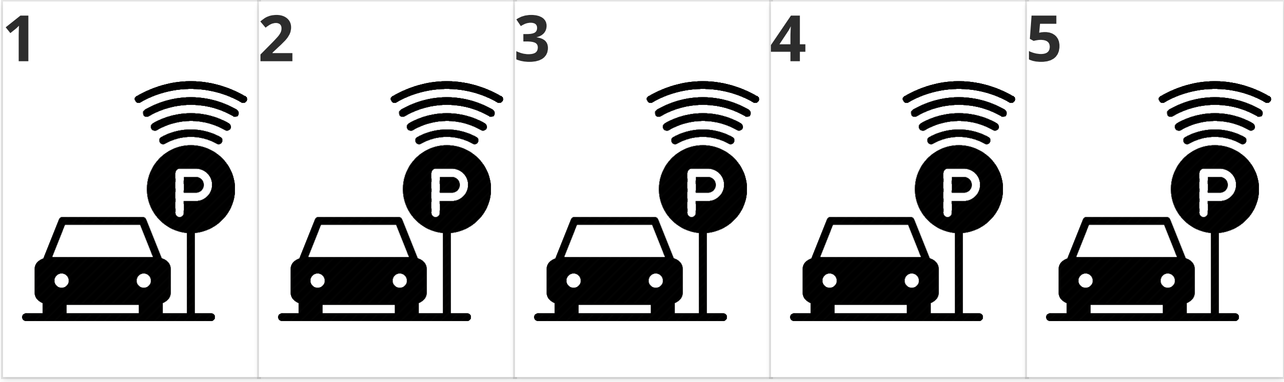


A static website hosted on S3, driven by data from EC2

Roundabout City Parking Spots Demo

An AWS Takehome

95085 Florencio Lights, XYZ, AB



253 Johnson Creek, XYZ, AB

SUMMARY

Design a smart parking solution that helps motorists find parking in downtown areas.

	Requirements	Solution
1.	Hardware/Sensor technology	Arduino Yun / simulated with EC2
2.	Connectivity technology	MQTT, websocket
3.	scalable data processing architecture	AWS IoT, Lambda, DynamoDB
4.	Parking data is made available	API Gateway, SNS, S3
5.	security and protection	IAM, Cognito, x.509 certificate authentication

COST ESTIMATION

Sensor	~65$/device
AWS IoT Core	$19 per device per year
S3 & DynamoDB	$6.8x12= 81.6 per device per year

Grand Total: $165 per device per year

Thank you

Bonus Slides

MANIFEST - MQTT TOPICS

- events:
  - `/roundabout/parkingspot/status/available`
  - `/roundabout/parkingspot/status/occupied`
- periodical(every x minutes, defaults 1m for demo purpose)
  - `/roundabout/parkingspot/heartbeat`

API

- Method 1, (only for realtime events)
  - subscribe to MQTT topics as in the demo
    - `/roundabout/parkingspot/status/available`
    - `/roundabout/parkingspot/status/occupied`
    - `/roundabout/parkingspot/heartbeat`
- Method 2: read DynamoDB directly through AWS API
- Method 3: API through AWS Gateway

```
http://<host>/prkingspot_lookup?deviceId=
<deviceId>&timestamp=<timestamp>
```