Pitt ECE x Microsoft

ECE 1894: Industry Project with the Azure Sphere Team

University of Pittsburgh

Introductions

Stephanie da Costa
Computer Engineering

Raheel Farouk
Computer Engineering

Noah Lichstein
Computer Engineering

University of Pittsburgh

ECE 1894: Course Structure

Azure Sphere Overview

The Seven Properties of Highly Secured Devices (2nd Edition)[1,2]

Galen Hunt, George Letey, and Edmund B. Nightingale
Microsoft Azure Sphere Team

ABSTRACT

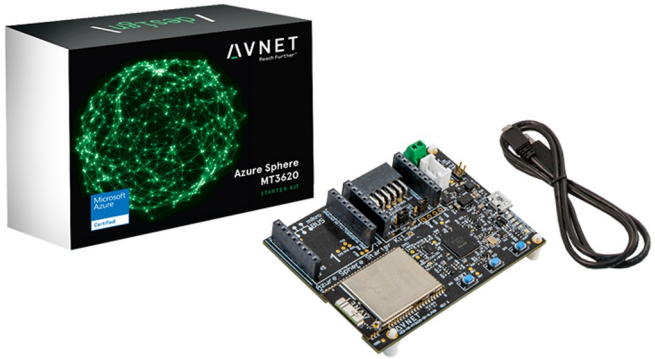
Many organizations building and deploying IoT devices largely underestimate the critical societal need to embody the highest levels of cybersecurity in every network-connected device. Every connected child's toy, every household's connected appliances, and every industry's connected equipment needs to be secured against network-based attacks. Until now, high development and maintenance costs have limited strong security to high-cost or high-margin devices.

Our mission is to bring high-integrity cybersecurity to every IoT device. We are especially concerned with the tens of billions of devices powered by microcontrollers. This class of devices is particularly ill-prepared for the security challenges of internet connectivity. Insufficient investments in the security needs of these and other price-sensitive devices have left consumers, enterprises, and society critically exposed to device security and privacy failures.

This paper makes two contributions to the field of device security. First, we identify seven properties we assert are required for all highly secured devices. Second, we describe our experiment working with a silicon partner to create a prototype highly secured microcontroller, codenamed Sopris. We evaluate Sopris against the seven properties framework and in a penetration test utilizing a red team of 150 top-tier hackers. Our experimental results suggest that even the most price-sensitive devices can and should be redesigned to achieve the high levels of device security critical to society's safety.

1. INTRODUCTION

The next decade promises the universal democratization of connectivity to every device. Significant drops in the cost of connectivity mean that every form of electrical device—every child's toy, every household's appliances, and every industry's equipment—will connect to the Internet. This Internet of Things (IoT) will drive huge economic efficiencies; it will enable countless innovations as digital transformation reaches across fields from childcare to eldercare, from hospitality to mining, from



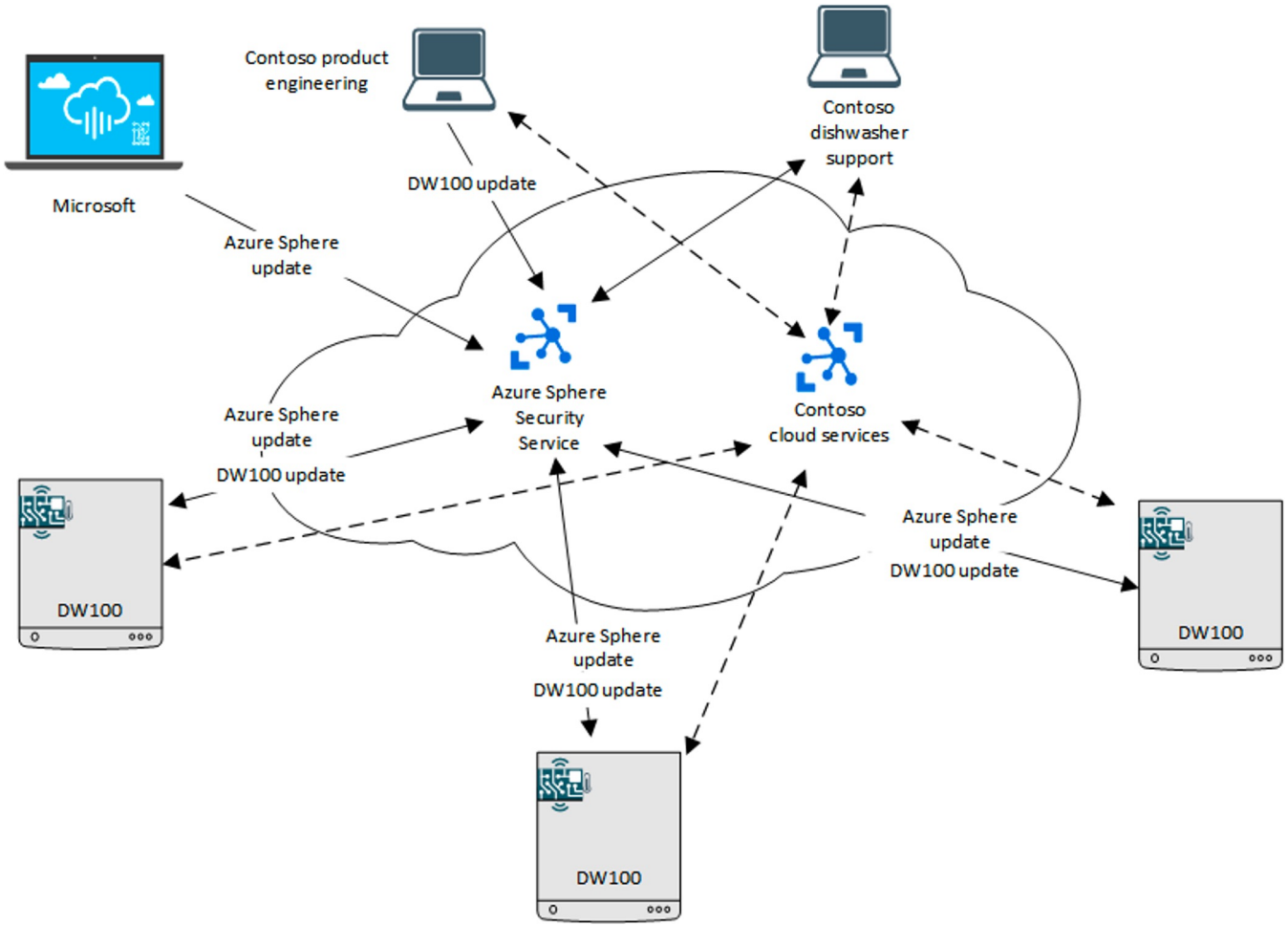
Property	Examples and *Questions to Prove the Property*
Hardware-based Root of Trust	Unforgeable cryptographic keys generated and protected by hardware. Physical countermeasures resist side-channel attacks.
	Does the device have a unique, unforgeable identity that is inseparable from the hardware?
Small Trusted Computing Base	Private keys stored in a hardware-protected vault, inaccessible to software. Division of software into self-protecting layers.
	Is most of the device's software outside the device's trusted computing base?
Defense in Depth	Multiple mitigations applied against each threat. Countermeasures mitigate the consequences of a successful attack on any one vector.
	Is the device still protected if the security of one layer of device software is breached?
Compartmentalization	Hardware-enforced barriers between software components prevent a breach in one from propagating to others.
	Does a failure in one component of the device require a reboot of the entire device to return to operation?
Certificate-based Authentication	Signed certificate, proven by unforgeable cryptographic key, proves the device identity and authenticity.
	Does the device use certificates instead of passwords for authentication?
Renewable Security	Renewal brings the device forward to a secure state and revokes compromised assets for known vulnerabilities or security breaches.
	Is the device's software updated automatically?
Failure Reporting	A software failure, such as a buffer overrun induced by an attacker probing security, is reported to cloud-based failure analysis system.
	Does the device report failures to its manufacturer?

Table 1. Required Properties of Highly Secure Devices with Examples.

image: https://www.avnet.com/wps/portal/us/products/new-product-introductions/npi/azure-sphere-mt3620-starter-kit-2-0/
paper: https://www.microsoft.com/en-us/research/wp-content/uploads/2017/03/SevenPropertiesofHighlySecureDevices.pdf

University of Pittsburgh

Azure Sphere Overview

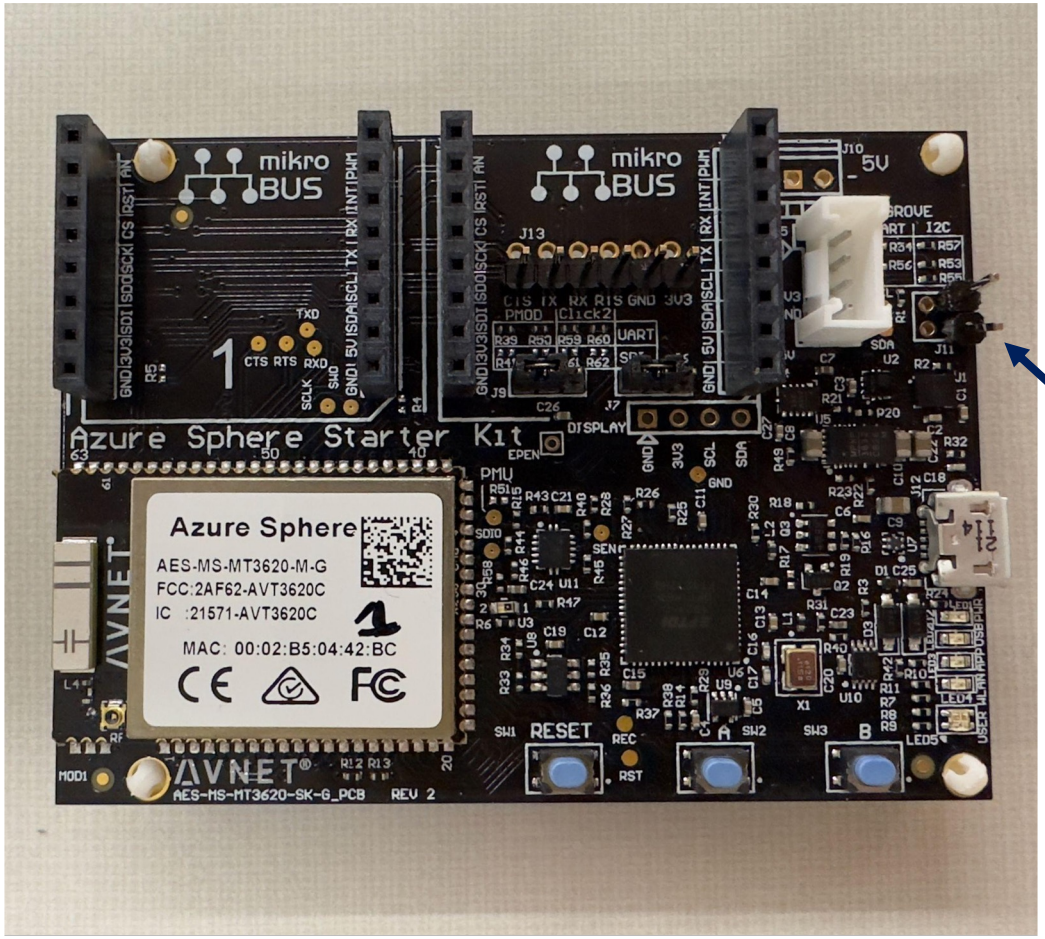


https://learn.microsoft.com/en-us/azure-sphere/product-overview/what-is-azure-sphere

University of Pittsburgh

Azure Sphere Overview

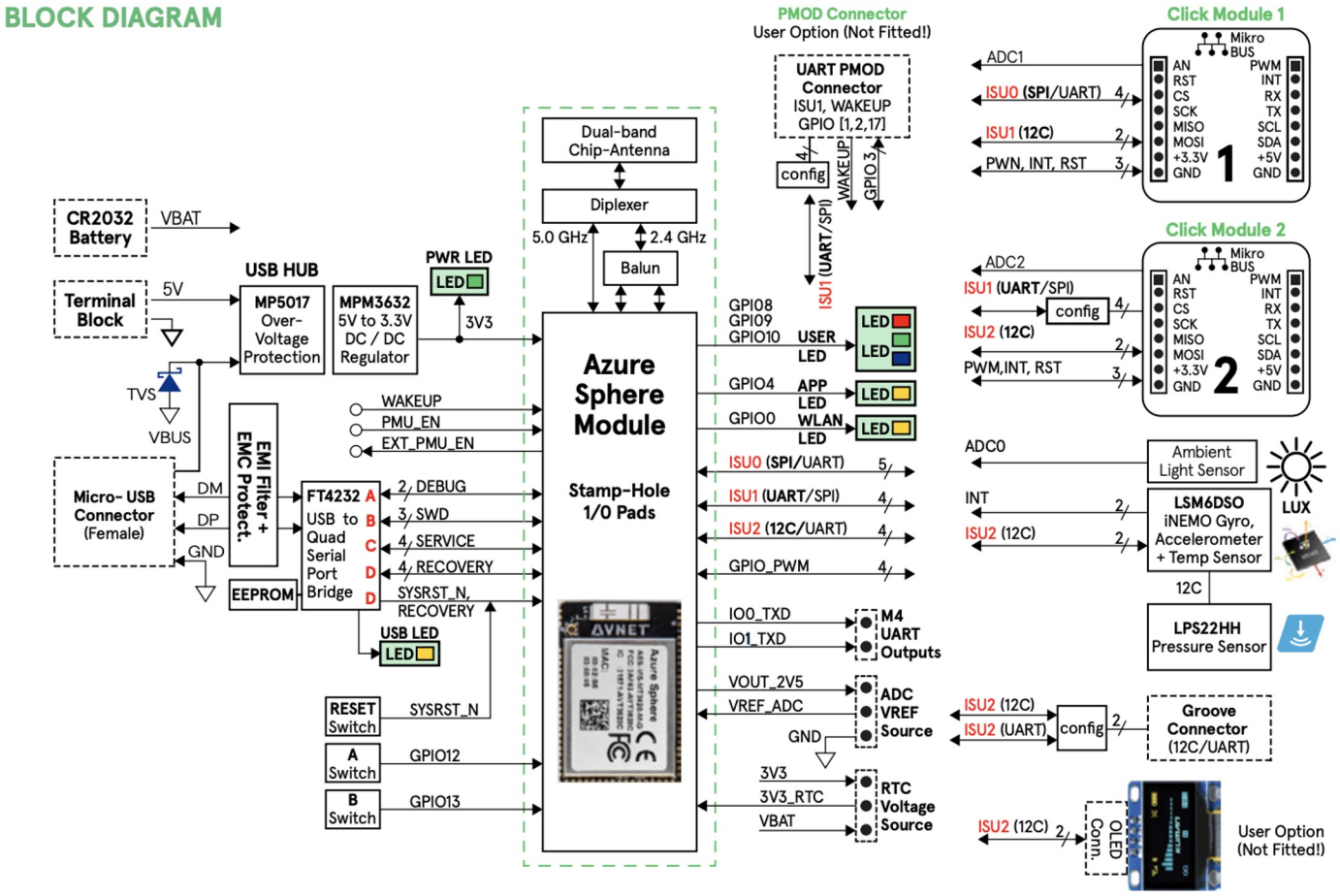
Azure Sphere MT3620 Starter Kit 2.0 from Avnet



Added headers for UART

University of Pittsburgh

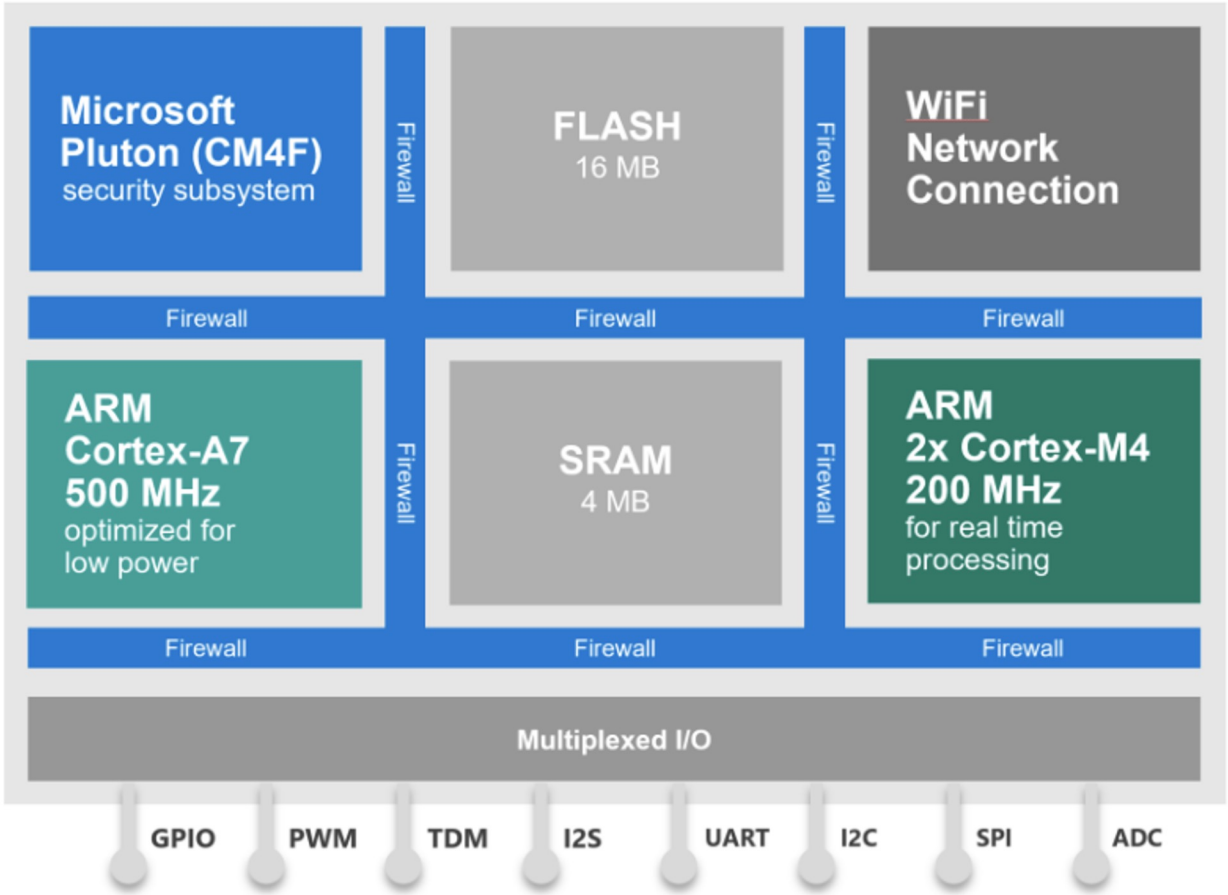
Avnet Board Block Diagram



https://www.avnet.com/wps/portal/us/products/new-product-introductions/npi/azure-sphere-mt3620-starter-kit-2-0/

University of Pittsburgh

Avnet Starter Kit Chipset: Azure Sphere MT3620 Module 2.0



https://learn.microsoft.com/en-us/azure-sphere/product-overview/what-is-azure-sphere

University of Pittsburgh

Azure Sphere Development Environment

Windows 11

Ubuntu

University of Pittsburgh

Building Azure Sphere Apps

app_manifest.json: the CMakeLists.txt of Azure Sphere Apps…

Building Azure Sphere Apps

…there's also CMakeLists.txt too

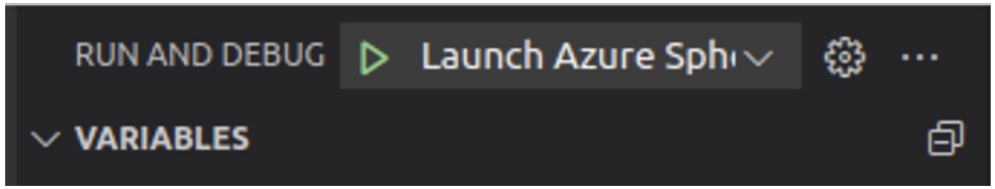
CMakeLists.txt

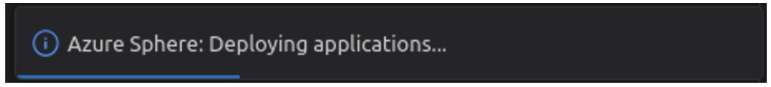
```
1   #  Copyright (c) Microsoft Corporation. All rights reserved.
2   #  Licensed under the MIT License.
3
4   cmake_minimum_required (VERSION 3.20)
5
6   project (Blink-test C)
7
8   azsphere_configure_tools(TOOLS_REVISION "22.11")
9   azsphere_configure_api(TARGET_API_SET "15")
10
11  # Create executable
12  add_executable (${PROJECT_NAME} main.c)
13  target_link_libraries (${PROJECT_NAME} applibs pthread gcc_s c)
14  azsphere_target_hardware_definition(${PROJECT_NAME} TARGET_DIRECTORY "HardwareDefinitions/mt3620_
15
16  azsphere_target_add_image_package(${PROJECT_NAME})
17
```

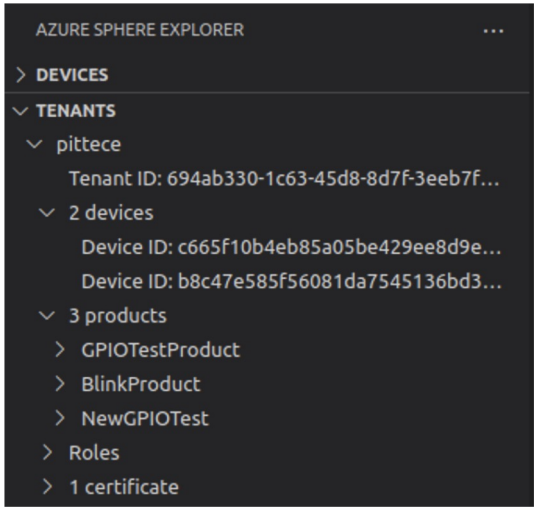
University of Pittsburgh

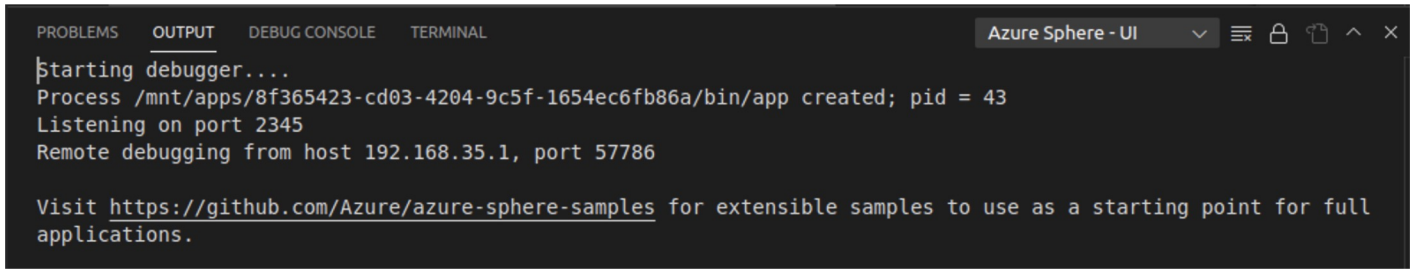
Building Azure Sphere Apps

VS Code Azure Sphere Extension





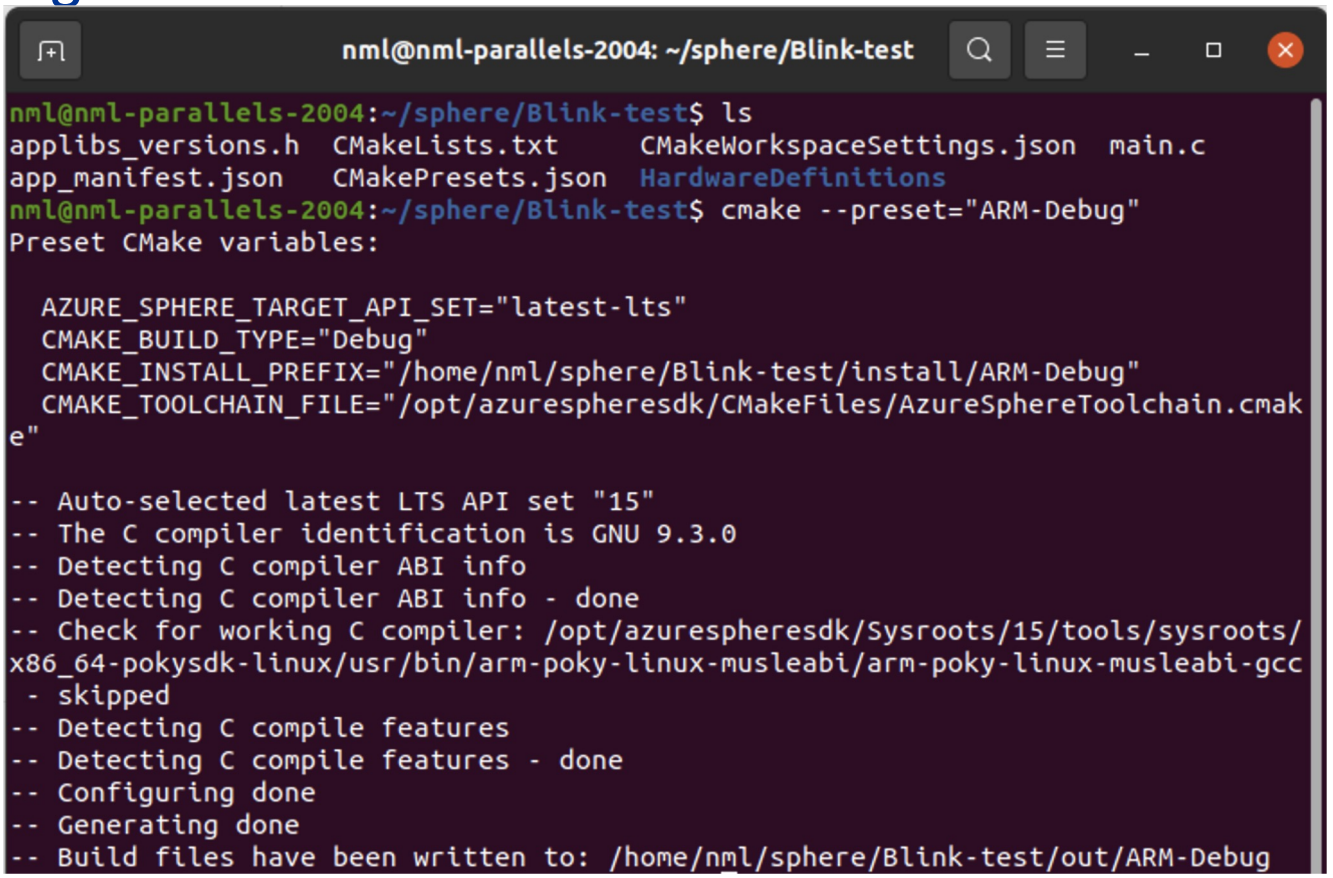




University of Pittsburgh

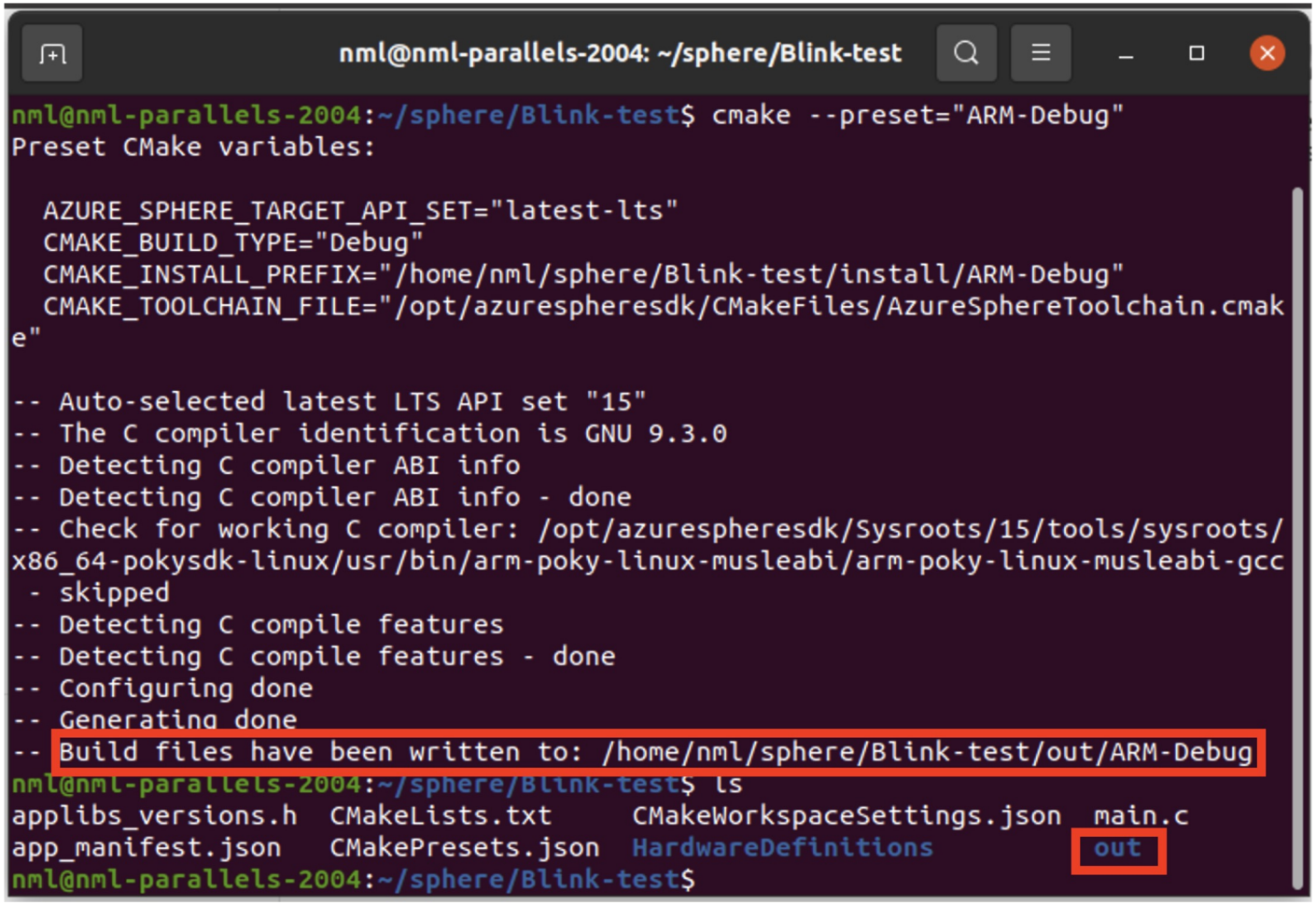
Building Azure Sphere Apps

Building with CLI



University of Pittsburgh

Building Azure Sphere Apps



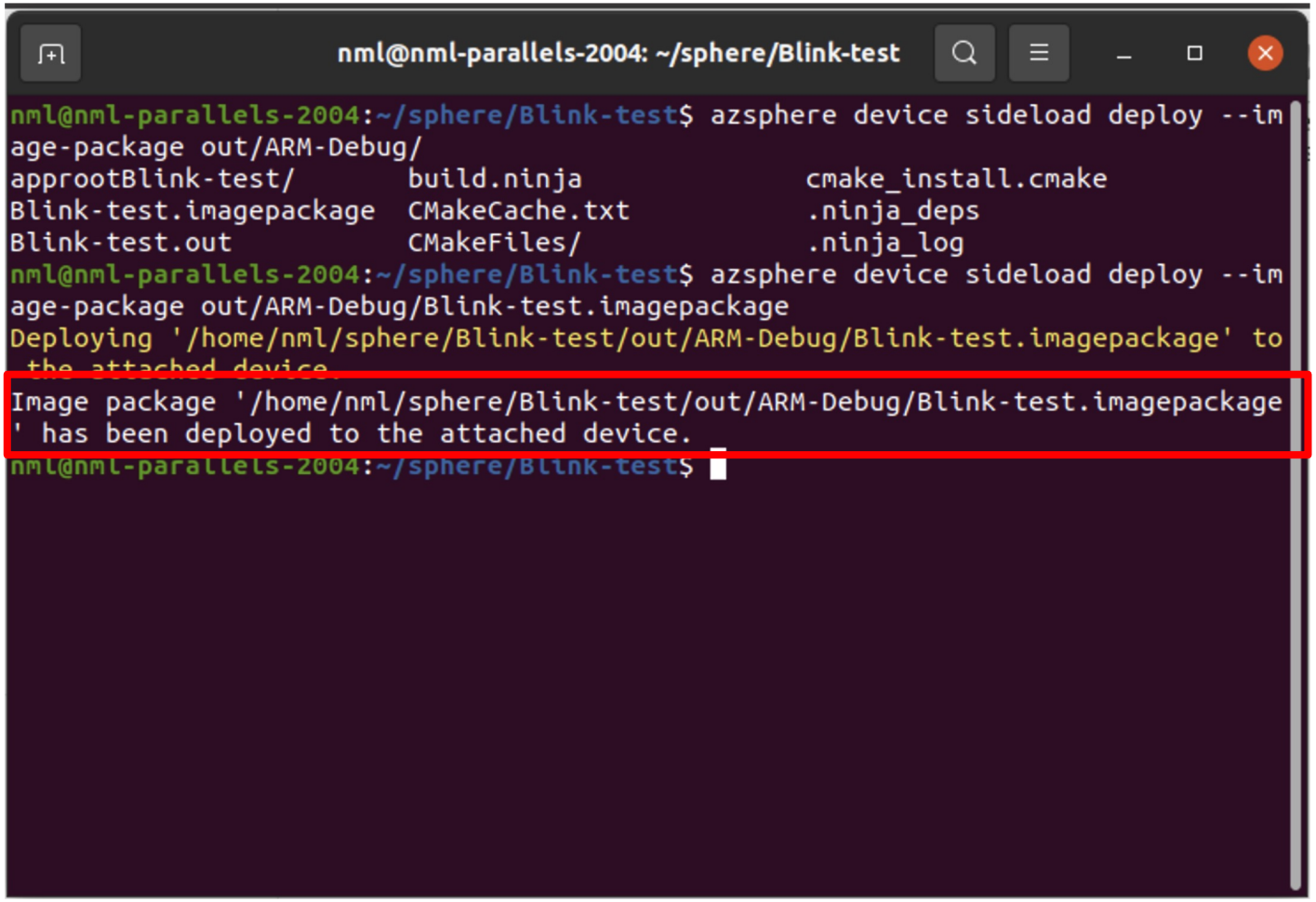
University of Pittsburgh

Building Azure Sphere Apps

```
nml@nml-parallels-2004:~/sphere/Blink-test$ ninja -C out/ARM-Debug
ninja: Entering directory `out/ARM-Debug'
[4/4] Generating Blink-test.imagepackage
INFO: Azure Sphere Utility version 22.11.5.13799
INFO: Copyright (C) Microsoft Corporation. All rights reserved.
INFO:
INFO: Start time (UTC): Wednesday, 26 April 2023 01:32:53
INFO: Creating image package.
INFO: Azure Sphere application image package written.
INFO: Appending metadata.
INFO: Wrote metadata:
INFO:   Section: Identity
INFO:     Image Type:           Application
INFO:     Component ID:         8f365423-cd03-4204-9c5f-1654ec6fb86a
INFO:     Image ID:             0197b107-62de-424b-8164-0aa59850272a
INFO:   Section: Signature
INFO:     Signing Type:         ECDsa256
INFO:     Cert:                 a8d5cc6958f48710140d7a26160fc1cfc31f5df0
INFO:   Section: Debug
INFO:     Image Name:           Blink-test
INFO:     Built On (UTC):       4/26/2023 1:32:54 AM
INFO:     Built On (Local):     4/25/2023 9:32:54 PM
INFO:   Section: Temporary Image
INFO:     Remove image at boot: False
INFO:     Under development:    True
INFO:     Image Name:           Blink-test
INFO:     Built On (UTC):       4/26/2023 1:32:54 AM
INFO:     Built On (Local):     4/25/2023 9:32:54 PM
INFO:   Section: Temporary Image
INFO:     Remove image at boot: False
INFO:     Under development:    True
INFO:   Section: ABI Depends
INFO:     Depends on:           ApplicationRuntime, version 15
INFO:
INFO: Packaging completed successfully.
INFO: Output file is at: /home/nml/sphere/Blink-test/out/ARM-Debug/Blink-test.im
agepackage
INFO: Command completed in 00:00:01.1971123.
INFO: Command ran in 0.915 seconds (init: 0.004, invoke: 0.911)
nml@nml-parallels-2004:~/sphere/Blink-test$
```

University of Pittsburgh

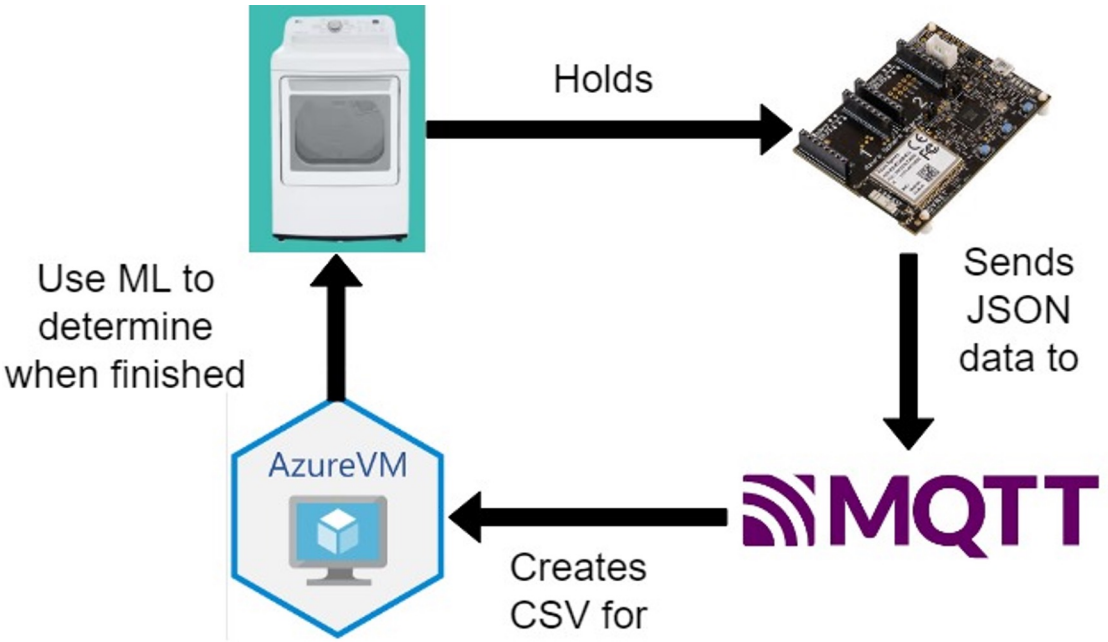
Building Azure Sphere Apps



University of Pittsburgh

Our Project

- Device to determine when dryer is finished
- Retrofit onto old machines
- Usable on top and side-loading dryers

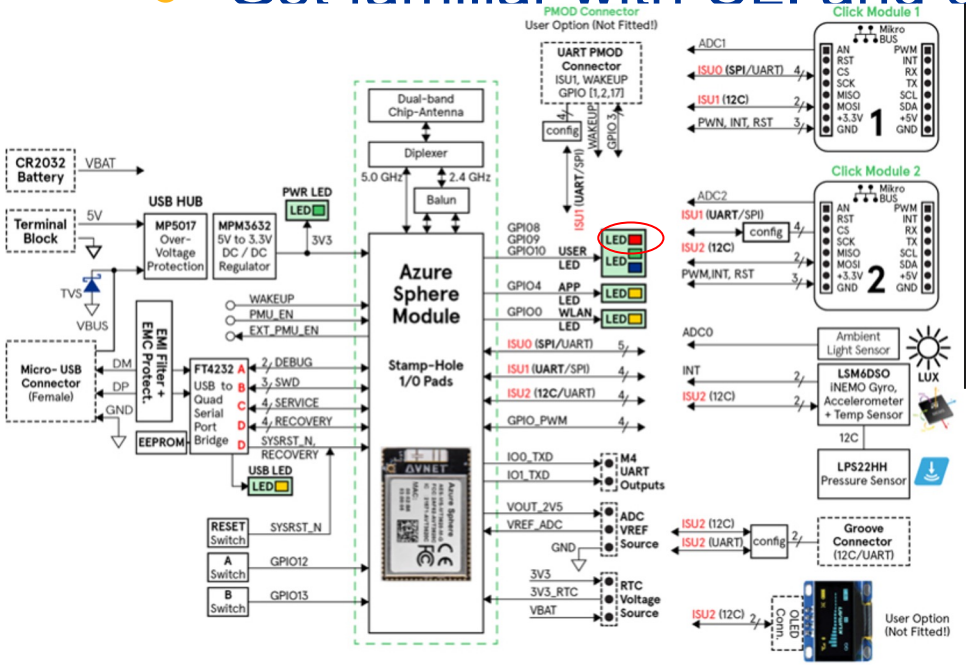


University of Pittsburgh

Timeline

Timeline: Hello World - Blink App

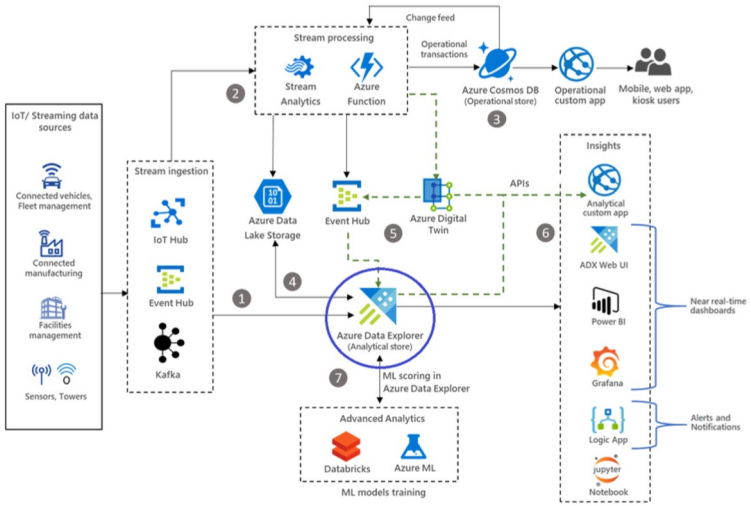
- Set up development environment
- Got familiar with CLI and device groups




```c
int main(void)
{
    int fd = GPIO_OpenAsOutput(TEMPLATE_LED, GPIO_OutputMode_PushPull, GPIO_Value_High);
    if (fd < 0) {
        Log_Debug(
            "Error opening GPIO: %s (%d). Check that app_manifest.json includes the GPIO used.\n",
            strerror(errno), errno);
        return ExitCode_Main_Led;
    }
    const struct timespec sleepTime = {.tv_sec = 1, .tv_nsec = 0};
    while (true) {
        GPIO_SetValue(fd, GPIO_Value_Low);
        nanosleep(&sleepTime, NULL);
        GPIO_SetValue(fd, GPIO_Value_High);
        nanosleep(&sleepTime, NULL);
    }
}
```

University of Pittsburgh

Timeline: IoT Hub and Digital Twin

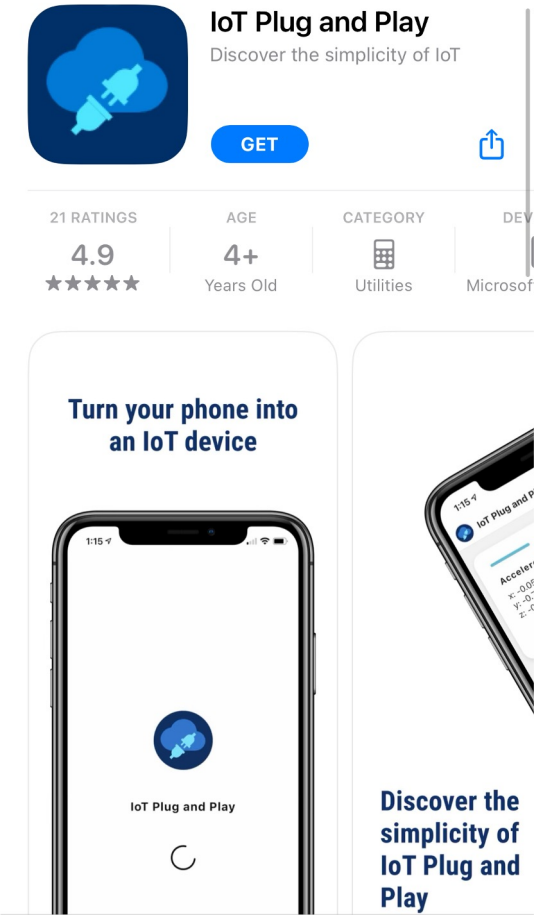



```
static void DeviceMoved(void)
{
    Log_Debug("INFO: Device moved.\n");

    time_t now;
    time(&now);

    Cloud_Result result = Cloud_SendThermometerMovedEvent(now);
    if (result != Cloud_Result_OK) {
        Log_Debug("WARNING: Could not send thermometer moved event to cloud: %s\n",
                  CloudResultToString(result));
    }
}
```

code snippet: https://github.com/Azure/azure-sphere-samples/tree/main/Samples/AzureIoT



University of Pittsburgh

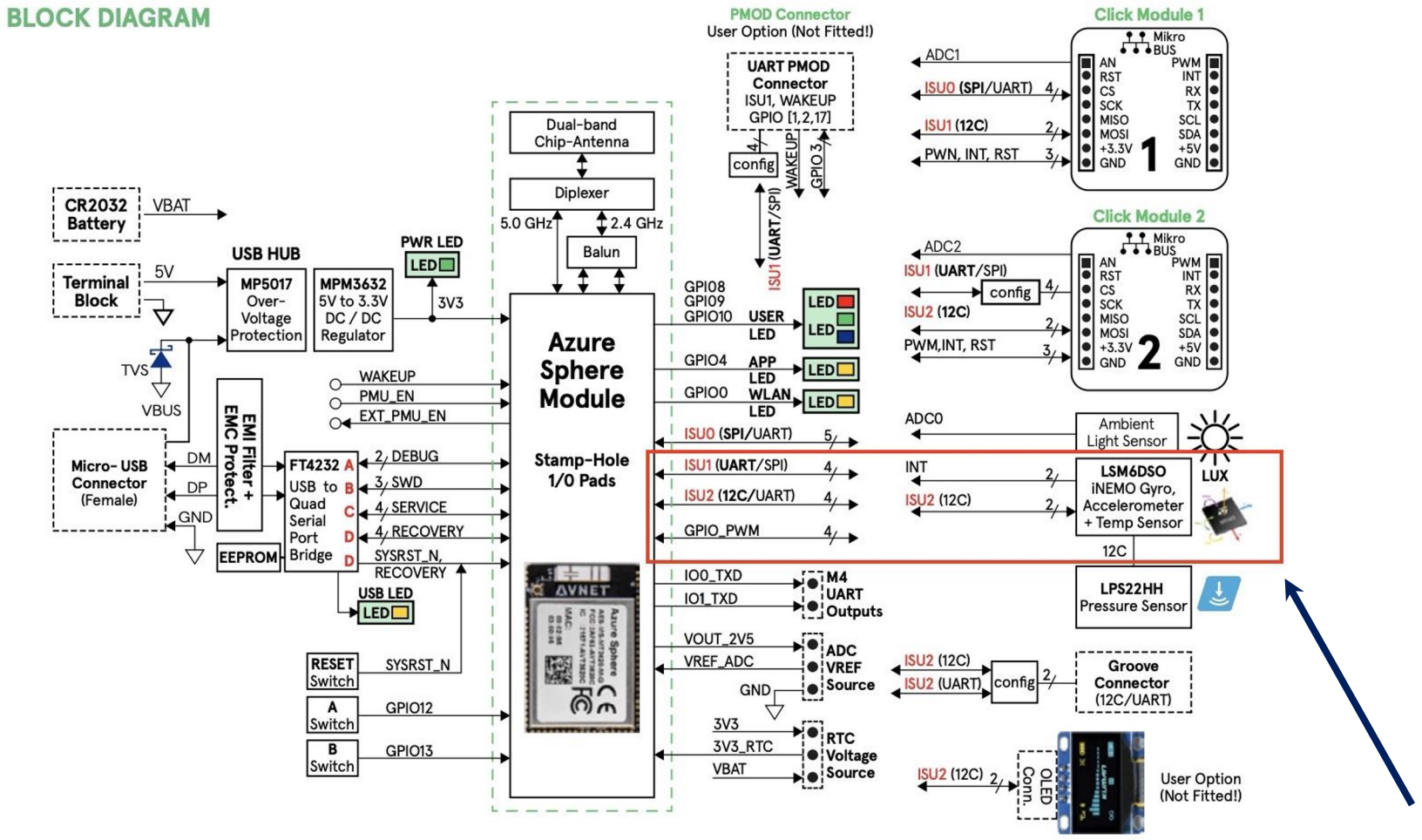
Timeline: Initial Dryer App Ideas After First Exploration with Sphere

Dryer ball implementation

- Put in with clothes - measure acceleration of device
- Issues with durability and signal connections
- Pivot to device on the outside of dryer

University of Pittsburgh

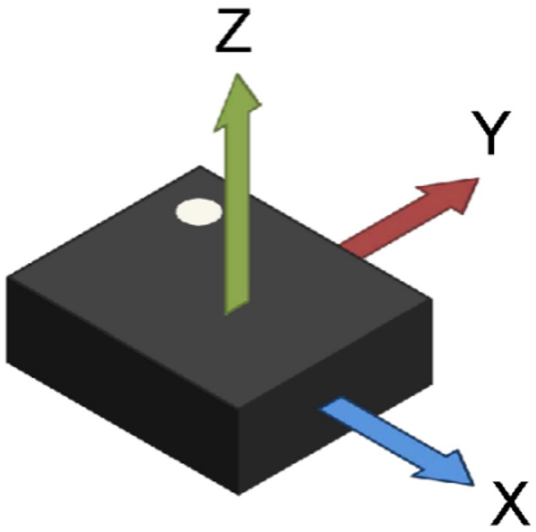
Timeline: Test Gyroscope

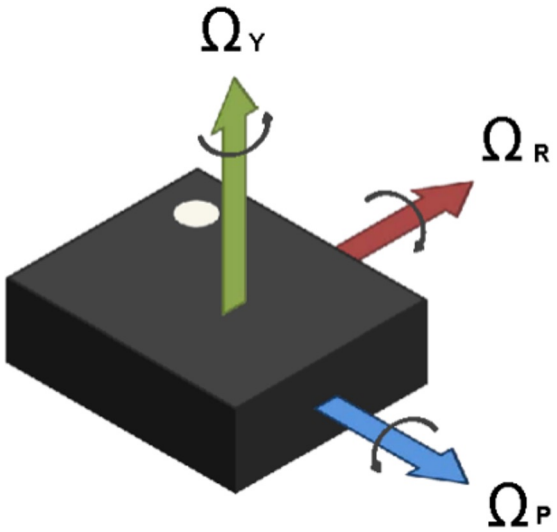


University of Pittsburgh

Timeline: Test Gyroscope

- Read from gyroscope and transmit the data in a JSON string - show output
- Tested sensitivity of on-board accelerometer and gyroscope
- Built our main application off of this code

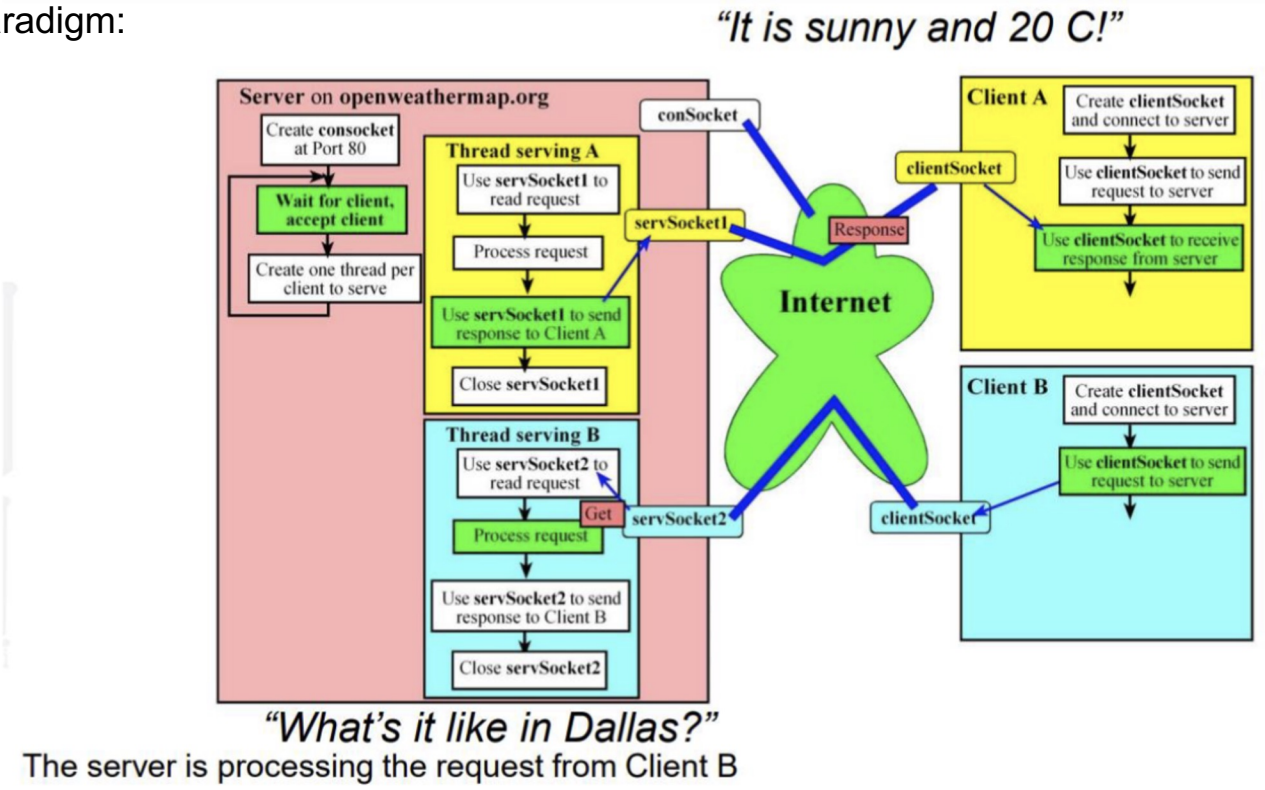




University of Pittsburgh

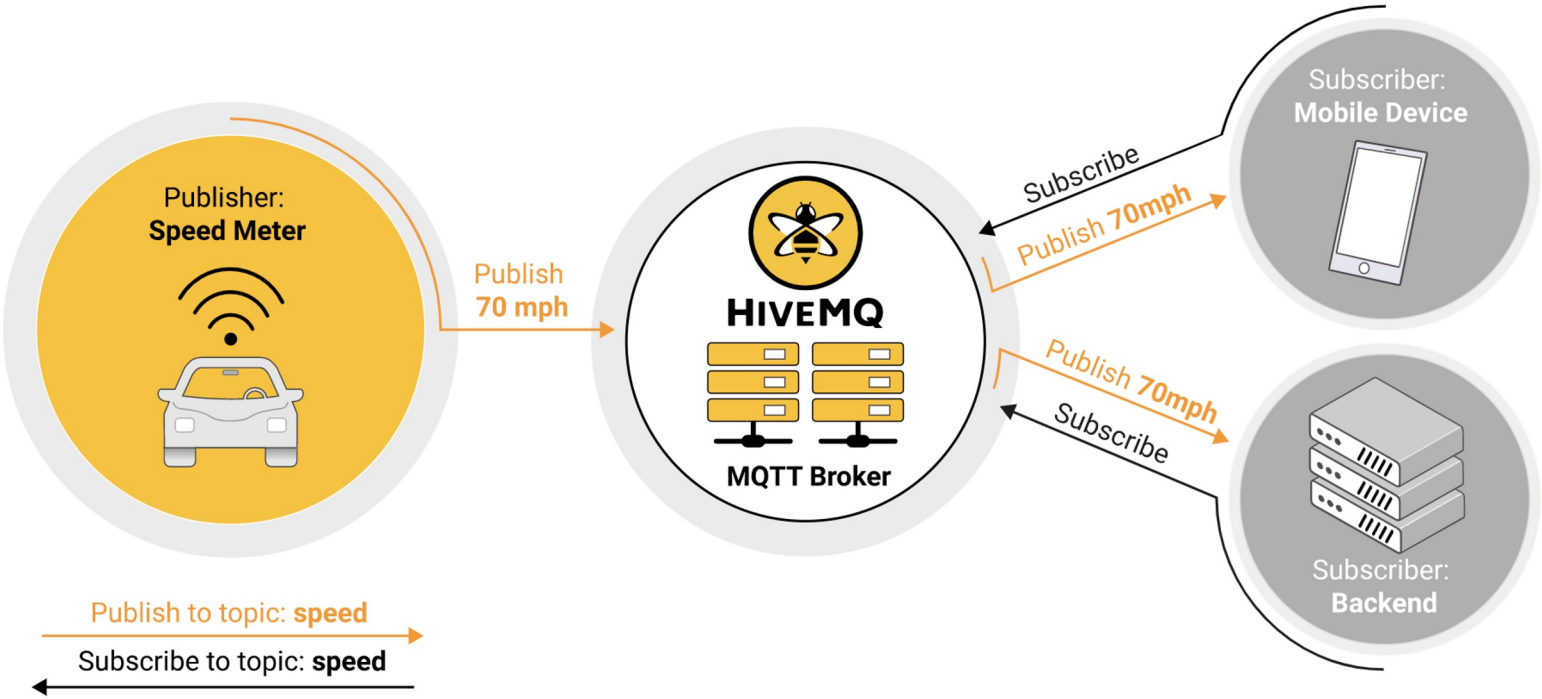
Timeline: Getting Data off Board

Typical HTTP client-server paradigm:



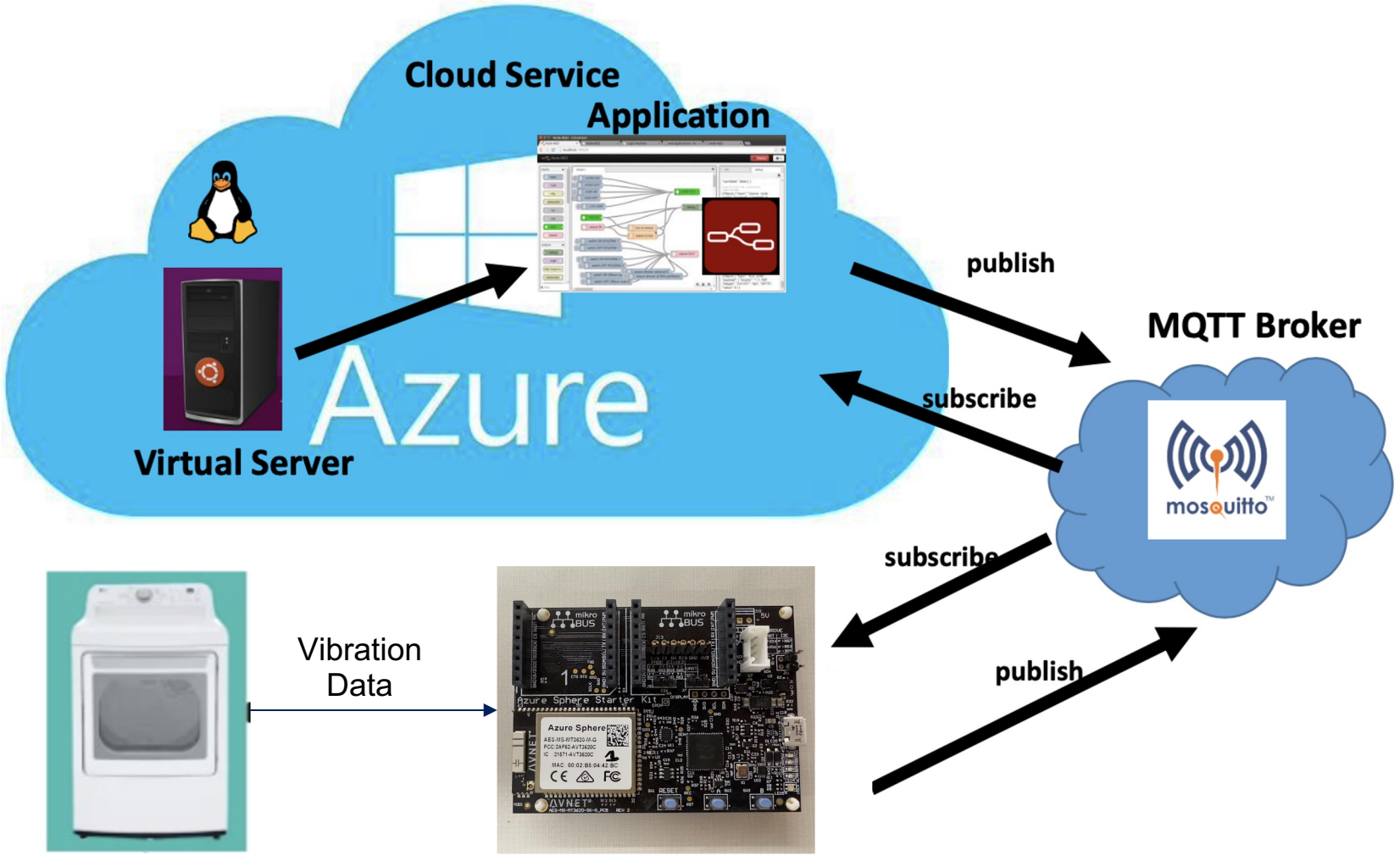
From Dr. Samuel Dickerson's ECE 1188 Lecture 23

Timeline: MQTT



https://www.hivemq.com/blog/mqtt-essentials-part2-publish-subscribe/

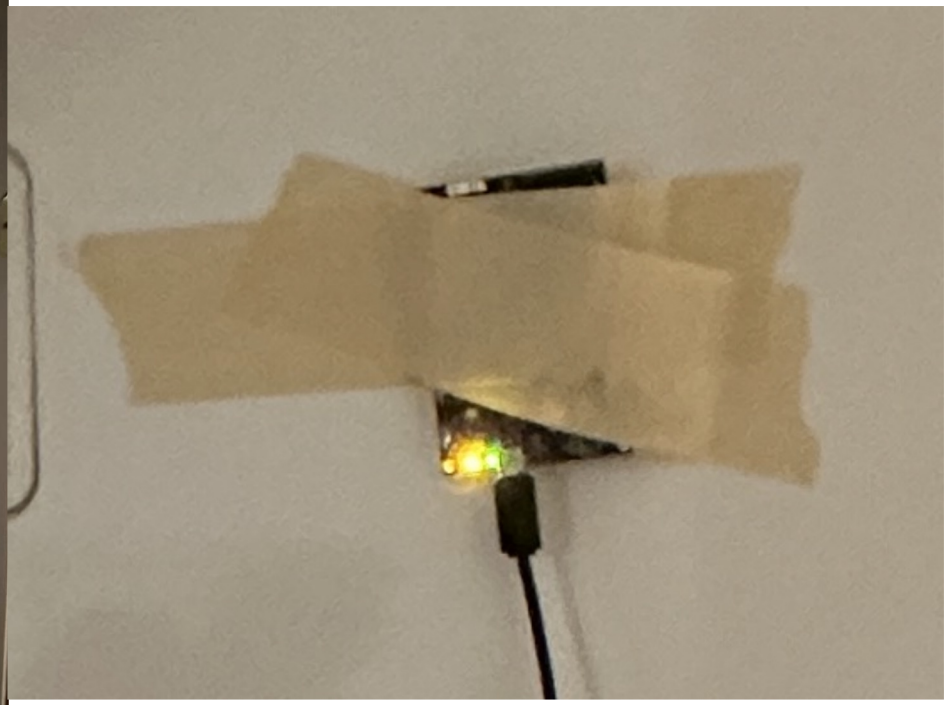
Timeline: Implementation Azure Server



University of Pittsburgh

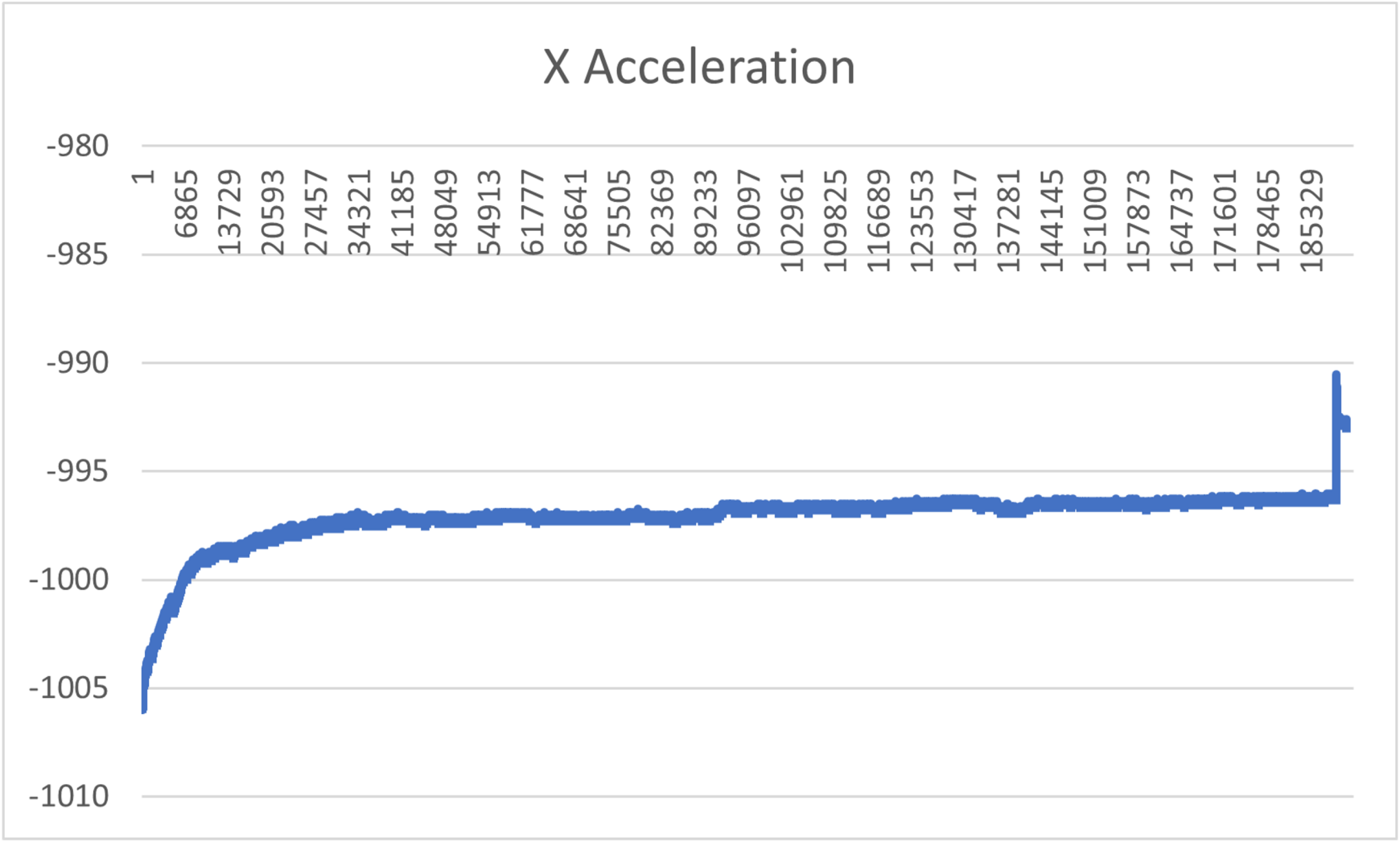
Adapted From Dr. Samuel Dickerson's ECE 1188 Cloud Computing Lecture

Timeline: Data Collection



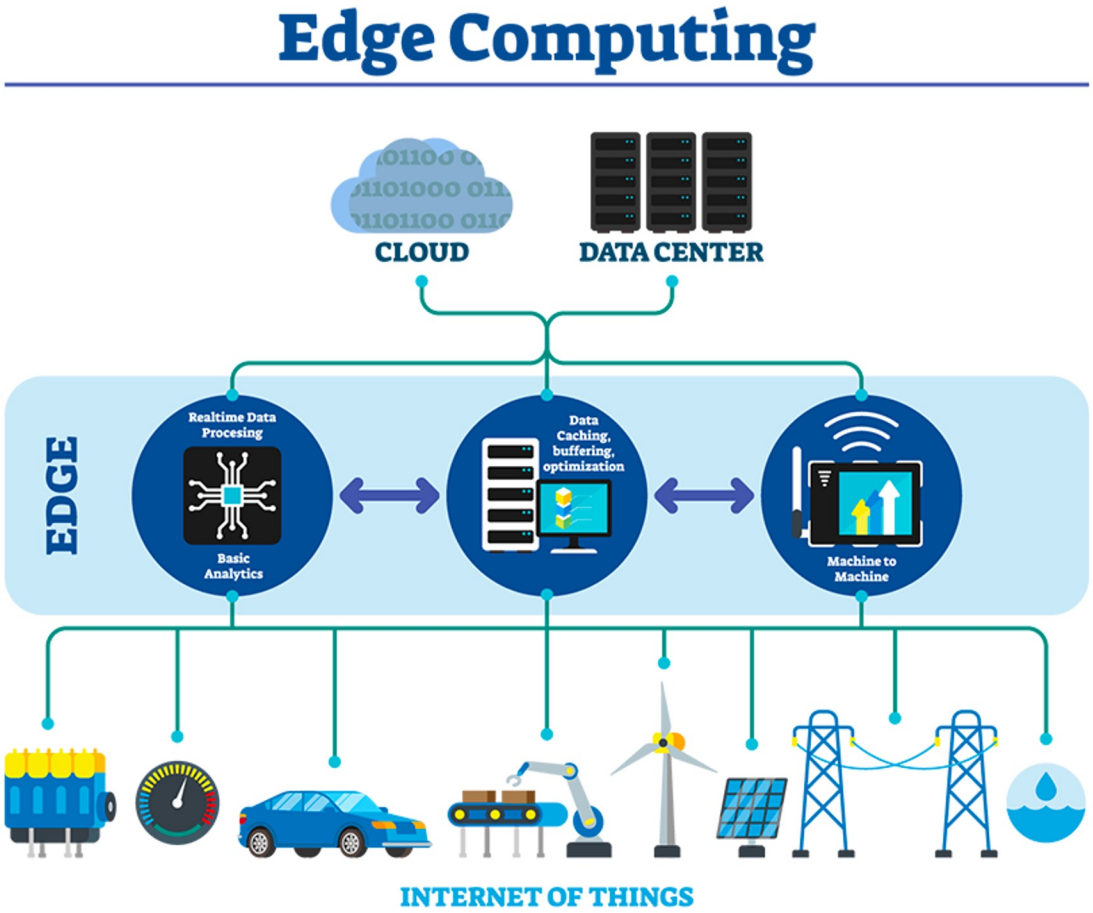
University of Pittsburgh

Timeline: Data Collection



University of Pittsburgh

Timeline: Edge Computing

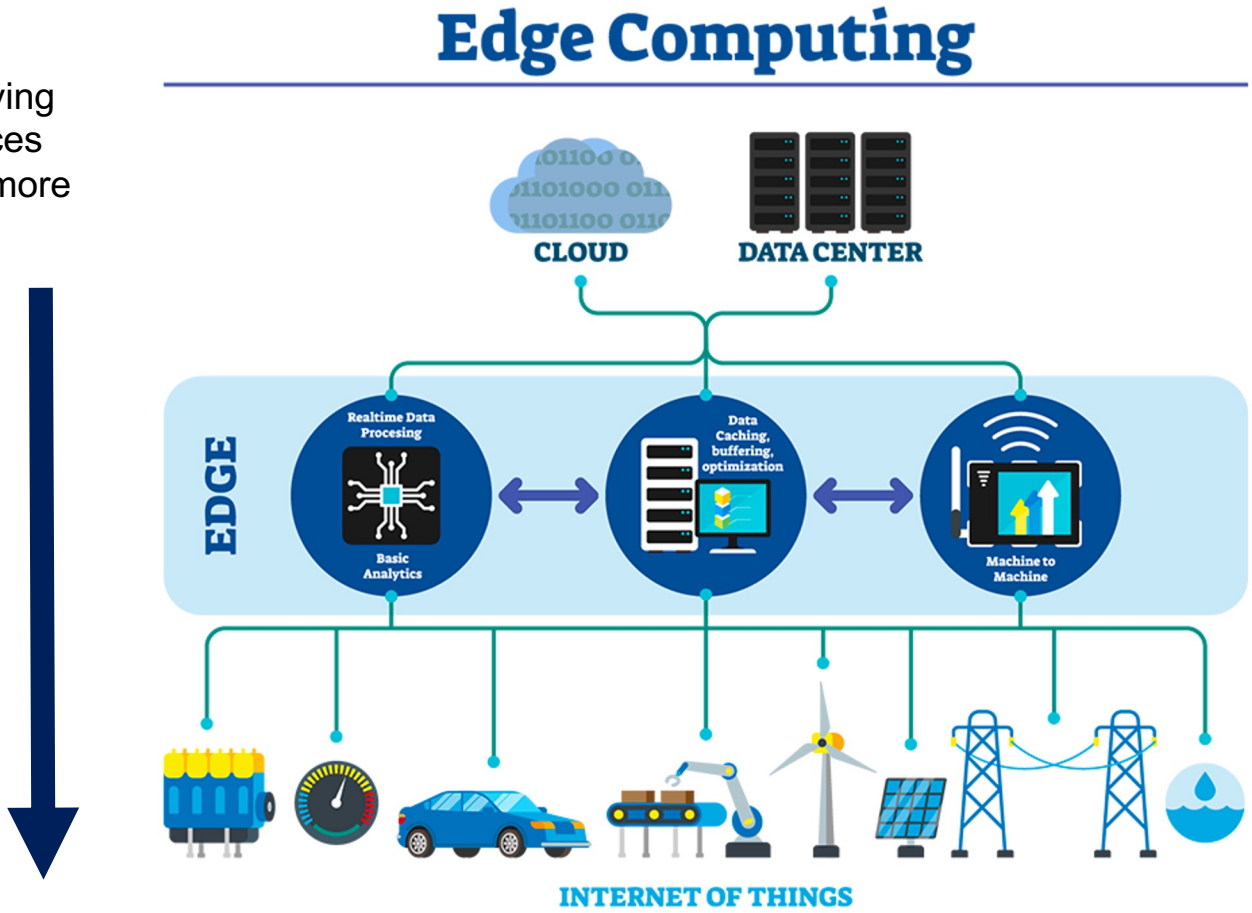


https://innovationatwork.ieee.org/real-life-edge-computing-use-cases/

University of Pittsburgh

Timeline: Edge Computing

Edge is moving down! Devices are getting more intelligent

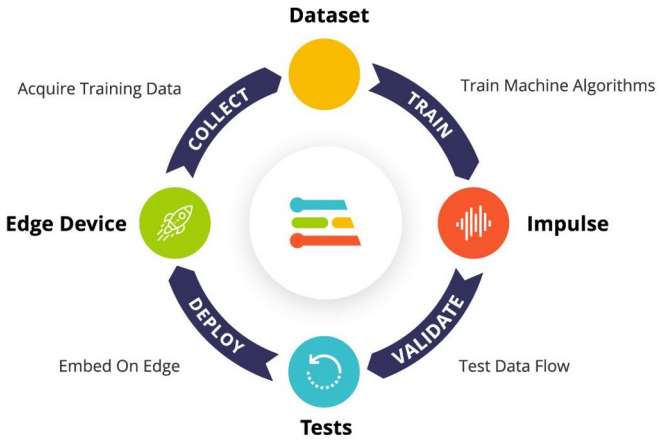


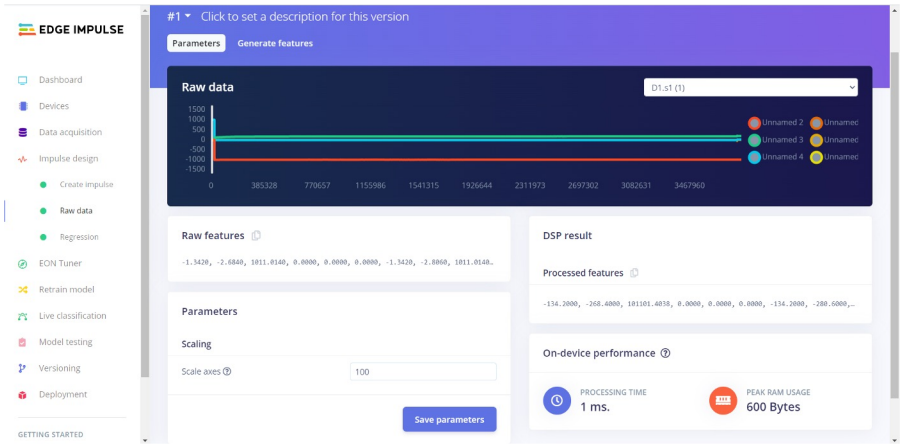
https://innovationatwork.ieee.org/real-life-edge-computing-use-cases/

University of Pittsburgh

Timeline: Edge Impulse

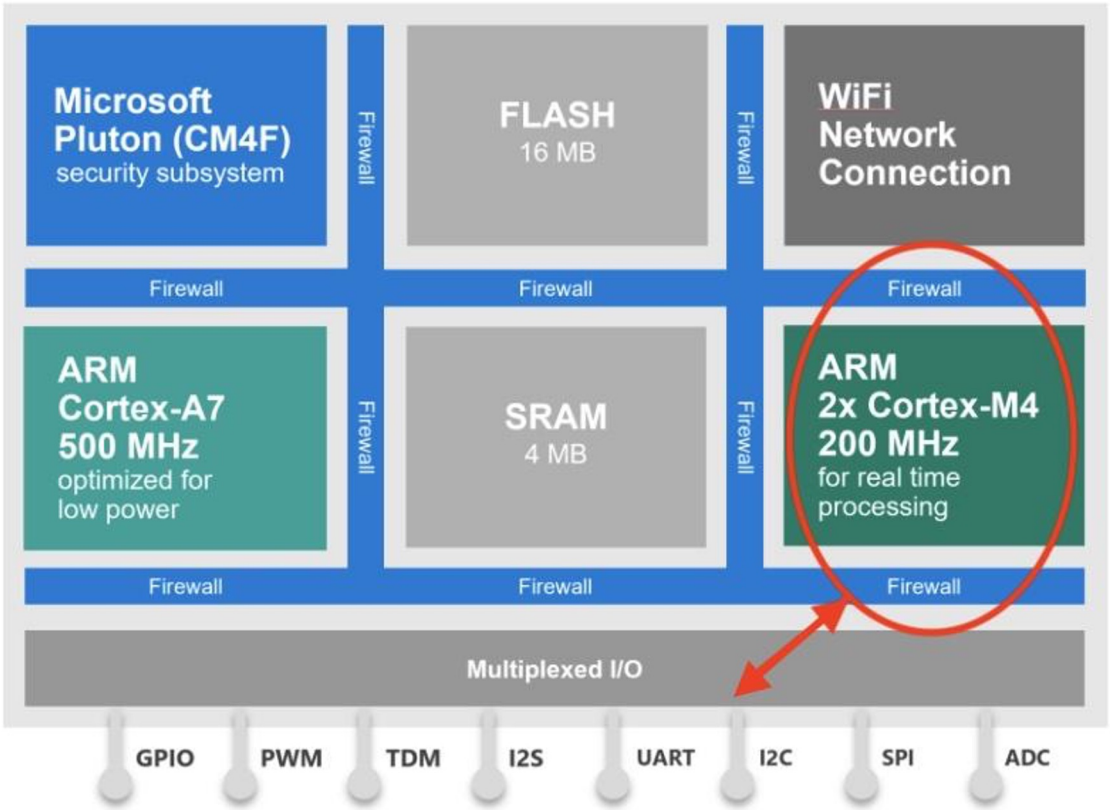
- Tried to use Edge Impulse to perform ML tasks on the Edge
- Generated a C++ Library to be used with the Azure Sphere Board





University of Pittsburgh

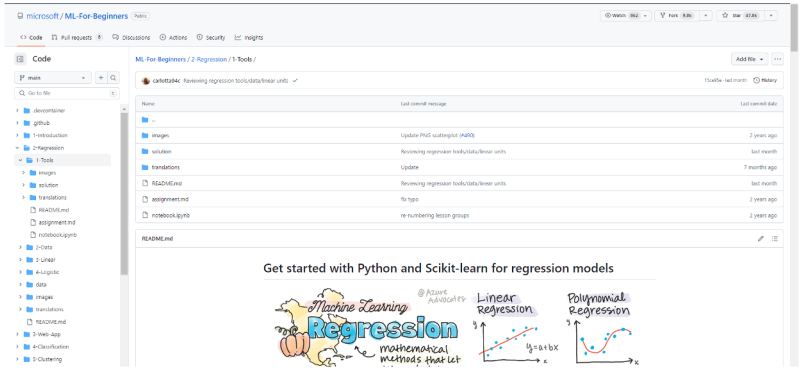
Timeline: Real-Time/Low Level C

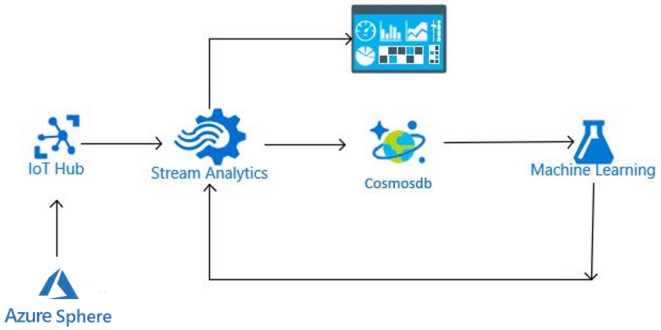


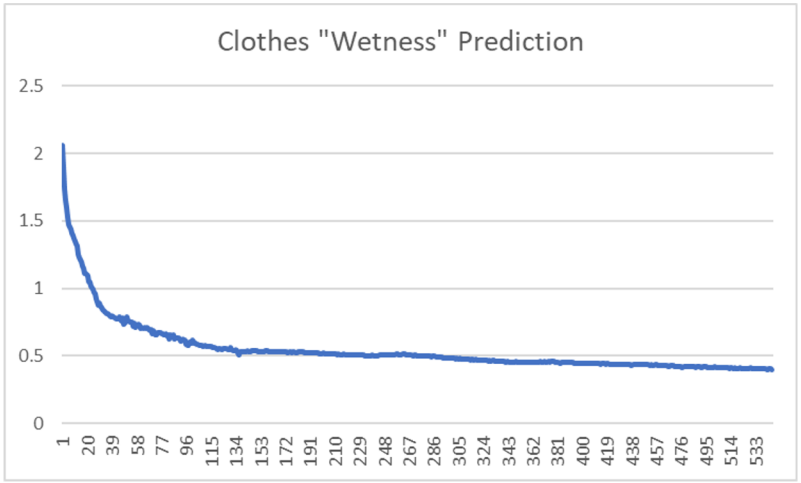
University of Pittsburgh

Timeline: Pivot to Python

- Thank You Microsoft!
- We pivoted to doing the Machine Learning on the cloud
- Did it work? Maybe



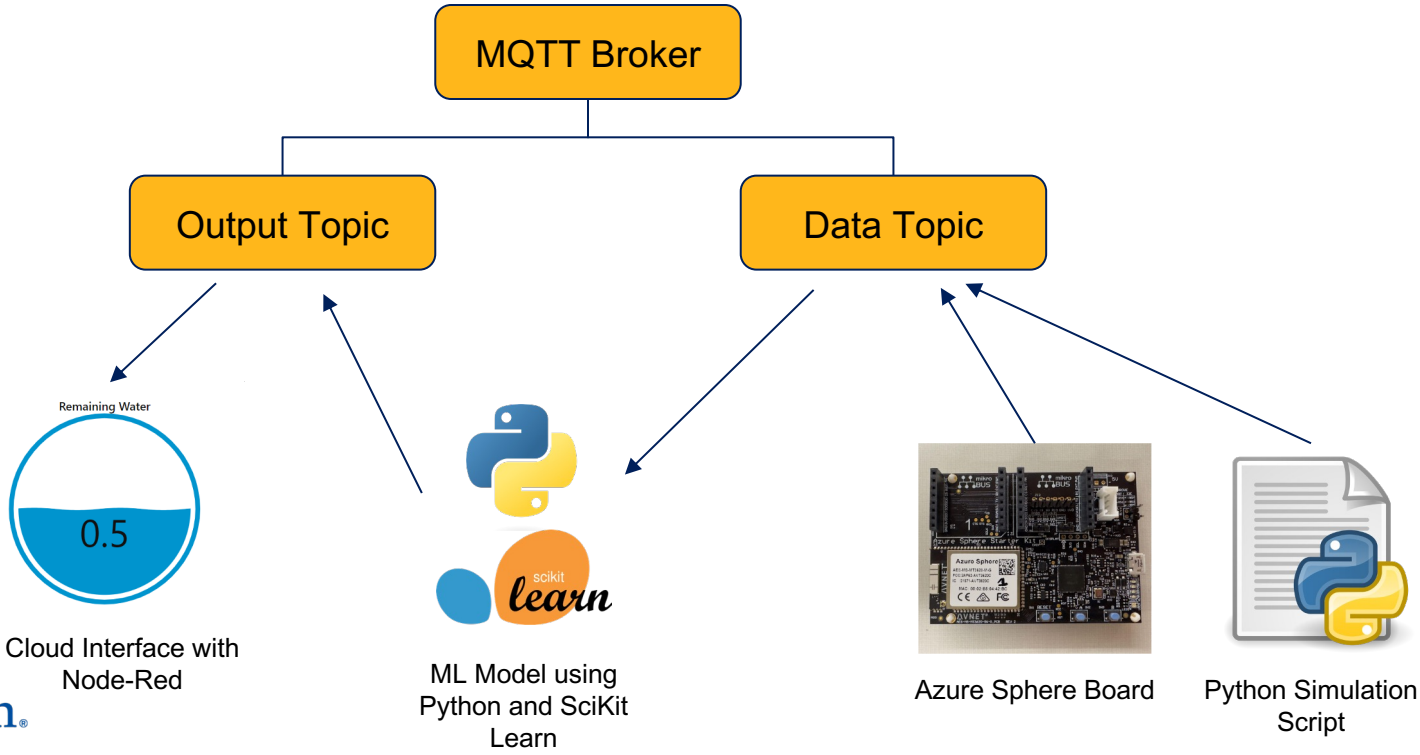




University of Pittsburgh

Timeline: Testing

- Phase 1: Testing the model by feeding data collected from the dryer
- Phase 2: Testing the model with real time data from the Azure Sphere Board*



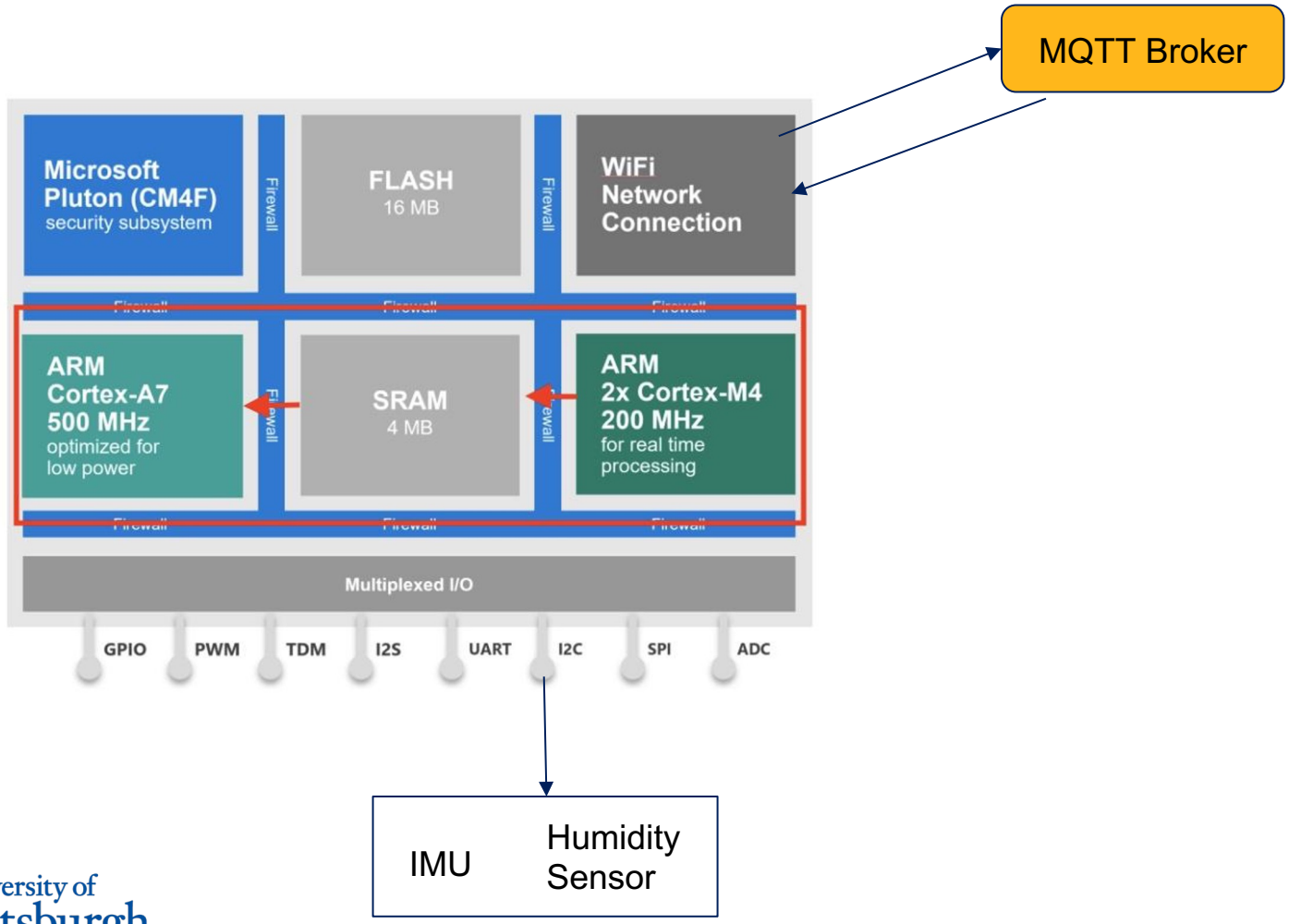
University of Pittsburgh

Outcomes - Progress So Far

- Testable product measuring acceleration
- ML Model using multiple regression
- Still in Phase 1 - Gathering additional data to inform our model

Future

Future: Humidity Sensor and Real Time Data Collection



University of Pittsburgh

Special Thank You

- Justin McCann
- Sabrina Miller
- Paige North
- Joseph Lloyd
- James Devine
- Mike Hall
- Dr. Samuel Dickerson

University of Pittsburgh

Lessons Learned

Questions