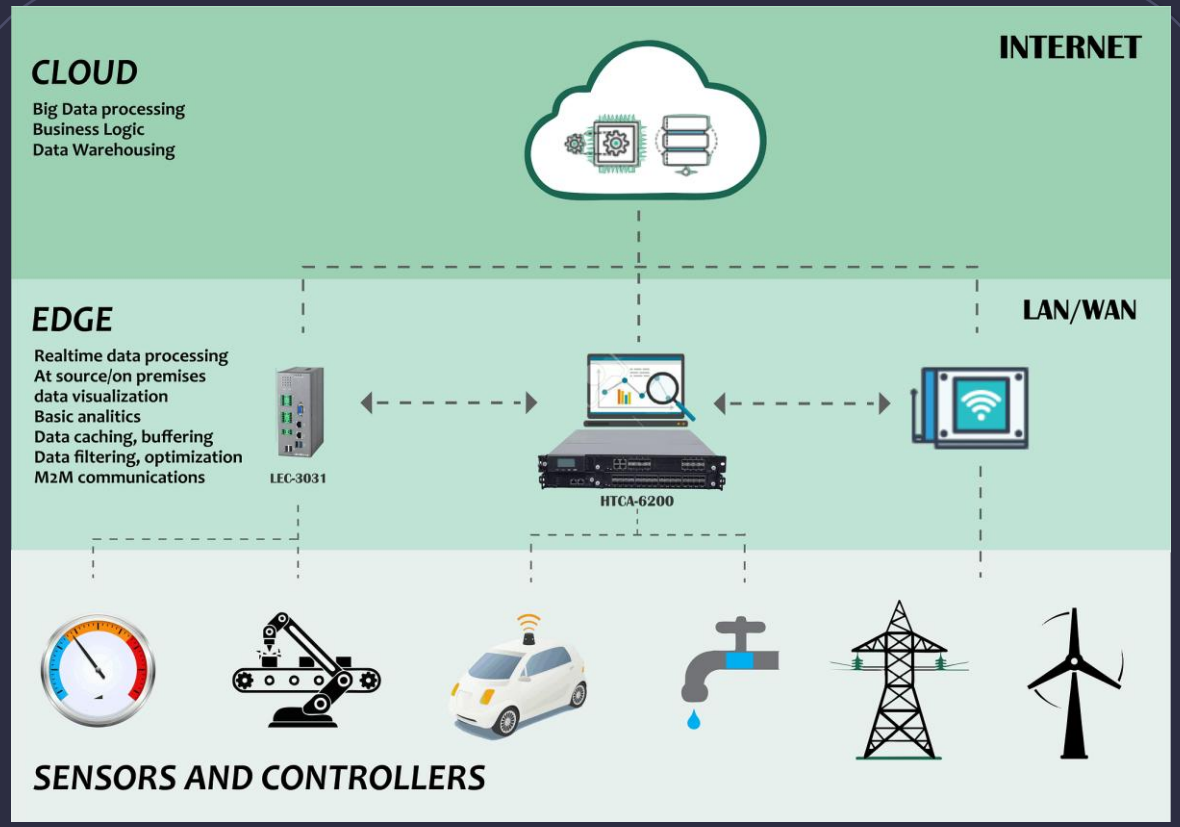
Internet of Things

System Design with Sensors
Baremetal Applications

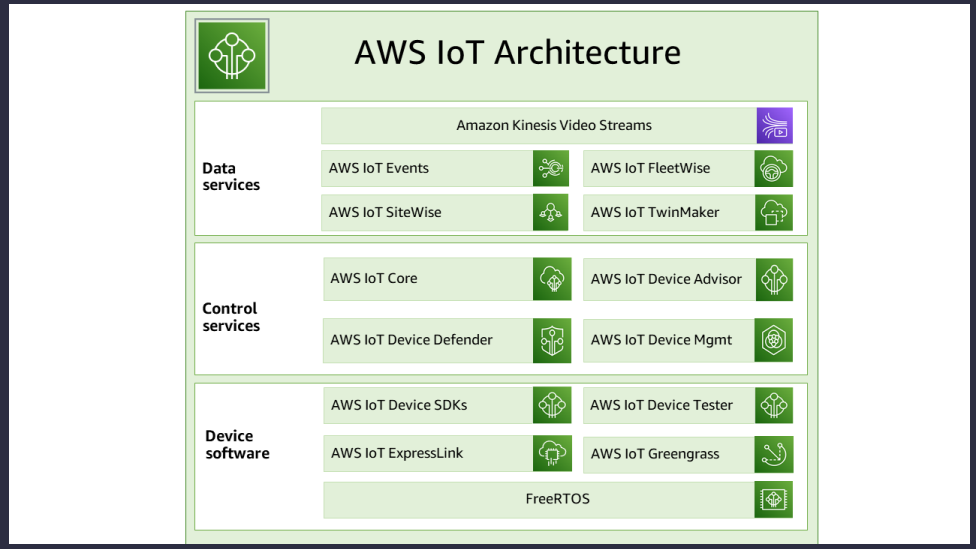
Fenerbahce University

How IoT Systems Work (End-to-End)

A practical mental model: device → network → edge/gateway → cloud → dashboard → actions



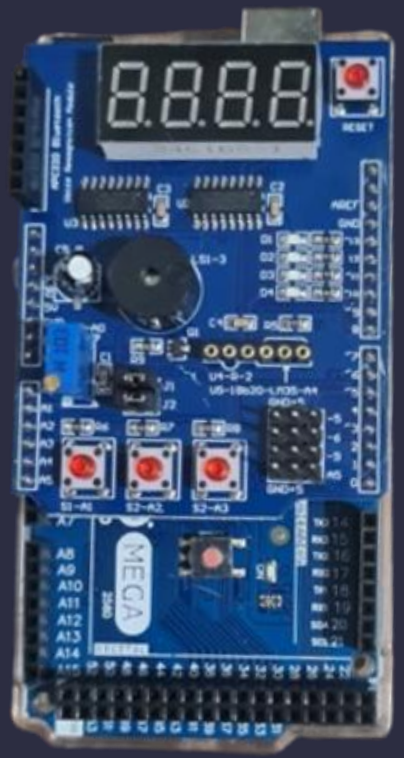
Edge–Fog–Cloud reference diagram

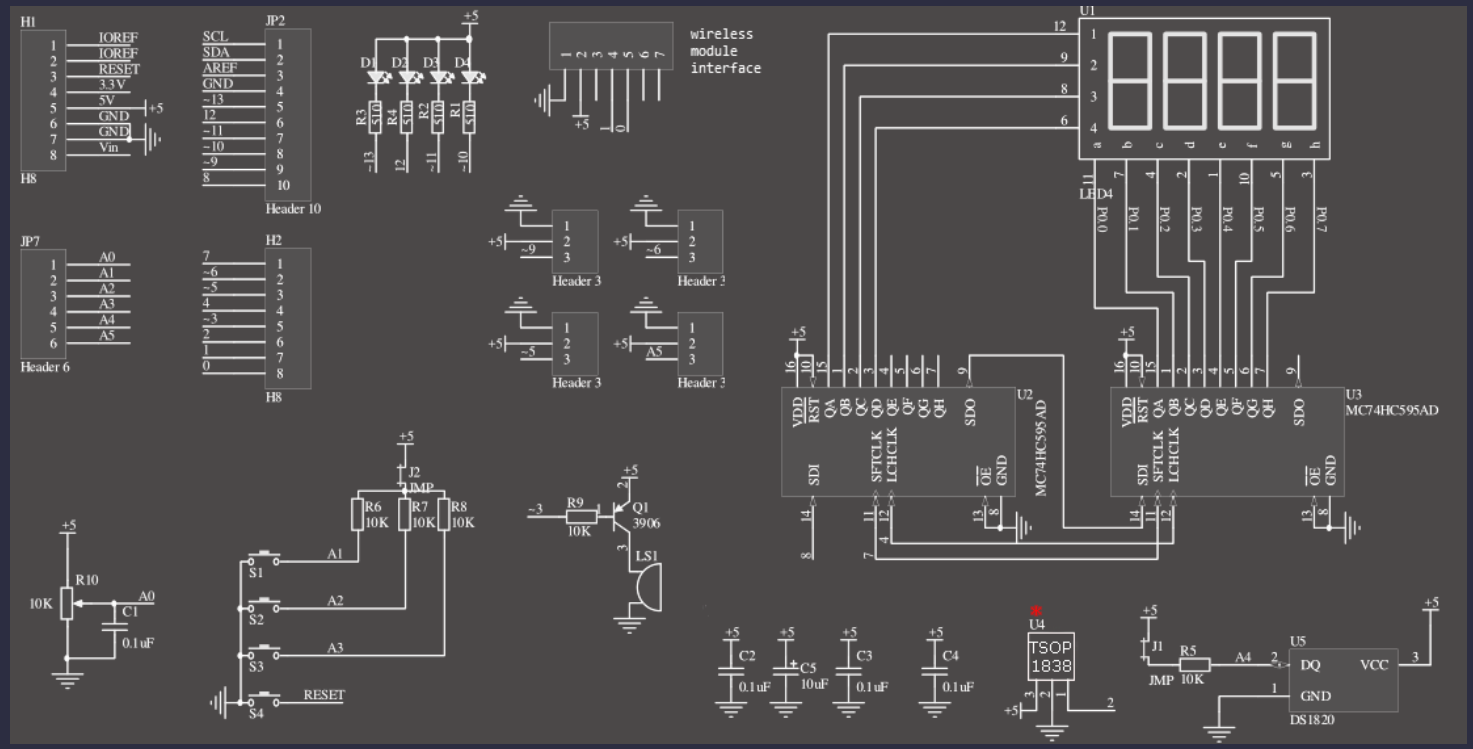


Example: AWS IoT “how it works” overview

System Design with Sensors I – Baremetal II

Arduino Mega and Shield Connection





System Design with Sensors I – Baremetal II

Simple Output

```
#define LED_PIN 13

void setup() {
  pinMode(LED_PIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_PIN, HIGH);
  delay(1000);
  digitalWrite(LED_PIN, LOW);
  delay(1000);
}
```

Program Arduino

System Design with Sensors I – Baremetal II

Simple Input & Output

```
int ledPin = 13;
int buttonPin = A1;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
}

void loop() {
  static bool ledDurum = LOW;
  if (digitalRead(buttonPin) == LOW) {
    ledDurum = !digitalRead(ledPin);
    digitalWrite(ledPin, ledDurum);
    delay(200);
  }
}
```

Program Arduino

System Design with Sensors I – Baremetal II

UART

```
int ledPin = 13;
int buttonPin = A1;
bool ledState = LOW;
bool lastButtonState = HIGH;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
  Serial.begin(9600);
}

void loop() {
  bool buttonState = digitalRead(buttonPin);

  if (lastButtonState == HIGH && buttonState == LOW) {
    ledState = !ledState;
    digitalWrite(ledPin, ledState);
    Serial.println(ledState ? "LED Closed" : "LED Open");
    delay(2000);
  }

  lastButtonState = buttonPin;
  delay(50);
}
```

Program Arduino

Open Putty

System Design with Sensors I – Baremetal II

Counter

```
int buttonPin = A1;
int buttonState = HIGH;
int lastButtonState = HIGH;
int pressCount = 0;

void setup() {
    pinMode(buttonPin, INPUT_PULLUP);
    Serial.begin(9600);
}

void loop() {
    buttonState = digitalRead(buttonPin);

    if (lastButtonState == HIGH && buttonState == LOW) {
        pressCount++;
        Serial.print("Butona basildi. Sayac: ");
        Serial.println(pressCount);
        delay(200);
    }

    lastButtonState = buttonState;
}
```

Program Arduino

Open Putty

System Design with Sensors I – Baremetal II

UART Command

```
int ledPin = 13;
char receivedChar;

void setup() {
    pinMode(ledPin, OUTPUT);
    Serial.begin(9600);
    Serial.println("Komut : 'A' = LED Ac, 'K' = LED Kapat");
}

void loop() {
    if (Serial.available() > 0) {
        receivedChar = Serial.read();

        if (receivedChar == 'A') {
            digitalWrite(ledPin, LOW);
            Serial.println("LED Acildi");
        }
        else if (receivedChar == 'K') {
            digitalWrite(ledPin, HIGH);
            Serial.println("LED Kapandi");
        }
        else {
            Serial.println("Gecersiz Komut! 'A' veya 'K' girin.");
        }
    }
}
```

Program Arduino

Open Putty

System Design with Sensors I – Baremetal II

UART Command

```
int ledPins[] = {10, 11, 12, 13};
char receivedChar;
bool ledStates[] = {LOW, LOW, LOW, LOW};

void setup() {
    for (int i = 0; i < 4; i++) {
        pinMode(ledPins[i], OUTPUT);
        digitalWrite(ledPins[i], ledStates[i]);
    }
    Serial.begin(9600);
    Serial.println("1-4 arasinda bir sayi girin.");
}

void loop() {
    if (Serial.available() > 0) {
        receivedChar = Serial.read();

        if (receivedChar >= '1' && receivedChar <= '4') {
            int ledIndex = receivedChar - '1';
            ledStates[ledIndex] = !ledStates[ledIndex];
            digitalWrite(ledPins[ledIndex], ledStates[ledIndex]);

            Serial.print("LED ");
            Serial.print(ledIndex + 1);
            Serial.print(" durumu: ");
            Serial.println(ledStates[ledIndex] ? "Acik" : "Kapali");
        }
        else {
            Serial.println("Gecersiz giris! 1-4 arasinda bir sayi girin.");
        }
    }
}
```

Program Arduino

Open Putty

System Design with Sensors I – Baremetal II

Buzzer

```
int buttonPins[] = {A1, A2, A3};
int buzzerPin = 3;
int delays[] = {2, 3, 4};

void setup() {
    for (int i = 0; i < 4; i++) {
        pinMode(buttonPins[i], INPUT_PULLUP);
    }
    pinMode(buzzerPin, OUTPUT);
    digitalWrite(buzzerPin, HIGH);
}

void loop() {
    for (int i = 0; i < 3; i++) {
        if (digitalRead(buttonPins[i]) == LOW) {
            for (int j = 0; j < 500 / delays[i]; j++) {
                digitalWrite(buzzerPin, LOW);
                delay(delays[i]);
                digitalWrite(buzzerPin, HIGH);
                delay(delays[i]);
            }
        }
    }
}
```

Program Arduino

System Design with Sensors I – Baremetal II

Seven Segment

```
#define LATCH_PIN 4
#define CLK_PIN   7
#define DATA_PIN  8

void SendDataToSegment(byte Segment_no, byte hexValue);

const byte DIGIT_MAP[] = {
  0xC0, // 0
  0xF9, // 1
  0xA4, // 2
  0xB0, // 3
  0x99, // 4
  0x92, // 5
  0x82, // 6
  0xF8, // 7
  0x80, // 8
  0x90  // 9
};

const byte SEGMENT_SELECT[] = {0xF1, 0xF2, 0xF4, 0xF8};

int counter = 0;

void SendDataToSegment(byte Segment_no, byte hexValue) {
  digitalWrite(LATCH_PIN, LOW);
  shiftOut(DATA_PIN, CLK_PIN, MSBFIRST, hexValue);
  shiftOut(DATA_PIN, CLK_PIN, MSBFIRST, Segment_no);
  digitalWrite(LATCH_PIN, HIGH);
}
```

```
void setup() {
  pinMode(LATCH_PIN, OUTPUT);
  pinMode(CLK_PIN, OUTPUT);
  pinMode(DATA_PIN, OUTPUT);
}

void loop() {
  unsigned long startTime = millis();
  while (millis() - startTime < 1000) {
    displayNumber(counter);
  }
  counter = (counter + 1) % 10000;
}

void displayNumber(int num) {
  int digits[4];

  digits[0] = num / 1000;         // Binler basamağı
  digits[1] = (num / 100) % 10;   // Yüzler basamağı
  digits[2] = (num / 10) % 10;    // Onlar basamağı
  digits[3] = num % 10;           // Birler basamağı

  for (int i = 0; i < 4; i++) {
    SendDataToSegment(SEGMENT_SELECT[i], DIGIT_MAP[digits[i]]);
    delay(2);
  }
}
```

Program Arduino

System Design with Sensors I – Baremetal II

Potentiometer

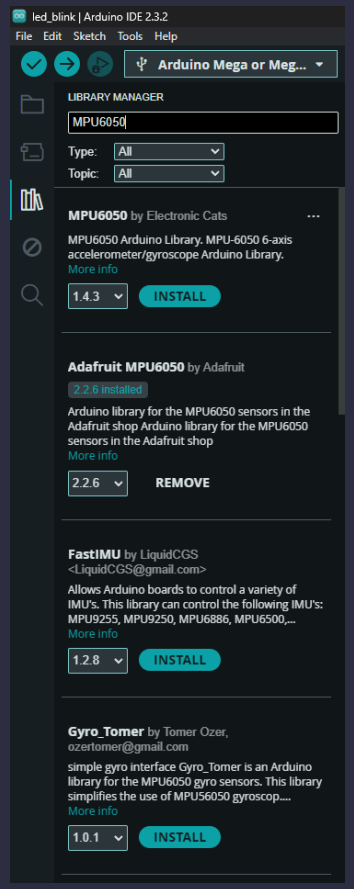
```
void setup() {
    Serial.begin(9600);
}

void loop() {
    int potValue = analogRead(A0);
    Serial.println(potValue);
    delay(500);
}
```

Program Arduino

Open Putty

System Design with Sensors I – Baremetal II

MPU6050 IMU

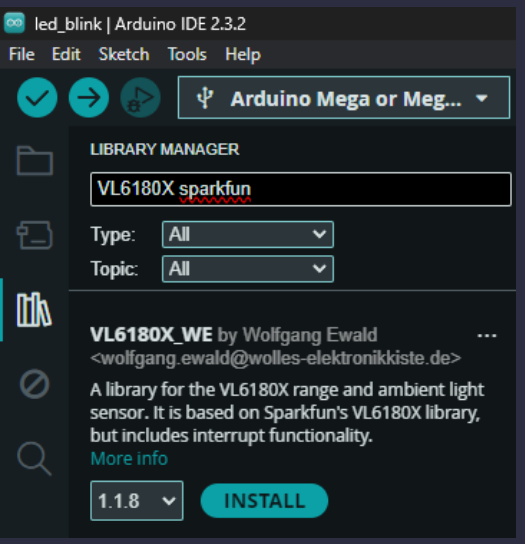



```ino
#include <Wire.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>

Adafruit_MPU6050 mpu;

void setup() {
    Serial.begin(9600);
    while (!Serial);  // Seri port açılmasını bekle (opsiyonel)

    Wire.begin();  // Arduino Mega için SDA=20, SCL=21
    if (!mpu.begin()) {
        Serial.println("MPU6050 bağlanamadı!");
        while (1);
    }

    Serial.println("MPU6050 Başlatıldı!");
}
```

```ino
void loop() {
    sensors_event_t a, g, temp;
    mpu.getEvent(&a, &g, &temp);

    // İvmeölçer ile açı hesaplama (derece cinsinden)
    float accelAngleX = atan2(a.acceleration.y, a.acceleration.z) * 180 / PI;
    float accelAngleY = atan2(a.acceleration.x, a.acceleration.z) * 180 / PI;

    // Jiroskop verilerini dereceye çevirme
    float gyroX = g.gyro.x * 180 / PI;
    float gyroY = g.gyro.y * 180 / PI;
    float gyroZ = g.gyro.z * 180 / PI;

    // Yaw, Pitch, Roll hesaplama (Temel Filtre)
    static float yaw = 0, pitch = 0, roll = 0;
    float dt = 0.01; // 10ms döngü süresi

    roll = 0.96 * (roll + gyroX * dt) + 0.04 * accelAngleX;
    pitch = 0.96 * (pitch + gyroY * dt) + 0.04 * accelAngleY;
    yaw += gyroZ * dt;

    // Seri porttan gönder
    Serial.print("Yaw: ");
    Serial.print(yaw);
    Serial.print(" | Pitch: ");
    Serial.print(pitch);
    Serial.print(" | Roll: ");
    Serial.println(roll);

    delay(10);
}
```

System Design with Sensors I – Baremetal II

TOF050C Laser Distance Measurement




```
#include <Wire.h>
#include <VL6180X_WE.h>
#define VL6180X_ADDRESS 0x29

VL6180xIdentification identification;
VL6180x sensor(VL6180X_ADDRESS);

void setup() {

  Serial.begin(9600); //Start Serial at 9600bps
  Wire.begin(); //Start I2C library

  sensor.getIdentification(&identification);
  printIdentification(&identification);

    if(sensor.VL6180xInit() != 0){
    Serial.println("FAILED TO INITALIZE");
  };

  sensor.VL6180xDefaultSettings();
  delay(100); // delay 0.1 s


}
```

```
void loop() {

  Serial.print("Distance measured (mm) = ");
  Serial.println( sensor.getDistance() );

  delay(500);
};

void printIdentification(struct VL6180xIdentification *temp){
  Serial.print("Model ID = ");
  Serial.println(temp->idModel);

  Serial.print("Model Rev = ");
  Serial.print(temp->idModelRevMajor);
  Serial.print(".");
  Serial.println(temp->idModelRevMinor);

  Serial.print("Module Rev = ");
  Serial.print(temp->idModuleRevMajor);
  Serial.print(".");
  Serial.println(temp->idModuleRevMinor);

  Serial.print("Manufacture Date = ");
  Serial.print((temp->idDate >> 3) & 0x001F);
  Serial.print("/");
  Serial.print((temp->idDate >> 8) & 0x000F);
  Serial.print("/1");
  Serial.print((temp->idDate >> 12) & 0x000F);
 }
```