TFB2093 Practical 07 (Tinkercad Edition)

Simulating an IoT Gateway with Two Arduinos

> **Tinkercad-only, step-by-step lecture sheet**: We'll mimic a Raspberry Pi gateway by using **two Arduinos** — one as the **Sensor Node**, one as the **Gateway**.

Learning outcomes

- Build a virtual circuit that demonstrates **device → gateway** data flow.
- Wire an **LM35** temperature sensor and stream readings as **CSV** lines.
- Use **SoftwareSerial** to send data from one Arduino to another.
- Display and "log" incoming data on the **gateway's Serial Monitor**.
- Understand how this maps to a real **Raspberry Pi** gateway.

---

0) What you'll build (concept)

Sensor Node (Arduino #1) reads temperature (LM35) → sends lines like `timestamp_ms,tempC` over a **TX wire**.

Gateway (Arduino #2) listens to that line → prints readable logs to **Serial Monitor**, blinks LED on high temp.

---

1) Parts list (Tinkercad)

- 2 × Arduino Uno R3
- 1 × Breadboard (half is enough)
- 1 × LM35 temperature sensor
- Jumper wires (male–male)
- (Optional) 1 × LED + 220Ω resistor for gateway pin 13 (onboard LED works too)

---

2) Wiring (step-by-step)

A) Sensor Node (Arduino #1) + LM35

1. Place **Arduino #1** and a **breadboard**.
2. **LM35 orientation** (flat face toward you):
3. **Left pin → 5V**
4. **Middle pin → A0**
5. **Right pin → GND**
6. This Arduino's **TX pin for the link** will be **D8** (via SoftwareSerial).

B) Gateway (Arduino #2)

1. Place **Arduino #2** nearby.
2. **Inter-Arduino link (one-way)**
3. **Sensor D8 (TX) → Gateway D10 (RX)**
4. **Sensor GND ↔ Gateway GND** (common ground is essential)
5. Do **not** connect the 5V lines between boards.
6. Gateway will blink **LED on pin 13** when temp is high (uses onboard LED).

Quick wiring table

From	To	Purpose
LM35 left	Arduino #1 5V	Sensor power
LM35 middle	Arduino #1 A0	Sensor output
LM35 right	Arduino #1 GND	Sensor ground
Arduino #1 D8	Arduino #2 D10	Data TX → RX
Arduino #1 GND	Arduino #2 GND	Common ground

---

3) Code

A) Sensor Node (Arduino #1)

```
#include <SoftwareSerial.h>
SoftwareSerial link(7, 8); // RX, TX  (we use TX on D8)
const int LM35_PIN = A0;

float readTempC() {
  int raw = analogRead(LM35_PIN);
  float volts = (raw * 5.0) / 1023.0;
  return volts * 100.0; // LM35: 10 mV/°C
}

void setup() {
  Serial.begin(9600); // optional local debug
  link.begin(9600);   // link to gateway
}

void loop() {
  float t = readTempC();
  unsigned long ts = millis();

  // Send CSV line: timestamp_ms,tempC
  link.print(ts);
  link.print(",");
  link.println(t, 2);
```

```
  // Optional debug:
  // Serial.print("Sent: "); Serial.print(ts); Serial.print(",");
Serial.println(t, 2);

  delay(1000);
}
```

B) Gateway (Arduino #2)

```
#include <SoftwareSerial.h>
SoftwareSerial link(10, 9); // RX, TX  (we use RX on D10)
const int LED_PIN = 13;     // onboard LED

void setup() {
  pinMode(LED_PIN, OUTPUT);
  Serial.begin(9600); // to PC (Tinkercad Serial Monitor)
  link.begin(9600);   // incoming from sensor
  Serial.println("Gateway ready. Waiting for data (ts,tempC)...");
}

void loop() {
  static String line = "";

  while (link.available()) {
    char c = link.read();
    if (c == '\n') {
      int comma = line.indexOf(',');
      if (comma > 0) {
        String ts = line.substring(0, comma);
        float tempC = line.substring(comma + 1).toFloat();

        // Pretty print + simple "log"
        Serial.print("ts="); Serial.print(ts);
        Serial.print(" ms, tempC="); Serial.println(tempC, 2);

        // Threshold alert
        digitalWrite(LED_PIN, tempC > 30.0 ? HIGH : LOW);
      }
      line = ""; // reset buffer
    } else if (c != '\r') {
      line += c;
    }
  }
}
```

---

4) Run it in Tinkercad (exact steps)

1. Create a **new Circuit**. Place two Arduinos and one LM35.
2. Do the wiring from Section 2.
3. Click Arduino #1 (Sensor Node) → **Code** → switch to **Text** → paste **Sensor code** → **Done**.
4. Click Arduino #2 (Gateway) → **Code** → **Text** → paste **Gateway code** → **Done**.
5. Click **Start Simulation**.
6. Open the **Serial Monitor** for **Arduino #2 (Gateway)** (monitor icon).
   You should see lines like: `ts=1234 ms, tempC=26.53`
7. Warm the LM35 with the mouse (or adjust environment) and watch **LED 13** toggle when temp > 30°C.

---

5) Mapping to a real Pi gateway (concept)

- In real hardware, the **Sensor Node** could be an Arduino sending serial or **MQTT** data.
- The **Gateway** here (Arduino #2) simulates the *receive* role.
- A **Raspberry Pi** would instead read serial or subscribe to MQTT, run Python (`pyserial`, `paho-mqtt`), and store/serve data (file/DB/REST/UI).

---

6) Discussion prompts

- Why must we share **GND** between boards?
- CSV vs JSON lines for streaming sensor data—pros/cons?
- Where would **data validation** and **timestamping** happen in a real gateway?
- If we replace the wire link with **MQTT**, what changes? (broker, topics, protocol)

---

7) Troubleshooting quick checks

- **No logs?** Confirm **Sensor D8 → Gateway D10** and **GND ↔ GND**.
- **Garbled values?** Ensure both ends use **9600 baud**.
- **Always ~0°C or ~100°C?** Re-check **LM35 pins** (left=5V, middle=A0, right=GND with flat face toward you).
- **LED not blinking?** Lower threshold (e.g., 28.0) or warm the sensor longer.

---

8) Stretch goals (optional)

- Replace LM35 with **LDR** or **potentiometer** to stream `ts,value`.
- Change CSV to **JSON**: `{"ts":1234,"tempC":26.53}` (harder parse).
- Add a simple **moving average** on the gateway.
- Add a one-time **CSV header**: `ts_ms,tempC`.

Demo checklist (show in your submission)

- Circuit screenshot (two Arduinos + LM35 + link wire + common GND)
- Gateway Serial Monitor showing **ts,tempC** lines
- LED reaction when temperature crosses threshold

2.1 Exact breadboard layout (with TMP36 fallback)

> **Tinkercad note:** Some libraries label the analog temperature sensor as **TMP36** (very common in Tinkercad). If you only find **TMP36**, use it instead of LM35 and change the math (see below).

LM35 vs TMP36 wiring & math

- **Wiring is the same** (Vcc, OUT, GND).
- **Math differs**:
- **LM35**: `°C = (analogRead * 5.0 / 1023.0) * 100.0`
- **TMP36**: `°C = ((analogRead * 5.0 / 1023.0) - 0.5) * 100.0` (because TMP36 has 500 mV offset)

Visual wiring checklist

- [ ] Flat face of sensor toward you
- [ ] Left → **5V**, Middle → **A0**, Right → **GND** (sensor → Arduino #1)
- [ ] **D8 (TX)** on Arduino #1 → **D10 (RX)** on Arduino #2
- [ ] **GND** ↔ **GND** between boards
- [ ] (Optional) LED+220Ω on Arduino #2 **D13 → resistor → LED → GND** (or use onboard LED)

3.1 Full code with headers & robustness

A) Sensor Node (Arduino #1) — LM35 version with one-time CSV header

```
#include <SoftwareSerial.h>
SoftwareSerial link(7, 8); // RX, TX (we use TX on D8)
const int LM35_PIN = A0;
bool headerPrinted = false;

float readTempC() {
  int raw = analogRead(LM35_PIN);
  float volts = (raw * 5.0) / 1023.0;
  return volts * 100.0; // LM35: 10 mV/°C
}

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

```
void loop() {
  if (!headerPrinted) {
    link.println("ts_ms,tempC");
    headerPrinted = true;
  }
  float t = readTempC();
  unsigned long ts = millis();
  link.print(ts); link.print(","); link.println(t, 2);
  delay(1000);
}
```

TMP36 math drop-in (if you used TMP36 sensor)

Replace `readTempC()` with:

```
float readTempC() {
  int raw = analogRead(A0);
  float volts = (raw * 5.0) / 1023.0;
  return (volts - 0.5) * 100.0; // TMP36: 500 mV offset
}
```

B) Gateway (Arduino #2) — more robust line parsing + threshold

```
#include <SoftwareSerial.h>
SoftwareSerial link(10, 9); // RX, TX (we use RX on D10)
const int LED_PIN = 13;     // onboard LED
const float HIGH_TEMP = 30.0; // °C threshold

void setup() {
  pinMode(LED_PIN, OUTPUT);
  Serial.begin(9600);
  link.begin(9600);
  Serial.println("Gateway ready. Waiting for data (ts_ms,tempC)...");
}

void loop() {
  static String line = "";
  while (link.available()) {
    char c = link.read();
    if (c == '
') {
      // skip headers or empty lines
      if (line.length() > 0 && line.indexOf("ts_ms") == -1) {
        int comma = line.indexOf(',');
        if (comma > 0) {
          String ts = line.substring(0, comma);
          String val = line.substring(comma + 1);
```

```
          float tempC = val.toFloat();

          // Log as pretty text and as CSV echo
          Serial.print("ts="); Serial.print(ts);
          Serial.print(" ms, tempC="); Serial.println(tempC, 2);
          Serial.print("CSV:"); Serial.print(ts); Serial.print(",");
Serial.println(tempC, 2);

          // Threshold alert
          digitalWrite(LED_PIN, tempC > HIGH_TEMP ? HIGH : LOW);
        }
      }
      line = ""; // reset
    } else if (c != '
') {
      line += c;
    }
  }
}
```

3.2 (Optional) Moving-average smoothing on Gateway

Smooth noisy readings before thresholding (window N=5).

```
// Add at top
#define N 5
float ring[N];
int idx = 0; int filled = 0;

float pushAndAvg(float x) {
  ring[idx] = x; idx = (idx + 1) % N; if (filled < N) filled++;
  float s = 0; for (int i = 0; i < filled; i++) s += ring[i];
  return s / max(1, filled);
}
```

Inside parsing block, replace `tempC` usage with:

```
float avg = pushAndAvg(tempC);
Serial.print("avgC="); Serial.println(avg, 2);
digitalWrite(LED_PIN, avg > HIGH_TEMP ? HIGH : LOW);
```

4.1 Exact Tinkercad run steps (with screenshots placeholders)

1. **Create Circuit** → search and place: *Arduino Uno*, *Arduino Uno*, *TMP36 (or LM35)*, *Breadboard*.

2. **Wire sensor** to Arduino #1: *(5V → Vcc, A0 → OUT, GND → GND)*.
   (Screenshot placeholder: Sensor wiring close-up)
3. **Data link**: Arduino #1 **D8** → Arduino #2 **D10**, **GND ↔ GND**.
   (Screenshot placeholder: D8→D10 and GND↔GND jumpers)
4. Paste **Sensor code** to Arduino #1 (Text mode), **Gateway code** to Arduino #2.
   (Screenshot placeholder: Code editor view)
5. **Start Simulation** → open **Serial Monitor of Arduino #2**.
   (Screenshot placeholder: Serial log showing ts_ms,tempC)
6. Warm sensor or adjust environment → observe **LED 13** threshold behavior.
   (Screenshot placeholder: LED on)

   CSV tip: Copy the `CSV:` lines from Serial Monitor and paste into a spreadsheet as `ts_ms,tempC`.

5.1 Common pitfalls & fixes (mini-FAQ)

- **Nothing prints:** Check baud rate (9600 ↔ 9600), verify D8→D10, and shared GND.
- **Always same temperature:** Ensure the correct sensor model math (LM35 vs TMP36).
- **LED never turns on:** Lower `HIGH_TEMP` to 28 or simulate a warmer environment longer.
- **Random characters / garbled lines:** Avoid using very high baud; 9600 is safe in Tinkercad.

---

6) Student tasks (ready-to-assign)

1. **Baseline demo:** Wire and run the two-Arduino gateway. Show 30 seconds of Serial output.
2. **Sensor swap:** Replace LM35/TMP36 with a **potentiometer**. Stream `ts_ms,adc` and toggle LED when `adc > 700`.
3. **Smoothing:** Add the moving-average filter and compare LED behavior with/without smoothing (2 screenshots + 3-line explanation).
4. **CSV export:** Copy 20 `CSV:` lines into a sheet and plot `tempC vs ts_ms`. Add a 1-sentence observation.

---

7) How this maps to a real Raspberry Pi (simple mental model)

- **Serial link** here ≈ **USB Serial** on a Pi.
- **Gateway printing** here ≈ **Pi Python script** storing to file/DB and serving a web API.
- **Two boards** here ≈ **Device + Gateway** in real IoT.

> In class, we use Tinkercad to master the **signals & data flow** first; later, we swap the Gateway Arduino for a Pi.

---

8) Assessment checklist (for quick grading)

- [ ] Correct wiring (sensor, D8→D10, common GND)
- [ ] Sensor code compiles and sends header + CSV lines
- [ ] Gateway prints parsed values (and CSV echo)

- [ ] LED reacts to threshold
- [ ] (Optional) Moving-average implemented and demonstrated

---

9) Bonus challenges (for fast finishers)

- **JSON mode:** Send `{\"ts\":123,\"tempC\":26.5}` and parse on gateway (hint: find positions of `:` and `,`).
- **Two-way link:** Add **ACK** back from Gateway (use its TX on D9 to Sensor RX on D7) and blink Sensor LED when ACK received.
- **Timestamp reset:** Add a button to zero the timer and send `ts=0` event.