TFB2093 Internet-of-Things

Practical 06: Interfacing with Analog Sensors & Actuators (TinkerCAD-Only)

> **Class duration:** ~2 hours
> **Target board:** Arduino Uno (in TinkerCAD Circuits)
> **Tools:** TinkerCAD Circuits → *New Circuit* → Arduino Uno R3 → Serial Monitor / Serial Plotter

---

1) Learning Objectives

By the end of this lab, you will be able to: - Wire **analog sensors** (potentiometer, LDR, LM35 temperature) to Arduino Uno in TinkerCAD. - Use `analogRead()` to capture sensor values (0–1023) and print to **Serial Monitor** & **Serial Plotter**. - Map raw readings to human-friendly units (%, °C, 0–180°). - Drive an **analog-controlled actuator** (servo) using sensor input. - Apply basic **smoothing/ denoising** to noisy analog data.

---

2) Bill of Materials (TinkerCAD parts)

- Arduino Uno R3
- Breadboard (small or half)
- **Potentiometer** (10 kΩ)
- **Photoresistor / LDR** + **10 kΩ** fixed resistor (voltage divider)
- **LM35** temperature sensor
- **Servo motor** (standard hobby servo)
- Jumper wires (M–M)

> **Ground rule for wiring:** Each sensor's **GND → Arduino GND**, **VCC → 5V**, **signal → analog pin (A0/A1/...)**

---

3) Fast Setup Guides (TinkerCAD)

A) Potentiometer → A0

- Potentiometer has 3 pins: **left = GND**, **middle = signal A0**, **right = 5V** (or swap left/right; direction just flips).
- In TinkerCAD: Wire **5V → + rail**, **GND → − rail**, then connect pot pins accordingly.

B) LDR (Photoresistor) Voltage Divider → A1

```
5V ─ LDR ─●─ A1
          |
```

```
        10kΩ
         |
GND
```

- Node ● goes to **A1**. - Choose **10 kΩ** as the fixed resistor (typical). Higher resistance → more sensitivity in low light.

C) LM35 → A2

- **Pinout (flat face toward you):** left=**Vout** → **A2**, middle=**GND**, right=**+5V**.
- LM35 outputs **10 mV/°C** → use conversion formula below.

D) Servo → D9

- **Red → 5V**, **Brown/Black → GND**, **Yellow/Orange (signal) → D9**.
- Use the **Servo** library.

---

4) Activity 1 — Read a Potentiometer & Print Percentage

Sketch (copy to TinkerCAD > Code > Text):

```
// Activity 1: Potentiometer on A0 → % on Serial Monitor
const int POT_PIN = A0;

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

void loop() {
  int raw = analogRead(POT_PIN);        // 0..1023
  float pct = (raw / 1023.0) * 100.0;   // map to 0..100%
  Serial.print("raw="); Serial.print(raw);
  Serial.print("\tpercent="); Serial.println(pct, 1);
  delay(50);
}
```

Try: Turn the knob and watch values change in **Serial Monitor**.

Extension: Use `map()` and `constrain()`

```
int raw = analogRead(A0);
int pct = map(raw, 0, 1023, 0, 100);
pct = constrain(pct, 0, 100);
```

---

5) Activity 2 — Serial Plotter (Real-Time Graph)

Tip: The classic Arduino Serial Plotter expects **one or more numeric series per line** separated by spaces/tabs. Labels are optional.

Sketch:

```
// Activity 2: Potentiometer waveform for Serial Plotter
const int POT_PIN = A0;

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

void loop() {
  int raw = analogRead(POT_PIN);
  // Print ONE value per line for a single trace:
  Serial.println(raw);
  // For multiple traces, print like: Serial.print(val1); Serial.print("\t");
Serial.println(val2);
  delay(20);
}
```

Observe: Open **Serial Plotter** and rotate the pot slowly. You'll see a smooth ramp up/down.

---

6) Activity 3 — LDR Light Level (A1)

Sketch:

```
// Activity 3: LDR on A1 with 10k divider → plot light level
const int LDR_PIN = A1;

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

void loop() {
  int raw = analogRead(LDR_PIN); // Higher raw typically = brighter (depends
on wiring)
  Serial.println(raw);
  delay(50);
}
```

Try: Wave a hand over the LDR, shine the TinkerCAD desk lamp, or change the environment light slider (component inspector).

Discussion prompt: Limitations of LDRs — slow response vs. photodiodes, temperature dependence, poor absolute accuracy, non-linear output, needs calibration, varies per part.

---

7) Activity 4 — LM35 Temperature in °C (A2)

Conversion (5V reference):
`tempC = (analogRead(A2) * 5.0 / 1023.0) * 100.0;`

Sketch:

```
// Activity 4: LM35 on A2 → Celsius
const int LM35_PIN = A2;

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

void loop() {
  int raw = analogRead(LM35_PIN);
  float voltage = raw * (5.0 / 1023.0); // Vout
  float tempC = voltage * 100.0;        // 10 mV per °C
  Serial.print("tempC="); Serial.println(tempC, 2);
  delay(250);
}
```

Note: In real hardware, ADC reference and wiring noise affect accuracy; in TinkerCAD, values are idealized.

---

8) Activity 5 — Servo Position from Potentiometer (A0 → D9)

Sketch:

```
// Activity 5: Pot controls Servo angle (0..180°)
#include <Servo.h>

const int POT_PIN = A0;
const int SERVO_PIN = 9;
Servo myServo;

void setup() {
  Serial.begin(9600);
  myServo.attach(SERVO_PIN);
}

void loop() {
  int raw = analogRead(POT_PIN);           // 0..1023
```

```
  int angle = map(raw, 0, 1023, 0, 180);   // map to degrees
  myServo.write(angle);
  Serial.print("raw="); Serial.print(raw);
  Serial.print("\tangle="); Serial.println(angle);
  delay(20);
}
```

Watch: Start Simulation → Servo sweeps as you turn the pot.

---

9) Optional: Smooth / Filter Noisy Analog Data

A) Moving Average (boxcar)

```
// N-point moving average
const int N = 10;
int buf[N];
int idx = 0;
long sum = 0;

int smoothRead(int pin) {
  sum -= buf[idx];
  buf[idx] = analogRead(pin);
  sum += buf[idx];
  idx = (idx + 1) % N;
  return sum / N;
}
```

Use as: `int raw = smoothRead(A0);`

B) Simple Median of 5

```
int median5(int pin){
  int v[5];
  for(int i=0;i<5;i++) v[i]=analogRead(pin);
  // insertion sort
  for(int i=1;i<5;i++){
    int k=v[i], j=i-1;
    while(j>=0 && v[j]>k){ v[j+1]=v[j]; j--; }
    v[j+1]=k;
  }
  return v[2];
}
```

When to use: Moving average for random noise; median for spike outliers.

---

10) Serial Plotter — Multi-Series Demo (Compare Raw vs. Smoothed)

```
// Plot raw vs. smoothed pot values
const int POT_PIN = A0;
const int N = 10;
int buf[N]; int idx=0; long sum=0;

int smoothA0(){
  sum -= buf[idx];
  buf[idx] = analogRead(POT_PIN);
  sum += buf[idx];
  idx = (idx + 1) % N;
  return sum / N;
}

void setup(){ Serial.begin(9600); }
void loop(){
  int raw = analogRead(POT_PIN);
  int sm  = smoothA0();
  Serial.print(raw); Serial.print('\t'); Serial.println(sm);
  delay(20);
}
```

Open **Serial Plotter** → you'll see **two lines** (raw & smoothed).

---

11) Troubleshooting Cheatsheet (TinkerCAD)

- **Flat lines at 0 or 1023?** Check wiring (VCC/GND reversed, wrong analog pin).
- **No serial output?** `Serial.begin(9600)` and **Start Simulation** first; open **Serial Monitor/Plotter**.
- **Jittery plot?** Add smoothing (Section 9) or increase `delay()`.
- **Servo twitching?** Ensure **GND is common** between servo and Arduino; avoid writing angles too fast; add small delay.
- **LM35 wrong temps?** Confirm pinout and formula (`tempC = voltage * 100`).

---

12) Reflection / Viva Questions

- Contrast **digital vs. analog** sensor interfacing (wiring, code, resolution, noise, latency).
- Why map/scale analog values? Give two examples (percent display, servo angle, temperature).
- Name two ways to **smooth** analog data; when is each preferable?
- Explain the **voltage divider** for the LDR and why we need it.

---

13) Quick Assessment (Instructor)

- ✅Pot wired to A0, % printed on Serial.
- ✅Plotter shows live waveform.
- ✅LDR divider works; values vary with light.
- ✅LM35 prints realistic °C.
- ✅Servo controlled by pot with sensible mapping.
- ✅Student explains smoothing and voltage divider.

---

14) Extension Ideas (If time remains)

- Add **threshold** to switch an LED on when light < X (from LDR).
- Map LM35 **°C → bar graph** on 8 LEDs using `map()`.
- Combine **two sensors** (e.g., pot & LDR) and plot both series.

Submission tip (if required): Export each circuit's link/screenshot + paste your code. Include a 2–3 line observation for each activity.