CEC Continuing Education Center

DesignNews

Understanding Industrial Controls with an ESP32

Day 3: Understanding Sensors

Sponsored by

DigiKey

informa markets

Webinar Logistics

- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click “Help” or submit a question asking for assistance.
- Participate in ‘Attendee Chat’ by maximizing the chat widget in your dock.

Dr. Don Wilcher

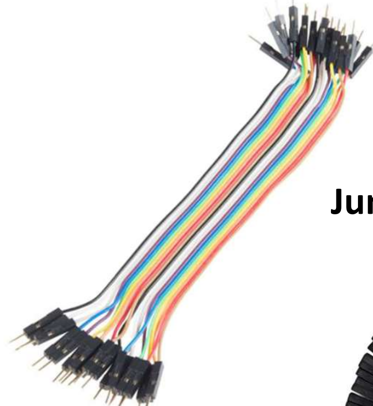
Visit 'Lecturer Profile' in your console for more details.

Course Kit and Materials

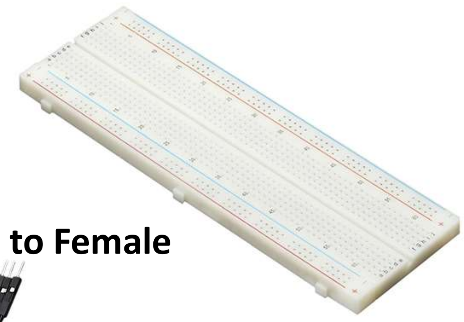
ESP32-DEVKITC-VIE

Adafruit Parts Pal Kit

Jumper Wires: Male to Male

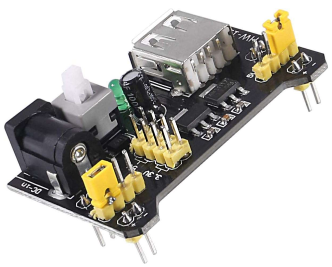


Solderless Breadboard x2



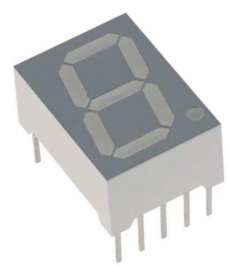
Jumper Wires: Male to Female

Solderless Breadboard Power Supply



5V Relay Module, 5V Indicator Light LED

7 Segment LED Display, Common Cathode



Standard Motor, 9100 RPM 6VDC

Agenda:

- Sensors Introduction
- Modeling Sensors
- A Simple Data Logging Technique
- Lab: Build An ESP32 Electronic Thermometer

CEC Continuing Education Center

DigiKey

Research Perspective

"Programmable logic controllers (PLCs) provide an ecosystem of relatively simple software logic, robust and ruggedized hardware, networks with controllable real-time behaviors, and extensive availability of interoperable components such as sensors and actuators" (Sehr et al., 2021).

CEC Continuing Education Center

DigiKey

Course Question
Can an ESP32 microcontroller contribute to the Industrial Controls field?

CEC Continuing Education Center

DigiKey

Sensors Introduction

- A sensor is a semiconductor device that samples the physical properties of natural environments.
- Physical properties of natural environments include:
  a) sound
  b) light
  c) temperature
  d) motion
- Through this physical property stimulus, a proportional current or voltage is produced.
- Another term used to describe sensors in process control environments is a transducer.

Continuing Education Center

Sensors Introduction. . .

- A transducer converts mechanical, sound, magnetic, thermal, and optical into:
  - a) electrical voltage signals
  - b) or electrical current signals.
- These electrical signals are wired to a microcontroller for processing and controlling actuators, audible, and visual alarm devices.

Question 1

What is a sensor?

a) **Electromechanical device that samples the physical properties of natural environments.**

b) **Electrical device that samples the physical properties of natural environments.**

c) **A semiconductor device that samples the physical properties of physical environments.**

d) **none of the above**

Sensors Introduction. . .

Microphone: Converts sound waves into electrical signals.

Piezoelectric Sensor: Converts mechanical stress into electrical energy.

Transducer Examples

Thermocouple: Converts heat (temperature differences) into electrical voltage.

Photodiode: Converts light intensity into electrical signals.

Sensors Introduction. . .

Thermistor: Converts temperature variations into resistance changes, which can be measured electrically.

Pressure Transducer: Converts pressure into an electrical signal.

Transducer Examples

Reed Switch: An electrical current flow occurs when a magnetic field is detected.

Modeling Sensors

- Modeling sensors mathematically involves understanding how sensors convert physical quantities (like temperature, pressure, or light intensity) into measurable electrical signals.
- This can be done by deriving mathematical relationships between the sensor’s input (physical stimulus) and output (electrical signal).
- Here’s an outline of the steps for creating a mathematical model for sensors:

Modeling Sensors. . .

1. Define the Sensor's Purpose and Input-Output Relationship

- Identify the physical quantity the sensor is intended to measure, like temperature, light, or pressure.
- Then, determine the output type (such as voltage or current) that the sensor will produce in response to the physical input

Modeling Sensors. . .

2. Determine the Sensor's Transfer Function

The transfer function is a mathematical equation that relates the input x (physical variable) to the output y (electrical signal). For many sensors, this can be a linear or nonlinear relationship:

- **Linear Sensors**: Often represented as $y = Kx + C$, where K is a sensitivity coefficient and C is a constant offset.
- **Nonlinear Sensors**: Can have polynomial, exponential, or logarithmic relationships. For example, a thermistor (a temperature sensor) might have an exponential response: $R(T) = R_0 e^{\beta\left(\frac{1}{T} - \frac{1}{T_0}\right)}$.

Modeling Sensors. . .

3. Model Dynamic Behavior (for Time-Dependent Responses)

Some sensors have time-dependent responses, meaning they take time to reach the final output value when the input changes. This dynamic behavior can be modeled with differential equations:

- **First-order Systems**: Many sensors are modeled as first-order systems if they gradually approach a final value. The relationship can be expressed as:

$$\tau \frac{dy}{dt} + y = Kx$$

  where τ is the time constant, representing how quickly the sensor responds.

- **Higher-order Systems**: For complex sensors with overshoot or oscillations, higher-order differential equations may be used.

Modeling Sensors. . .

4. Account for Non-Idealities (Noise, Offset, Drift)

Real sensors have imperfections. Common non-idealities include:

- **Noise**: Random fluctuations that can be modeled statistically (e.g., Gaussian noise).
- **Offset**: A constant bias that shifts the output.
- **Drift**: A slow change in sensor characteristics over time, often modeled as a function of time or environmental factors.

Modeling Sensors. . .

What is Gaussian Noise?

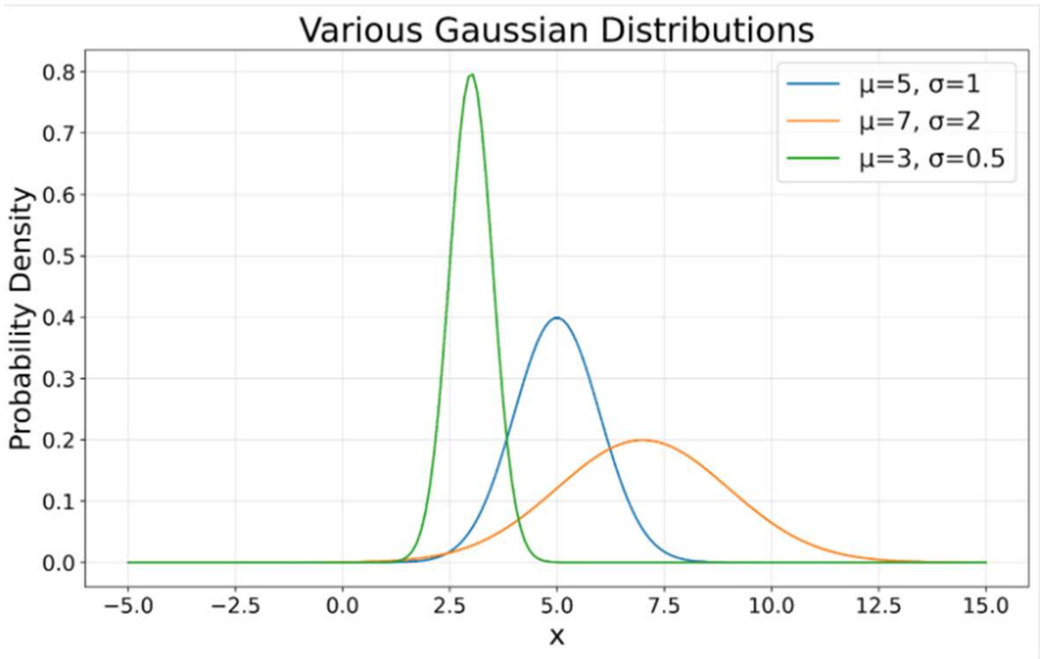
Known as normal noise is a random transient that follows a Gaussian Distribution.

Modeling Sensors. . .

What is Gaussian Noise?. . .

Gaussian distribution is used to predict probabilities and make inferences about data.

Note: An inference is a conclusion or educated guess that is based on evidence and reasoning



Modeling Sensors. . .

5. Calibrate and Linearize (if Needed)

For sensors with nonlinear responses, calibration may be necessary to map the raw sensor output to a physical quantity. Linearization techniques, like Taylor expansion or lookup tables, can approximate the response in a useful range.

Taylor Series Expansion Definition

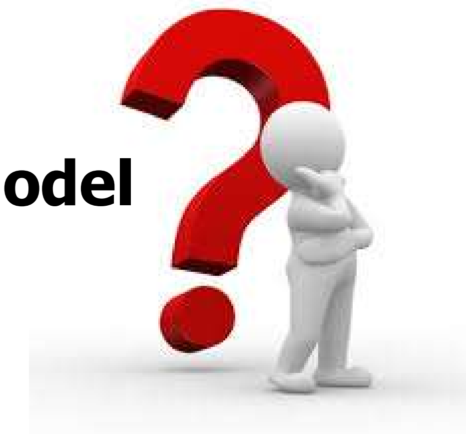
In general, for a function that is infinitely differentiable at a point a, the Taylor series expansion of $f(x)$ around a is:

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

where:

- $f^{(n)}(a)$ is the n-th derivative of f evaluated at a,
- $n!$ (n factorial) is the product of all positive integers up to n.

Question 2



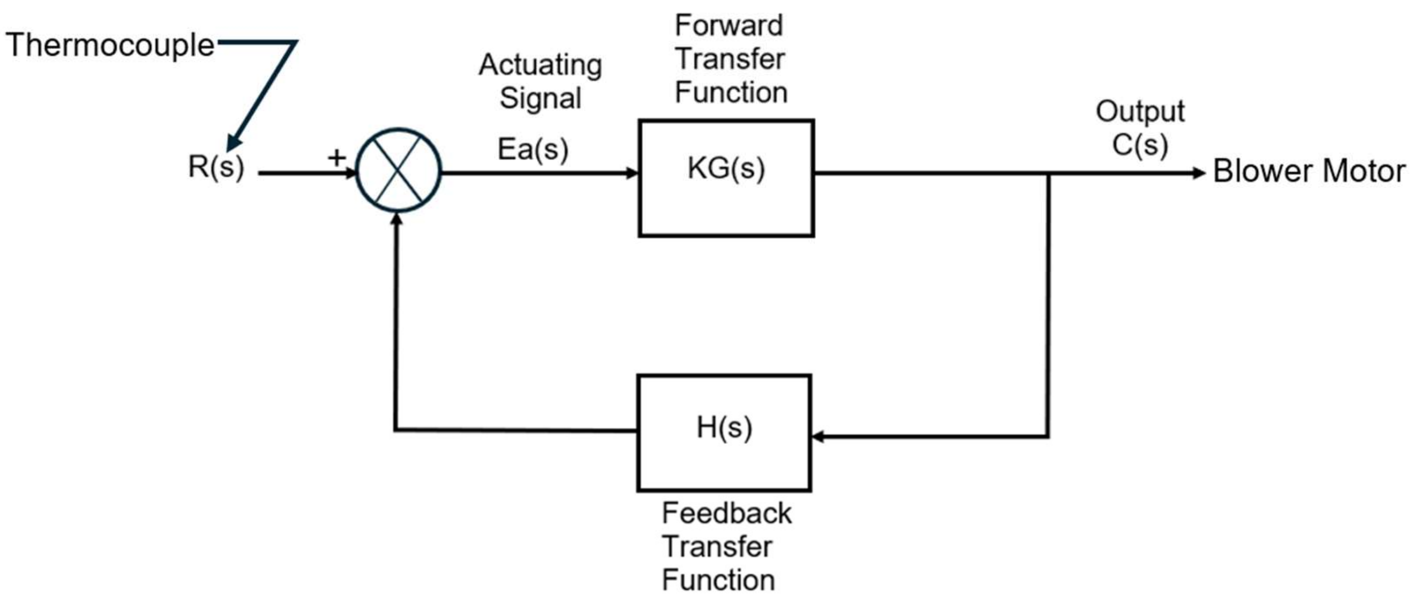
What is the first step in creating a mathematical model for sensors?

- **a) Determine the Sensor's Transfer Function**
- **b) Define the Sensor's Purpose and Input-Output Relationship**
- **c) Model Dynamic Behavior (for Time-Dependent Responses)**
- **d) Account for Non-Idealities (Noise, Offset, Drift)**

Modeling Sensors. . .

Example Application:
An Industrial Furnace Temperature Controller

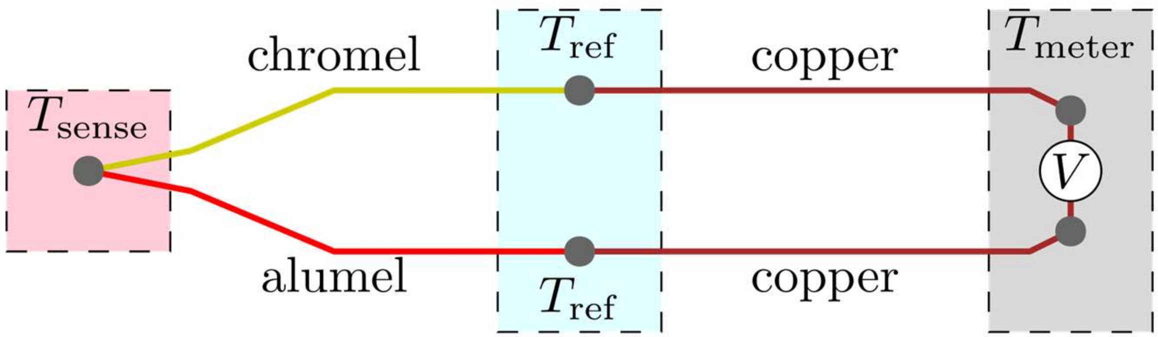
System Model



Modeling Sensors. . .

Example: Temperature Sensor Modeling. . .

K-Type Thermocouple Construction:



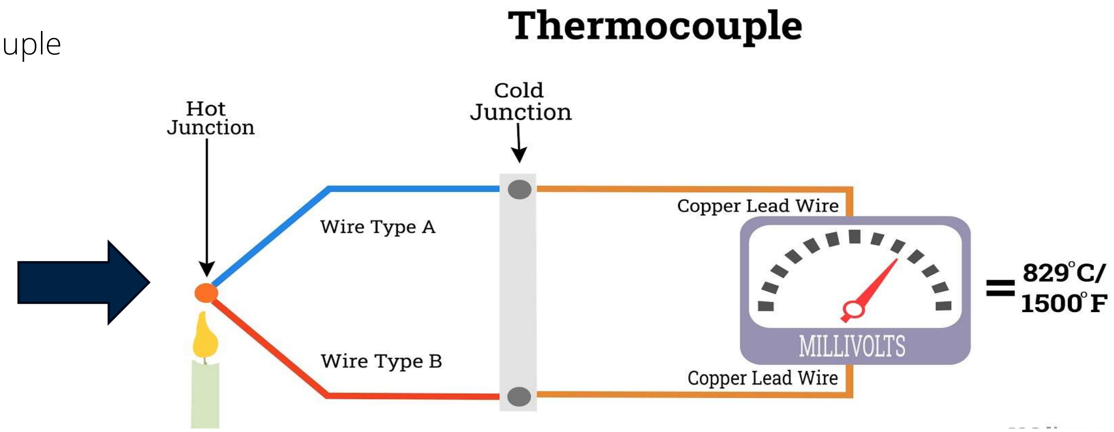
The measured voltage turns out to be

$$V = \int_{T_{\text{ref}}}^{T_{\text{sense}}} \left(S_+(T) - S_-(T)\right)\, dT,$$

where S_+ and S_- are the Seebeck coefficients of the conductors attached to the positive and negative terminals of the voltmeter, respectively (chromel and alumel in the figure).

Modeling Sensors. . .

Example: Temperature Sensor Modeling



Modeling Sensors. . .

Example: Temperature Sensor Modeling. . .

Consider a thermocouple, which measures temperature by producing a voltage V proportional to the temperature difference between two junctions:

$$V = S(T_{\text{hot}} - T_{\text{ref}})$$

where S is the Seebeck coefficient (sensitivity), T_{hot} is the temperature at the measurement junction, and T_{ref} is the reference temperature.

If the thermocouple has a dynamic response, it may be modeled with a first-order system equation:

$$\tau \frac{dV}{dt} + V = S(T_{\text{hot}} - T_{\text{ref}})$$

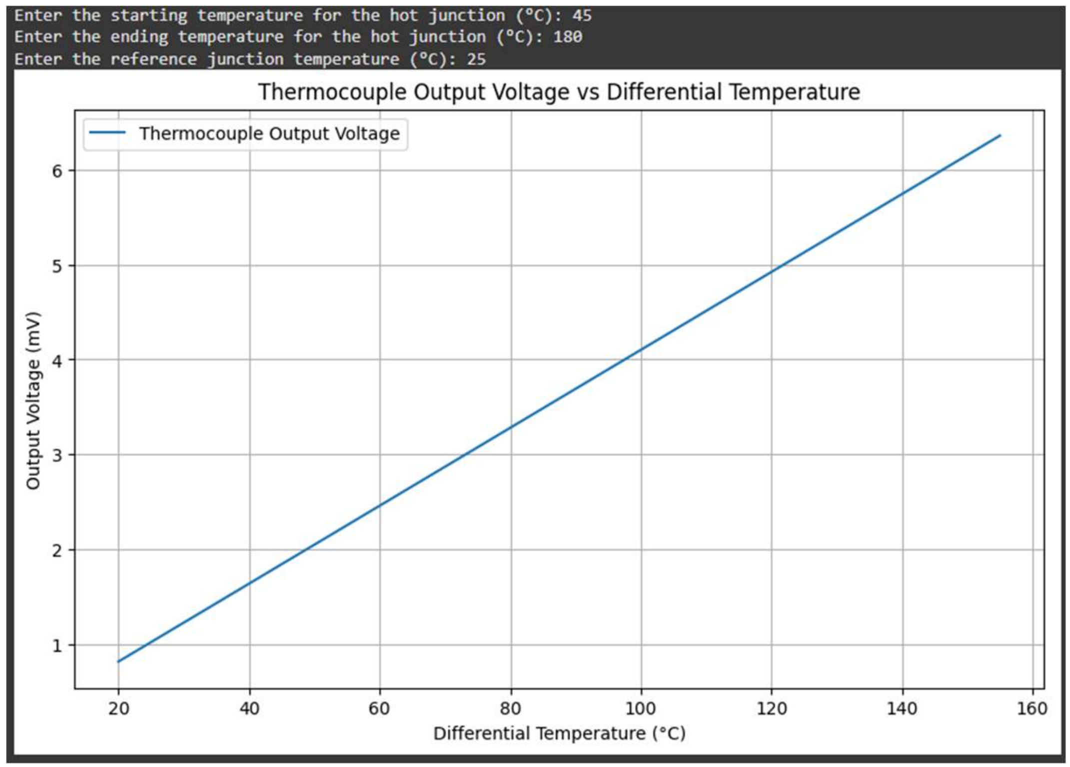
Note:
By establishing these mathematical models, sensor behavior can be simulated and systems optimized. Systems relying on sensor inputs can perform predictive maintenance or calibration tasks for enhanced accuracy.

Modeling Sensors. . .

Example: Temperature Sensor Modeling. . .

An Interactive Thermocouple Sensor Model creating in Python.

Note:
This chart can be used to calibrate homebrew/experimental Thermocouple Sensors.

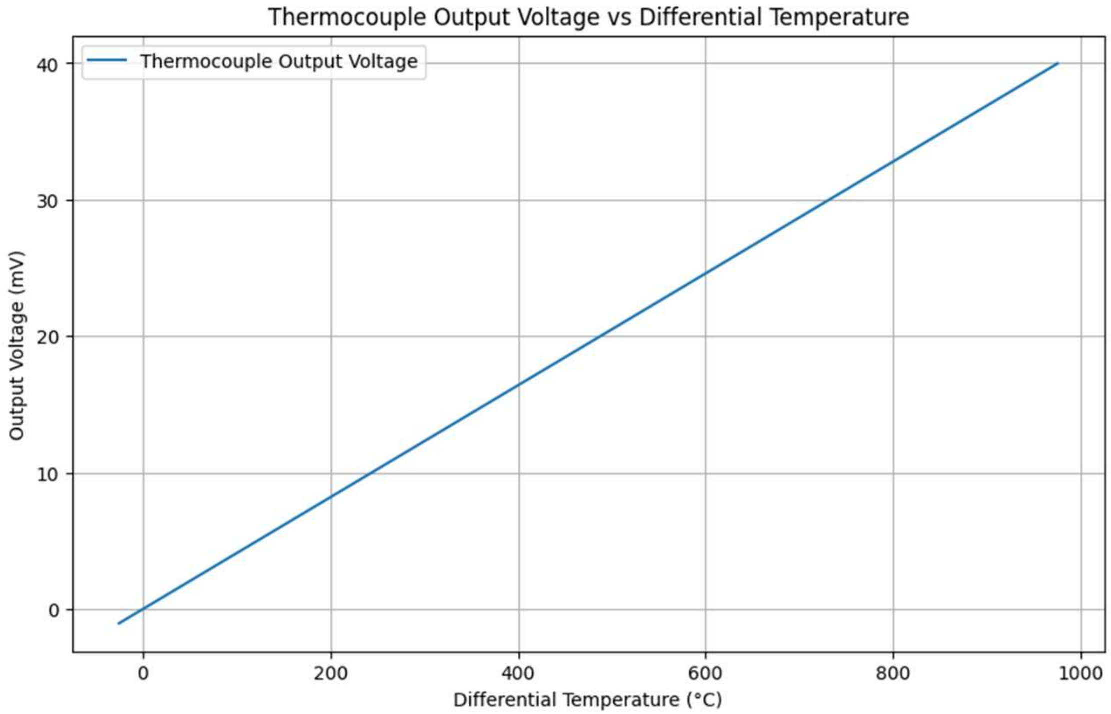


Modeling Sensors. . .

Example: Temperature Sensor Modeling. . .

A Thermocouple Sensor Model created in MATLAB.

Note:
This chart can be used to calibrate homebrew/experimental Thermocouple Sensors.

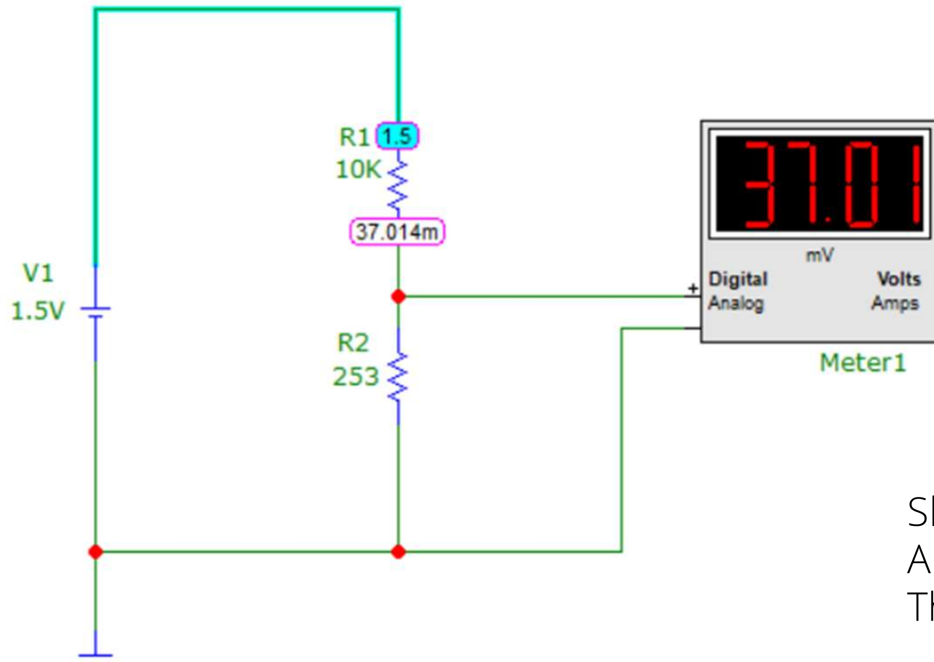


Modeling Sensors. . .

Example: Temperature Sensor Modeling. . .

Homebrew/Experimental Sensor Thermocouple Sensor circuit modeled in Microcap.

A 280Ω potentiometer will be wired as a rheostat.



Slider Control Simulates Applied Temperature to Thermocouple.

CEC Continuing Education Center

DigiKey

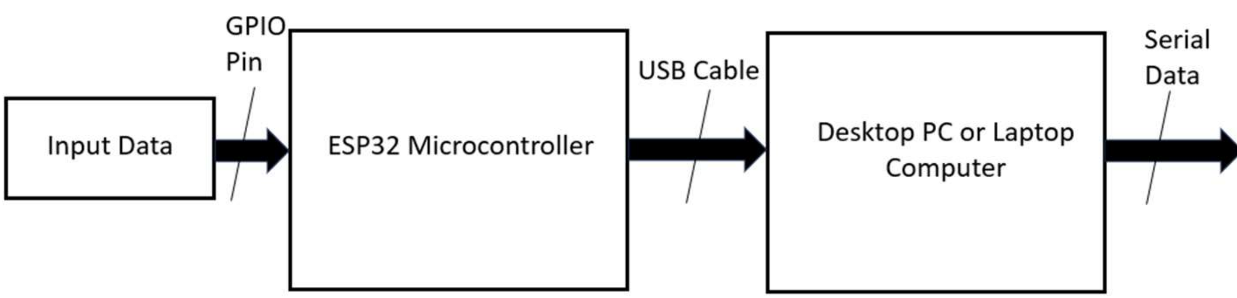
Question 3

What are the 2 dissimilar metals used in a K-Type Thermocouple?

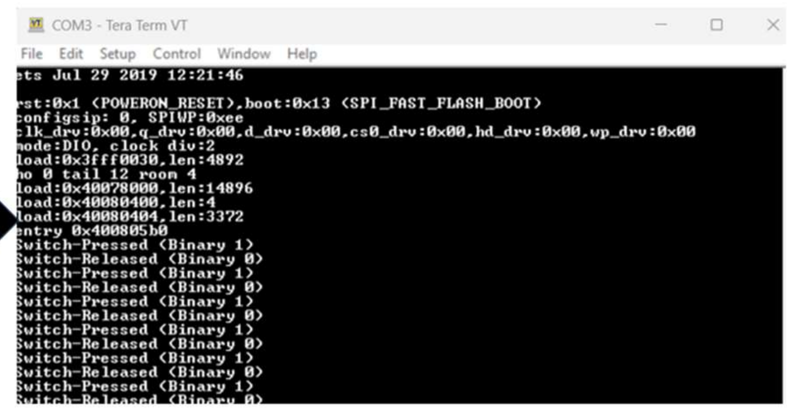
a) chromel, bimetal
b) alumel, bimetal
c) chromel, alumel
d) none of the above

A Simple Data Logging Technique

Concept Diagram



Serial Software Monitor: Logging Data



Web link

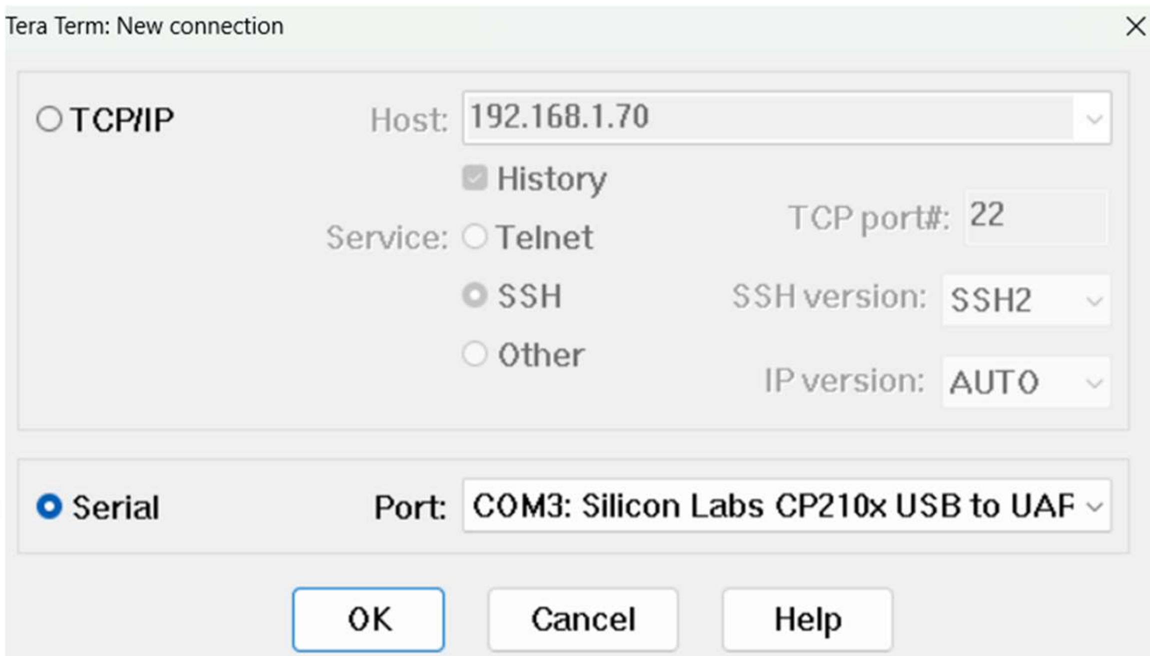
https://teratermproject.github.io/index-en.html

Tera Term Serial Monitor Software

A Simple Data Logging Technique. . .

Setup:
Step 1- Establishing a serial connection

Click Here!



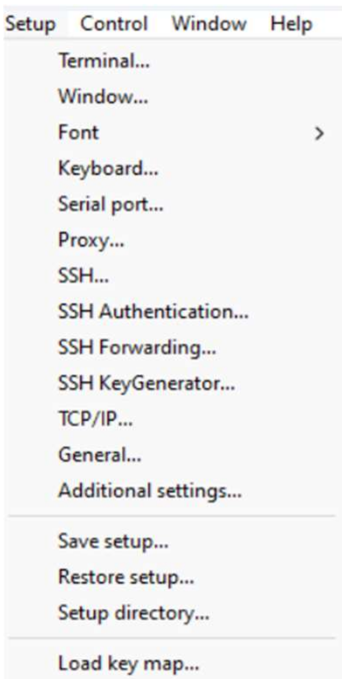
CEC Continuing Education Center

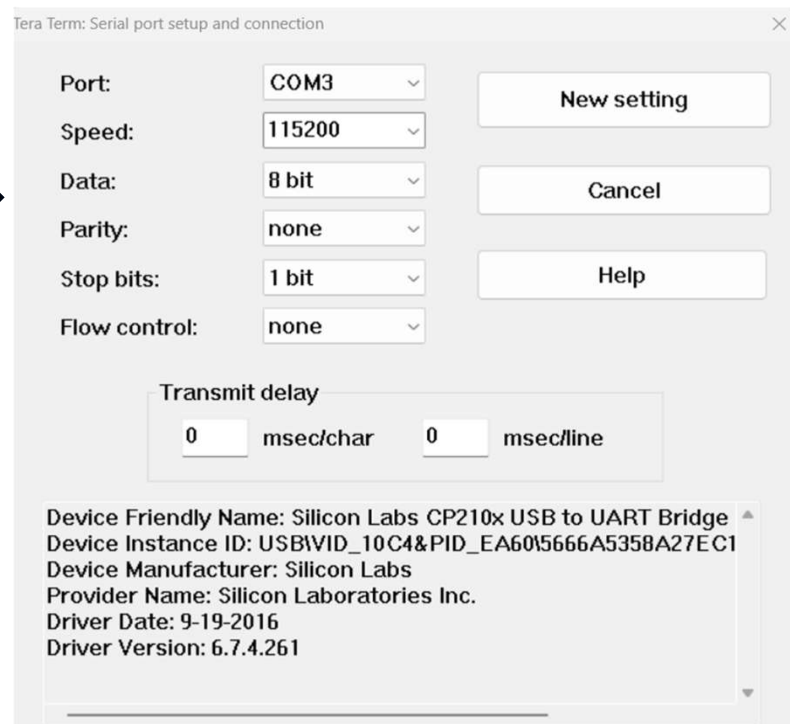
DigiKey

A Simple Data Logging Technique. . .

Setup:
Step 2- Setup Serial Port.

Click Here!





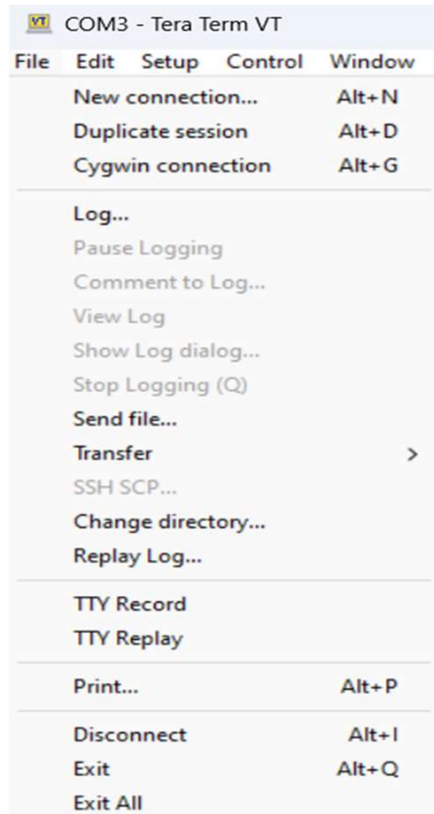
Click Here after Speed has been changed!

Note:
After setting up the serial port, reset the ESP32. Press/release the EN button on the board to reset the ESP32.

A Simple Data Logging Technique. . .

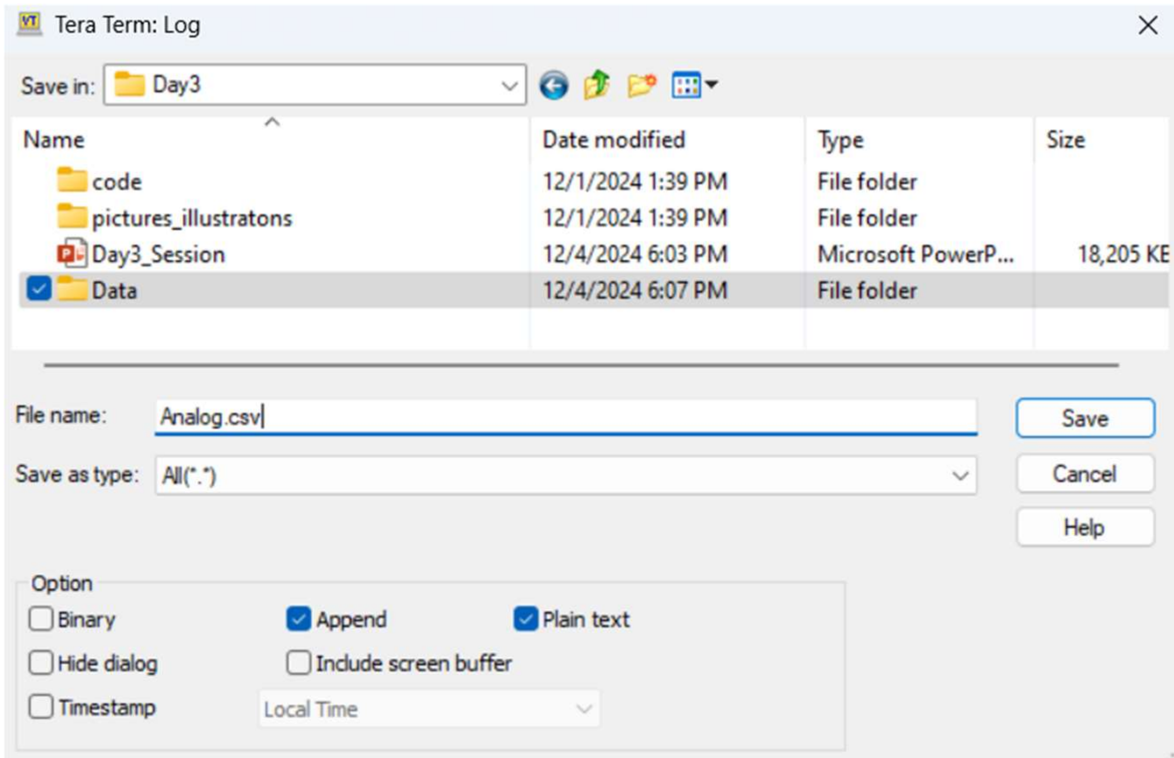
Setup:
Step 3- Create a file to log the data.

Click Here!



A Simple Data Logging Technique. . .

Setup:
Step 4- Save a file to log the data.

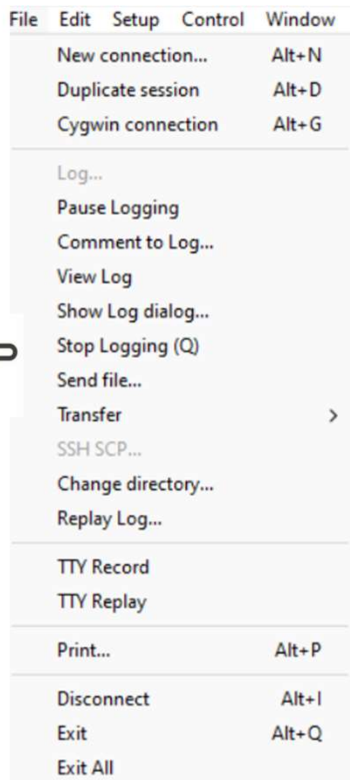


Type the name of the data file Here!

Click Here to save data file!

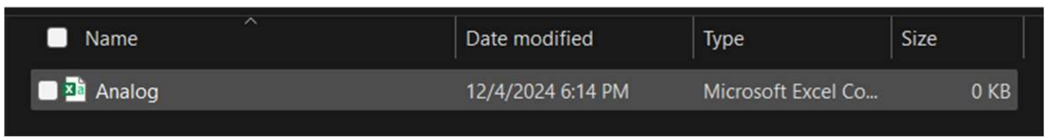
A Simple Data Logging Technique. . .

Setup:
Step 4- Stop the data logging event.



Click Here!

Logged Data



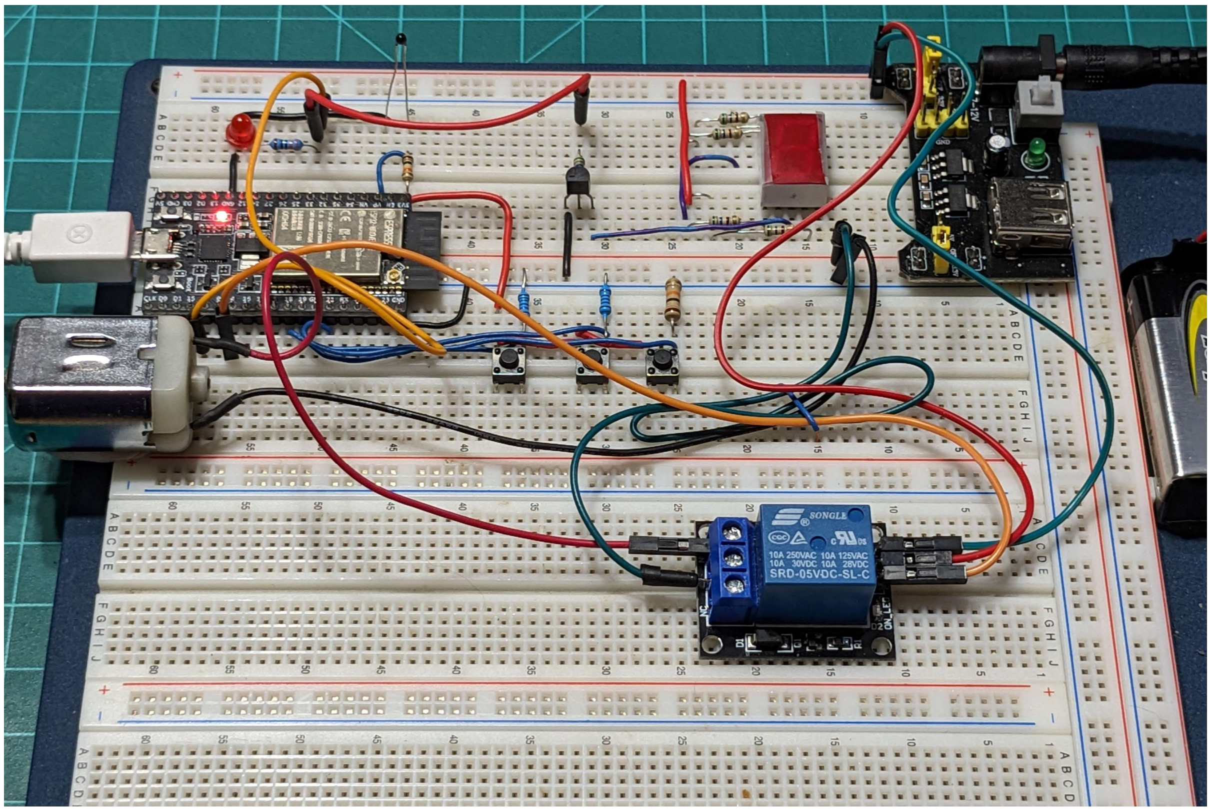
Continuing Education Center

Question 4

What is the serial monitor software's name for displaying and logging ESP32 microcontroller data?

- **a) Hyper-Terminal**
- **b) Putty**
- **c) Tera-Term**
- **d) none of the above**

Lab: Build An ESP32 Electronic Thermometer



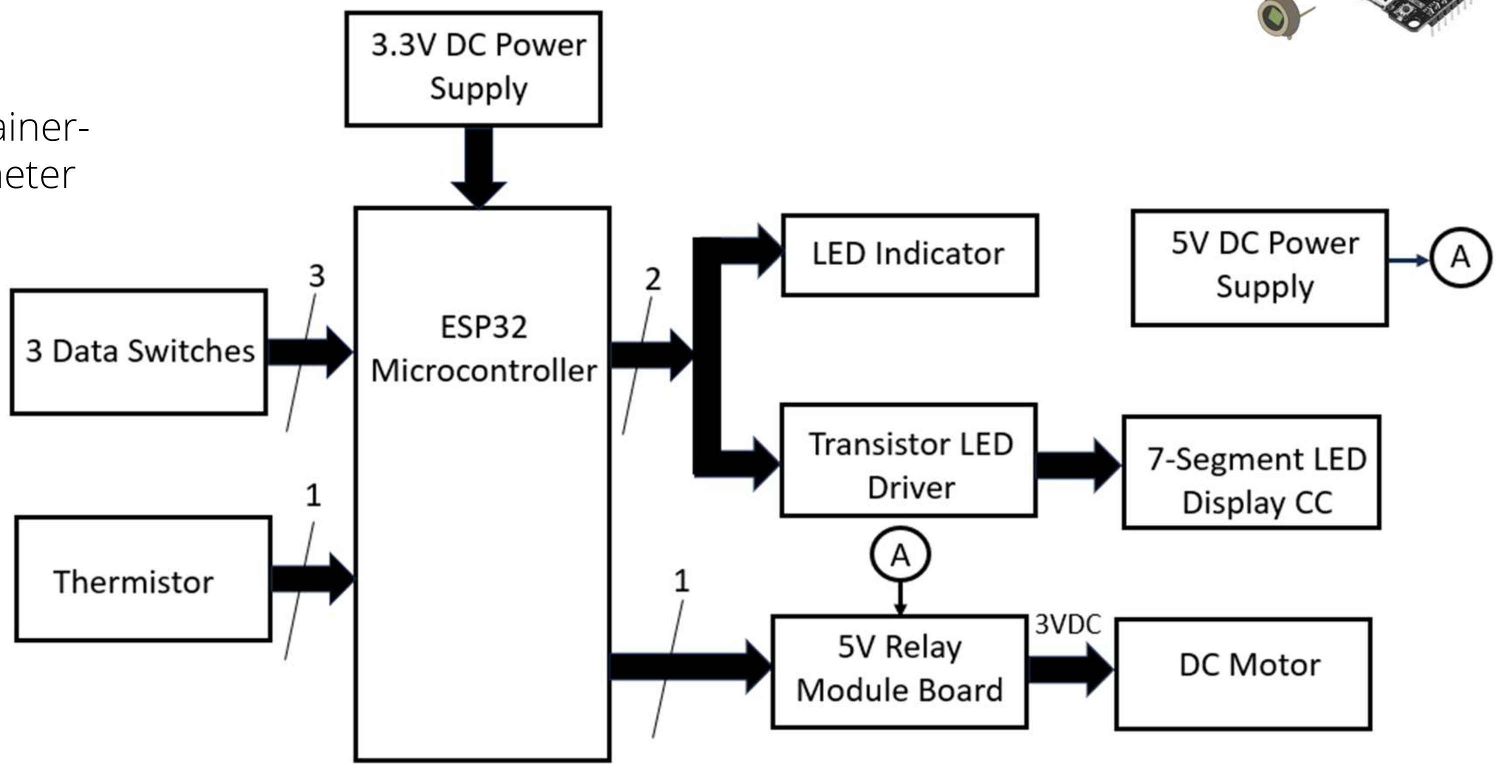
Lab: Build An ESP32 Electronic Thermometer…

Participant Learning Objectives:

- Participants will learn to wire a thermistor circuit using an ESP32 Micro Trainer, off-the-shelf electronic components, and a solderless breadboard.
- Participants will learn to install and set up the Mu programming platform.
- Participants will learn to set up a software terminal monitor for displaying and logging temperature data.
- Participants will learn to program and test their ESP32 Micro Trainer using the MicroPython language.
- Participants will learn to display and log temperature data using the ESP32 Micro Trainer’s thermistor circuit on a software terminal monitor.

Lab: Build An ESP32 Electronic Thermometer…

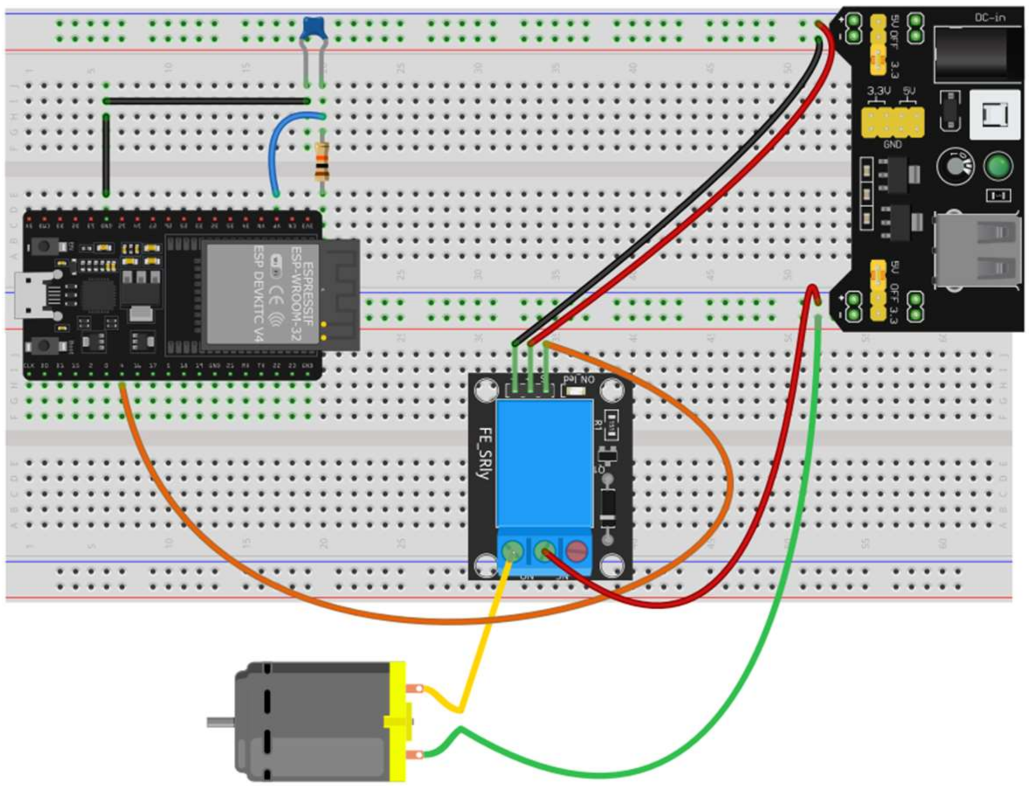
The ESP32 Micro Trainer-Electronic Thermometer Block Diagram



Note:
The Breadboard 3V DC Power Supply rails will power the DC Motor.

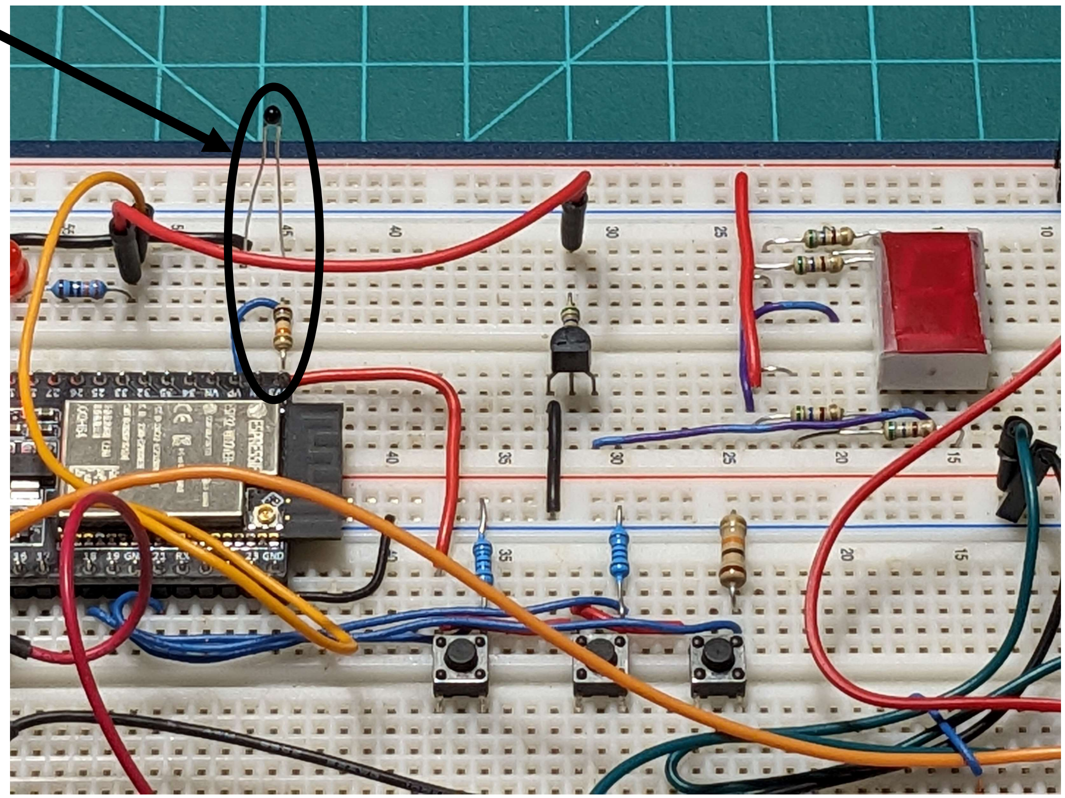
Lab: Build An ESP32 Electronic Thermometer…

Electrical Wiring Diagram: Location of the Thermistor and the 10KΩ resistor on the solderless breadboard.



Lab: Build An ESP32 Electronic Thermometer. . .

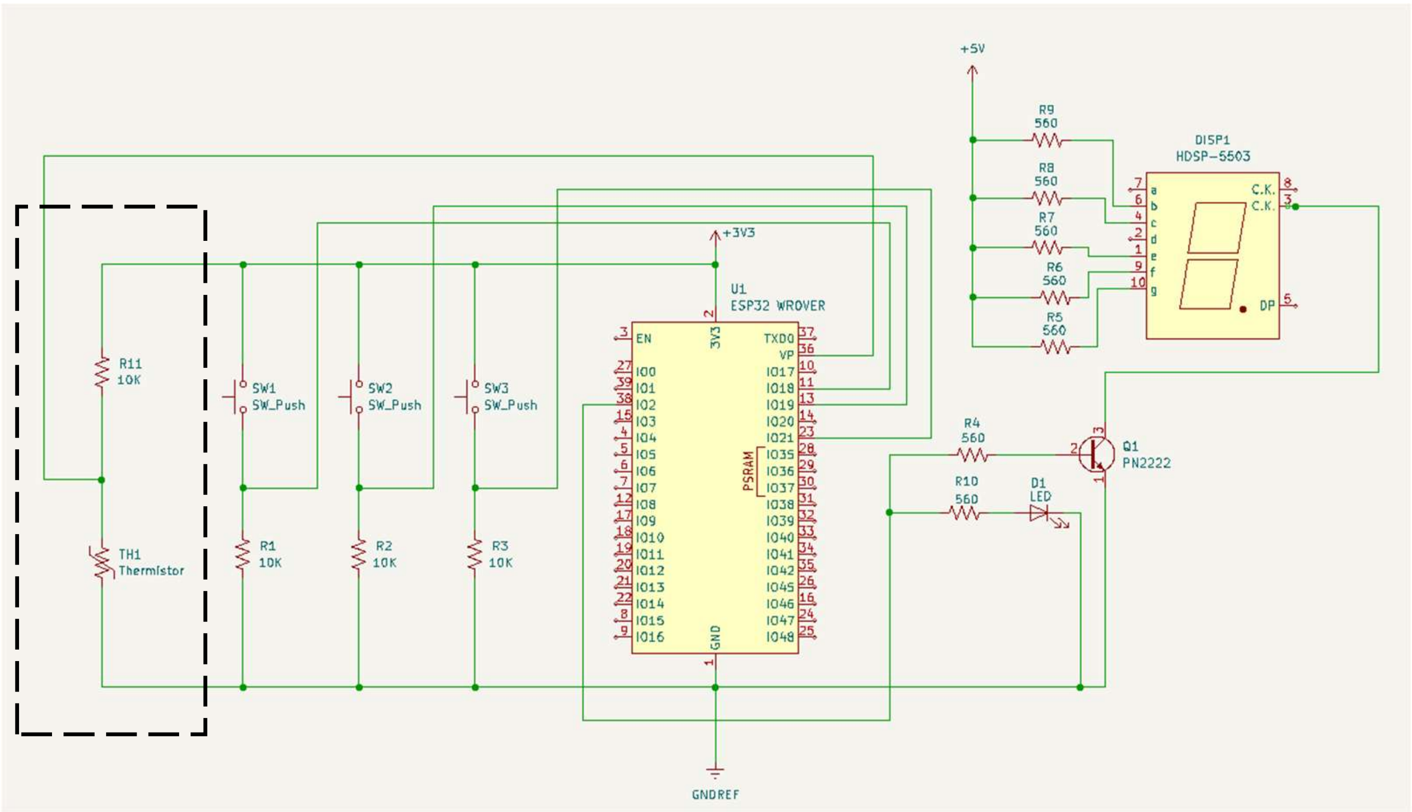
Thermistor
Circuit Added
Here!



Thermistor Circuit-
Temperature
Measuring Mode

Lab: Build An ESP32 Electronic Thermometer. . .

Thermistor Circuit wired to GPIO36 pin of the ESP32



Electronic Thermometer Circuit Schematic Diagram

Lab: Build An ESP32 Electronic Thermometer. . .

Electronic Thermometer MicroPython Code

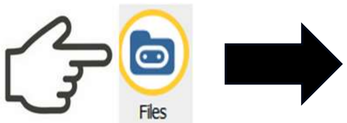
```
# Electronic Thermometer - csv format
from machine import Pin, ADC
import time
import math

adc = ADC(Pin(36))
adc.atten(ADC.ATTN_11DB)
adc.width(ADC.WIDTH_12BIT)

try:
    # Print CSV headers
    print("ADC value,Voltage (V),Temperature (°C)")
    while True:
        adcValue = adc.read()
        voltage = adcValue / 4095 * 3.3
        Rt = 10 * voltage / (3.3 - voltage)
        tempK = 1 / (1 / (273.15 + 25) + (math.log(Rt / 10)) / 3950)
        tempC = tempK - 273.15
        # Print values in CSV format
        print(f"{adcValue},{voltage:.2f},{tempC:.2f}")
        time.sleep_ms(1000)
except:
    pass
```

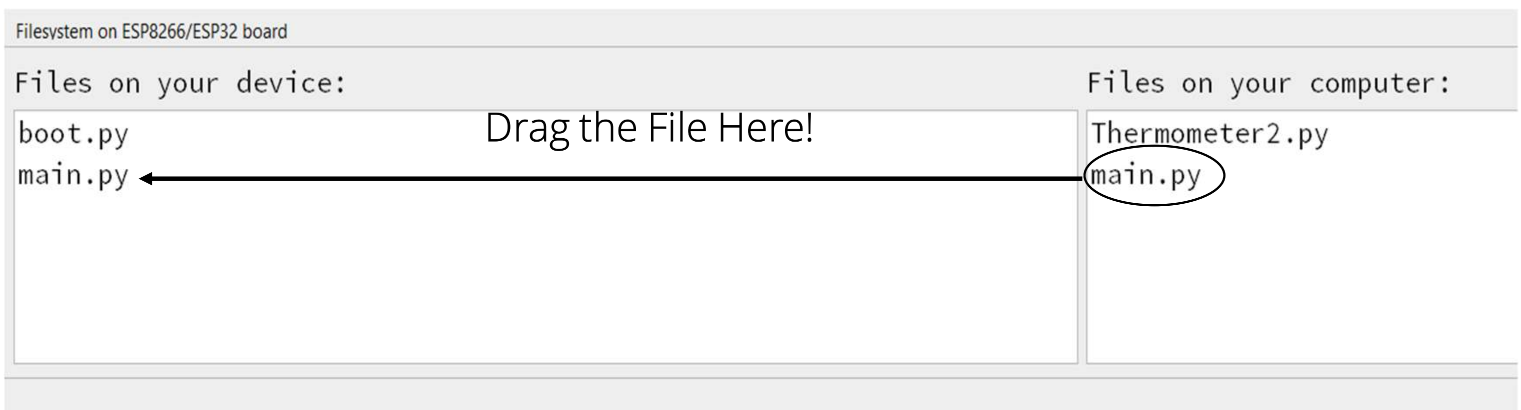
Lab: Build An ESP32 Electronic Thermometer. . .

Burning the Electronic Thermometer MicroPython code onto the ESP32 microcontroller.

Note:
Initially, main.py will not be present in the "Files on your device" location. After performing this task, the main.py will be overwritten with a new MicroPython code.

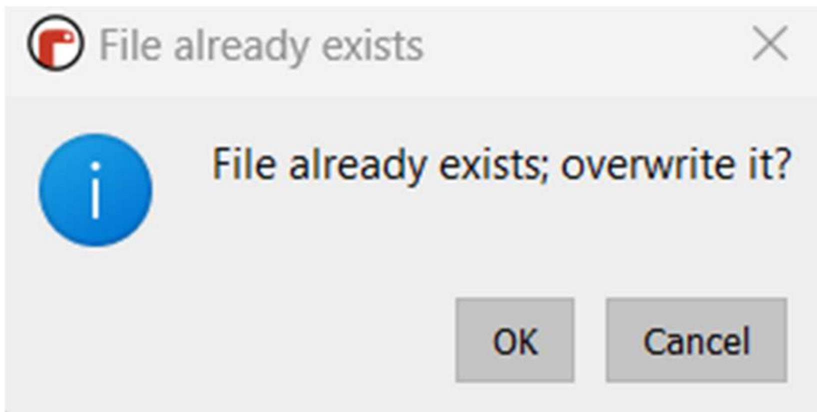
Click Here!





Lab: Build An ESP32 Electronic Thermometer. . .

Burning the Electronic Thermometer MicroPython code onto the ESP32 microcontroller.

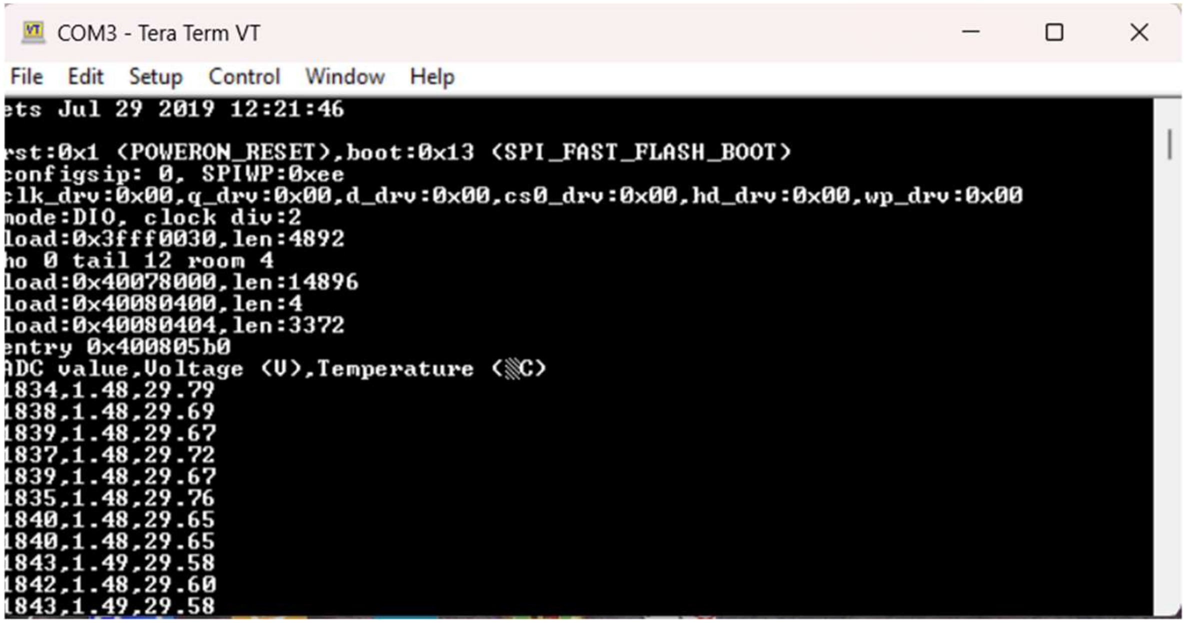


Click OK!

Lab: Build An ESP32 Electronic Thermometer. . .

Follow the Setup steps in the: A Simple Data Logging Technique section of this slide deck. Tera Term Serial Monitor Software is used to display and log the temperature data.

Scrolling Thermistor Temperature Data!



Lab: Build An ESP32 Electronic Thermometer. . .

Logged Thermistor Temperature data:csv file

	A	B	C
1	ADC_value	Voltage (V)	Temperature (˚C)
2	1795	1.45	30.69
3	1798	1.45	30.62
4	1799	1.45	30.59
5	1799	1.45	30.59
6	1798	1.45	30.62
7	1801	1.45	30.55
8	1802	1.45	30.52
9	1801	1.45	30.55
10	1803	1.45	30.5
11	1798	1.45	30.62
12	1802	1.45	30.52
13	1799	1.45	30.59
14	1801	1.45	30.55
15	1798	1.45	30.62
16	1798	1.45	30.62
17	1797	1.45	30.64
18	1797	1.45	30.64
19	1776	1.43	31.13
20	1631	1.31	34.58
21	1519	1.22	37.38
22	1472	1.19	38.59
23	1444	1.16	39.33

Note:
The column header information was added to the csv file.

Question 5

What ESP32 GPIO pin is used to read the thermistor analog voltage data?

- **a) 2**
- **b) 18**
- **c) 36**
- **d) 4**

Thank you for attending

Please consider the resources below:

Sehr, M.A, Lohstroh, M., Weber, M., Ugaide, I., Witte, M., Neidig, J., Hoeme, S., Niknami, M., & Lee, E.A. (2021). Programmable logic controllers in the context of industry 4.0. *IEEE Transactions On Industrial Informatics* 17(5), 3523 – 3535. https://ieeexplore.ieee.org/document/9134804

Wilcher, D. (2024). *Understanding industrial controls with an esp32*. GitHub. https://github.com/DWilcher/DesignNews-WebinarCode/blob/main/December_24_Webinar_Code.zip

Thank You

Sponsored by