# Project Overview

DIXELL -> JETSON NANO -> AWS CLOUD

13-12-2023

Contents

# Project Walkthrough

## Overview:

The aim of this project was to interface Dixell RS485 connection to Jetson Nano and using edge computing on jetson, we log and push the data to AWS cloud server. Jetson Nano was programmed to acquire data from XR70CX using the RS485 connection. Python script along with minimalmodbus library worked to get data from the Dixell Temperature Sensor and the AWS-SDK was implemented for pushing data to AWS IoT Core.

Following is the view of the final result developed at end of this project:
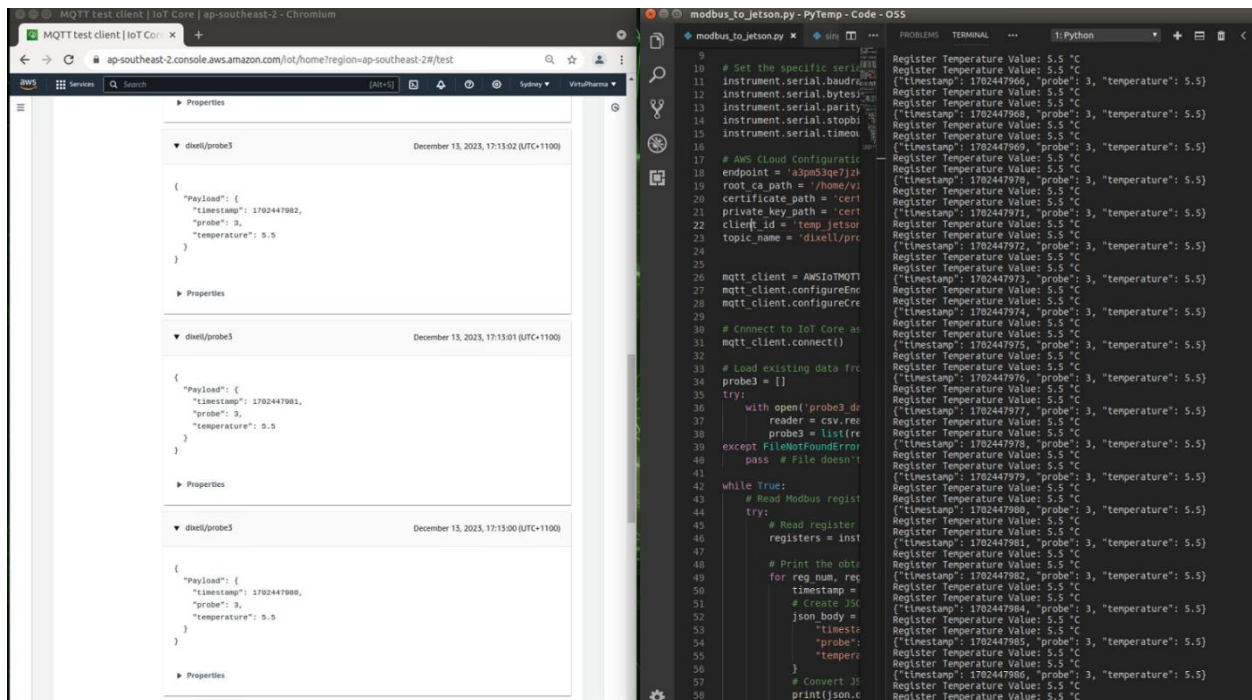


*Figure 1 (AWS) -> (Python) -> (Debug Console)*

## Jetson Nano

The Jetson Nano endpoint consists of:

### RS485 to Jetson

The data from sensor was connected to the Jetson Nano using RS485 to USB serial converter and the device shows up as USB serial device occupying a **/dev/ttyUSB0** port on the Jetson Nano. The fride itself utilizes MODBUS protocol to this provided a method to communicate MODBUS commands to sensor and read the current data.

The protocol has following parameters:

Baud Rate = 9600

Data Bits = 8

Parity Bit = None

Stop Bit = 1

USB Port = As assigned by the Nano -> (/dev/ttyUSB0)

These parameters were added to the MODBUS node settings in python script to ensure proper communication between both.

## Jetson to Local CSV Endpoint

The incoming data is logged into a csv file for further computational requirements that may arise further down the timeline. Incoming data is read from the registers, scaled accordingly and then stored in the csv file along with the timestamp in unix format. For each session, if the file already has data, the code reads this file and then appends the new data at the end of last session acquired data.
The csv file can be read by opening it in LibreOffice or using the python script I've made for this purpose.
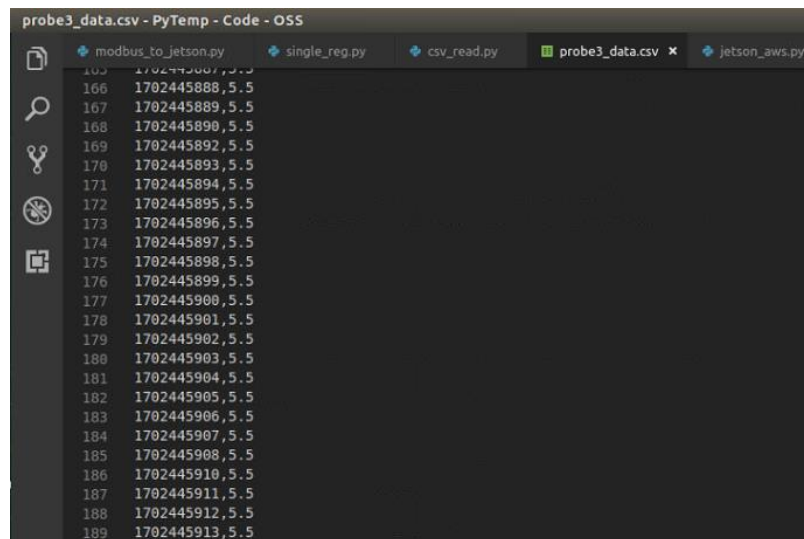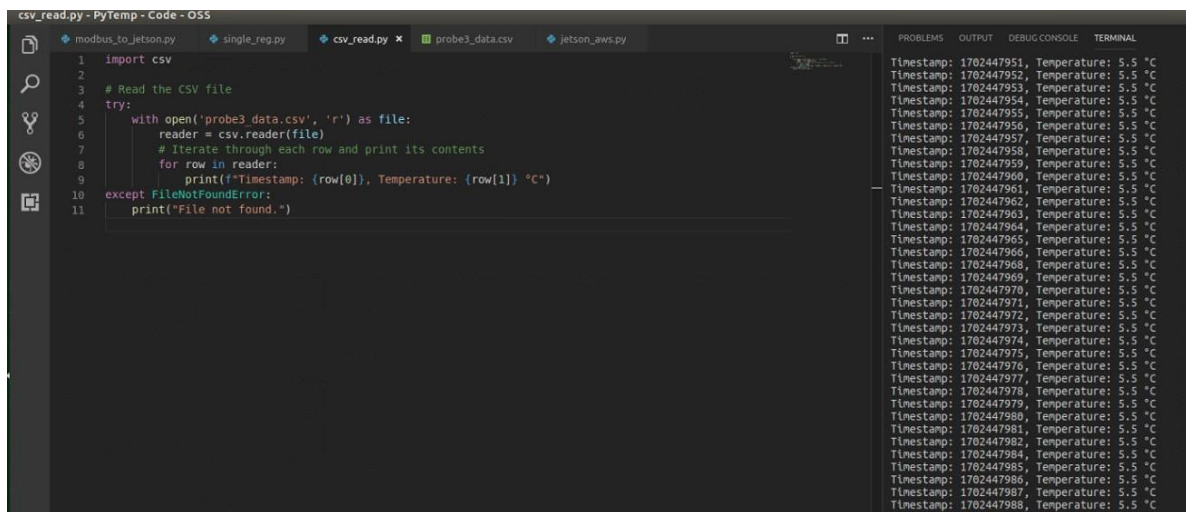


*Figure 2 CSV file direct Reading*



*Figure 3 CSV reading using python*

## Jetson to AWS Iot Core Endpoint

Sending data from Jetson Nano to AWS cloud involved using the MQTT protocol by first filtering the incoming data upon MODBUS read, extracting the temperature values and sending it over to AWS using MQTT.

The MQTT protocol was setup in AWS by creating and IoT core "Thing" and using it to manage MQTT data.

Inside the python script the MQTT was implemented using the AWS SDK, and necessary certificates were attached to configure secure TLS communication between AWS and Jetson Nano. After the setup, data was pushed to the relevant AWS IoT Core topic and can now be further utilized as setup.

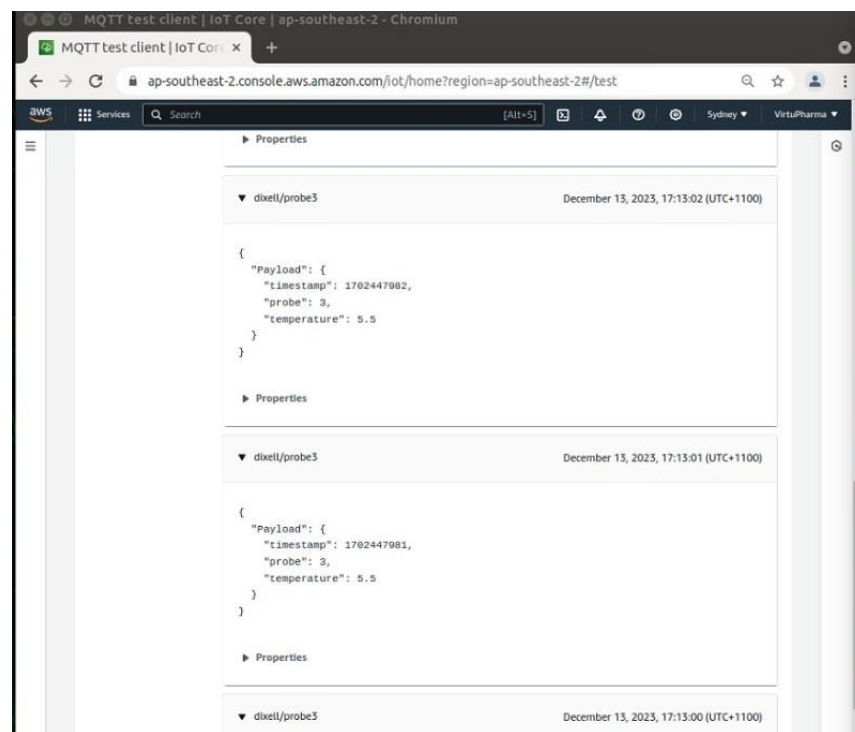And the resultant data received at the AWS showed Probe 3 Temperature values (as only those were connected).



*Figure 4 AWS IoT Core Endpoint*

## Commands

Once the Jetson Nano is powered up the PyTemp folder on the desktop contains all the necessary files and certificates. It also contains the csv file that logs the data.

For easier access I've constructed a bash script that starts the whole process with ease. It's on desktop and can be executed by opening terminal on desktop and executing following commands.

**cd Desktop**

**./begin.sh**