Cloud-based Control of a Production Plant

Satyam Uttamkumar Dudhagara
Abdalla Mohamed
Nouhaila Houssaini
Denys Skrytskyi

Projektlabor AT+ 2022

Cloud-based Control of a Production Plant

Projektlabor AT+

Satyam Uttamkumar Dudhagara (420424)

Abdalla Mohamed (420445)

Nouhaila Houssaini (420354)

Denys Skrytskyi (420523)

Prof. Dr. Ping Zhang
M.Sc. Andreas Köhler

Lehrstuhl für Automatisierungstechnik
Fachbereich Elektrotechnik und Informationstechnik

March 19, 2024

Contents

List of Figures

1 Introduction

Nowadays, many industrial systems use a cloud-based controller that executes on a cloud platform using control algorithms [12]. Indeed, to control operations, a cloud-based controller analyzes information, produces control instructions, and transfers them to the processes required.

Our project aims to develop a logic controller for a model factory to achieve the specifications and implement the logic controller in the cloud. For that, we will need to investigate several cloud performances and choose the appropriate one, design the logic controller, implement it in the cloud and finally test and evaluate the performance. In our case, we will use Node-RED as a programming tool that uses graphical flows and nodes in a browser-based flow editor. We also will use Amazon Web Service (AWS) as a computing service providing us with cloud services as it is cost-saving, adaptable, and secure compared to other platforms such as SoftPLC, which executes a program in the same manner as a hardware PLC. The advantage of executing a PLC program this way is, that the PLC status can be displayed in real-time. More details will be mentioned further in the report [11].

During the lab experiment, the input signals were evaluated in the Programmable Logic Controller (PLC) and the output signals were written to the actuators. We developed a Signal Interpreted Petri Net (SIPN) and implemented that in a ladder logic graphical language in the SIMATIC S7 software where we programmed the PLC [1].

However, for our project, we will be focusing on programming cloud control instead of the PLC. Later, this was considered as an interface for sending and receiving input/output signals and the Petri Net was implemented directly in the cloud without any need to convert into any different language. AWS will get the input signals and send the output back to the PLC.

2 The Plant and the Variables

2.1 Plant Overview

The plant used in this project is a production line that can be summarized into 5 parts as can be seen in Fig. [1]. The first part is named Feeder. The Feeder will hold the workpieces represented in Fig. [2] and this is the first step in the production line. The capacity of the Feeder is 5 workpieces at a time. When the system is started, the Feeder will transfer the workpiece into the conveyor belt, which will move the workpiece to the next station or task.

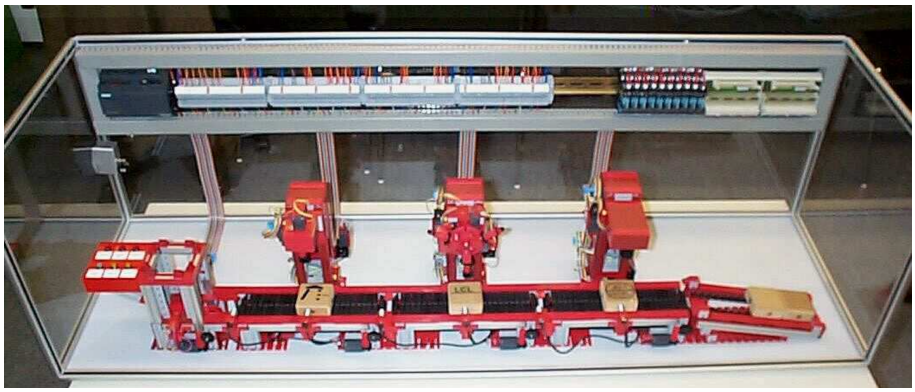

Figure 1: The whole production line

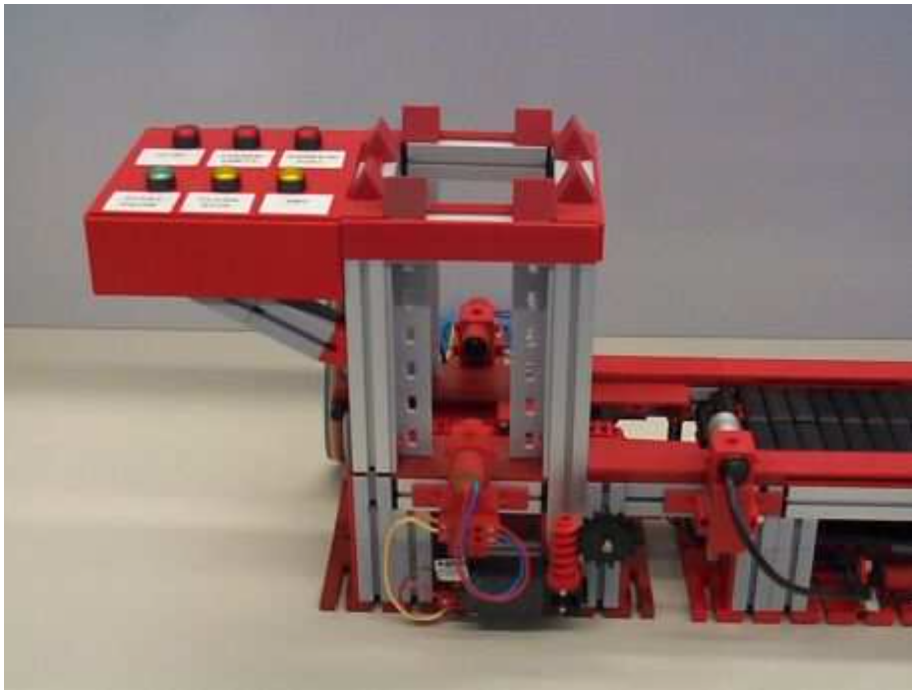

Figure 2: The Feeder

The next step is the Drilling machine as shown in Fig. [3]. The conveyor belt will move the workpiece until it's exactly under the Drilling machine, this is

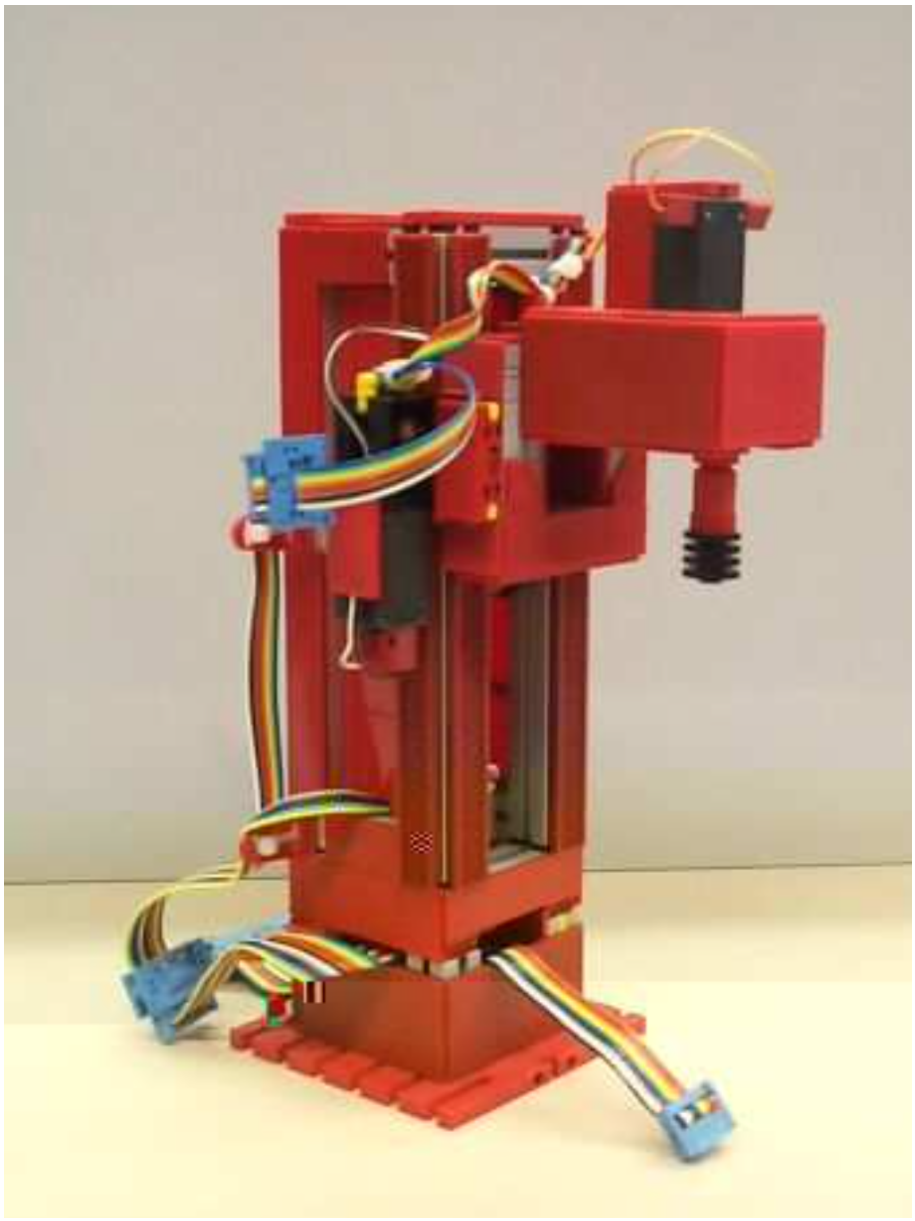

Figure 3: The Drilling machine

done with the help of some inductive proximity sensors. Here, the Drilling tool will move downwards until touching the workpiece, then it will return to its initial position. But there is no way for the motor to know when to stop. This problem can be solved by two mechanical limit switches attached to both ends

of the motor path. After this step, the conveyor belt (Fig. [4]) will move the workpiece to the next step, which is the Vertical Milling machine.

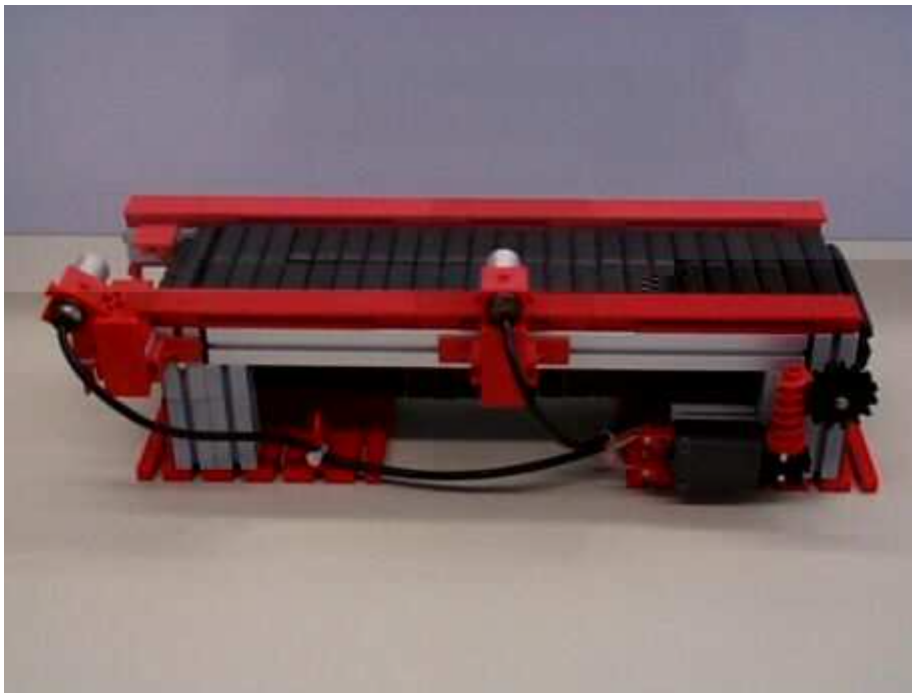

Figure 4: The Conveyor belt

The Vertical Milling machine works similar to the Drilling machine but with some additional steps. The vertical mill contains three different tools as shown in Fig. [5]. And each of them has to do some work on the workpiece. So, in the beginning, the first tool will move downwards until touching the workpiece, and it will move back to its initial position. A motor will run in the Vertical Milling machine that will rotate and change the milling machine's tool. Then the steps are repeated with the second tool. After this tool has its work done, the tool will be changed again to the third and last tool. The job of the vertical machine will be over when the initial tool is returned to its initial position (pointing downwards). Then the conveyor belt will move the workpiece to the next step, which is the Horizontal Milling machine.

The Horizontal Milling machine again differs from the Drilling machine slightly. Instead of having its tool moving just downwards and upwards, the tool has to

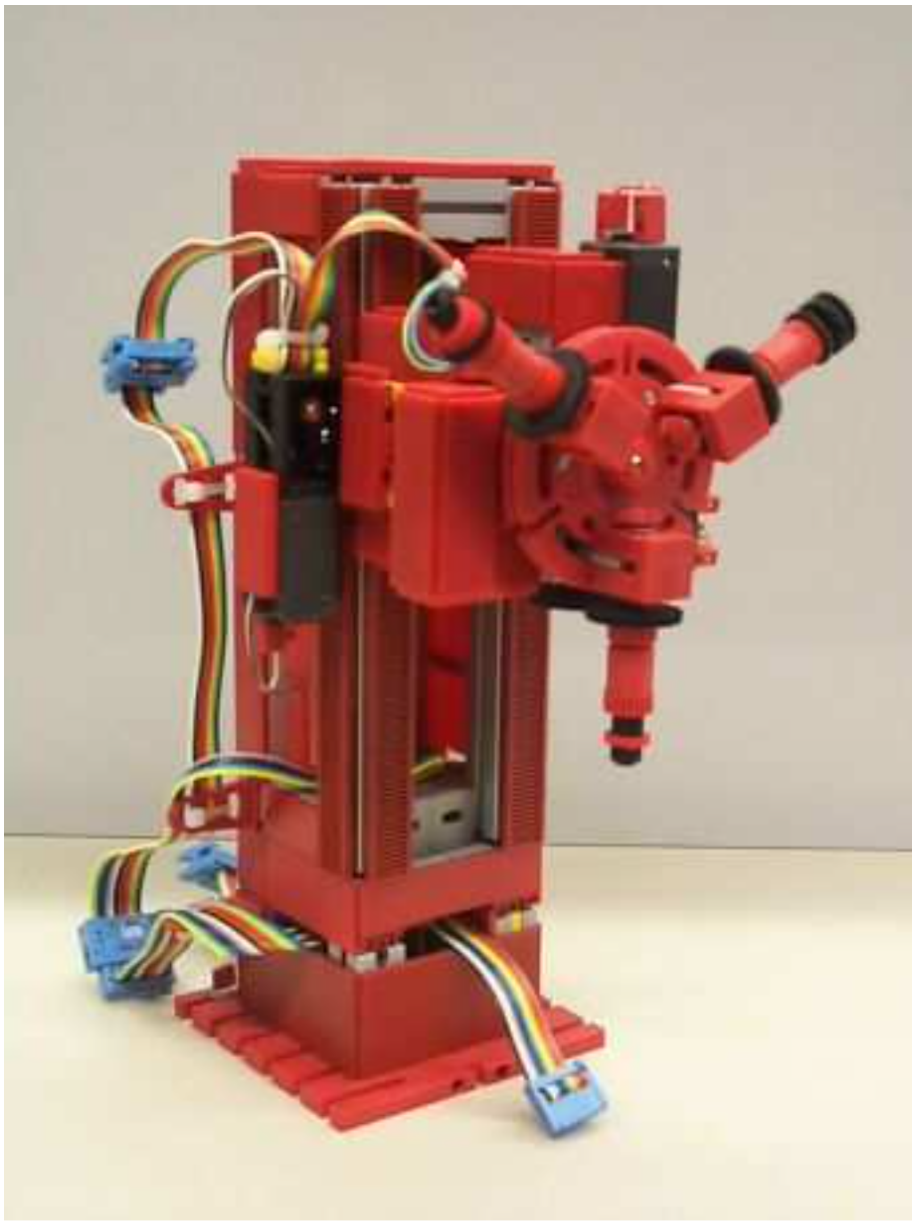

Figure 5: The Vertical Milling machine

also move in a horizontal direction after it moves downwards. Fig. [6] describes this well. So, it can form a line onto the workpiece. After these steps, the tool will move back to its initial position and that is when this step is over. The workpiece will then move to its final station with the help of a conveyor belt

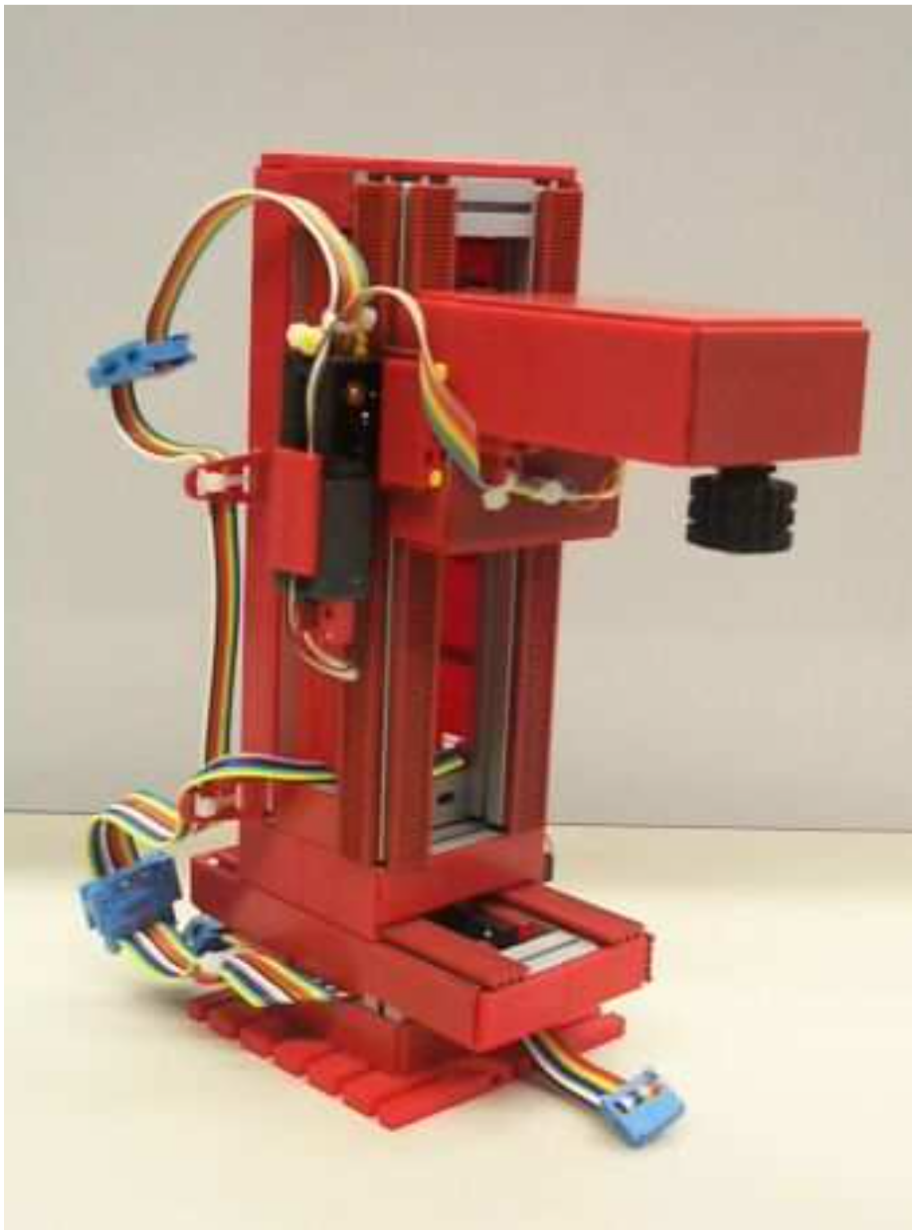

Figure 6: The Horizontal Milling machine

again. The last place for the workpiece is the storage area. If the storage area has space available, the workpiece will be moved to the storage area, otherwise, it will remain on the conveyor belt of the Horizontal mill until a space is freed.

The storage area (Fig. [7]) has a capacity of holding four workpieces. The storage area has a light barrier sensor on its entrance that sends “1” if there is an object over it. So, a signal of “1” here means the storage is full.

Figure 7: The Storage

2.2 The Plant Variables

The programming of this plant is based on the sensor readings and actions are performed by motors to achieve certain tasks. For that, there has to be a way to assign an address to every component in the plant. This address will help in knowing which sensors are sending which signals. Signals can be sent to specific motors, based on their address, to perform the required actions. The address assigned to the components is based on their connection to the PLC. The inputs to the PLC are the sensors. Using sensors, information from the process while it is running can be obtained. This input data will be used by the CPU inside the PLC and based on the program written, some signals will be evaluated and sent as an output. The output signals are sent to the actuators. The actuators in this plant are mainly conveyor belts and machine motors. Thus, a logical signal of ”0” or ”1” is received from the sensors. Also, the signals sent to the

actuators are either a "0" or "1". A list of all the variables and their addresses is shown in Fig. [8] and [9].

Address	Variable name	Meaning of binary zero	Meaning of binary one
E 0.0	stop_button	STOP-Button is actuated	STOP-Button is not actuated
E 0.1	start_button	START-Button is not actuated	START-Button is actuated
E 0.2	clear_button	CLEAR-Button is not actuated	CLEAR-Button is actuated
E 0.3	init_button	INIT-Button is not actuated	INIT-Button is actuated
E 0.6	feeder_ready	Conveyor chain in interim position	Conveyor chain in initial position
E 0.7	feeder_empty	Feeder is not empty	Feeder is empty
E 1.2	drill_top	Drill head is at upper limit	Drill head is not at upper limit
E 1.3	drill_bottom	Drill head is at lower limit	Drill head is not at lower limit
E 1.4	drill_entrance	No workpiece is at beginning of conveyor belt	Workpiece is at beginning of conveyor belt
E 1.5	drill_position	No workpiece is in front of drill	Workpiece is in front of drill
E 2.2	vmill_top	Mill head is at upper limit	Mill head is not at upper limit
E 2.3	vmill_bottom	Mill head is at lower limit	Mill head is not at lower limit
E 2.4	vmill_entrance	No workpiece is at beginning of conveyor belt	Workpiece is at beginning of conveyor belt
E 2.5	vmill_position	No workpiece is in front of vertical mill	No workpiece is in front of vertical mill
E 2.6	vmill_tool_ready	Tool is in working position	Tool is in not working position
E 3.0	hmill_back	Mill head is at back limit	Mill head is not at back limit
E 3.1	hmill_front	Mill head is at front limit	Mill head is not at front limit
E 3.2	hmill_top	Mill head is at upper limit	Mill head is not at upper limit
E 3.3	hmill_bottom	Mill head is at lower limit	Mill head is not at lower limit
E 3.4	hmill_entrance	No workpiece is at beginning of conveyor belt	Workpiece is at beginning of conveyor belt
E 3.5	hmill_position	No workpiece is in front of horizontal mill	Workpiece is in front of horizontal mill

Figure 8: Definition of Inputs

Address	Variable name	Meaning of binary zero	Meaning of binary one
A 0.0	stop_signal	LED STOP off	LED STOP on
A 0.1	start_signal	LED START off	LED START on
A 0.2	clear_signal	LED CLEAR-STOP off	LED CLEAR-STOP on
A 0.3	init_signal	LED INIT off	LED INIT on
A 0.5	feeder_signal	LED FEEDER EMPTY off	LED FEEDER EMPTY on
A 0.6	feeder_motor	Motor conveyor chain off	Motor conveyor chain on
A 1.2	drill_motor_up	Motor up off	Motor up on
A 1.3	drill_motor_down	Motor down off	Motor down on
A 1.4	drilling_motor	Motor tool off	Motor tool on
A 1.5	drill_conveyor	Motor conveyor belt off	Motor conveyor belt on
A 2.2	vmill_motor_up	Motor up off	Motor up on
A 2.3	vmill_motor_down	Motor down off	Motor down on
A 2.4	vmilling_motor	Motor tool off	Motor tool on
A 2.5	vmill_conveyor	Motor conveyor belt off	Motor conveyor belt on
A 2.7	vmill_change_right	Motor tool changer off	Motor tool changer on
A 3.0	hmill_motor_back	Motor backwards off	Motor backwars on
A 3.1	hmill_motor_front	Motor forwards off	Motor forwards on
A 3.2	hmill_motor_up	Motor up off	Motor up on
A 3.3	hmill_motor_down	Motor down off	Motor down on
A 3.4	hmilling_motor	Motor tool off	Motor tool on
A 3.5	hmill_conveyor	Motor conveyor belt off	Motor conveyor belt on

Figure 9: Definition of Outputs

2.3 Task Description

First, the plant was run using different control languages locally. The plant was tested and it was running perfectly in real-time without any issues. In this project, the idea is to run the same plant using cloud control. Instead of running the plant locally, a program is uploaded to a cloud, which can be used online to monitor and run the plant from anywhere. This requires a lot of different connections. The first thing is to establish a connection between the plant and the PLC. To connect the PLC to the cloud a Message Queuing Telemetry Transport (MQTT) protocol in the Node-RED application is used. MQTT is a lightweight,

publish-subscribe network protocol that transports messages between devices. In the cloud, the program is run based on the MQTT messages received from the sensors via Node-RED. The calculation and the evaluation parts are carried out in the cloud. The results, which are output signals, will be sent back to Node-RED and again back to the PLC which will run the plant. Further details about Node-RED and the cloud used will be discussed in the coming chapters. Also, since these computations and connections take time and are related to the internet connection as well, some problems will be raised like time lags. This will also be discussed in later chapters when showing the testing results of the plant.

3 Comparison of Cloud Services

Modern cloud computing came into existence in the 2000s and has since revolutionized technology and scaled applications and processes to a multitude of possibilities. Amazon Web Service was one of the first modern cloud computing platforms, established in 2002. But at the present day, we have various options to select from when deciding on creating a cloud application. Controlling plants from the cloud is the latest trend leading to Industrial Internet-of-things and thus Industry 4.0. A detailed comparison can be easily found on the web, so here is a brief comparison justifying the selection of a particular cloud platform for this Project [11].

Siemens S7 provides software modules that can be used with Amazon Web Service, Microsoft Azure Cloud, etc. But, this project tries to control the plant without the pre-built services for direct connection so that different cloud services can be compared and evaluated [2].

3.1 IBM Cloud

IBM Cloud was the first choice, and it led to the discovery of many cloud features that are currently being used by bigger companies. IBM Watson Internet of Things (IoT) package installed in the Node-RED is used to send messages to a "Quickstart" service which directly lets one visualize the incoming data. This data could be then used in the IBM Cloud Foundry Application to design the control algorithm. The disadvantages here were that the Node-RED packages for IBM Watson IoT were outdated and could not be used with Siemens S7 PLC. Also, the service demanded high-level programming in python or java which would be cumbersome for a new user to understand and modify the program. Since the program is a logic control program, it would be better to understand if it was like Ladder Diagram, Sequential Function Chart, etc. Although we tried to use the free plan, the IBM cloud is relatively expensive compared to other cloud platforms if a high volume of data is to be communicated and/or stored [8].

3.2 Amazon Web Service

Amazon Web Service was the next platform to be utilized and tested and it allowed us to program the PLC just like a SIPN, which is the first step before writing logic programs in Ladder Logic or any other programming languages of PLC. Amazon IoT Events is a special cloud service that allows making detector models analogical to SIPNs which can be used to create a graphical program that can be easily understood by anyone and easy to modify. The Node-RED MQTT nodes helped in communicating with this detector model over specified "topic" which is further elaborated in coming chapters. The AWS is also cheap relatively for large volume data communication. This is why AWS is superior in this case because it aligns with our interests and objectives [3].

4 Connection with Node-RED

4.1 Architecture

The purpose of our work is to operate the plant from the cloud platform. This leads to one of the most challenging parts of the project, which is to be able to read and write data to/from the plant not from a local PC but using cloud base platform like AWS. The complete data flow diagram can be seen in Fig. [10].

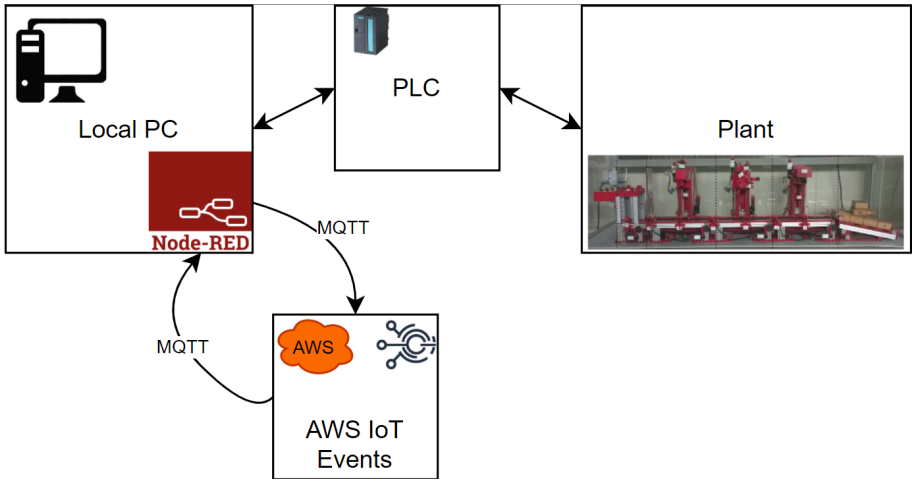

Figure 10: The Network Architecture

4.2 Node-RED connection with Siemens S7-300

To start with, firstly we need to be able to get data from Plant. This was done by using Node-RED.

Node-RED is a programming tool for wiring together hardware devices, Application Program Interfaces (APIs), and online services in new and interesting ways. Node-RED provides a browser-based flow editor that makes it easy to wire together flows using the wide range of nodes in the palette [9]. The term "flow" is used to informally describe a single set of connected nodes. So a flow (tab) can contain multiple flows (sets of connected nodes). A "node" is a drag-and-drop block that can have numerous functions. Flows can be then deployed to the run-time in a single click. JavaScript functions can be created within the editor using a rich text editor. A built-in library allows you to save useful functions, templates or flows for re-use. Node-RED is built on Node.js, taking full advantage of its event-driven, non-blocking model. This makes it ideal to run at the edge of the network on low-cost hardware such as the Raspberry Pi as well as in the cloud.

It is installed in our case locally on the PC, as it has a direct connection to the plant we want to operate. However, it can be installed locally using Docker or could be run directly in the Cloud. After installing and running it, Node-RED will be running locally on this address "http://127.0.0.1:1880/". The next step is to connect the plant to the Node-RED. It is done by installing "Node-RED-contrib-s7" in "Manage Palette" [10]. After this, we need to set the connection using "s7 in" block and add connection properties like IP Address, Port, Slot, etc. The "s7 in" block is depicted in Fig. [11].

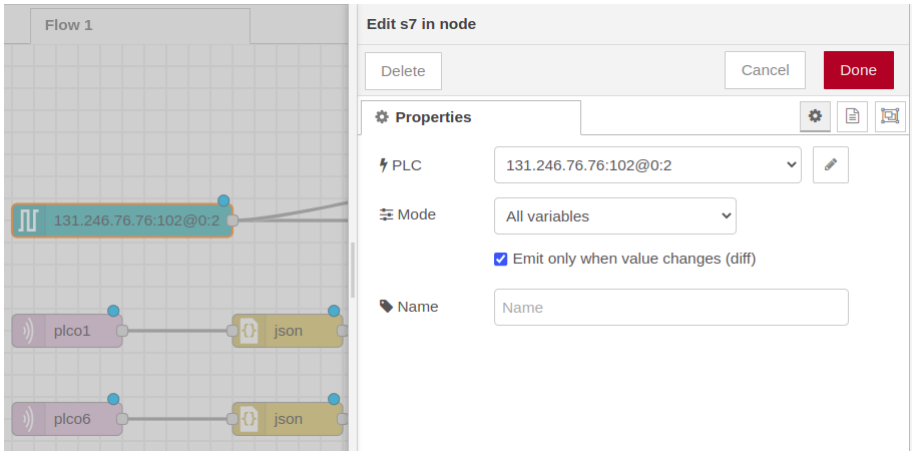

Figure 11: S7-in Node

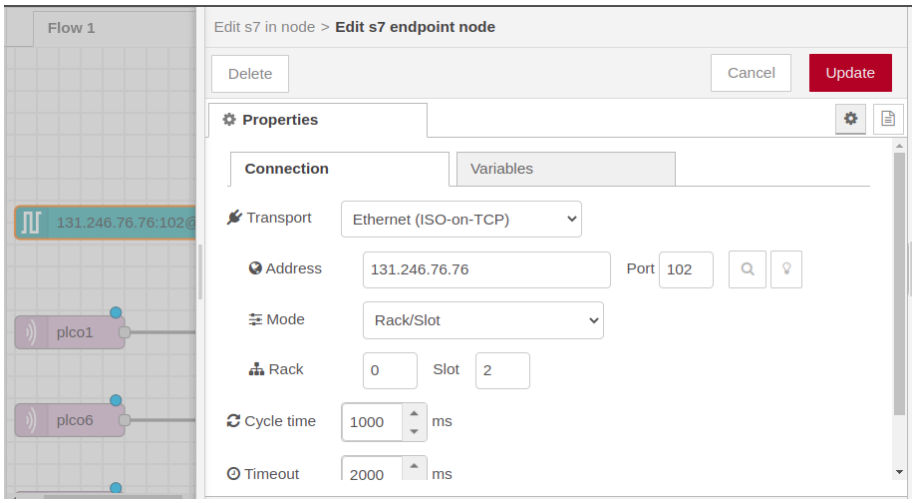

Figure 12: Editing S7-endpoint inside S7-in

After the connection is established between the PLC and the Node-RED application, we need to add all the required plant variables, which can be then read and written from/to respectively.As can be seen in Fig. [12], the Connection and Variables are defined and added respectively in the "s7 in" block. This block has different modes: "Single variable", "All variables", and to have all variables in a single JavaScript Object Notation (JSON) object choose "All variables". The variable list can be referred from Fig. [13]. This lets us send all variables at once and, later, in AWS implementation there will be no need to define multiple rules in the AWS. We can use one rule and action in AWS for the whole data set. The list of all variables could be seen in the previous chapter. You may add a "debug" block to see Debug Message changes after, e.g., putting a workpiece in front of the sensor, which variable you already added on "s7 in" block.

Also, from Node-RED you may send data back to the plant. To do that you could add the "Inject" block and choose a boolean type to be set in the variable of any actuator of the plant. Note: the PLC should have no saved/loaded program, but only variables. Do not forget to add the "Inject" block with the opposite boolean to turn off the actuator.

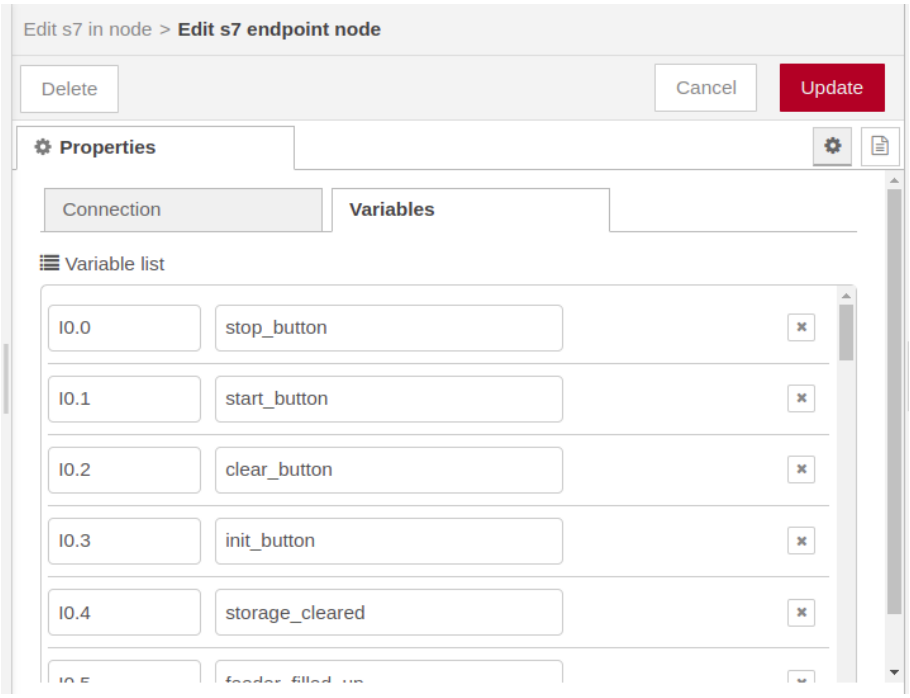

Figure 13: S7 variables

4.3 Node-RED and AWS over MQTT

The connection to AWS cloud is done using MQTT protocol, specifically, the "MQTT in" block in our flow. To operate and monitor each signal and actuator an MQTT request is to be thus added for each one of them. The MQTT protocol now needs to be set up with AWS cloud address, by that, we could connect out plant to cloud through the Node-RED. The MQTT node is depicted in Fig. [14] Detailed AWS implementation is discussed in Chapter "Using AWS Cloud Platform".

The flow after setting up all the nodes and connecting them, looks like as shown in Fig. [15].

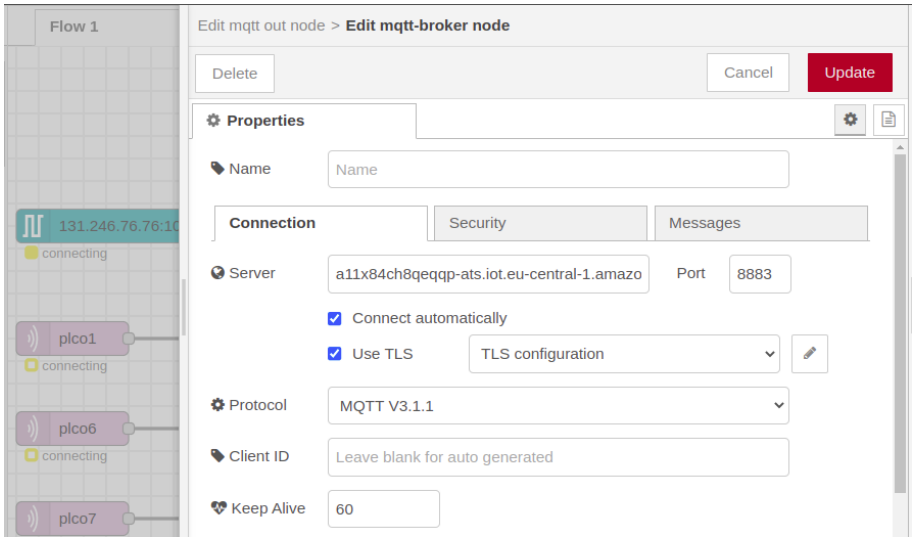

Figure 14: Editing MQTT broker in MQTT out node

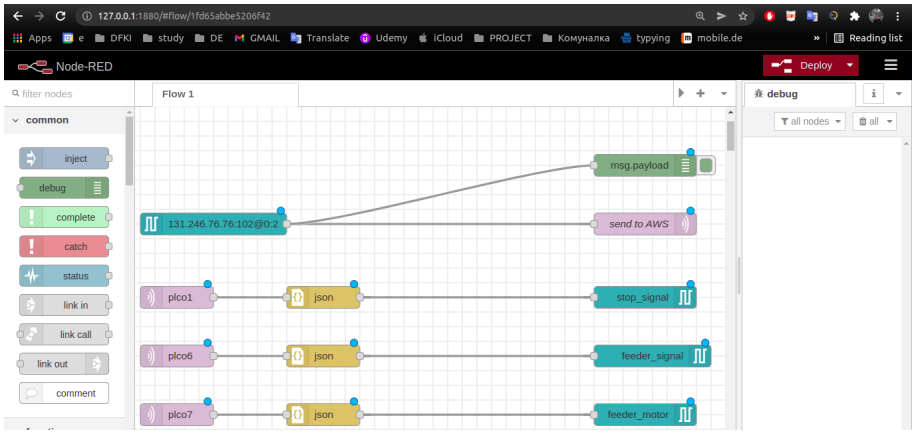

Figure 15: Node-RED flow created for application with all necessary nodes

5 Using AWS Cloud Platform

5.1 Introduction to Amazon Web Services

Amazon Web Service (AWS) is an Information Technology (IT) service management company dealing with on-demand cloud computing platforms for industrial or individual purposes with various features incorporating APIs to individuals, companies, and governments on a metered basis [5].

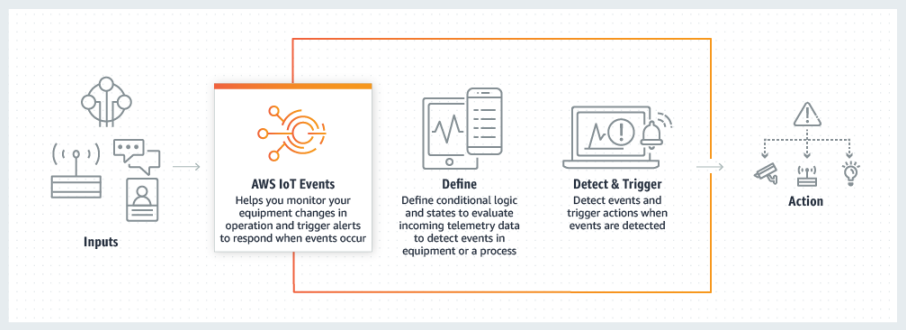

Figure 16: AWS IOT Events Architecture

5.2 Creating a "Thing" with AWS IoT Core

AWS provides various services for various applications. We make use of the AWS IoT Core and AWS IoT Events service (Fig. [16] describes the architecture of this service) in our project. AWS IoT Core lets us create a "thing" on the cloud. This IoT thing is a representation and record of the physical device, i.e., the Siemens PLC S7-300 for our case in the cloud. Any physical device has to have a thing to work with AWS IoT services [6]. To create a thing [7], we follow the steps listed below:

Step 1: Login to AWS IoT Console and navigate to IoT Core.

Step 2: Open the "**Manage**" tab on the left and then select "**create**" to make a new thing. For our Project, we only require a single thing (Single PLC is to be linked) so click on "**Create a Single Thing**".

Step 3: Now give a name to the newly created thing and generate necessary Certificates and Keys (use the recommended option). This is used to authenticate a device from the cloud. After generation, AWS will allow the user to download all the necessary keys and certificates for that thing. Note that this is the only time a user can download them.

Step 4: Download and save all the Certificates and Keys on your PC. Also, activate this certificate after downloading on the same page. Now you can publish this thing.

Step 5: The next step is to attach a policy to this thing that will let us define allowable functions for this thing. To create a policy, go to the "**Secure**" tab on the left and select "**Policies**". Here click on "**Create**" and give a name to your policy.

Step 6: In the "Statements" box, write the Action as "**iot:***" and Amazon Resource Name (ARN) as "*". This defines that this policy has all IoT permissions and there is no specific resource for all actions. Select the "**Allow**" Effect and create the policy.

Step 7: Now navigate to Manage > Things > "Your newly created thing" > Security > Policies. Here, attach the newly created policy to this thing. Now, the AWS IoT core is all set for our new device to communicate with other cloud services.

5.3 Testing the "Thing"

The AWS IoT Core thing is now ready to be tested with the Node-RED flow after completing the necessary steps discussed in the previous chapter to connect the MQTT in/out nodes to our thing in the AWS by uploading necessary certificates and keys.

The topic name written in the Node-RED flow's MQTT out node can be directly used to see the incoming messages in the cloud. For our application, we get a JSON format message about various I/O in our PLC as Boolean values. The "**Test**" tab on the left in the AWS IoT Core page lets us test our message reception. The topic name can be written in the "**Subscribe**" section and all incoming messages can be read after subscribing to this topic. The subscription tab is depicted in Fig. [18].

An example of the JSON object we receive from our MQTT out node is shown in Fig. [17].

Now following the same procedure for an MQTT out node in the Node-RED, we can publish some data on this topic and see the actions being implemented in our device. Since the topic "plco1" is linked to the Drill conveyor of our plant, publishing a "true"/"1" message on this topic should turn the conveyor motor on. An example of testing our "publish to" is shown in Fig. [19].

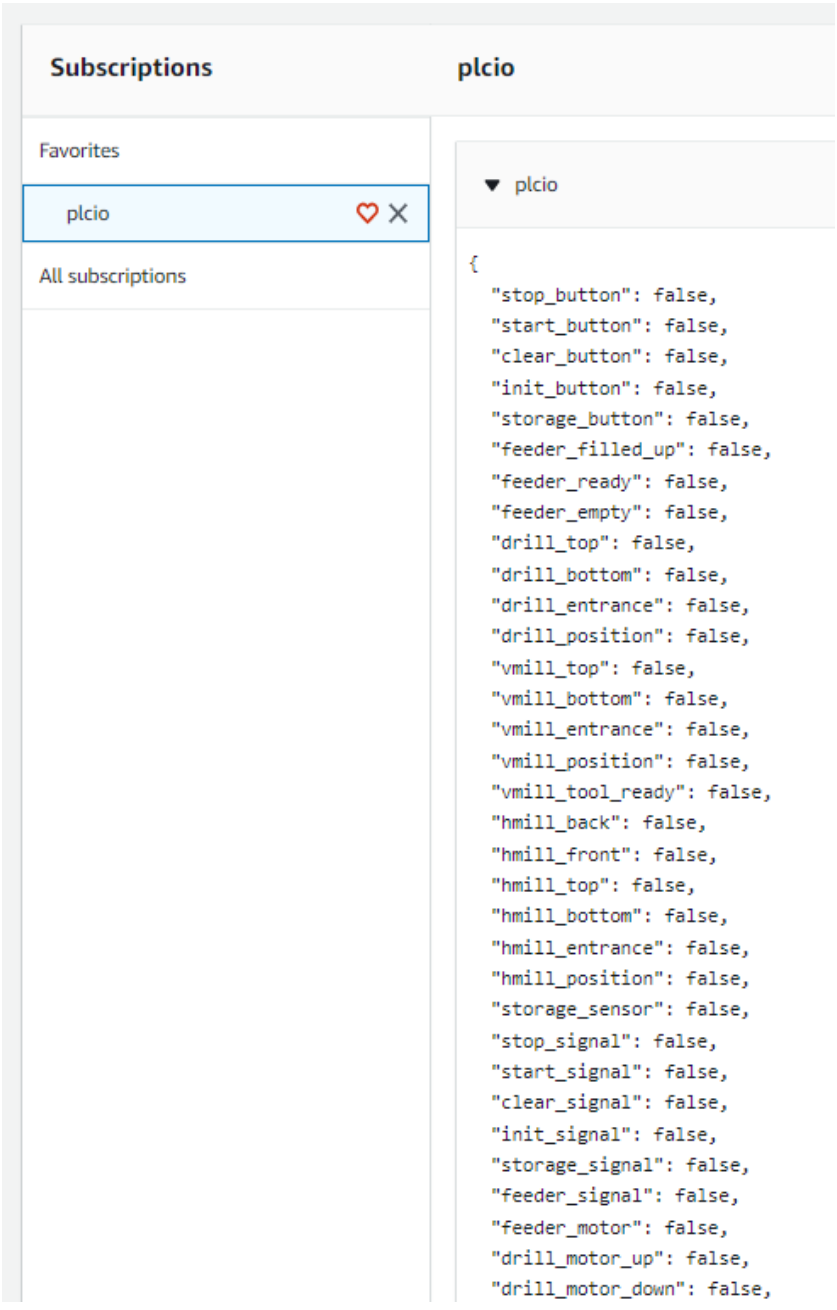

Figure 17: Sample JSON object message received from Node-RED

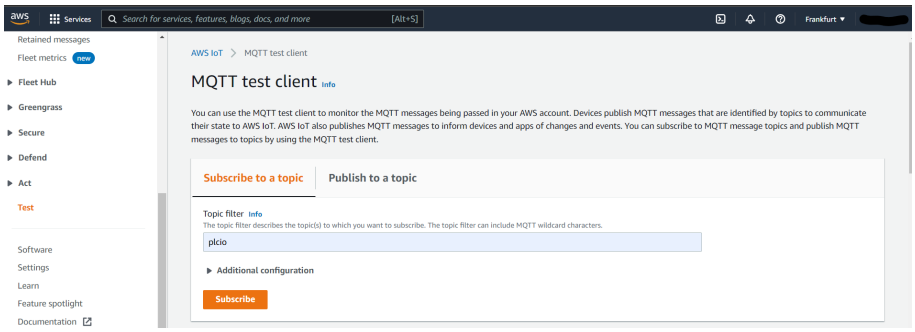

Figure 18: Testing the subscription to a topic

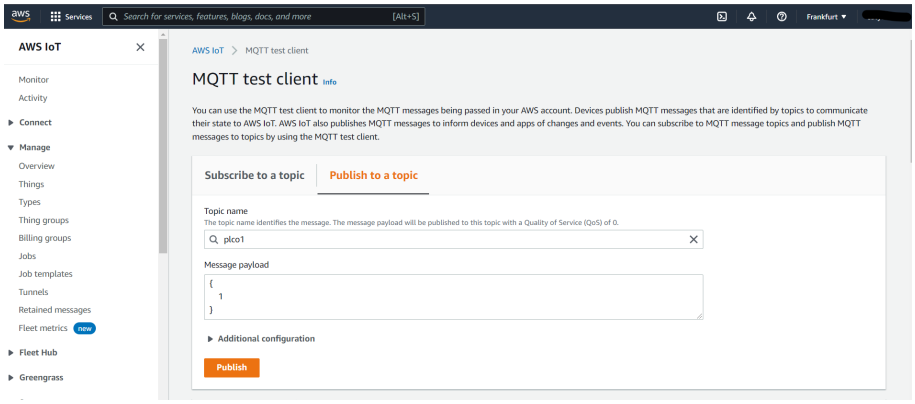

Figure 19: Testing the publication to a topic

5.4 Creating an Act Sequence

With this feature, one can set or define a rule for the incoming messages. The "Rule" tab as shown in Fig. [20] lets one route the device data to many AWS Services available. Creating a rule with the rule query statement being "SELECT * FROM 'plcio'", where "plcio" is the incoming message topic, allows us to select the whole JSON object that we receive as a message from Node-RED. This can be further optimized by selecting only the required data by some SQL FROM/WHERE specifications. In our case, we only receive all the I/O variables as boolean values and all at once.

Now, we have a multitude of options to use this message further in one or many of the AWS Services simultaneously. AWS IoT Events is an industrially

recognized service that lets the user make a detector model to monitor and trigger certain events. This service is very similar to developing a digital "Petri Net" for the logic control of the plant.

Thus, in the "Actions" pane for this rule, we select "Send a message to an IoT Events Input" where we can create a new input for our IoT Events [4]. The input name for our detector model is "plant_start". While creating this input, we need to specify a typical input message this input may receive from our IoT core. Thus, here we upload a sample JSON file that we receive as input in the subscriber topic "plcio" or MQTT out node topic in Node-RED, which has all the variables in it.

The Action is set and now our AWS can make something useful out of the incoming data in the IoT Events Service. This is where we design the control logic of our plant.

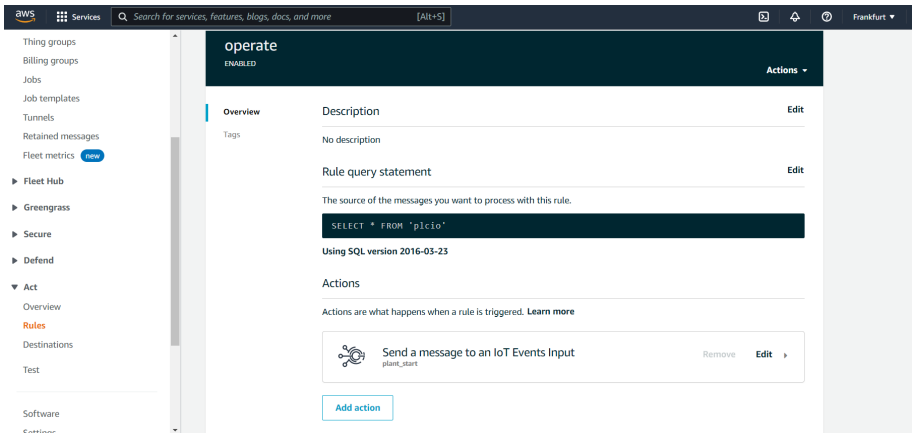

Figure 20: Setting a new rule

5.5 Control Logic in Detector Model

A detector model allows us to define, detect and trigger events based on received inputs. A detector model can be created from the AWS IoT Events Service webpage. Here, we use the same input that we created in our rule action i.e., "plant_start". The details on how a detector model is made can be seen in the documentation of AWS IoT Events. This is very similar to a Petri Net as you can add states where a particular event happens. For example, publishing a "true"/"1" message to an MQTT topic where you need to run a conveyor in our case. Then, there are transition states which check for conditions continuously once the detector model enters a state before it. The analogy to a Petri Net

can be noted here. A transition event can be defined as a conditional statement from one of the inputs; some timer/counter triggers; or as message requests etc. For example, if we need to stop the Drill conveyor once the drill position sensor detects an object, we can define the transition event as

```
$input.plant_start.drill_position == true
```

If this condition is satisfied, the transition occurs and the conveyor which was running in the previous state can be stopped in the next state by again publishing "false"/"0" to the same topic.
Therefore, the whole control logic can be developed here based on input signal conditions and publishing outputs to the actuators directly in the same model. The detector model example for a test run of the plant is shown in Fig. [21].

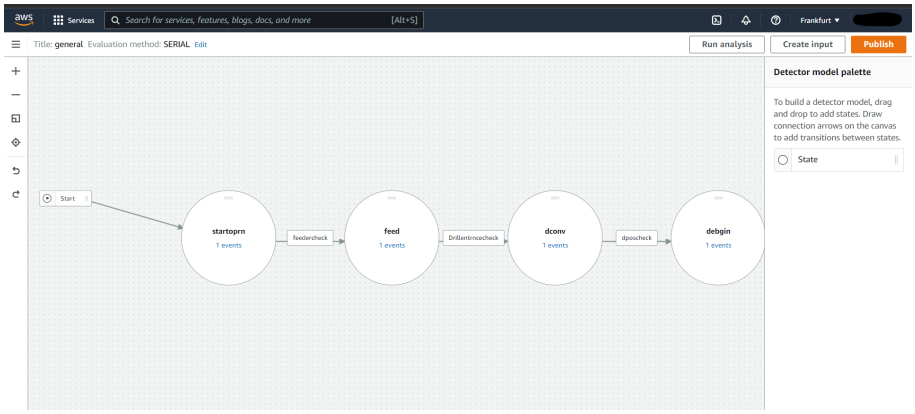

Figure 21: Sample Detector Model in AWS IoT Events

The circular shapes are the "States", and the rectangular boxes are the "Transition states" in the figure shown above. It is important to note that the initial trigger of the detector model occurs only when it receives an input, i.e., the model remains in the "Start" state shown above until it detects an input of any form. Also, it is better to export the model every time some changes are made. This is because once the final state is reached in the detector model, it is not possible to go back again if the model is not reversible. Thus, deleting the current model and importing the same model again will reset the state back to "Start". It should also be kept in mind that the input name should not be changed in the process, otherwise there will be no communication between the IoT core and IoT events.

6 Results and Future Scope

After finishing the whole control logic in the detector model, one can directly test the operation of the whole plant by sending a simple input message from the plant. The "start_button" is set to "1" for example by pressing it physically on the plant and thus a JSON object will be sent to the detector model making it move from the "Start" state to the first state in the model. Thereafter, it starts operation by checking transition logic and switching states if they are satisfied.

The major obstacle in the whole cloud-control implementation is the delay in signal reception and publication. These delays cause the conveyors and motors to stop later than required, causing a lag in the plant and even completely ignoring some instantaneous input signals such as reed switches, position sensors, etc. which switches to "1" or "0" only for some small time-instant, which is overlooked by the communication lag.

Thus, an implementation of the control logic for a simple operation is achieved. But for complex operations in the plant, such as simultaneous machining in the plant, the delays and bottlenecks play a very important role and need to be eliminated. This creates scope for further research on overcoming these delays by incorporating event- and time-based triggers simultaneously. Also, limitation on the message size from an MQTT out node is of concern allowing us to limit the number of Boolean variables that we can send to the IoT Events' inputs in a single message. All these limitations can be overcome with some research on other AWS services. But, a very detailed implementation of this control logic on the cloud-based on the analogy to a Petri Net is implemented in the project making it easier to transform and publish PLC programs directly on the cloud with a better understanding of the currently active states in the plant from any part of the world.

Bibliography

[1] Siemens AG 2006. *Ladder Logic (LAD) for S7-300 and S7-400 Programming. SIMATIC*. Siemens AG, 2006. ISBN A5E00706949-01.

[2] Siemens AG 2001 2017. *Programmable controller S7-300. SIMATIC*. Siemens AG, 2017. ISBN A5E00105505-AJ.

[3] Brian Adler. Private and Hybrid Clouds: 9 Use Cases and Implementation Advice. doi: https://www.flexera.com/blog/cloud/private-and-hybrid-clouds-9-use-cases-and-implementation-advice/.

[4] Inc. Amazon Web Services. What is aws iot events?, . URL `https://docs.aws.amazon.com/iotevents/latest/developerguide/what-is-iotevents.html`.

[5] Inc. Amazon Web Services. What is aws iot?, . URL `https://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html`.

[6] Inc. Amazon Web Services. Aws iot core, . URL `https://docs.aws.amazon.com/timestream/latest/developerguide/IOT-Core.html`.

[7] Inc. Amazon Web Services. Setting up aws iot, . URL `https://docs.aws.amazon.com/iot/latest/developerguide/iot-moisture-setup.html`.

[8] OpenJS Foundation Contributors. Running on ibm cloud, . URL `https://nodered.org/docs/getting-started/ibmcloud`.

[9] OpenJS Foundation Contributors. Running node-red locally, . URL `https://nodered.org/docs/getting-started/local`.

[10] OpenJS Foundation Contributors. node-red-contrib-s7, . URL `https://flows.nodered.org/node/node-red-contrib-s7`.

[11] Christian Rigg. AWS vs Azure vs Google Cloud vs IBM Cloud: What's the best cloud environment? doi: https://www.tomsguide.com/features/aws-vs-azure-vs-google-cloud-vs-ibm-cloud-whats-the-best-cloud-environment.

[12] Siemens. Industrial cloud computing. 2022. doi: https://www.plm.automation.siemens.com/global/en/our-story/glossary/industrial-cloud-computing/58773.

Appendices

A JSON file for the flow created in Node-RED

Note: The Certificate, Key and Certificate Authority (CA) names have been marked as "***" for privacy at the end of this JSON file.


```
[
   {
       "id": "1fd65abbe5206f42",
       "type": "tab",
       "label": "Flow 1",
       "disabled": false,
       "info": "",
       "env": []
   },
   {
       "id": "028a0f4e839597bc",
       "type": "s7 in",
       "z": "1fd65abbe5206f42",
       "endpoint": "e63e982caac36b4d",
       "mode": "all",
       "variable": "DI14",
       "diff": true,
       "name": "",
       "x": 240,
       "y": 240,
       "wires": [
           [
               "898181db18e017d9",
               "f3e533a82c839d1a"
           ]
       ]
   },
   {
       "id": "898181db18e017d9",
       "type": "mqtt out",
       "z": "1fd65abbe5206f42",
       "name": "send to AWS",
       "topic": "plcio",
       "qos": "",
       "retain": "",
       "respTopic": "",
       "contentType": "",
       "userProps": "",
```

```
        "correl": "",
        "expiry": "",
        "broker": "dc65e55f6bd20655",
        "x": 830,
        "y": 240,
        "wires": []
    },
    {
        "id": "0ddacc10c54572d1",
        "type": "mqtt in",
        "z": "1fd65abbe5206f42",
        "name": "",
        "topic": "plco1",
        "qos": "0",
        "datatype": "auto",
        "broker": "dc65e55f6bd20655",
        "nl": false,
        "rap": true,
        "rh": 0,
        "inputs": 0,
        "x": 190,
        "y": 340,
        "wires": [
            [
                "f0496e6ff90cae11"
            ]
        ]
    },
    {
        "id": "f0496e6ff90cae11",
        "type": "json",
        "z": "1fd65abbe5206f42",
        "name": "",
        "property": "payload",
        "action": "",
        "pretty": false,
        "x": 390,
        "y": 340,
        "wires": [
            [
                "f5f2a08dfba88ce6"
            ]
        ]
    },
    {
        "id": "f78d8d8a7651b6c6",
```

```
        "type": "s7 out",
        "z": "1fd65abbe5206f42",
        "endpoint": "e63e982caac36b4d",
        "variable": "hmilling_motor",
        "name": "",
        "x": 840,
        "y": 900,
        "wires": []
    },
    {
        "id": "0bfc35d7cf35b24b",
        "type": "mqtt in",
        "z": "1fd65abbe5206f42",
        "name": "",
        "topic": "plco21",
        "qos": "0",
        "datatype": "auto",
        "broker": "dc65e55f6bd20655",
        "nl": false,
        "rap": true,
        "rh": 0,
        "inputs": 0,
        "x": 190,
        "y": 900,
        "wires": [
            [
                "2772016adf2717cd"
            ]
        ]
    },
    {
        "id": "2772016adf2717cd",
        "type": "json",
        "z": "1fd65abbe5206f42",
        "name": "",
        "property": "payload",
        "action": "",
        "pretty": false,
        "x": 390,
        "y": 900,
        "wires": [
            [
                "f78d8d8a7651b6c6"
            ]
        ]
    },
```

```
{
    "id": "f3e533a82c839d1a",
    "type": "debug",
    "z": "1fd65abbe5206f42",
    "name": "",
    "active": true,
    "tosidebar": true,
    "console": false,
    "tostatus": false,
    "complete": "false",
    "statusVal": "",
    "statusType": "auto",
    "x": 830,
    "y": 160,
    "wires": []
},
{
    "id": "f5f2a08dfba88ce6",
    "type": "s7 out",
    "z": "1fd65abbe5206f42",
    "endpoint": "e63e982caac36b4d",
    "variable": "stop_signal",
    "name": "",
    "x": 830,
    "y": 340,
    "wires": []
},
{
    "id": "b0386f99f6f87167",
    "type": "mqtt in",
    "z": "1fd65abbe5206f42",
    "name": "",
    "topic": "plco20",
    "qos": "0",
    "datatype": "auto",
    "broker": "dc65e55f6bd20655",
    "nl": false,
    "rap": true,
    "rh": 0,
    "inputs": 0,
    "x": 190,
    "y": 980,
    "wires": [
        [
            "695751e19d5098fd"
        ]
```

```
        ]
    },
    {
        "id": "695751e19d5098fd",
        "type": "json",
        "z": "1fd65abbe5206f42",
        "name": "",
        "property": "payload",
        "action": "",
        "pretty": false,
        "x": 390,
        "y": 980,
        "wires": [
            [
                "481485789e802c80"
            ]
        ]
    },
    {
        "id": "481485789e802c80",
        "type": "s7 out",
        "z": "1fd65abbe5206f42",
        "endpoint": "e63e982caac36b4d",
        "variable": "hmill_conveyor",
        "name": "",
        "x": 840,
        "y": 980,
        "wires": []
    },
    {
        "id": "e2242d036d54f0d8",
        "type": "s7 out",
        "z": "1fd65abbe5206f42",
        "endpoint": "e63e982caac36b4d",
        "variable": "feeder_signal",
        "name": "",
        "x": 840,
        "y": 420,
        "wires": []
    },
    {
        "id": "34b9f5584f1582b3",
        "type": "mqtt in",
        "z": "1fd65abbe5206f42",
        "name": "",
        "topic": "plco6",
```

```
        "qos": "0",
        "datatype": "auto",
        "broker": "dc65e55f6bd20655",
        "nl": false,
        "rap": true,
        "rh": 0,
        "inputs": 0,
        "x": 190,
        "y": 420,
        "wires": [
            [
                "0dc3d95ec87cc8ee"
            ]
        ]
    },
    {
        "id": "0dc3d95ec87cc8ee",
        "type": "json",
        "z": "1fd65abbe5206f42",
        "name": "",
        "property": "payload",
        "action": "",
        "pretty": false,
        "x": 390,
        "y": 420,
        "wires": [
            [
                "e2242d036d54f0d8"
            ]
        ]
    },
    {
        "id": "06eec8b143a427fb",
        "type": "mqtt in",
        "z": "1fd65abbe5206f42",
        "name": "",
        "topic": "plco7",
        "qos": "0",
        "datatype": "auto",
        "broker": "dc65e55f6bd20655",
        "nl": false,
        "rap": true,
        "rh": 0,
        "inputs": 0,
        "x": 190,
        "y": 500,
```

```
        "wires": [
            [
                "ac414478e9f81959"
            ]
        ]
    },
    {
        "id": "ac414478e9f81959",
        "type": "json",
        "z": "1fd65abbe5206f42",
        "name": "",
        "property": "payload",
        "action": "",
        "pretty": false,
        "x": 390,
        "y": 500,
        "wires": [
            [
                "efde4c34b5966871"
            ]
        ]
    },
    {
        "id": "efde4c34b5966871",
        "type": "s7 out",
        "z": "1fd65abbe5206f42",
        "endpoint": "e63e982caac36b4d",
        "variable": "feeder_motor",
        "name": "",
        "x": 830,
        "y": 500,
        "wires": []
    },
    {
        "id": "fe39f24a0ceb2a19",
        "type": "s7 out",
        "z": "1fd65abbe5206f42",
        "endpoint": "e63e982caac36b4d",
        "variable": "drilling_motor",
        "name": "",
        "x": 840,
        "y": 580,
        "wires": []
    },
    {
        "id": "74f5c5070459e50e",
```

```
        "type": "mqtt in",
        "z": "1fd65abbe5206f42",
        "name": "",
        "topic": "plco10",
        "qos": "0",
        "datatype": "auto",
        "broker": "dc65e55f6bd20655",
        "nl": false,
        "rap": true,
        "rh": 0,
        "inputs": 0,
        "x": 190,
        "y": 580,
        "wires": [
            [
                "7d9a05ac68a45783"
            ]
        ]
    },
    {
        "id": "7d9a05ac68a45783",
        "type": "json",
        "z": "1fd65abbe5206f42",
        "name": "",
        "property": "payload",
        "action": "",
        "pretty": false,
        "x": 390,
        "y": 580,
        "wires": [
            [
                "fe39f24a0ceb2a19"
            ]
        ]
    },
    {
        "id": "552b892cd27f78d7",
        "type": "mqtt in",
        "z": "1fd65abbe5206f42",
        "name": "",
        "topic": "plco11",
        "qos": "0",
        "datatype": "auto",
        "broker": "dc65e55f6bd20655",
        "nl": false,
        "rap": true,
```

```
        "rh": 0,
        "inputs": 0,
        "x": 190,
        "y": 660,
        "wires": [
            [
                "fa57bcae7e0068cc"
            ]
        ]
    },
    {
        "id": "fa57bcae7e0068cc",
        "type": "json",
        "z": "1fd65abbe5206f42",
        "name": "",
        "property": "payload",
        "action": "",
        "pretty": false,
        "x": 390,
        "y": 660,
        "wires": [
            [
                "79b9a61e943b2fd9"
            ]
        ]
    },
    {
        "id": "79b9a61e943b2fd9",
        "type": "s7 out",
        "z": "1fd65abbe5206f42",
        "endpoint": "e63e982caac36b4d",
        "variable": "drill_conveyor",
        "name": "",
        "x": 840,
        "y": 660,
        "wires": []
    },
    {
        "id": "fa0749f1d299c5e9",
        "type": "inject",
        "z": "1fd65abbe5206f42",
        "name": "",
        "props": [
            {
                "p": "payload"
            }
```

```
        ],
        "repeat": "",
        "crontab": "",
        "once": false,
        "onceDelay": 0.1,
        "topic": "",
        "payload": "false",
        "payloadType": "bool",
        "x": 1030,
        "y": 500,
        "wires": [
            []
        ]
    },
    {
        "id": "a73594e83f479f06",
        "type": "inject",
        "z": "1fd65abbe5206f42",
        "name": "",
        "props": [
            {
                "p": "payload"
            }
        ],
        "repeat": "",
        "crontab": "",
        "once": false,
        "onceDelay": 0.1,
        "topic": "",
        "payload": "true",
        "payloadType": "bool",
        "x": 1030,
        "y": 580,
        "wires": [
            []
        ]
    },
    {
        "id": "98b8ba55a51a642f",
        "type": "mqtt in",
        "z": "1fd65abbe5206f42",
        "name": "",
        "topic": "plco14",
        "qos": "0",
        "datatype": "auto",
        "broker": "dc65e55f6bd20655",
```

```
        "nl": false,
        "rap": true,
        "rh": 0,
        "inputs": 0,
        "x": 190,
        "y": 740,
        "wires": [
            [
                "1a1241e805713616"
            ]
        ]
    },
    {
        "id": "1a1241e805713616",
        "type": "json",
        "z": "1fd65abbe5206f42",
        "name": "",
        "property": "payload",
        "action": "",
        "pretty": false,
        "x": 390,
        "y": 740,
        "wires": [
            [
                "9388e5caefb51307"
            ]
        ]
    },
    {
        "id": "d30d2b7b86fd9e69",
        "type": "s7 out",
        "z": "1fd65abbe5206f42",
        "endpoint": "e63e982caac36b4d",
        "variable": "vmill_conveyor",
        "name": "",
        "x": 840,
        "y": 820,
        "wires": []
    },
    {
        "id": "716edf99898adc3c",
        "type": "mqtt in",
        "z": "1fd65abbe5206f42",
        "name": "",
        "topic": "plco15",
        "qos": "0",
```

```
        "datatype": "auto",
        "broker": "dc65e55f6bd20655",
        "nl": false,
        "rap": true,
        "rh": 0,
        "inputs": 0,
        "x": 190,
        "y": 820,
        "wires": [
            [
                "92c2935ecb0517a0"
            ]
        ]
    },
    {
        "id": "92c2935ecb0517a0",
        "type": "json",
        "z": "1fd65abbe5206f42",
        "name": "",
        "property": "payload",
        "action": "",
        "pretty": false,
        "x": 390,
        "y": 820,
        "wires": [
            [
                "d30d2b7b86fd9e69"
            ]
        ]
    },
    {
        "id": "9388e5caefb51307",
        "type": "s7 out",
        "z": "1fd65abbe5206f42",
        "endpoint": "e63e982caac36b4d",
        "variable": "vmilling_motor",
        "name": "",
        "x": 840,
        "y": 740,
        "wires": []
    },
    {
        "id": "e63e982caac36b4d",
        "type": "s7 endpoint",
        "transport": "iso-on-tcp",
        "address": "131.246.76.76",
```

```
"port": "102",
"rack": "0",
"slot": "2",
"localtsaphi": "01",
"localtsaplo": "00",
"remotetsaphi": "01",
"remotetsaplo": "00",
"connmode": "rack-slot",
"adapter": "",
"busaddr": "2",
"cycletime": "1000",
"timeout": "2000",
"name": "",
"vartable": [
    {
        "addr": "I0.0",
        "name": "stop_button"
    },
    {
        "addr": "I0.1",
        "name": "start_button"
    },
    {
        "addr": "I0.2",
        "name": "clear_button"
    },
    {
        "addr": "I0.3",
        "name": "init_button"
    },
    {
        "addr": "I0.4",
        "name": "storage_cleared"
    },
    {
        "addr": "I0.5",
        "name": "feeder_filled_up"
    },
    {
        "addr": "I0.6",
        "name": "feeder_ready"
    },
    {
        "addr": "I0.7",
        "name": "feeder_empty"
    },
```

```
    {
        "addr": "I1.2",
        "name": "drill_top"
    },
    {
        "addr": "I1.3",
        "name": "drill_bottom"
    },
    {
        "addr": "I1.4",
        "name": "drill_entrance"
    },
    {
        "addr": "I1.5",
        "name": "drill_position"
    },
    {
        "addr": "I2.2",
        "name": "vmill_top"
    },
    {
        "addr": "I2.3",
        "name": "vmill_bottom"
    },
    {
        "addr": "I2.4",
        "name": "vmill_entrance"
    },
    {
        "addr": "I2.5",
        "name": "vmill_position"
    },
    {
        "addr": "I2.6",
        "name": "vmill_tool_ready"
    },
    {
        "addr": "I3.0",
        "name": "hmill_back"
    },
    {
        "addr": "I3.1",
        "name": "hmill_front"
    },
    {
        "addr": "I3.2",
```

```
        "name": "hmill_top"
    },
    {
        "addr": "I3.3",
        "name": "hmill_bottom"
    },
    {
        "addr": "I3.4",
        "name": "hmill_entrance"
    },
    {
        "addr": "I3.5",
        "name": "hmill_position"
    },
    {
        "addr": "I3.6",
        "name": "storage_sensor"
    },
    {
        "addr": "A0.0",
        "name": "stop_signal"
    },
    {
        "addr": "A0.5",
        "name": "feeder_signal"
    },
    {
        "addr": "A0.6",
        "name": "feeder_motor"
    },
    {
        "addr": "A1.2",
        "name": "drill_motor_up"
    },
    {
        "addr": "A1.3",
        "name": "drill_motor_down"
    },
    {
        "addr": "A1.4",
        "name": "drilling_motor"
    },
    {
        "addr": "A1.5",
        "name": "drill_conveyor"
    },
```

```
            {
                "addr": "A2.2",
                "name": "vmill_motor_up"
            },
            {
                "addr": "A2.3",
                "name": "vmill_motor_down"
            },
            {
                "addr": "A2.4",
                "name": "vmilling_motor"
            },
            {
                "addr": "A2.5",
                "name": "vmill_conveyor"
            },
            {
                "addr": "A2.7",
                "name": "vmill_change_right"
            },
            {
                "addr": "A3.0",
                "name": "hmill_motor_back"
            },
            {
                "addr": "A3.4",
                "name": "hmilling_motor"
            },
            {
                "addr": "A3.5",
                "name": "hmill_conveyor"
            }
        ]
    },
    {
        "id": "dc65e55f6bd20655",
        "type": "mqtt-broker",
        "name": "",
        "broker": "a11x84ch8qeqqp-ats.iot.eu-central-1.amazonaws.
            ↪ com",
        "port": "8883",
        "tls": "a8ed97223a25cffc",
        "clientid": "",
        "autoConnect": true,
        "usetls": true,
        "protocolVersion": "4",
```

```
        "keepalive": "60",
        "cleansession": true,
        "birthTopic": "",
        "birthQos": "0",
        "birthPayload": "",
        "birthMsg": {},
        "closeTopic": "",
        "closeQos": "0",
        "closePayload": "",
        "closeMsg": {},
        "willTopic": "",
        "willQos": "0",
        "willPayload": "",
        "willMsg": {},
        "sessionExpiry": "",
        "credentials": {}
    },
    {
        "id": "a8ed97223a25cffc",
        "type": "tls-config",
        "name": "",
        "cert": "",
        "key": "",
        "ca": "",
        "certname": "***",
        "keyname": "***",
        "caname": "***",
        "servername": "",
        "verifyservercert": false,
        "alpnprotocol": "",
        "credentials": {}
    }
]
```


B JSON file of the detector model in AWS IoT Events


```
{
    "detectorModelDefinition": {
        "states": [
            {
                "stateName": "startoprn",
                "onInput": {
```

```
            "events": [],
            "transitionEvents": [
                {
                    "eventName": "feedercheck",
                    "condition": "$input.plant_start.
                        ↪ feeder_empty == false",
                    "actions": [],
                    "nextState": "feed"
                }
            ]
        },
        "onEnter": {
            "events": [
                {
                    "eventName": "begin",
                    "condition": "$input.plant_start.
                        ↪ start_button == true",
                    "actions": [
                        {
                            "iotTopicPublish": {
                                "mqttTopic": "plco13",
                                "payload": {
                                    "contentExpression": "0",
                                    "type": "STRING"
                                }
                            }
                        }
                    ]
                }
            ]
        },
        "onExit": {
            "events": []
        }
    },
    {
        "stateName": "feed",
        "onInput": {
            "events": [],
            "transitionEvents": [
                {
                    "eventName": "Drillentrncecheck",
                    "condition": "$input.plant_start.
                        ↪ drill_entrance == true",
                    "actions": [],
                    "nextState": "dconv"
```

```
                }
            ]
        },
        "onEnter": {
            "events": [
                {
                    "eventName": "feederrun",
                    "condition": "true",
                    "actions": [
                        {
                            "iotTopicPublish": {
                                "mqttTopic": "plco7",
                                "payload": {
                                    "contentExpression": "1",
                                    "type": "STRING"
                                }
                            }
                        }
                    ]
                }
            ]
        },
        "onExit": {
            "events": []
        }
    },
    {
        "stateName": "dconv",
        "onInput": {
            "events": [],
            "transitionEvents": [
                {
                    "eventName": "dposcheck",
                    "condition": "$input.plant_start.
                        ↪ drill_position == true",
                    "actions": [],
                    "nextState": "debgin"
                }
            ]
        },
        "onEnter": {
            "events": [
                {
                    "eventName": "runconv",
                    "condition": "true",
                    "actions": [
```

```
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco7",
                            "payload": {
                                "contentExpression": "0",
                                "type": "STRING"
                            }
                        }
                    },
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco11",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    }
                ]
            }
        ]
    },
    "onExit": {
        "events": []
    }
},
{
    "stateName": "debgin",
    "onInput": {
        "events": [],
        "transitionEvents": [
            {
                "eventName": "drillbottomcheck",
                "condition": "$input.plant_start.
                    ↪ drill_bottom == false",
                "actions": [],
                "nextState": "ddownstop"
            }
        ]
    },
    "onEnter": {
        "events": [
            {
                "eventName": "drilldown",
                "condition": "true",
                "actions": [
```

```
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco11",
                            "payload": {
                                "contentExpression": "0",
                                "type": "STRING"
                            }
                        }
                    },
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco10",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    },
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco9",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    }
                ]
            }
        ]
    },
    "onExit": {
        "events": []
    }
},
{
    "stateName": "ddownstop",
    "onInput": {
        "events": [],
        "transitionEvents": [
            {
                "eventName": "dtopchk",
                "condition": "$input.plant_start.
                    ↪ drill_top == false",
                "actions": [],
                "nextState": "dtonext"
```

```
            }
        ]
    },
    "onEnter": {
        "events": [
            {
                "eventName": "drilldownstop",
                "condition": "true",
                "actions": [
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco9",
                            "payload": {
                                "contentExpression": "0",
                                "type": "STRING"
                            }
                        }
                    },
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco8",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    }
                ]
            }
        ]
    },
    "onExit": {
        "events": []
    }
},
{
    "stateName": "dtonext",
    "onInput": {
        "events": [],
        "transitionEvents": [
            {
                "eventName": "vmillcheck",
                "condition": "$input.plant_start.
                    ↪ vmill_entrance == true",
                "actions": [],
                "nextState": "untitled_state_7"
```

```
                }
            ]
        },
        "onEnter": {
            "events": [
                {
                    "eventName": "stopdrill",
                    "condition": "true",
                    "actions": [
                        {
                            "iotTopicPublish": {
                                "mqttTopic": "plco8",
                                "payload": {
                                    "contentExpression": "0",
                                    "type": "STRING"
                                }
                            }
                        },
                        {
                            "iotTopicPublish": {
                                "mqttTopic": "plco10",
                                "payload": {
                                    "contentExpression": "0",
                                    "type": "STRING"
                                }
                            }
                        },
                        {
                            "iotTopicPublish": {
                                "mqttTopic": "plco11",
                                "payload": {
                                    "contentExpression": "1",
                                    "type": "STRING"
                                }
                            }
                        }
                    ]
                }
            ]
        },
        "onExit": {
            "events": []
        }
    },
    {
        "stateName": "untitled_state_7",
```

```
            "onInput": {
                "events": [],
                "transitionEvents": [
                    {
                        "eventName": "vposcheck",
                        "condition": "$input.plant_start.
                            ↪ vmill_position == true",
                        "actions": [],
                        "nextState": "untitled_state_1"
                    }
                ]
            },
            "onEnter": {
                "events": [
                    {
                        "eventName": "stop conv",
                        "condition": "true",
                        "actions": [
                            {
                                "iotTopicPublish": {
                                    "mqttTopic": "plco11",
                                    "payload": {
                                        "contentExpression": "0",
                                        "type": "STRING"
                                    }
                                }
                            },
                            {
                                "iotTopicPublish": {
                                    "mqttTopic": "plco15",
                                    "payload": {
                                        "contentExpression": "1",
                                        "type": "STRING"
                                    }
                                }
                            }
                        ]
                    }
                ]
            },
            "onExit": {
                "events": []
            }
        },
        {
            "stateName": "untitled_state_1",
```

```
"onInput": {
    "events": [],
    "transitionEvents": [
        {
            "eventName": "vmbotmchck",
            "condition": "$input.plant_start.
                ↪ vmill_bottom == false",
            "actions": [],
            "nextState": "untitled_state_2"
        }
    ]
},
"onEnter": {
    "events": [
        {
            "eventName": "vbegin",
            "condition": "true",
            "actions": [
                {
                    "iotTopicPublish": {
                        "mqttTopic": "plco15",
                        "payload": {
                            "contentExpression": "0",
                            "type": "STRING"
                        }
                    }
                },
                {
                    "iotTopicPublish": {
                        "mqttTopic": "plco14",
                        "payload": {
                            "contentExpression": "1",
                            "type": "STRING"
                        }
                    }
                },
                {
                    "iotTopicPublish": {
                        "mqttTopic": "plco13",
                        "payload": {
                            "contentExpression": "1",
                            "type": "STRING"
                        }
                    }
                }
            ]
```

```
            }
        ]
    },
    "onExit": {
        "events": []
    }
},
{
    "stateName": "untitled_state_2",
    "onInput": {
        "events": [],
        "transitionEvents": [
            {
                "eventName": "vmtopcheck",
                "condition": "$input.plant_start.
                    ↪ vmill_top == false",
                "actions": [],
                "nextState": "untitled_state_3"
            }
        ]
    },
    "onEnter": {
        "events": [
            {
                "eventName": "vbototop",
                "condition": "true",
                "actions": [
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco13",
                            "payload": {
                                "contentExpression": "0",
                                "type": "STRING"
                            }
                        }
                    },
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco12",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    }
                ]
```

```
            }
        ]
    },
    "onExit": {
        "events": []
    }
},
{
    "stateName": "untitled_state_3",
    "onInput": {
        "events": [],
        "transitionEvents": [
            {
                "eventName": "toolready",
                "condition": "$input.plant_start.
                    ↪ vmill_tool_ready == false",
                "actions": [],
                "nextState": "untitled_state_4"
            }
        ]
    },
    "onEnter": {
        "events": [
            {
                "eventName": "vrot",
                "condition": "true",
                "actions": [
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco12",
                            "payload": {
                                "contentExpression": "0",
                                "type": "STRING"
                            }
                        }
                    },
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco14",
                            "payload": {
                                "contentExpression": "0",
                                "type": "STRING"
                            }
                        }
                    },
                    {
```

```
                        "iotTopicPublish": {
                            "mqttTopic": "plco16",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    }
                ]
            }
        ]
    },
    "onExit": {
        "events": []
    }
},
{
    "stateName": "untitled_state_4",
    "onInput": {
        "events": [],
        "transitionEvents": [
            {
                "eventName": "vmbotchck2",
                "condition": "$input.plant_start.
                    ↪ vmill_bottom == false",
                "actions": [],
                "nextState": "untitled_state_5"
            }
        ]
    },
    "onEnter": {
        "events": [
            {
                "eventName": "2ndtime",
                "condition": "true",
                "actions": [
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco14",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    },
                    {
```

```
                        "iotTopicPublish": {
                            "mqttTopic": "plco13",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    }
                ]
            }
        ]
    },
    "onExit": {
        "events": []
    }
},
{
    "stateName": "untitled_state_5",
    "onInput": {
        "events": [],
        "transitionEvents": [
            {
                "eventName": "vmtopchk2",
                "condition": "$input.plant_start.
                    ↪ vmill_top == false",
                "actions": [],
                "nextState": "untitled_state_6"
            }
        ]
    },
    "onEnter": {
        "events": [
            {
                "eventName": "vmbottotop",
                "condition": "true",
                "actions": [
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco13",
                            "payload": {
                                "contentExpression": "0",
                                "type": "STRING"
                            }
                        }
                    },
                    {
```

```
                        "iotTopicPublish": {
                            "mqttTopic": "plco12",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    }
                ]
            }
        ]
    },
    "onExit": {
        "events": []
    }
},
{
    "stateName": "untitled_state_6",
    "onInput": {
        "events": [],
        "transitionEvents": [
            {
                "eventName": "vmtopready2",
                "condition": "$input.plant_start.
                    ↪ vmill_tool_ready == false",
                "actions": [],
                "nextState": "untitled_state_7jhg"
            }
        ]
    },
    "onEnter": {
        "events": [
            {
                "eventName": "vmtopwait",
                "condition": "true",
                "actions": [
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco12",
                            "payload": {
                                "contentExpression": "0",
                                "type": "STRING"
                            }
                        }
                    },
                    {
```

```
                        "iotTopicPublish": {
                            "mqttTopic": "plco14",
                            "payload": {
                                "contentExpression": "0",
                                "type": "STRING"
                            }
                        }
                    },
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco16",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    }
                ]
            }
        ]
    },
    "onExit": {
        "events": []
    }
},
{
    "stateName": "untitled_state_7jhg",
    "onInput": {
        "events": [],
        "transitionEvents": []
    },
    "onEnter": {
        "events": [
            {
                "eventName": "vmill",
                "condition": "true",
                "actions": [
                    {
                        "iotTopicPublish": {
                            "mqttTopic": "plco14",
                            "payload": {
                                "contentExpression": "1",
                                "type": "STRING"
                            }
                        }
                    }
```

```
                        ]
                    }
                ]
            },
            "onExit": {
                "events": []
            }
        }
    ],
    "initialStateName": "startoprn"
},
"detectorModelDescription": null,
"detectorModelName": "fffinal",
"evaluationMethod": "SERIAL",
"key": null,
"roleArn": "arn:aws:iam::768753832381:role/service-role/
    ↪ operator"
}
```