Digital System Design PA18

IOT REMOTE WEATHER STATION AND CENTRALIZED CONTROL CENTER

Javana Muhammad Dzaki	2306161826
Benedict Aurelius	2306209095
Syahmi Hamdani	2306220532
Muhamad Rey Kafaka Fadlan	2306250573

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

DEPARTEMEN TEKNIK ELEKTRO

TABLE OF CONTENTS

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

DEPARTEMEN
TEKNIK
ELEKTRO

Introduction

In-depth dive into the reasoning behind our project and to be achieved end goals

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia
DEPARTEMEN TEKNIK ELEKTRO

Background

Indonesia, as a disaster-prone country located on the Pacific Ring of Fire, recorded 1,300 disasters as of September 2024. To address this challenge, a more advanced and responsive system is needed, such as the development of IoT-based Weather Stations integrated with a control center. This technology enables real-time weather monitoring with high accuracy using sensors for temperature, humidity, and light intensity. The data transmitted directly to the control center can be used to detect weather patterns, predict disasters, and provide early warnings to reduce the impact of disasters.

Through historical data collection and continuous monitoring, this system can improve disaster prediction accuracy and support more effective mitigation efforts. The implementation of IoT technology and a centralized control system is expected to build a strong disaster management ecosystem, reducing risks, economic losses, and saving more lives in the future.

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia
DEPARTEMEN TEKNIK ELEKTRO

Project Description

The IoT Weather Station project is designed to monitor weather parameters such as humidity, temperature, and daylight intensity. The humidity sensor calculates the air's moisture level based on the readings of water mass and volume produced. The temperature sensor measures the environmental temperature in Celsius. The daylight sensor determines whether it is currently day or night. This data is automatically collected through sensors, processed, and sent to the control center using a Finite State Machine (FSM) mechanism. The system can read data from I/O files, process it, and store the observation results in CSV format to facilitate analysis and reporting.

Objectives

- Developing a weather monitoring system based on FSM.
- Developing an encoding mechanism capable of converting sensor data into 64-bit data packets.
- Designing a system that can read and execute instructions from the control center using an instruction port.

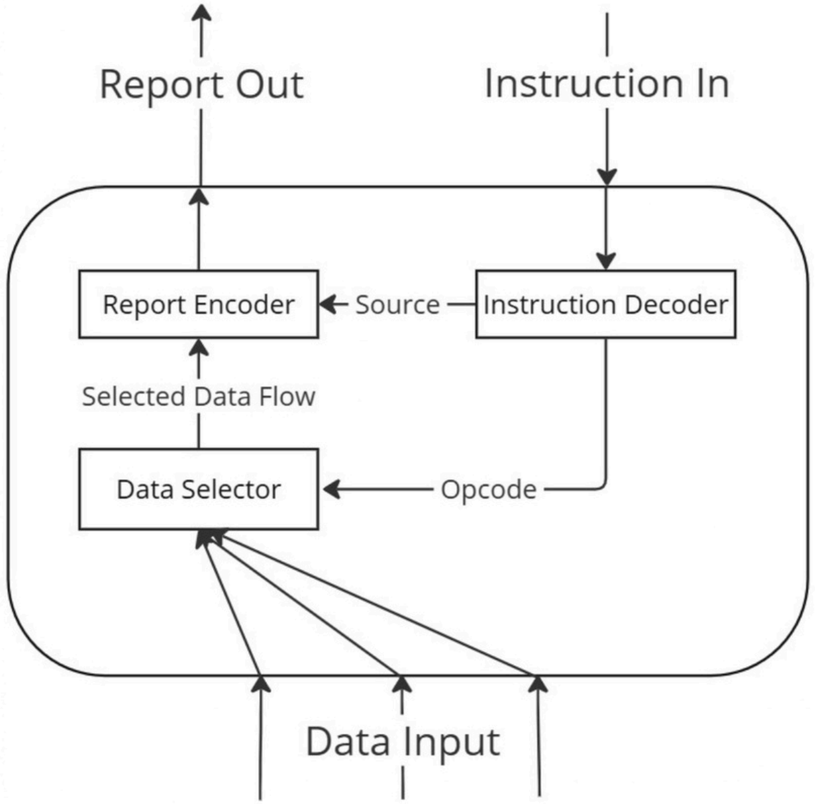
UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

DEPARTEMEN
TEKNIK
ELEKTRO

Implementation

Technical explanation of our project in VHDL,
system overview and operation details

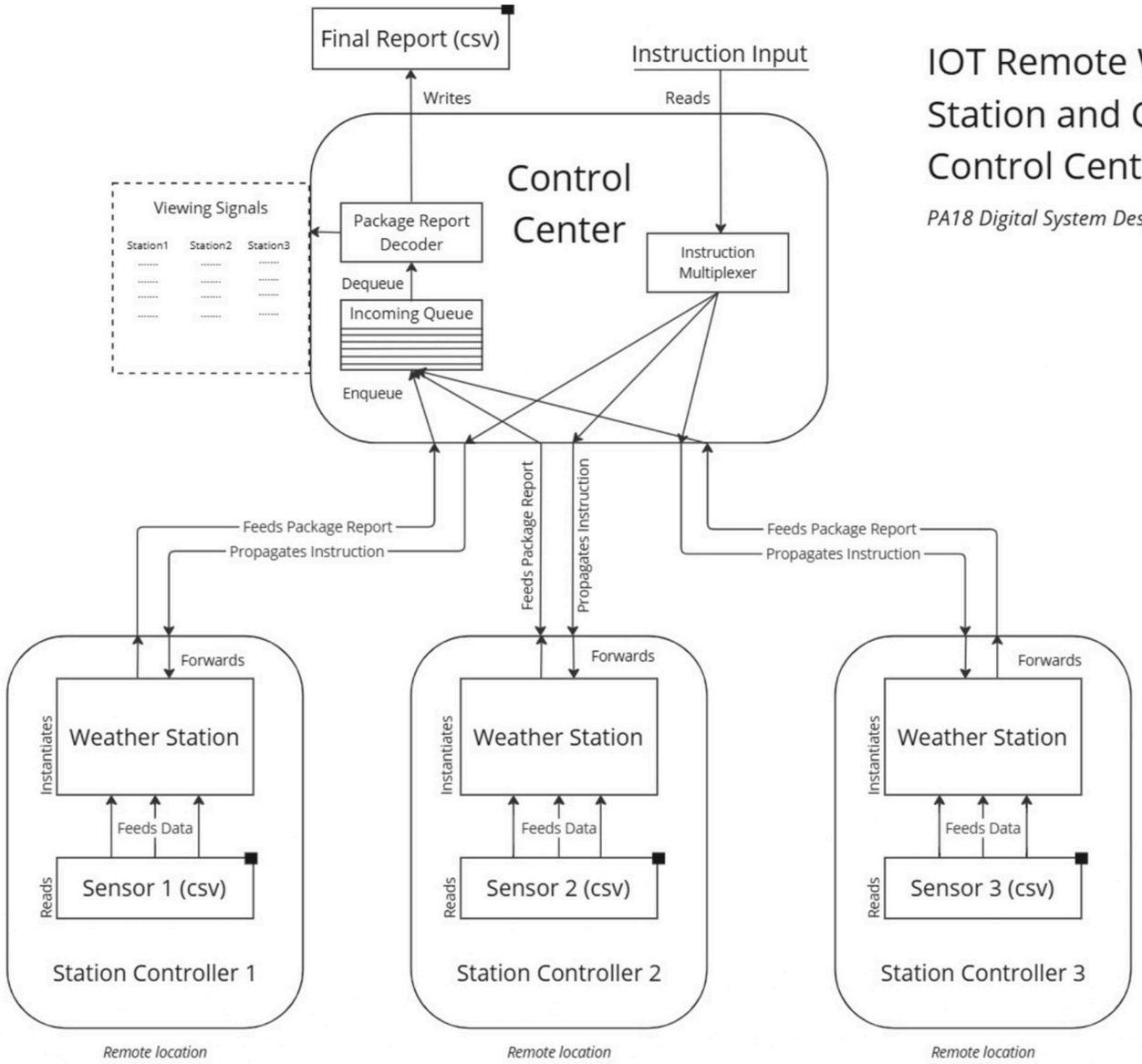
Schematics



The inner workings of a Weather Station

PA18 Digital System Design

Schematics



IOT Remote Weather Station and Centralized Control Center

PA18 Digital System Design

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia
DEPARTEMEN TEKNIK ELEKTRO

Components

- **Control Center**: a component that acts as the central hub for monitoring station controllers 1 through 3 and is capable of providing instructions to the stations. As the central control, the control center decodes package reports received from the station controllers, forwards them to the viewing signal, and logs them in a final report CSV. **Control Center Testbench** tests the inner workings of Control Center and ensure it complies with the specification. it applies complex packet report queue and dequeue decoding system to handle simultaneous incoming reports.

- There are three entities that control the stations: **station_controller_1**, **station_controller_2,** and **station_controller_3**. Each station is responsible for reading sensor data from a CSV file (**sensor_1**, **sensor_2**, and **sensor_3**). This data is then forwarded to the Weather Station for processing.

- **Weather Station**: a component that functions as a state machine to process data from the sensors. It encodes sensor data, processes operational instructions, and generates report packages containing information such as the data source, system status, opcode, timestamp, and sensor data.

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia
DEPARTEMEN TEKNIK ELEKTRO

Operation Codes

Opcode	Instruction	Description
000001	Idle	Idle
000010	Run	All sensors running
000011	Run but stop temp	All sensors running except temperature
000100	Run but stop daylight	All sensors running except light
000101	Run but stop moist	All sensors running except moisture
000110	Run temp only	Only temperature active
000111	Run daylight only	Only light active
001000	Run moist only	Only moisture active

Encoding System

AA BB CCCCCC DDDDDD E(16) F(16) G(16)

Bit	Data
63 - 62 (2 bit)	Source Station ID
61 - 60 (2 bit)	Status
59 - 54 (6 bit)	Opcode
53 - 48 (6 bit)	Timestamp
47 - 32 (16 bit)	Data temp
31 - 16 (16 bit)	Data light
15 - 0 (16 bit)	Data moist

HH II J KKKKKKKKKKK

Bit	Description
HH	Sensor’s Type
II	Status
J	Signed (Negative/Positive)
K (12 Bit)	Signed Integer Value

Instruction

AA BBBBBB

Bit	Description
AA	Station Selector
BBBBBB	Operation Code

Other Details

Status	Description
00	No Status
01	Running
10	Stopped

Sensor Type	Description
00	No Type
01	Temperature
10	Daylight
11	Moisture

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

DEPARTEMEN
TEKNIK
ELEKTRO

Periherals

Miscellaneous elements that supplements and assist our project implementation

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia
DEPARTEMEN TEKNIK ELEKTRO

Raw Data Source

We use a CSV file to transmit the data source to the weather station. The data includes temperature in degrees Celsius, daylight, which contains a true or false value represented by 0 or 1 to classify whether it is day or night, as well as water mass and water volume to calculate the air humidity level at that time in the respective area.

	A	B	C	D
1	Suhu	Daylight	Mass_H2O	Volume
2	25	1	15	2
3	28	0	20	1
4	30	1	25	1
5	27	1	18	2
6	32	0	22	1
7	26	1	17	2
8	29	1	21	1
9	35	0	24	1
10	33	1	19	2
11	31	0	23	1
12	28	1	16	2
13	30	1	22	1
14	34	0	25	1
15	25	1	18	2
16	29	1	20	1
17	32	0	23	1
18	27	1	19	2
19	31	0	24	1
20	28	1	17	2
21	33	0	21	1
22	35	1	25	1
23	29	0	22	1
24	26	1	16	2

Calculation Formulas

As mentioned above, we will calculate the air humidity level from the obtained water mass and water volume. This can be calculated using the formula:

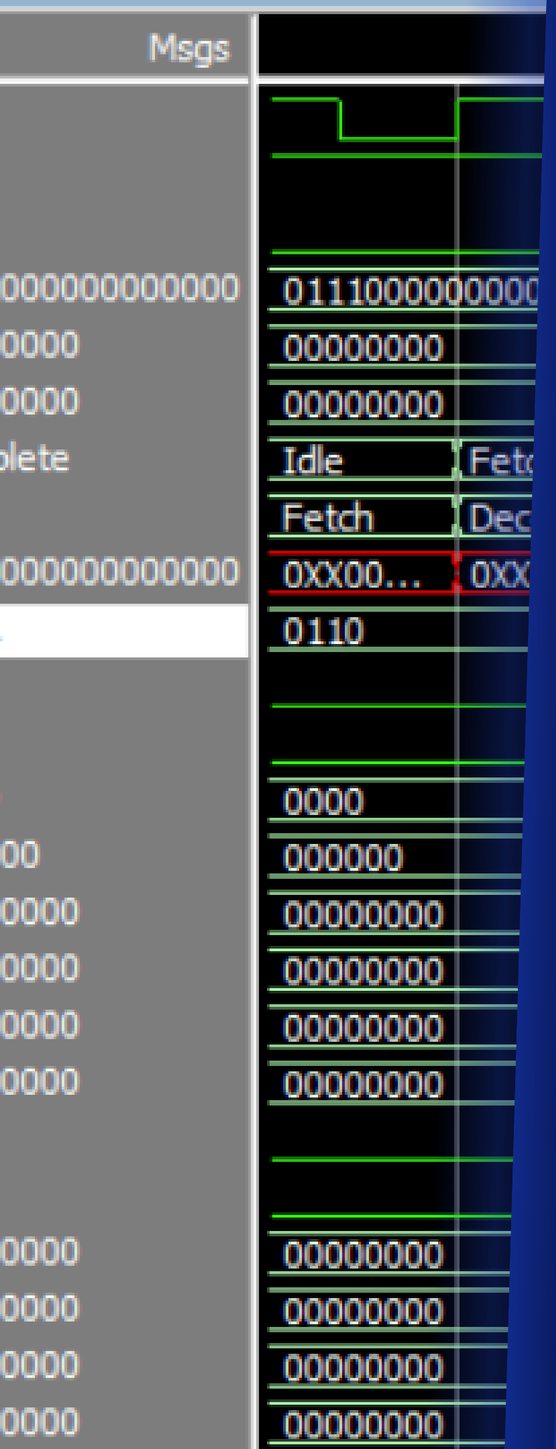
moisture = (water mass * 100) / water volume

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

DEPARTEMEN
TEKNIK
ELEKTRO

Testing & Analysis

Project-component benchmarking for performance analysis and testing

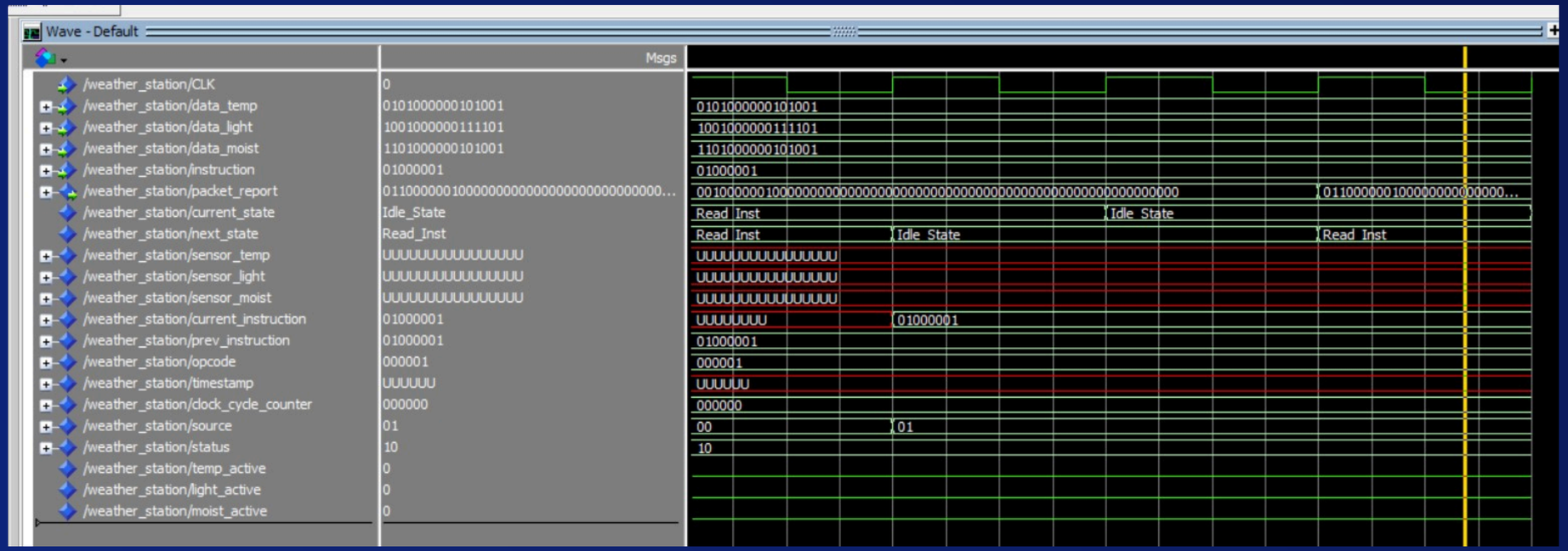


System Testing

After completing the design and implementation, the next step is testing. In this project, testing involves three types: wave test, FSM, and CSV. The wave test is a commonly used method in digital design testing. FSM (Finite State Machine) is used to ensure that the operational states of components function correctly. Meanwhile, CSV is a data format in databases where each entry is separated by a comma (,) or a semicolon (;). The testing results are then recorded and analyzed.

Results - Weather Station

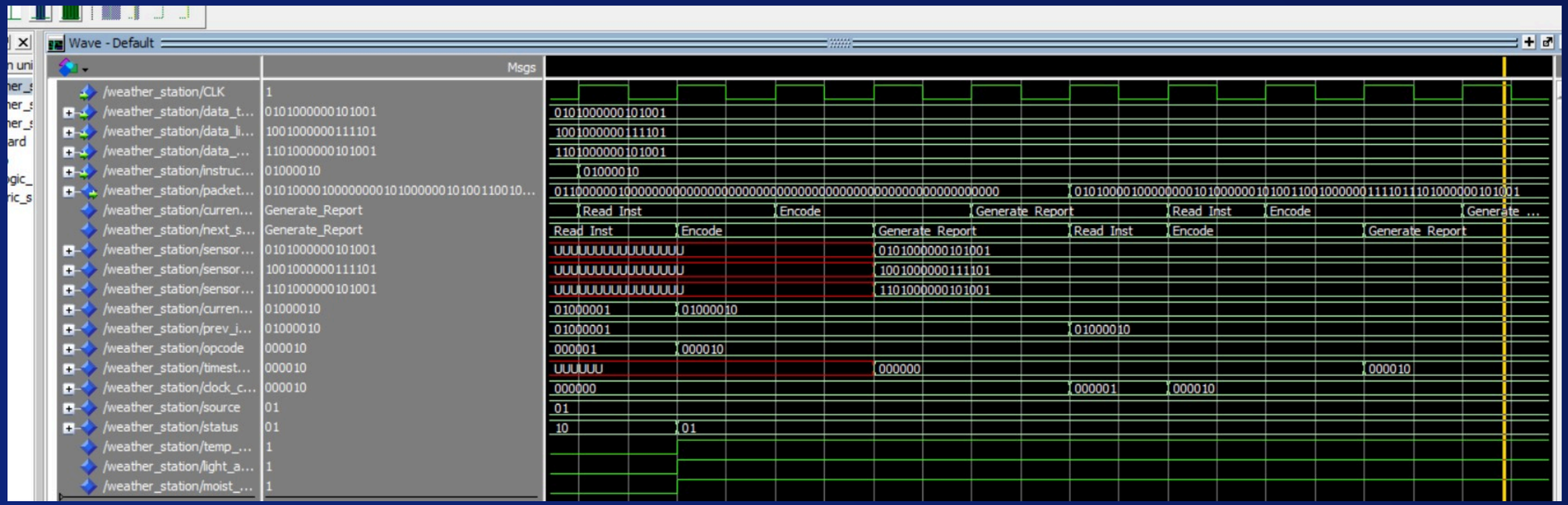
Instruction 01000001



At startup, the weather station enters Idle mode. The instruction provided is 01000001 (Source: 01, Opcode: 000001 for Idle). This creates a Read-Idle loop in the weather station, where it waits for the next instruction.

Results - Weather Station

Instruction 01000010



After receiving the instruction 01000010 (Source: 01, Opcode: 000010 for Run All Sensors), the weather station begins reading data from all sensor ports and performs encoding as well as delivering a 64-bit packet report. Additionally, it uses the "Source" field as the station's identifier and increments the refresh cycle count.

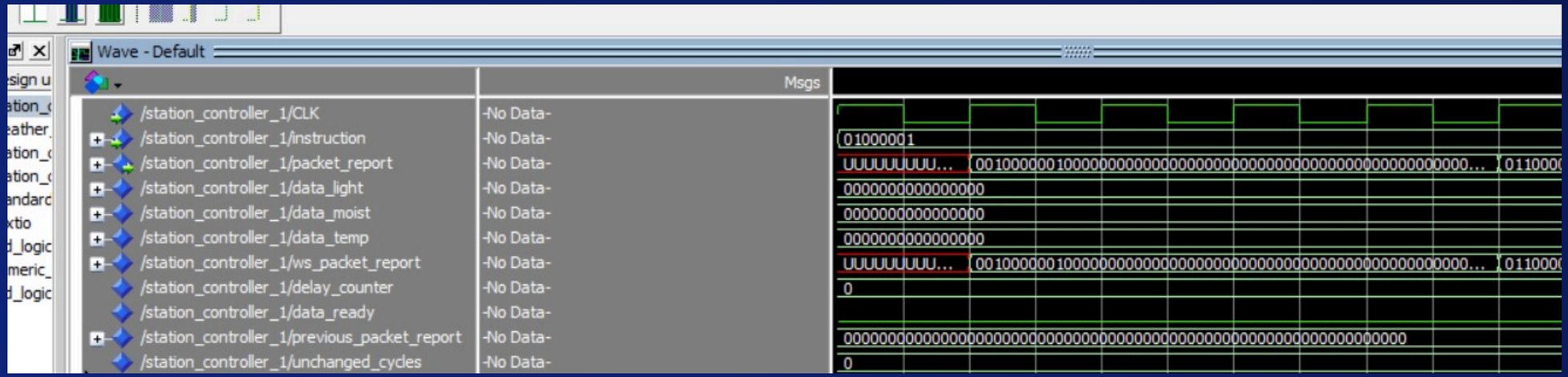
Results - Weather Station

Instruction 01000011

When the instruction 01000011 (Source: 01, Opcode: 000011 for Run Without Temperature) is received, the packet report proceeds as usual. However, the 12-bit temperature data section consistently shows 0, and its status is set to 10 (stopped).

Results - Station Controller

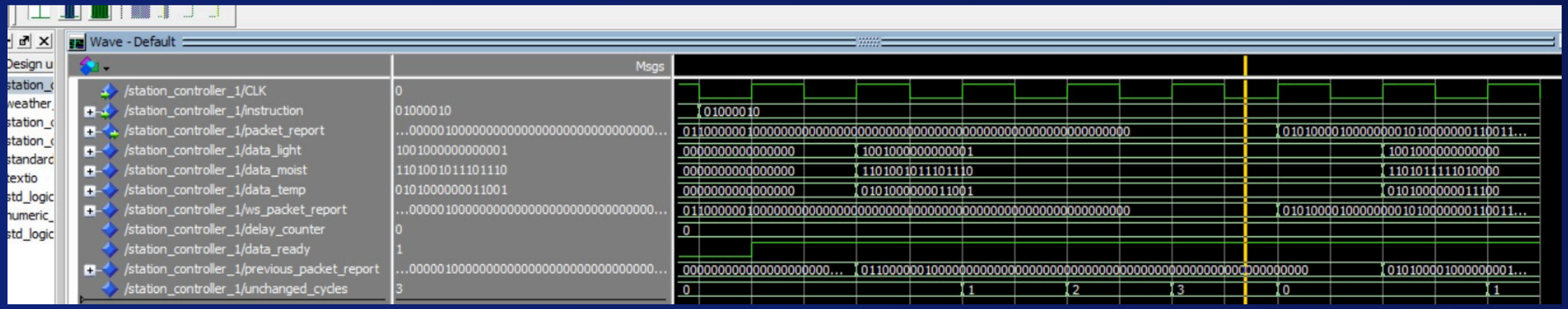
Instruction 01000001



At startup, the weather station is instantiated by the station controller and undergoes port mapping. When the instruction 01000001 (Source: 01, Opcode: 000001 for Idle) is issued, the instruction is propagated to the weather station. Meanwhile, the controller itself delays CSV reading.

Results - Station Controller

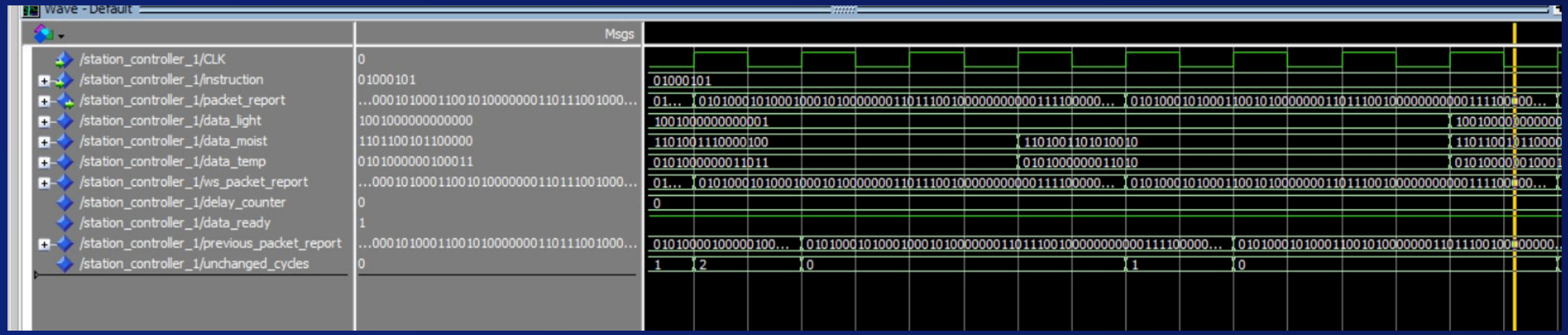
Instruction 01000010



When the controller receives the instruction 01000010 (Source: 01, Opcode: 000010 for Run All), it reads the CSV file and passes the data to the weather station. The weather station then encodes the data according to the opcode and generates a package report.

Results - Station Controller

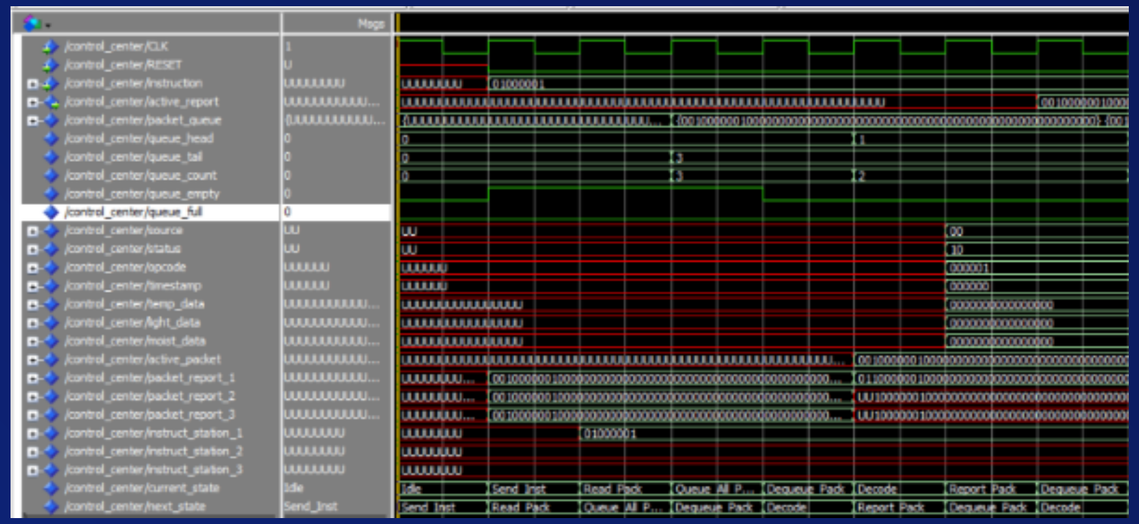
Instruction 01000101



When the controller receives the instruction 01000101 (Source: 01, Opcode: 01000101 for Run But Stop Moist), it reads the CSV file and forwards the data to the weather station. However, the moisture sensor is stopped, with its status set to 10, and the moisture data section consistently reads 0. Additionally, synchronization of CSV sensor readings is observed, with a delay of 3 clock cycles each time the report changes.

Results - Control Center

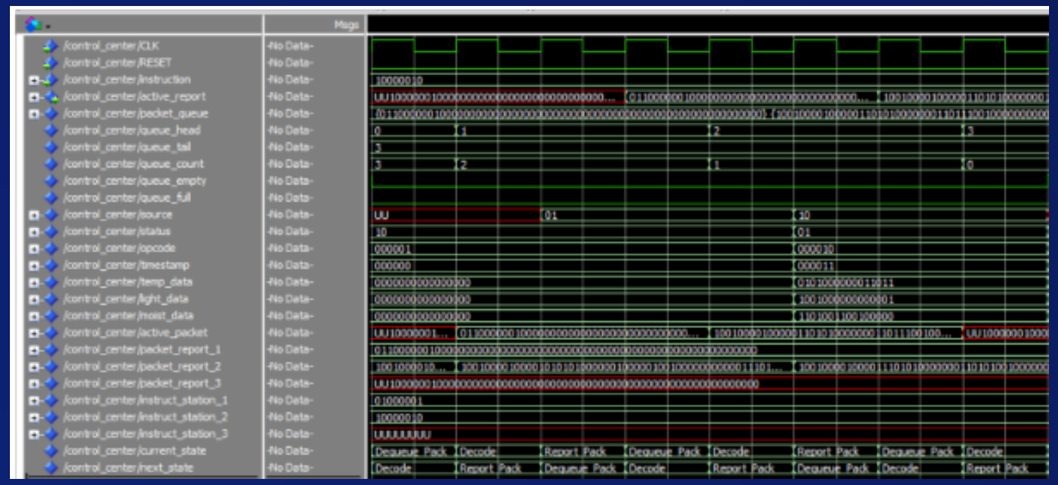
Instruction 01000001



When the control center is first initialized, the state is idle. When an instruction is received from the reporter/testbench of control center, the state will move to send_inst (send instructions to weather station and it's controller). Then, after weather station sends the report packet back, the weather station will move to read packet state and then put them to a packet queue. Because station 1 is told to idle here, the packet report will be a blank one.

Results - Control Center

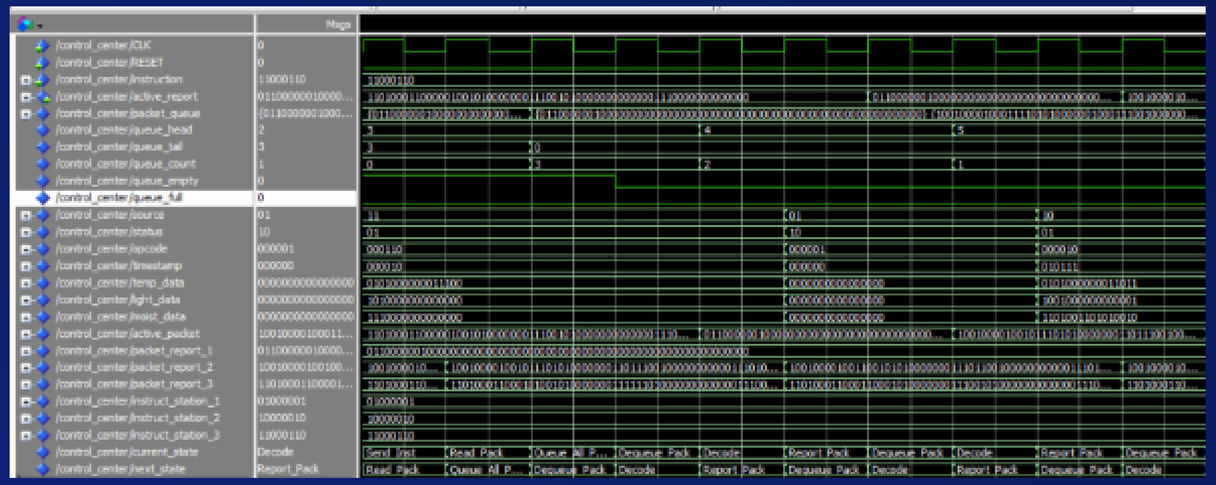
Instruction 10000010



Here, the control center gives instruction 10000010, where station 2 will run all sensors and report the encoded packet to the control center. Here after the instruction is given, the weather station will start to run sensors and generate the packet.The packet received by control center will then be put on a packet queue, before finally reporting.

Results - Control Center

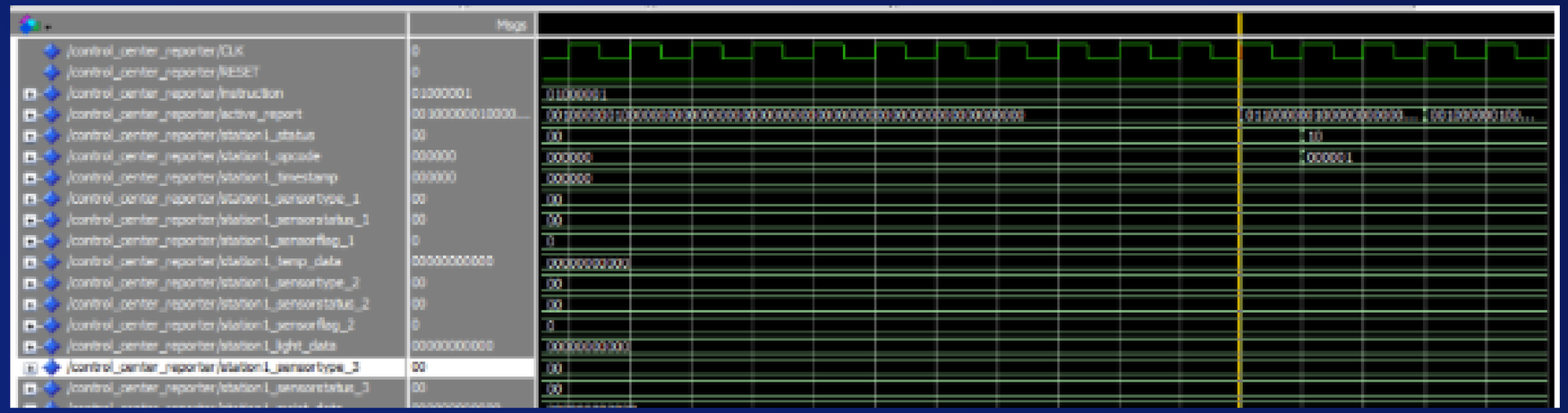
Instruction 11000110



Here control center gives out another instruction. This time for station 3, with the opcode is 11000110, where only the temperature sensor will be active. After the instruction is given, the weather station will generate packet and feeds it back to control center. The packet will be read and be put on a queue, before the process of dequeueing, decoding, and finally be able to reported and written on a CSV output file.

Results - Reporter (tb)

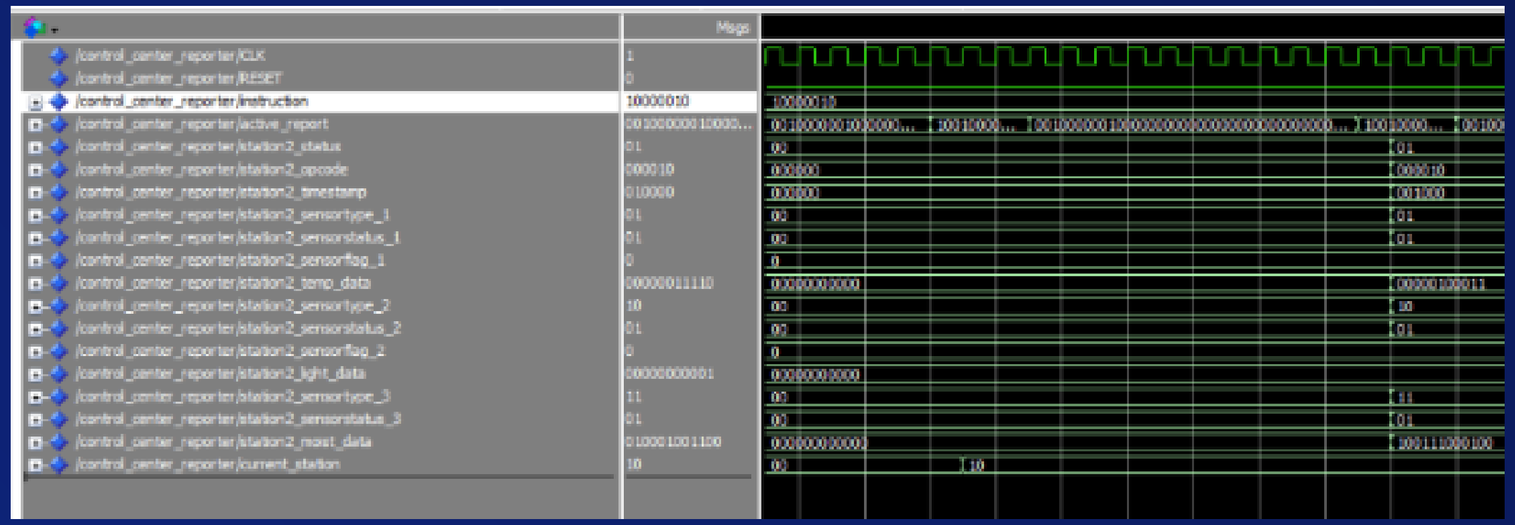
Instruction 01000001



When starting the reporter (testbench process), the signal on the control center reporter is not updated yet. Once given instruction of 01000001 (Source: Station 1, Opcode: 000001, Go Idle), station 1 will stay at idle state until a run instruction is givien by the control center reporter. Other than that, the status signal for station 1 in the reporter will be updated to idle (10).

Results - Reporter (tb)

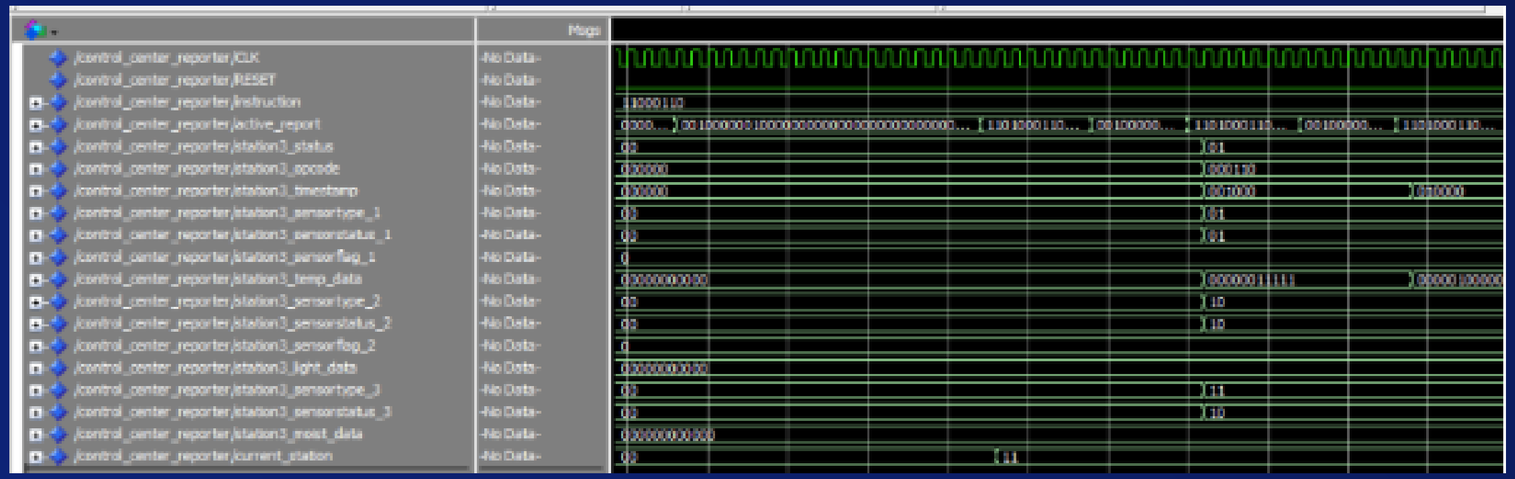
Instruction 10000010



After entering the instruction 10000010 (Source: Station 2, Opcode: 000010, Run all sensors), station 2 will start to run all sensors. The signal for station 2 in the reporter will also be updated. It is shown that the active report is always changing. This is because control center will one by one put the report packets from station 1 to 3 into a packet queue, where later it will be dequeued in which the dequeued packet will be the active report.

Results - Reporter (tb)

Instruction 11000110



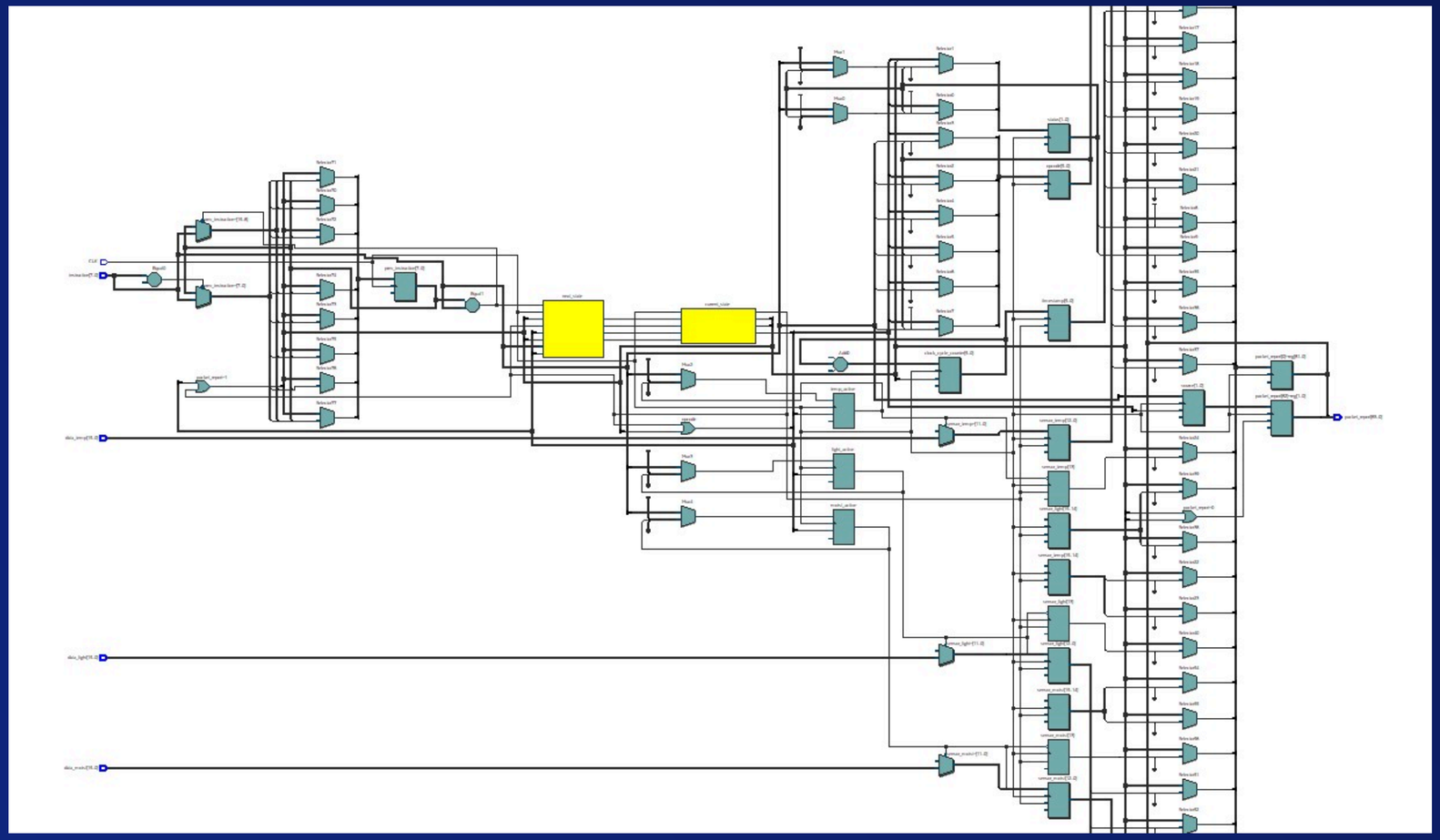
Lastly the reporter will hand out instruction 11000110 (Source: Station 3, Opcode: 000110, Run Temp Only). Here, station 2 will only run it's temperature sensor only and generate a packet that only has the temperature sensor as active. Same as in previous instruction, the active report will one by one change from station 1-3 reports, following the system of ordered queue and dequeue.

Results - Reporter (tb)

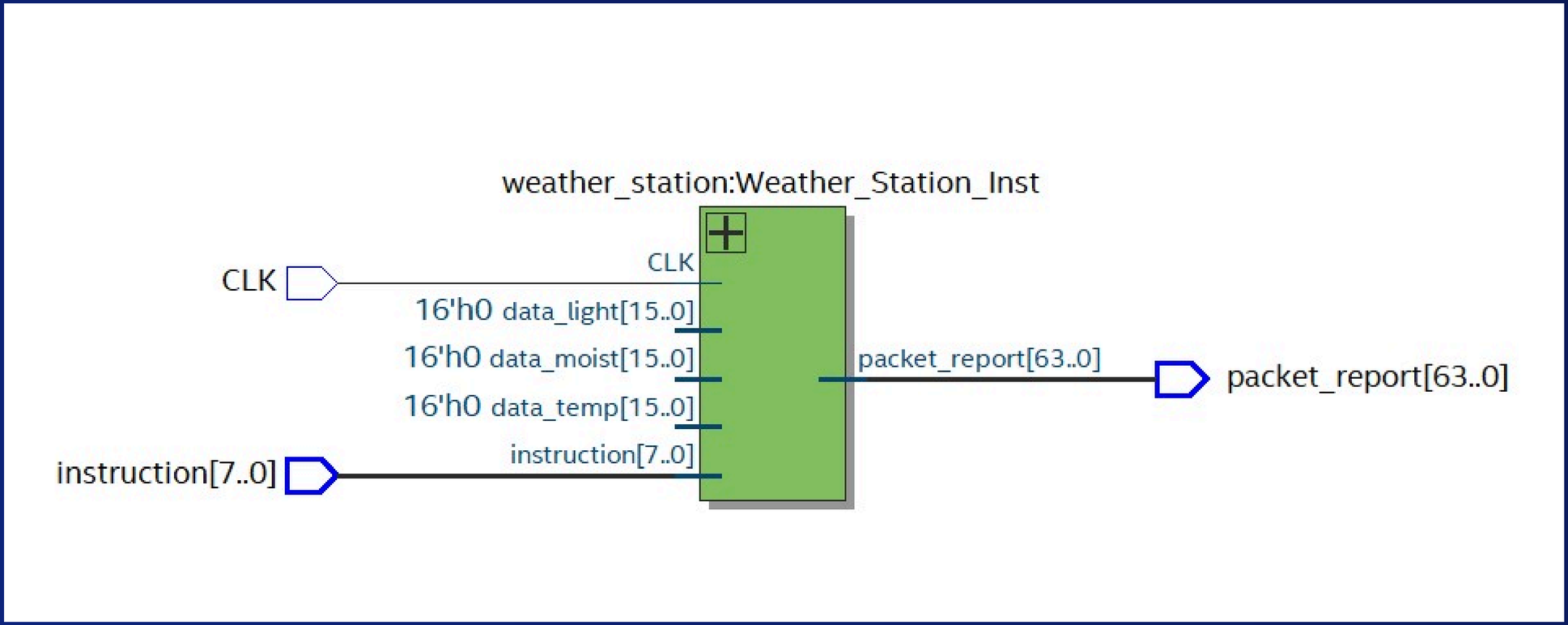
Source	Status	Opcode	Timestamp	SensorType1	SensorStatus1	SensorFlag1	Temp Data	SensorType2	SensorStatus2	SensorFlag2	Light Data	SensorType3	SensorStatus3	Moisture Data
1	Idle	1	0	Unknown	No Status	0	0	Unknown	No Status	0	Night	Unknown	No Status	0
2	Running	2	2	Temperature	Enabled	0	28	Light	Enabled	0	Night	Moisture	Enabled	850
1	Idle	1	0	Unknown	No Status	0	0	Unknown	No Status	0	Night	Unknown	No Status	0
2	Running	2	8	Temperature	Enabled	0	35	Light	Enabled	0	Night	Moisture	Enabled	2500
3	Running	6	2	Temperature	Enabled	0	28	Light	Disabled	0	Night	Moisture	Disabled	0
1	Idle	1	0	Unknown	No Status	0	0	Unknown	No Status	0	Night	Unknown	No Status	0
2	Running	2	16	Temperature	Enabled	0	30	Light	Enabled	0	Day	Moisture	Enabled	1100
3	Running	6	8	Temperature	Enabled	0	31	Light	Disabled	0	Night	Moisture	Disabled	0
1	Idle	1	0	Unknown	No Status	0	0	Unknown	No Status	0	Night	Unknown	No Status	0
2	Running	2	22	Temperature	Enabled	0	26	Light	Enabled	0	Night	Moisture	Enabled	1900
3	Running	6	16	Temperature	Enabled	0	32	Light	Disabled	0	Night	Moisture	Disabled	0
1	Idle	1	0	Unknown	No Status	0	0	Unknown	No Status	0	Night	Unknown	No Status	0
2	Running	2	28	Temperature	Enabled	0	35	Light	Enabled	0	Day	Moisture	Enabled	2200
3	Running	6	22	Temperature	Enabled	0	31	Light	Disabled	0	Night	Moisture	Disabled	0

The image above is the output of the writing done by the reporter component to a csv file named control_center_report.csv. As we can see, real-time packet report from all three stations is written in order. We can also see that station 1 is on the idle state, station 2 runs all sensors, and station 3 runs temperature sensor only.

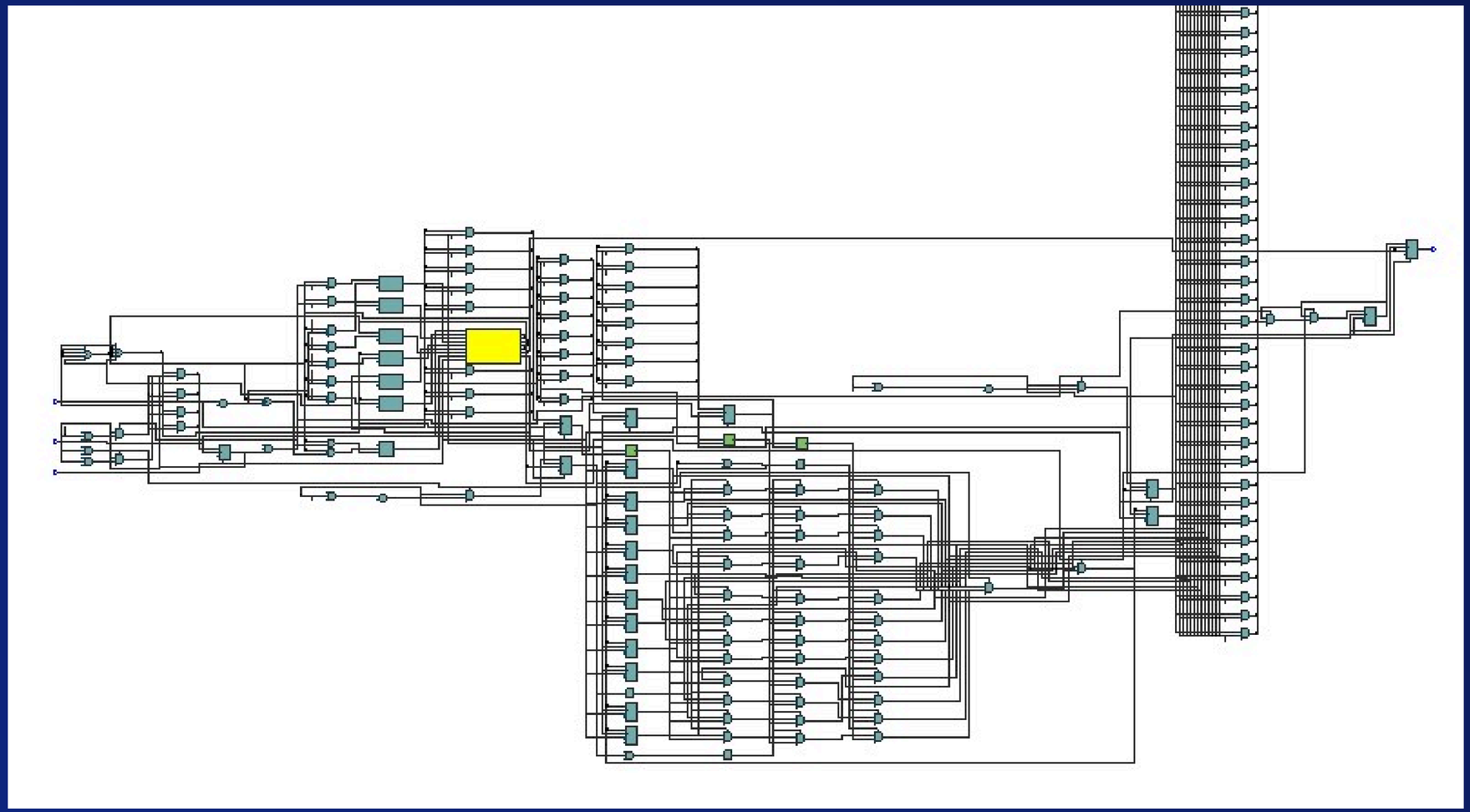
RTL Design - Weather Station



RTL Design - Station Controller



RTL Design - Control Center



UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

DEPARTEMEN
TEKNIK
ELEKTRO

Closing

Project conclusion, documentation, and VHDL component source repositories

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia
DEPARTEMEN TEKNIK ELEKTRO

Conclusion

The project titled IoT Remote Weather Station & Centralized Control Center, which our team has designed, has been successfully executed with satisfying results. This IoT Weather Station & Control Center is also capable of implementing modules and theories of digital system design. The processes of sensor reading, decoding, and reporting have also been implemented effectively. Overall, the project titled IoT Remote Weather Station & Centralized Control Center has run successfully, and we hope it can contribute to the development of more advanced and responsive systems for monitoring and managing disasters in Indonesia.

Documentation

GitHub Repository

https://github.com/javendzk/FinalProjectPSD-PA18

Project Report

https://docs.google.com/document/d/1ZEUtrkhklmSTeEz2k1lhla97zF-vLWoD6fMVpV34-uA

REFERENCES

Ricardo-Jasinski, “Ricardo-Jasinski/VHDL-CSV-file-reader: VHDL package for reading formatted data from comma-separated-values (CSV) files,” GitHub, https://github.com/ricardo-jasinski/vhdl-csv-file-reader

“Ada 1.300 Bencana Alam di ri Sampai September 2024, Ini Rinciannya: Databoks,” Pusat Data Ekonomi dan Bisnis Indonesia, https://databoks.katadata.co.id/demografi/statistik/66d7d7a492e96/ada-1300-bencana-alam-di-ri-sampai-september-2024-ini-rinciannya

Digilab, “Digital System Design,” Digilab UI, https://learn.digilabdte.com/books/digital-system-design

“VHDL structural modeling style,” Surf, https://surf-vhdl.com/vhdl-syntax-web-course-surf-vhdl/vhdl-structural-modeling-style/

“Mastering loops in VHDL: Enhancing flexibility and performance in hardware design,” FPGA Insights, https://fpgainsights.com/fpga/mastering-loops-in-vhdl-enhancing-flexibility-and-performance-in-hardware-design

IEEE, Design of encoding and decoding of hamming code based on VHDL. IEEE Xplore, https://ieeexplore.ieee.org/document/9443744

“Implementing a finite state machine in VHDL” All About Circuits, https://www.allaboutcircuits.com/technical-articles/implementing-a-finite-state-machine-in-vhdl/