Lab 2 - Hydroelectric Dam

GreEn-ER Industrial Control systems Sandbox (G-ICS)
2022 - Wago - Asean-Factori

Table of contents

Introduction **2**

Work to do **2**

Control specifications 2

Configuration **3**

Variables configuration 3

Compilation and programming **4**

Program writing 4

Ladder 4

Structured Text 4

Scada Interfacing **5**

HMI Creation 5

Bonus Exercise **5**

Conclusion **5**

Appendices **6**

Introduction

During this lab, you will program a PLC used to command a simplified hydroelectric dam. To do that, you will use *Ladder* and *Structured Text*. You will also create a HMI and simulate the process. This training will be accompanied by a video about the subject of the lab and a correction.

Work to do

During this lab, you will have to :

- Configure the variables
- Write a control program for the PLC
- Create an HMI to visualize the process
- Check the functioning with the simulation

Control specifications

The control specification concerns a system based on a river hydroelectric dam. The goal is to produce electricity with the water flow. There are four water level sensors on the dam :

1. Min_Warning
2. Min_Level
3. Max_Level
4. Max_Warning

When the level is between Min_Level and Max_Level, the water spills out into a penstock (a large pipe that is used to force the flow to be greater than the normal river flow). The water then turns a turbine which drives the generator to produce electricity.

If the water level is below Min_Level, the process continues to run but it can be stopped by an employee. If the water reaches Min_Warning level, there is not enough water, and the water valve must be closed.

If the water level reaches Max_Level, the process continues to run but it can be stopped by an employee. If the water reaches Max_Warning, there is too much water, the excess water must be drained off by a river to avoid breaking the turbine.

A sensor is placed near the alternator to check if there is an error on the generator or the turbine. If there is one, the water must be emptied via a river and no water should go through the penstock.

If someone pushes the "*Emergency Stop*" button, the penstock valve is closed and the water must be drained off into the river.

You can see a scheme of the process in the figure 1 just below.

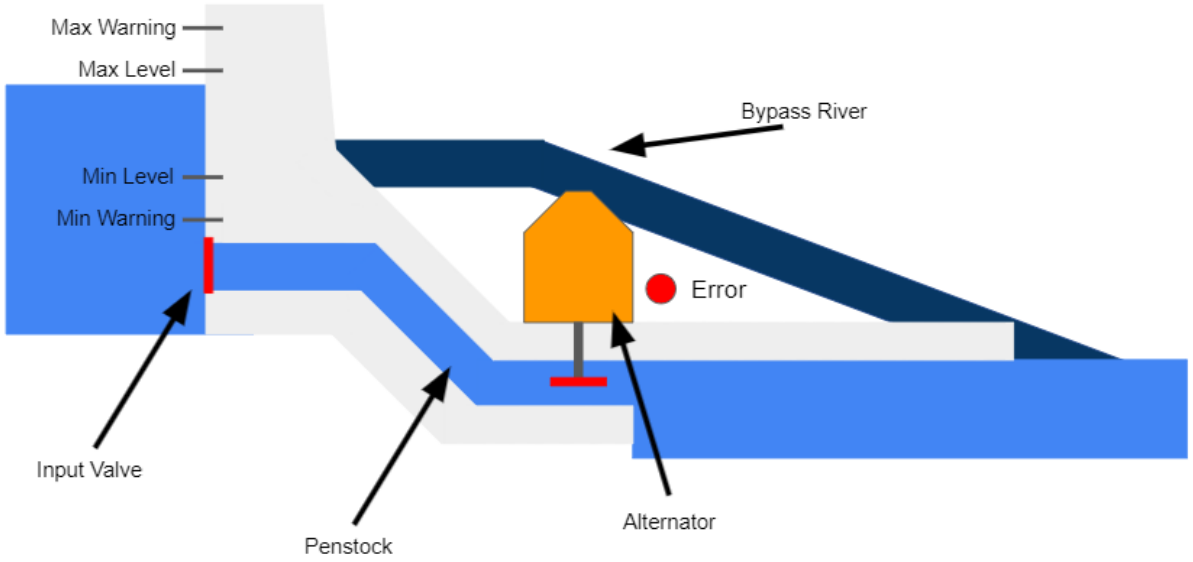


Figure 1 : Scheme of the process

Configuration

As in the last lab, you can use the ***750-8212*** controller, the 16 digital inputs module ***750-1405***, and the 16 digital outputs module ***750-1505***.

Variables configuration

The inputs are as follows :

- Min_Warning → Absolute minimum level of water
- Min_Level → Minimum level of water (needs an acknowledgment to close the penstock valve)
- Max_Level → Maximum level of water (needs an acknowledgment to close the penstock valve)
- Max_Warning → Absolute maximum level of water
- Level_Ack → Button used to acknowledge that the level reached min or max
- Alt_Error → Sensor that detects if there is an error on the alternator
- Em_Stop → Emergency stop

The outputs are as follows :

- Water_Bp → Open the valve to let the water goes out (water bypass)
- Open_Valve → Close the penstock valve

Exercise : Declare the variables autonomously. You can read the lab0 if you don't remember how to do it.

Compilation and programming

Program writing

Ladder

As said in the introduction, you will use *Ladder* and *Structured Text* to program the PLC. As we have two outputs, the first one (*Open_Valve*) will be controlled by a *Ladder* diagram, and the second one (*Water_Bp*) will be controlled by a *ST* program.

Exercise : Try to write the *Open_Valve* command program. It is really similar to the transitions created in the last lab for the grafcet. If you need more ladder tools, you can look in the "*toolbox*" tab of the right-window of the software, in the "*Ladder Elements*" category.

Structured Text

The first part of the program is now completed. In this part, we will see how to use the basics of *Structured Text* language.

To create a ST program, right click on "*application*" in the program structure window, and choose "*POU*". Select ST in the "*Implementation Language*" drop-down menu.

The screen is divided in two parts. At the top, you can create intern variables. We don't use this in this lab but notice it for the next one. Below this window, you have the empty programming window. This is where you will write the ST program.

ST is not so far from C programming. You can use basic functions (If, While, For, Case…). In our case, we will only use "*If*".

In our case, we want to open the bypass valve (Water_Bp = True) when :

- Max Warning is reached
- Max Level is reached and an employee validate by pushing *Level_Ack*
- The Emergency Stop button is pushed

Each of these two cases could appear independently, so we should use an "OR" between them. In the second case, there are two conditions so we should use an "AND". Finally, we can write this program :

```
IF ((Em_Stop = TRUE) OR (Max_Warning = TRUE)) THEN
    Water_Bp := TRUE;
ELSE
    Water_BP := FALSE;
END_IF
```

You must always put an "ELSE" case to avoid the program being blocked in "TRUE". You could also use a "CASE" condition to do the same job. Once it is completed, remember to add your two programs in the "*task configuration*", and compile your project to check for errors.

Scada Interfacing

HMI Creation

To create a visualization for this project, you can create a cross-sectional diagram of the dam, or a top-view diagram. We should see the dam, the alternator, the penstock, and the river. All inputs are represented by buttons. A solution is visible on the appendix 2.

Bonus Exercise

For the moment, we are simulating the potential errors with only one button. In real life, many errors could occur. Change “Alt_Error” so that is it an output, and Write a new ST program which manage two new errors :

- The alternator is not rotating while water should come from the penstock
- There is an overcharge in the alternator

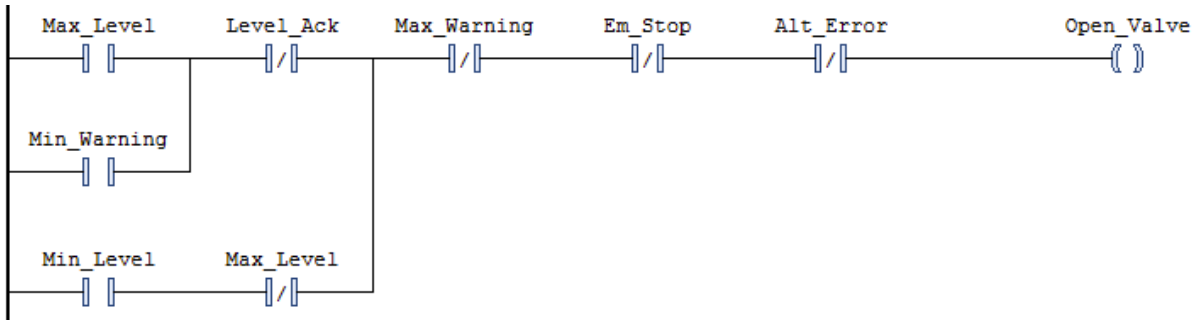
These errors are detected by two sensors (input), and they activate one output (*Alt_Error)*. If necessary, adapt the rest of the program to make it work.

Conclusion

In this lab, you learned how to use two different languages together. In the next lab, we will mix all the skills you learned in the three labs to create a complex industrial process.

Appendices

Appendix 1 : Ladder Solution



Appendix 2 : HMI Solution