Control Systems Technologies

Logic Control Project - Washing Machine

Baraghini Nicholas 0000825055

University of Bologna
Automation Engineering Bachelor Degree
Academic Year : 2019/2020

Contents

1 USER INSTRUCTIONS **1**

1.1 Description of the Machine 1

2 PROJECT STRUCTURE **3**

3 GENERALIZED ACTUATORS **4**

3.1 Definition 4

4 POLICIES **7**

4.1 PREWASH_WASH_POLICY 7

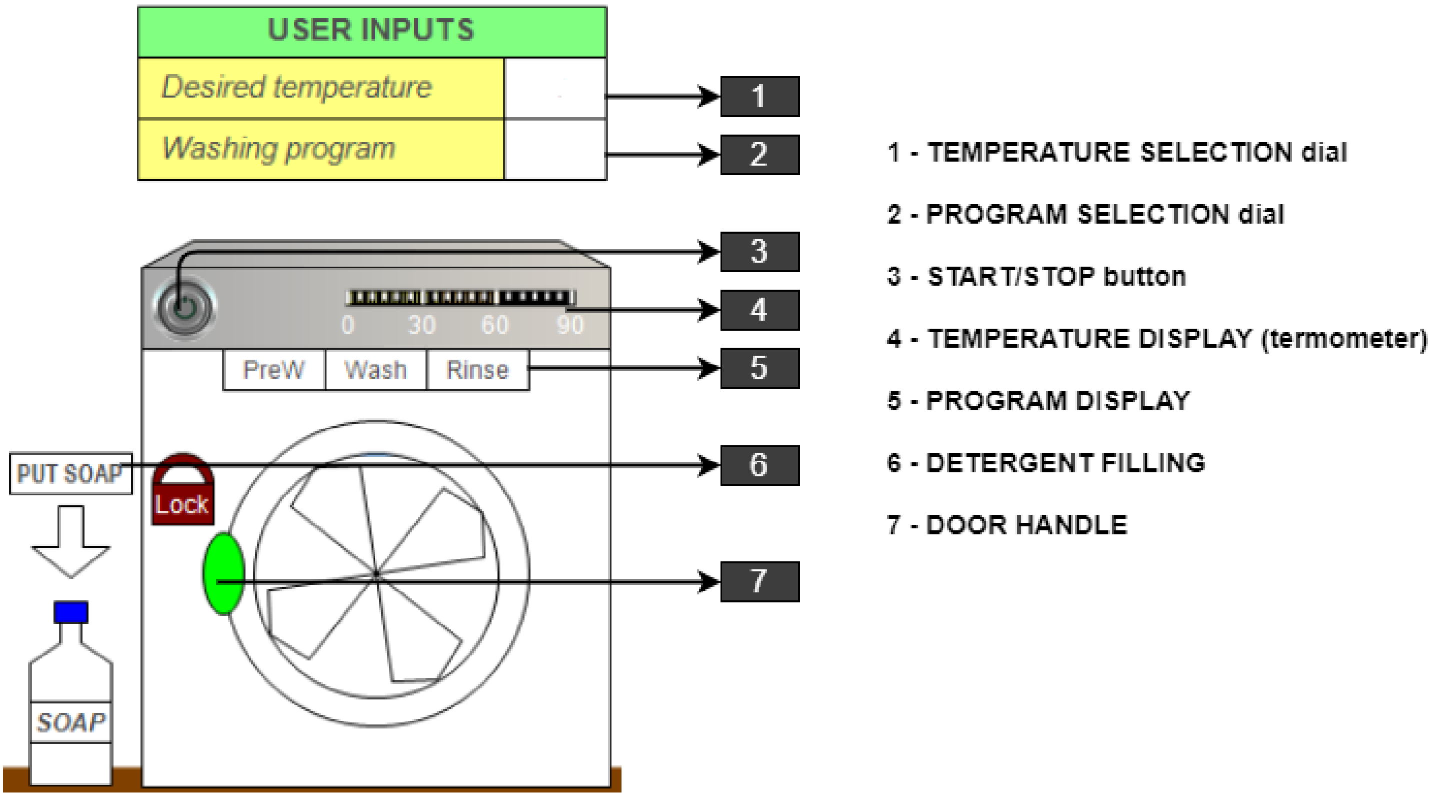
4.2 Main POLICY 10

5 FAULT MANAGEMENT **12**

Chapter 1

USER INSTRUCTIONS

1.1 Description of the Machine



Temperature Selection

The keypad is used to select a desired temperature among 30°C, 60°C and 90°C. Other temperature cannot be selected.

Program Selection

The keypad is used to select the desired program among three different program possible :

1. Pre-wash : the machine will perform a wash procedure without the usage of soap;
2. Wash : the machine will perform a single wash procedure. Soap is required;
3. Pre-wash & Wash : the machine will perform an initial pre-wash and then a wash procedure. Soap is required;

Other programs cannot be selected.

Start/Stop Button

This button allows to start or stop the working of the machine. the machine will start only :

- The temperature and program selected are correct;
- The door is closed;
- If required by the program: the soap case is full;
- The start button is active;
- All these conditions are verified at the same time;

If start/stop button, while it's running, is deactivated, then the machine will start a stop procedure which consist in :

- If the machine is filling the drum, it will finish the water intake procedure and then drain the water and turn off;
- Otherwise the machine will finish the pre-washing or washing procedure, drain the drum avoiding the rinse and turn off;
- If the machine has to perform both, pre-wash and wash, program 3, the machine will end the process in action without performing also the subsequent process avoiding the rinse and turn off;

Temperature and Program display

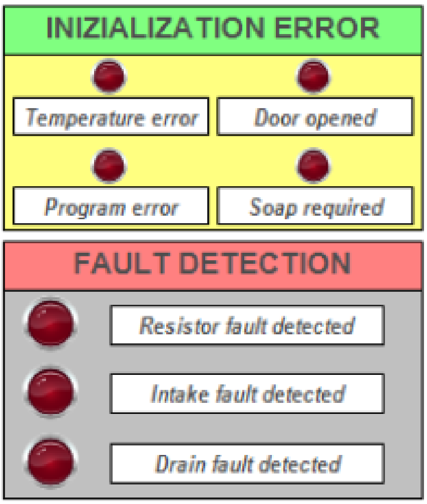
The temperature display is essentially a thermometer which shows the actual temperature of the water inside the machine's drum. The program display is composed by three leds each led assigned to a process of the machine which can be : Pre-wash, Wash or Rinse.

Detergent Filling

By pressing the button assigned to the soap case then the soap container will fill up automatically.

Door Handle

Before starting the door of the bull's eye must be closed. During the running of the machine the door will be locked in order to avoid that it will open during the correct functioning of the washing machine. At the end of the whole washing or pre-washing procedure the bull's eye door will be unlocked to permit the user to take off the clothes inside the drum.



The machine presents also a display of leds subdivided in two categories :

- **Initialization Errors**, which group all the leds assigned to external errors, so then to errors that do not concern with the functioning of the internal component of the machine.
  These leds then provide a 'warning' to the user that the machine cannot start its process because of a lack of information and/or resource and/or an incorrect configuration of its components;
- **Fault Detection**. These leds are related to internal problem of the machine. So with these leds the machine warn the user that it's broken and it cannot performed the task required as desired. Once these leds are on only a direct service from an operator can reset them after having adjusted or replaced the broken component.

Chapter 2

PROJECT STRUCTURE

This project has been developed by a Generalized Actuators and Policies approach. The control code of the washing machine is developed by five Generalized Actuators, four of them controlled by a policy developed only to manage the washing or prewashing process called PREWASH_WASH_POLICY; And the remaining generalized actuator together with the PREWASH_WASH_POLICY are controlled by a general policy which communicate with the HMI. The hierarchical structure of the software elements :

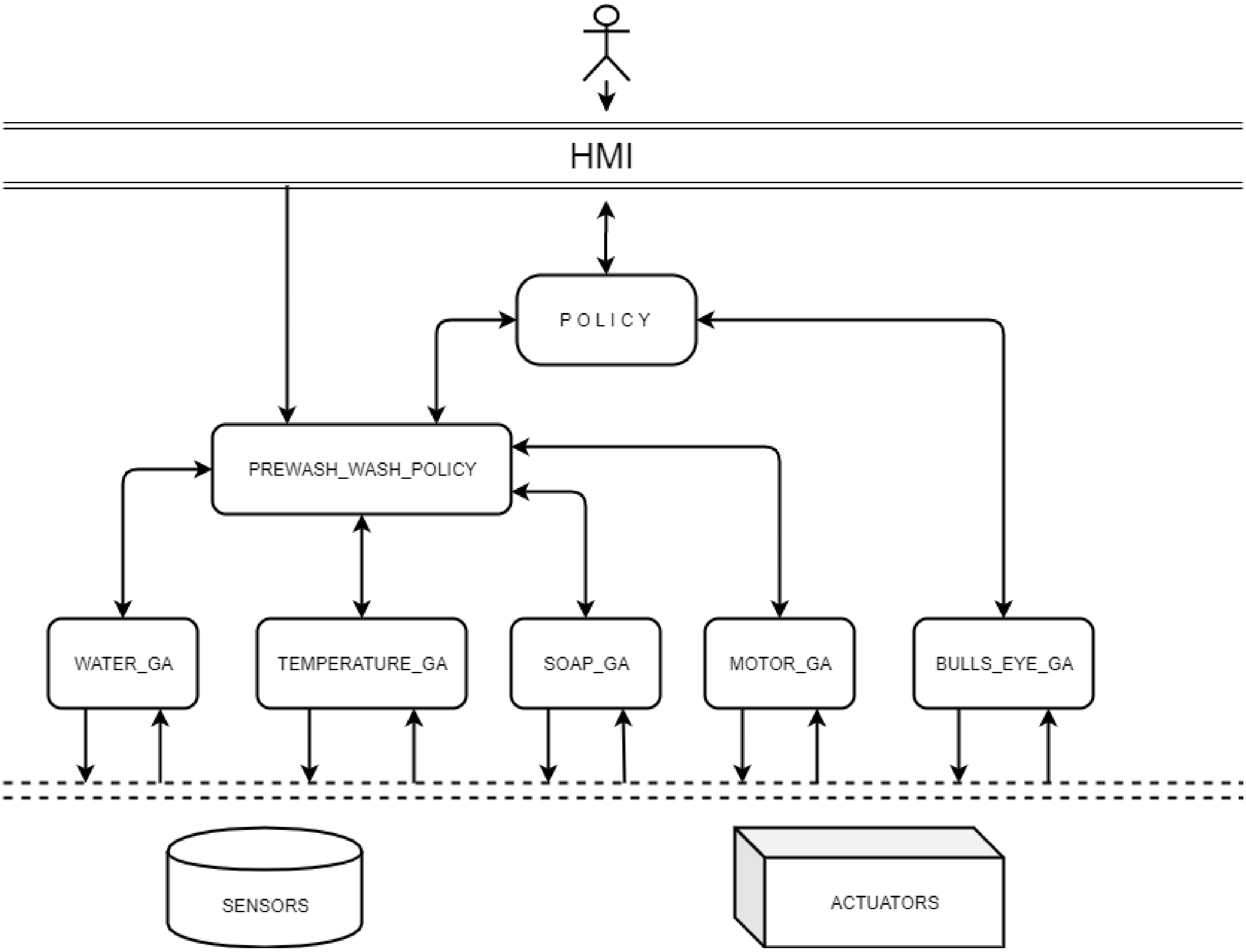


Figure 2.1: Function Block Hierarchy

The GAs and PREWASH_WASH_POLICY have not a direct communication with the HMI (only in case a 'STOP' condition is requested by the user then the order directly goes to the PREWASH_WASH_POLICY);

Chapter 3

GENERALIZED ACTUATORS

3.1 Definition

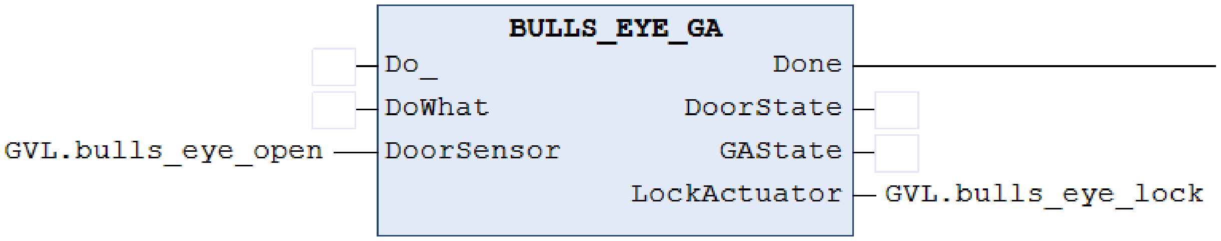


Figure 3.1: Function Block of the BULLS_EYE_GA

BULLS_EYE_GA: this GA manage the control of the bull eye door of the washing machine. It's a Do-Done architecture GA . It receive data from the sensor whose state is referred to the information which tells if the door is open or not. And it manage the actuator linked to the lock of the bull eye's door;

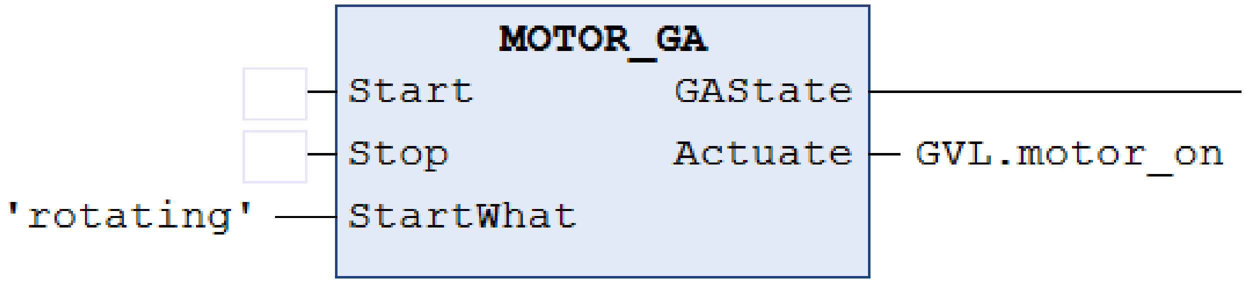


Figure 3.2: Function Block of the MOTOR_GA

MOTOR_GA: it's the GA which has to turn on or turn off the motor of the washing machine drum during the different phases of prewash or wash. It's developed as a Start-Stop GA since the motor does not need to perform any kind of position or speed or acceleration control but it has just to run for a predetermined time interval;

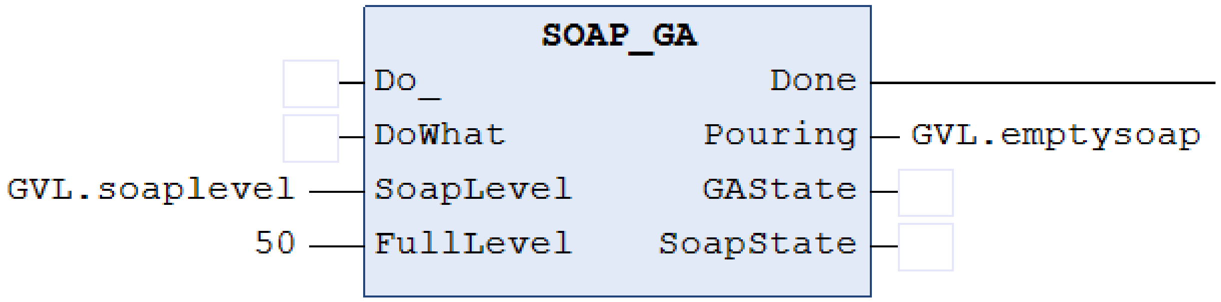


Figure 3.3: Function Block of the SOAP_GA

SOAP_GA: this GA was produced in order to fulfil the task to check the soap level thanks to the sensor which returns back the current level of the soap introduced inside the washing machine; And, moreover, to pour the soap inside the drum of the washing machine during the washing phase by the actuator assigned to empty the soap when turn it on. Since it's a GA which has to perform a non-continuous action the it's developed as a Do-Done GA;

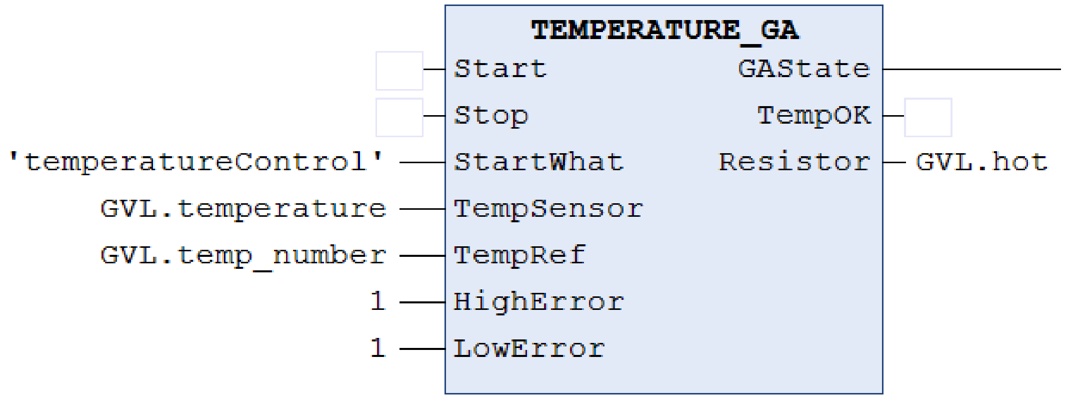


Figure 3.4: Function Block of the TEMPERATURE_GA

TEMPERATURE_GA: through an hysteresis control with an upper and lower bound of one Celsius degree, this GA manage the temperature control of the water inside the drum of the washing machine during its normal working conditions. For this reason it's a Start-Stop GA , which control a temperature sensor and a resistor. The main task of this GA is to maintain the actual temperature as close as possible to the reference temperature, chosen by the user among 30, 60 or 90 Celsius degree, while it's active;

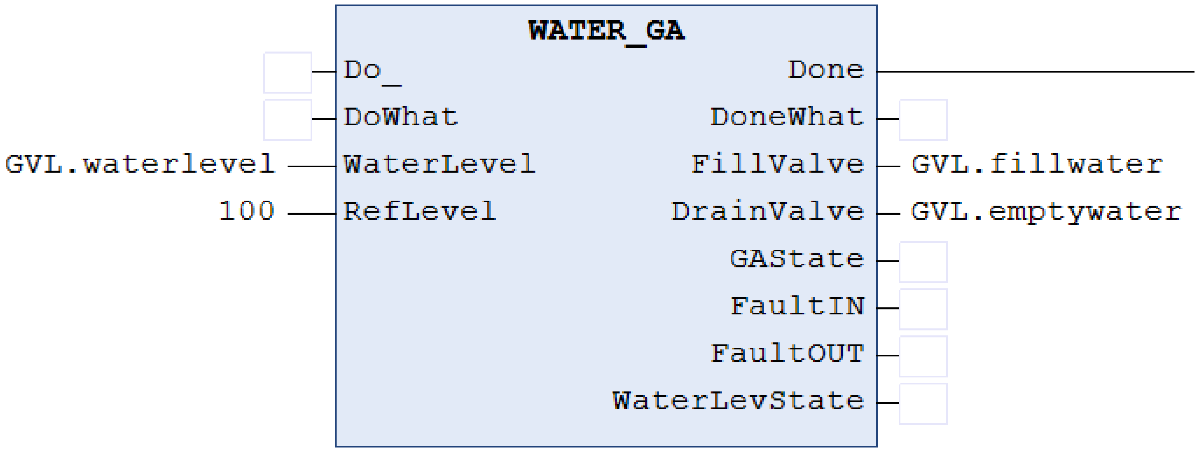
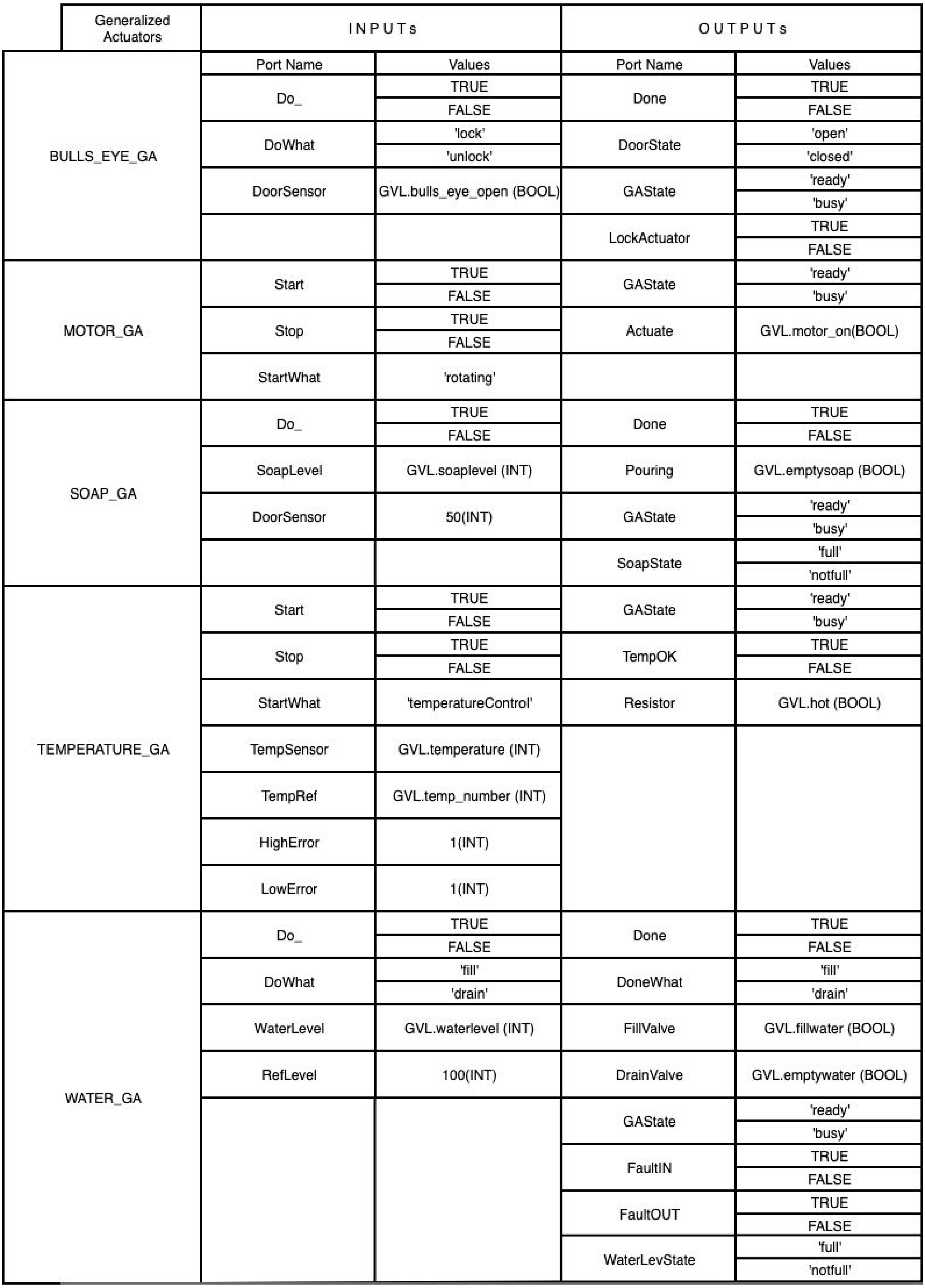


Figure 3.5: Function Block of the WATER_GA

WATER_GA: it's the GA assigned to control the water level inside the drum of the washing machine. It's a Do-Done GA which can fill up the drum thanks to the actuator linked to the valve that permit the water to enter inside the washing machine; Then it can measure, with a particular sensor assigned, the actual level of water in the washing machine drum ; And , when it's required, it can drain out the water by a second actuator linked to a valve which manage the water leaking. In this GA is also able to detect fault on the two valves;



Generalized Actuators	INPUTs		OUTPUTs	
	Port Name	Values	Port Name	Values
BULLS_EYE_GA	Do_	TRUE	Done	TRUE
		FALSE		FALSE
	DoWhat	'lock'	DoorState	'open'
		'unlock'		'closed'
	DoorSensor	GVL.bulls_eye_open (BOOL)	GAState	'ready'
				'busy'
			LockActuator	TRUE
				FALSE
MOTOR_GA	Start	TRUE	GAState	'ready'
		FALSE		'busy'
	Stop	TRUE	Actuate	GVL.motor_on(BOOL)
		FALSE		
	StartWhat	'rotating'		
SOAP_GA	Do_	TRUE	Done	TRUE
		FALSE		FALSE
	SoapLevel	GVL.soaplevel (INT)	Pouring	GVL.emptysoap (BOOL)
	DoorSensor	50(INT)	GAState	'ready'
				'busy'
			SoapState	'full'
				'notfull'
TEMPERATURE_GA	Start	TRUE	GAState	'ready'
		FALSE		'busy'
	Stop	TRUE	TempOK	TRUE
		FALSE		FALSE
	StartWhat	'temperatureControl'	Resistor	GVL.hot (BOOL)
	TempSensor	GVL.temperature (INT)		
	TempRef	GVL.temp_number (INT)		
	HighError	1(INT)		
	LowError	1(INT)		
WATER_GA	Do_	TRUE	Done	TRUE
		FALSE		FALSE
	DoWhat	'fill'	DoneWhat	'fill'
		'drain'		'drain'
	WaterLevel	GVL.waterlevel (INT)	FillValve	GVL.fillwater (BOOL)
	RefLevel	100(INT)	DrainValve	GVL.emptywater (BOOL)
			GAState	'ready'
				'busy'
			FaultIN	TRUE
				FALSE
			FaultOUT	TRUE
				FALSE
			WaterLevState	'full'
				'notfull'

Figure 3.6: Input Output table of the generalized actuators' function blocks

Chapter 4

POLICIES

Then there's the policy, which has to manage and coordinate the GAs in order to perform the task required by the user. In order to do that , it has been developed a function block which has to manage the washing and prewashing working phases : This function block control all the GAs except the BULLS_EYE_GA , and it's called to service by the POLICY. It has the task to detect fault on the resistor and it can receive a direct order by the HMI if the machine is required to stop its functionalities. Furthermore it handle the fault condition in order to turn off the washing machine in a reset or safe state of the machine.

4.1 PREWASH_WASH_POLICY

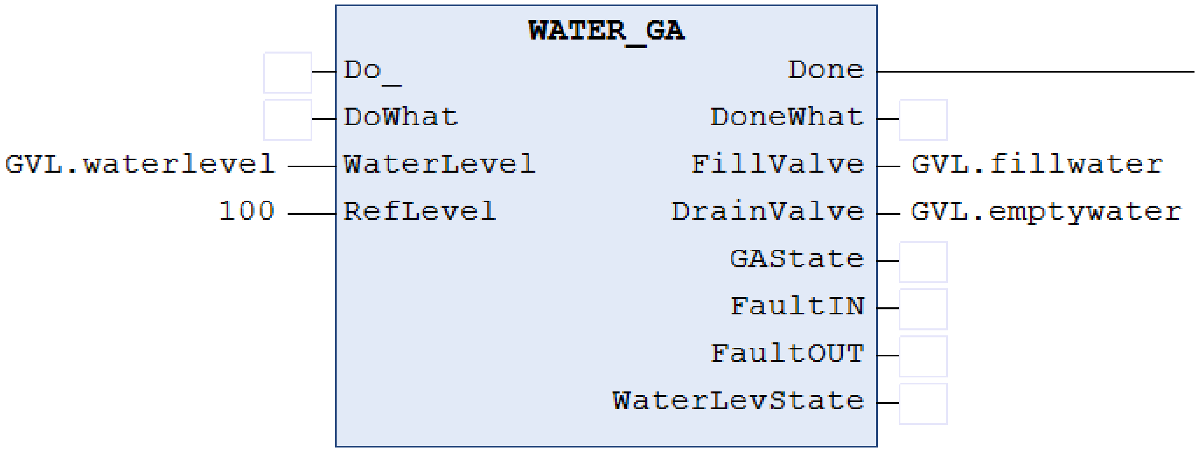
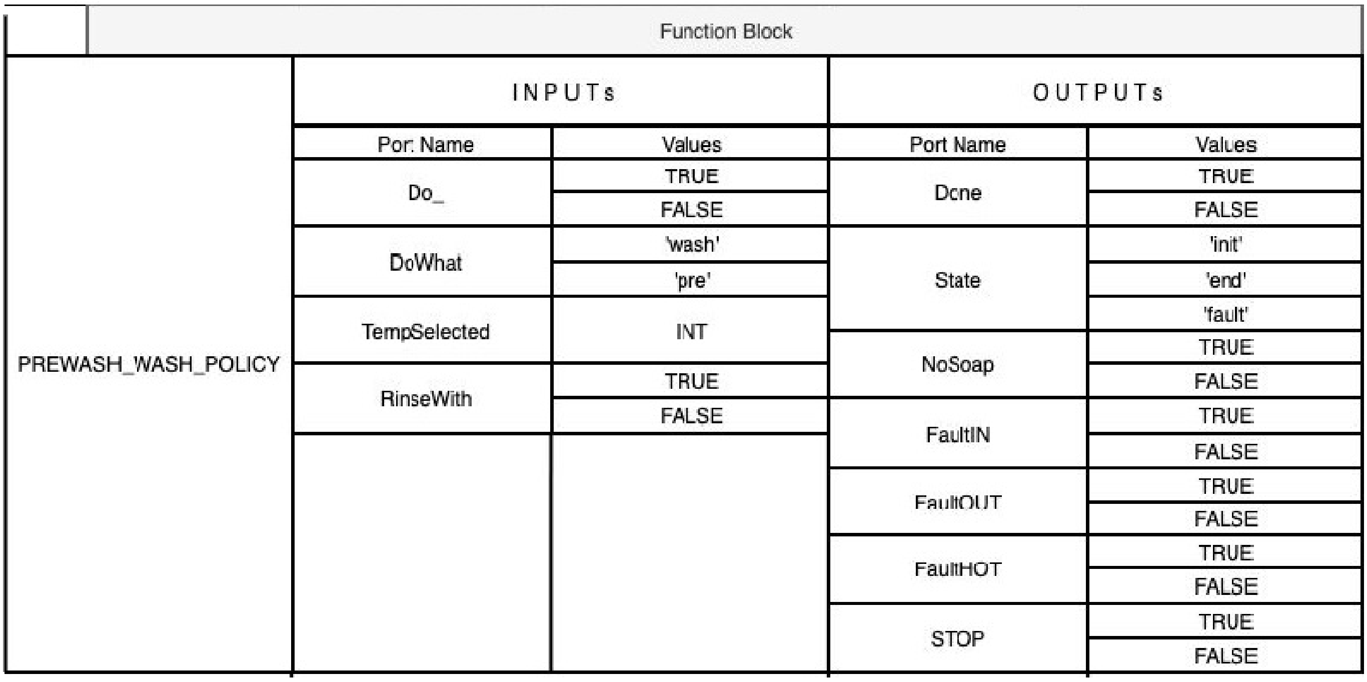


Figure 4.1: Function Block of the WATER_GA

The PREWASH_WASH_POLICY working steps are:

- At the beginning it checks for STOP: the washing machine can be stopped by turning off the run button while it's running;
- If no stop condition are verified and the POLICY has required its service then the PREWASH_WASH_POLICY calls the WATER_GA requiring to fill up the drum of the machine with water; During the water intake, the WATER_GA will also check in parallel if a fault in the water_in_valve has occurred or not and the PREWASH_WASH_POLICY, during this operation, is continuously checking if the WATER_GA has detected some fault or not;
- If a fault occur then the PREWASH_WASH_POLICY will end its work and tell the POLICY turn off the machine;

- If no fault or stop command are occurred then the PREWASH_WASH_POLICY start the TEMPERATURE_GA in order to heat the water until reaching the temperature selected by the user; Also during the time this step is active other two steps are kept active by a branch architecture in order to check for fault on the resistor, one, and for check for stop command, the other;
- Once the temperature has reached the desired temperature then a parallel structure is ran with three branches:
  - Starts the motor;
  - Maintain the temperature at the reference temperature by starting the TEMPERATURE_GA;
  - If the POLICY has requested for a wash then the SOAP_GA is called in order to poor the soap inside the drum, otherwise this branch does nothing ;
- The MOTOR_GA and the TEMPERATURE_GA are kept active for 5 seconds;
- After five second the parallel structure close and the PREWASH_WASH_POLICY call the WATER_GA to drain out the water inside the drum; During the water drain, the WATER_GA will also check in parallel if a fault in the water_out_valve has occurred or not and the PREWASH_WASH_POLICY, during this operation, is continuously checking if the WATER_GA has detect some fault or not;
- If a fault or a stop command occur then the PREWASH_WASH_POLICY will end its work and tell the POLICY turn off the machine;
- If no fault or stop command are occurred then the PREWASH_WASH_POLICY start the MOTOR_GA in order to rinse; And after a rinse of 10 seconds the PREWASH_WASH_POLICY send a ‘Done’ flag in order to tell the POLICY that it has finished the process, and It goes to an idle condition until the POLICY request for other services;



	Function Block			
	INPUTs		OUTPUTs	
	Por: Name	Values	Port Name	Values
PREWASH_WASH_POLICY	Do_	TRUE	Dcne	TRUE
		FALSE		FALSE
	DoWhat	'wash'	State	'init'
		'pre'		'end'
	TempSelected	INT		'fault'
			NoSoap	TRUE
	RinseWith	TRUE		FALSE
		FALSE	FaultIN	TRUE
				FALSE
			FaultOUT	TRUE
				FALSE
			FaultHOT	TRUE
				FALSE
			STOP	TRUE
				FALSE

Figure 4.2: Input Output table of the PREWASH_WASH_POLICY function block

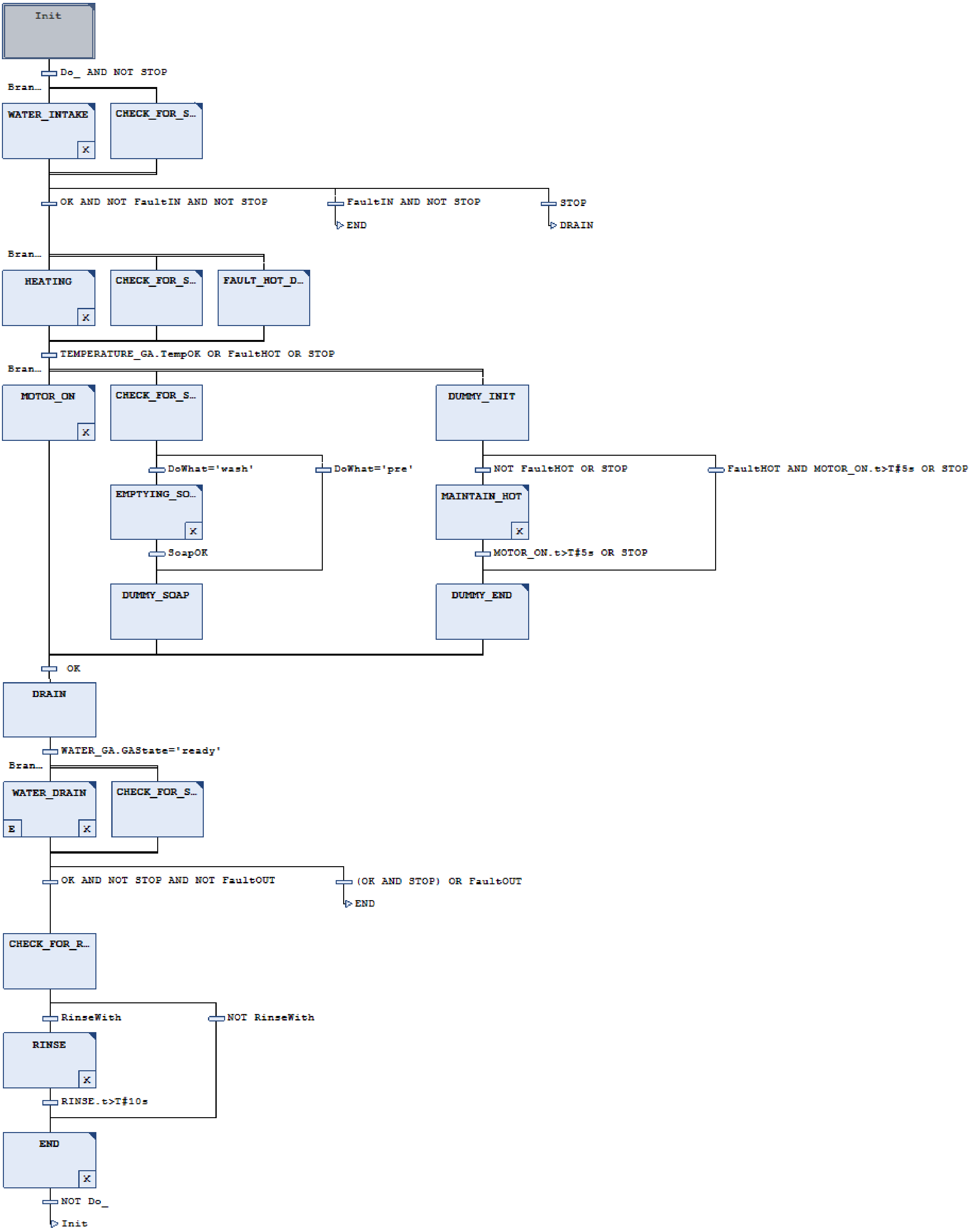
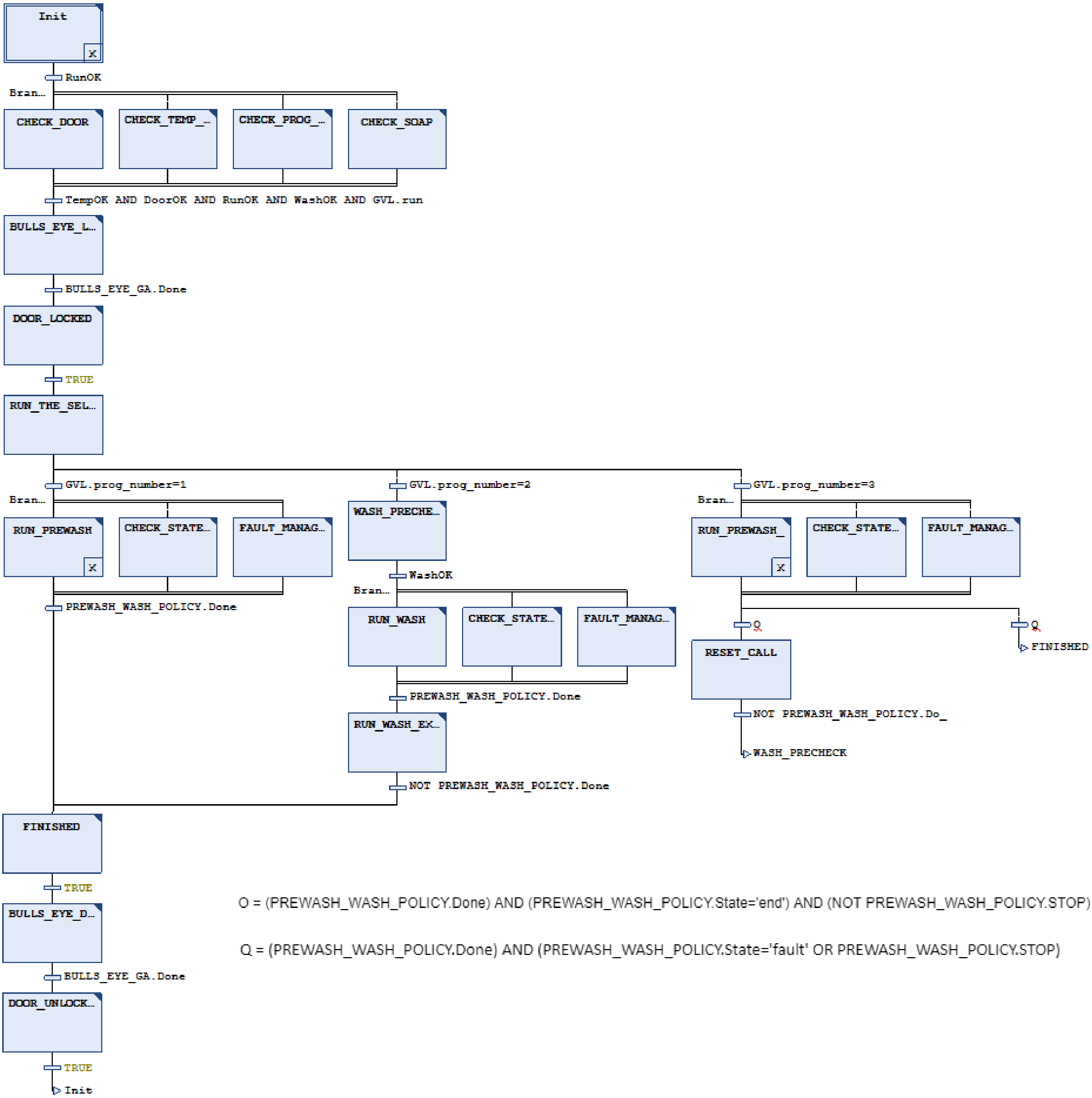


Figure 4.3: PREWASH_WASH_POLICY SFC code

4.2 Main POLICY

The main Policy's working steps are :

- The washing machine will not start working if a fault in the two valves was detected in the previous working phases and it has not been reset yet;
- Check if the initialization requirements are met: as the correctness of the program and temperature number inserted, if the door is close or not (requiring for door state by exploiting BULLS_EYE_GA) and check if the soap(requiring for the soap level exploiting SOAP_GA) is enough to proceed with the washing, when program 2 or 3 are selected; These check of conditions are verified at the same time by exploiting a parallel structure with a branch architecture of four states (one checking the door close, one checking the temperature number , the other for the program number and the last for the soap level check);
- Once the starting conditions are verified at the same time, and the user push the run button then the door is locked by calling the BULL_EYE_GA, and an alternative choice structure in order to run the program selected by the user;
- In order to perform the prewash or wash procedure steps the PREWASH_WASH_POLICY is called: the policy at the call of the PREWASH_WASH_POLICY, specify if a prewash or a wash procedure needs to be performed and if it requires a rinse or not ; This last specification it's particularly needed for the program 3 where prewash and wash have both to be performed, and so the PREWASH_WAS_POLICY is called two times : the first one to perform the prewash and the second one to perform the wash ; But in between these two call the rinse would be useless and so the policy has also to specify to the PREWASH_WASH_POLICY if a rinse should be performed or not; During the 'busy' state of the PREWASH_WASH_POLICY, which means that the PREWASH_WASH_POLICY is working, the communication with the POLICY is still active : The POLICY cannot order to the PREWASH_WASH_POLICY to perform other task during its 'busy' state but the PREWASH_WASH_POLICY feedback the POLICY with information regarding the working phase it's performing (prewash, wash or rinse) and inform the POLICY if a fault is detected or not. This kind of behaviour is achieved by a parallel structure among three states :
  - Manage the call of the PREWASH_WASH_POLICY;
  - Check the working phases on which the PREWASH_WASH_POLICY is working on ;
  - Check if some fault have been detected or not;
- Once the PREWASH_WASH_POLICY has finished, then the POLICY call for the BULL_EYE_GA to unlock the door and it restart;



O = (PREWASH_WASH_POLICY.Done) AND (PREWASH_WASH_POLICY.State='end') AND (NOT PREWASH_WASH_POLICY.STOP)

Q = (PREWASH_WASH_POLICY.Done) AND (PREWASH_WASH_POLICY.State='fault' OR PREWASH_WASH_POLICY.STOP)

Figure 4.4: POLICY SFC code

Chapter 5

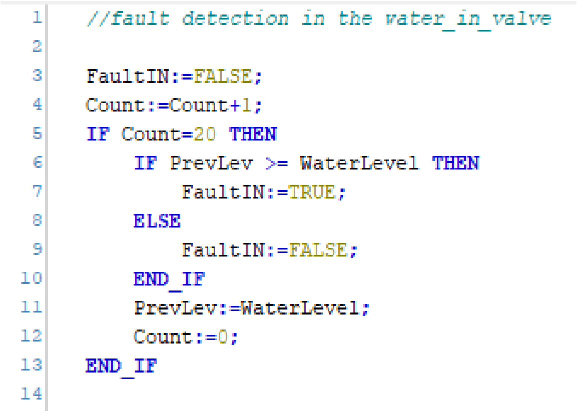
FAULT MANAGEMENT

The washing machine can detect three kind of faults:

- Fault in the water intake valve;
- Fault in the water drain valve;
- Fault in the resistor;

The first two faults are detected by the WATER_GA: During the Intake or drain phases the WATER_GA is called for service by the PREWASH_WASH_POLICY with the request to fill up the machine drum or to empty it accordingly; During these two actions, when the WATER_GA activate the water_in_valve or the water_out_valve, in parallel is checked if these two valves are working or not ; And if the valve activated is found with fault then the WATER_GA end its action with a flag variable which indicate that a fault has occurred; Then the PREWASH_WASH_POLICY will detect the change in state of the fault flag of the WATER_GA variable, stop the process and communicate the POLICY to turn off the machine; The WATER_GA detect the fault by these procedure:

- Fault Intake valve:
  - In the WATER_GA are defined as internal variable : Count (an INT variable that will be used as a counter, initialized at 0)and PrevLev (an INT variable which has to memorize the previous water level measured by the sensor, it's initialized at 0);
  - Reset the flag variable which keeps track of the fault state of the water_in valve (defined as a Boolean variable called FaultIN which it's turned TRUE when the fault is detected);
  - Increase by +1 the counter;
  - An IF condition control when the counter reaches '20' ;
  - When the IF condition is verified, so the counter has counted up to 20, then, another IF condition control the level of the water by comparing it with its previous level and its actual level: In this case if the previous level is found higher or equal to the actual level of the water then the FaultIN variable is set to TRUE in order to stop the WATER_GA action and communicate to the higher policies that a fault in the intake valve has been detected;
  - Then the counter is restored to 0 and the PrevLev memorize the actual value of the water;



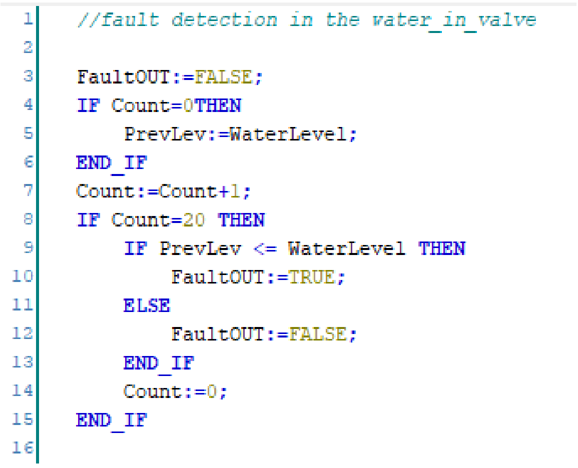
```
//fault detection in the water_in_valve

FaultIN:=FALSE;
Count:=Count+1;
IF Count=20 THEN
    IF PrevLev >= WaterLevel THEN
        FaultIN:=TRUE;
    ELSE
        FaultIN:=FALSE;
    END_IF
    PrevLev:=WaterLevel;
    Count:=0;
END_IF

```

Figure 5.1: Structured Text code for the detection of the fault in the water_in valve

- Fault drain valve:
  - Reset the flag variable which keeps track of the fault state of the water_out_valve (defined as a Boolean variable called FaultOUT which it's turned TRUE when the fault is detected);
  - The PrevLev memorize the actual value of the water only if the counter assume the value 0;
  - Increase by +1 the counter;
  - An IF condition control when the counter reaches '20' ;
  - When the IF condition is verified, so the counter has counted up to 20, then, another IF condition control the level of the water by comparing it with its previous level and its actual level: In this case if the previous level is found lower or equal to the actual level of the water then the FaultOUT variable is set to TRUE in order to stop the WATER_GA action and communicate to the higher policies that a fault in the drain valve has been detected;
  - Then the counter is restored to 0;



```
//fault detection in the water_in_valve

FaultOUT:=FALSE;
IF Count=0THEN
    PrevLev:=WaterLevel;
END_IF
Count:=Count+1;
IF Count=20 THEN
    IF PrevLev <= WaterLevel THEN
        FaultOUT:=TRUE;
    ELSE
        FaultOUT:=FALSE;
    END_IF
    Count:=0;
END_IF
```

Figure 5.2: Structured Text code for the detection of the fault in the water_out valve

The logic inside these two algorithms is:
The PLC has a cycle period of 100ms, and so each 0.1s it will run the task; In order to record if the water level is changing properly with the required action, then each 2s the actual value of the water level is compared with the level assumed 2s before, and the result of this comparison is analysed in order to detect if a fault has occurred or not ;

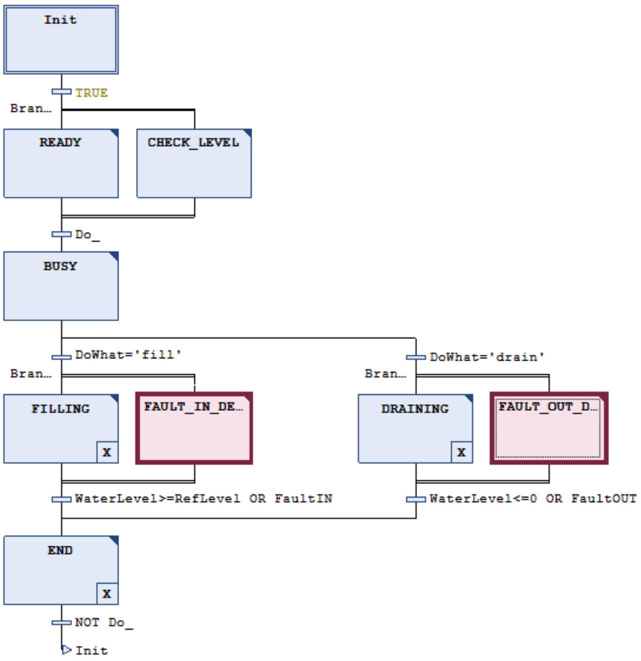
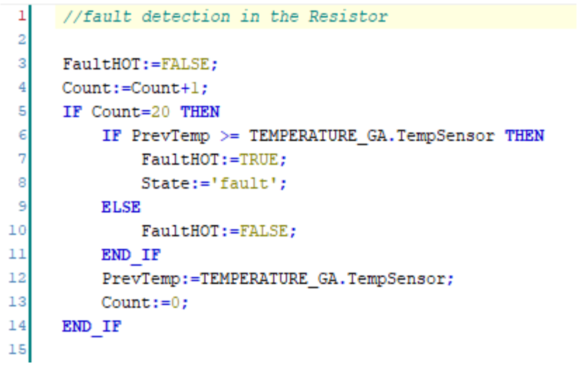


Figure 5.3: SFC code of the WATER_GA underlined the two blocks which contains the ST code related to the detection of fault in the two water valves

- The fault related to the malfunctioning of the resistor, which has to heat up the water, is detected by the PREWASH_WASH_POLICY by applying basically the same logic exploited for the detection of faults in the water_in_valve and water_out_valve; The algorithm:
  - In the PREWASH_WASH_POLICY are defined as internal variable : Count (an INT variable that will be used as a counter, initialized at 0)and PrevTemp (an INT variable which has to memorize the previous water temperature measured by the sensor, it's initialized at 0);
  - Reset the flag variable which keeps track of the fault state of the resistor (defined as a Boolean variable called FaultHOT which it's turned TRUE when the fault is detected);
  - Increase by +1 the counter;
  - An IF condition control when the counter reaches '20' ;
  - When the IF condition is verified, so the counter has counted up to 20, then, another IF condition control the temperature of the water by comparing it with its previous temperature and its actual temperature: if the previous temperature is found higher or equal to the actual temperature of the water then the FaultOUT variable is set to TRUE in order to communicate to the POLICY that a fault in the resistor has been detected;
  - Then the counter is restored to 0 and the PrevTemp memorize the actual temperature of the water;



```
//fault detection in the Resistor

FaultHOT:=FALSE;
Count:=Count+1;
IF Count=20 THEN
    IF PrevTemp >= TEMPERATURE_GA.TempSensor THEN
        FaultHOT:=TRUE;
        State:='fault';
    ELSE
        FaultHOT:=FALSE;
    END_IF
    PrevTemp:=TEMPERATURE_GA.TempSensor;
    Count:=0;
END_IF
```

Figure 5.4: Structured Text code for the detection of the fault in the Resistor

A big difference on the behaviour of the machine in relation with the water_in_valve, water_out_valve and resistor fault is that: in the first two faults we need to stop the prewash or wash operation, because otherwise the machine or the objects inside that need to be washed can be damaged; On the other hand, if the resistor is lame, the washing machine can finish its working process, and the HMI will advise the user that the prewashing or washing procedure has finished without heating the water; Furthermore the machine can be used with a fault in the resistor and the indicator light referred to the fault on the resistor will be kept on in order to indicate that the washing and prewashing are performed using the water at its ambient temperature;