Mini Project

SC42155 Modelling of Dynamical Systems

Maria de Neves de Fonseca, 5611679
Qingyi Ren, 5684803

October 2022
Delft University of Technology

I. Modelling of a complex system

In our task, we are given a complex system driven by a pendulum and a voltage source.

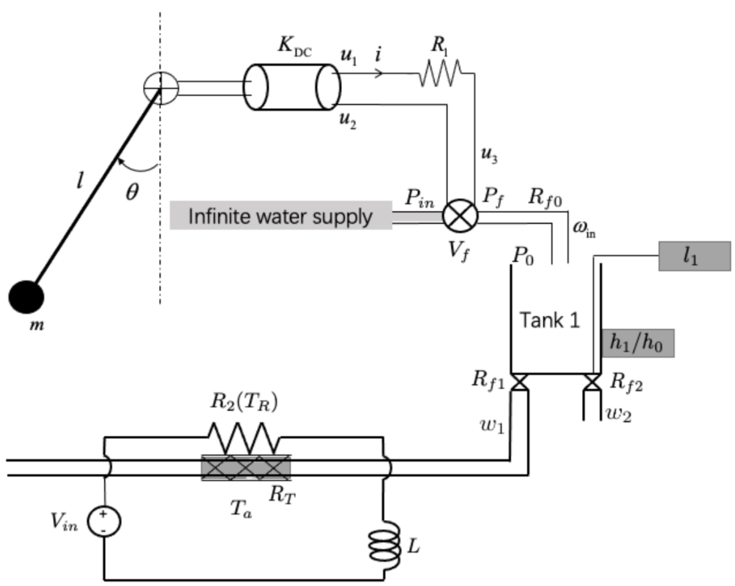


Fig. 1. Complex system driven by a pendulum and a voltage source.

For better analysis and description, we divide the whole complex system into two subsystems, notated as Subsystem 1 and Subsystem 2. The Subsystem 1 contains a pendulum, an ideal DC motor(including DC motor electric circuit with resistor R_1), an ideal pressure valve, Tank 1(including two valves: valve 1 and valve 2) and a water level sensor, which is shown in Fig 2. Then the valve 1 of Subsystem is connected with Subsystem 2 by a pipe, and lets water flow from Subsystem 1 to Subsystem 2. For Subsystem 2, it has a heat-sink (attached to a thermal resistor $R_T(a)$ connected in series), a voltage source V_{in}, an electrical resistance R_2, a inductance L, as shown in Fig 3.

A. Modelling of Subsystem 1

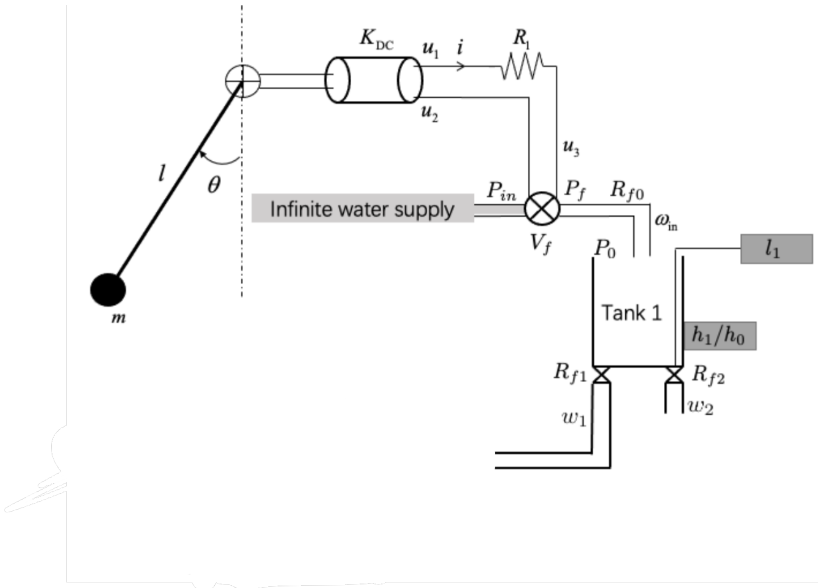


Fig. 2. Subsystem 1.

First, we move to the ideal DC motor part, a pendulum is connected to the input port of DC motor, which creates the input torque τ to drive DC motor. According force analysis of the pendulum, we have

$$\vec{\tau} = \vec{r} \times \vec{F} = \vec{r} \times m\vec{g} = mgl\sin\theta$$

where m is the mass of pendulum, l represents the length of pendulum, and θ means the angle between the pendulum and vertical direction. The behaviour of the DC motor can be described as

$$\tau_{elec} = -K_{DC} \cdot i$$
$$u_1 - u_2 = K_{DC} \cdot \dot{\theta}$$

where $\dot{\theta}$ represents angular velocity of pendulum.

The pendulum is not assumed to swing infinitely, since the motor also produces a torque contrary to the one produced by the pendulum. With this the actual torque provided to the system decreases as

$$\tau_{in} = \tau - \tau_{elec} = mgl\sin\theta + K_{DC} \cdot i$$

and eventually the pendulum stops. To obtain the value of the angular velocity $\dot{\theta}$ we solve the following differential equation from the force balance

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\sin(\theta) = 0 \Leftrightarrow ml^2\ddot{\theta} = -mlg\sin(\theta) - K_{DC} \cdot i \Leftrightarrow \ddot{\theta} = \frac{-mlg\sin(\theta) - K_{DC} \cdot i}{ml^2}$$

so now we have that

$$\theta = \int \dot{\theta} d\theta = \int\int \ddot{\theta} d\theta$$

In the electrical circuit, with the current going through R_1, the voltage $u_3 - u_1$ is

$$u_3 - u_1 = -R_1 \cdot i$$

Then the voltage $u_3 - u_2$ controls the valve V_f, generating pressure change on the right side pipe. Here it assumes that V_f is fed water by an infinite water supply, which means the input water pressure is constant.

$$P_{in} - P_f = \alpha \left| u_3 - u_2 \right|$$

where P_{in} is a constant, α is a positive parameter and P_f is water pressure of pipe connecting with V_f. The pipe leads water into the Tank 1.

We can obtain the voltage difference $u_3 - u_2$ and current i as follows

$$u_3 - u_2 = (u_3 - u_1) + (u_1 - u_2) = -R_1 \cdot i + K_{DC} \cdot \dot{\theta} \Leftrightarrow i = \frac{K_{DC}}{R_1} \cdot \dot{\theta} - \frac{P_{in} - P_f}{\alpha R_1}$$

With the resistance of the open pipe R_{f0}, we can compute the input volumetric flow rate as (assuming it's linear to simplify)

$$w_{in} = \frac{1}{R_{f0}}(P_f - P_0)$$

where P_0 is ambient pressure. This leads to $P_f = w_{in} R_{f0} + P_0$. Looking at the valve as a transformer we can relate the pressure and flow with the voltage and current from the circuit as follows

$$(P_{in} - P_f) = \alpha \left| u_3 - u_2 \right|$$
$$(P_{in} - P_f)\omega_{in} = \left| u_3 - u_2 \right| i$$

which leads to

$$\omega_{in} = \frac{i}{\alpha}$$

with α bring some positive parameter. From this $P_f - P_0 = \frac{i}{\alpha} R_{f0}$ and thus, by having $P_{in} - P_f = (P_{in} - P_0) - (P_f - P_0)$, we can define

$$P_{in} - P_f = (P_{in} - P_0) - \frac{i}{\alpha} R_{f0}$$

and therefore the current in the electric circuit is now

$$i = \frac{K_{DC}}{R_1} \cdot \dot{\theta} - \frac{(P_{in} - P_0) - \frac{i}{\alpha} R_{f0}}{\alpha R_1} \Leftrightarrow i = \frac{K_{DC} \cdot \dot{\theta}\alpha - (P_{in} - P_0)}{\alpha R_1 - \frac{R_{f0}}{\alpha}}$$

For the valves on the bottom of the tank we have that their respective flow rate is defined as (assuming the discharge through ω_2 is zero)

$$w_1 = \frac{1}{R_{f1}} \sqrt{P_1 - P_0}$$
$$w_2 = \frac{1}{R_{f2}} \sqrt{P_1 - P_0} = 0$$

where R_{f1} and R_{f2} represents the fluid resistances of valve 1 and valve 2; P_1 represents the pressure of the bottom of Tank 1. No information about the fluid capacitance C_1 of Tank 1 is provided, but the level of the water in the Tank 1, l_1, is measured by a water level sensor. Therefore, we get the pressure of the bottom of the Tank 1 as $P_1 = \rho g l_1 + P_0$. Note that even though nothing is said about the fluid capacitance we assume it exists and it is C_1. With this we can provide an equation for $\dot{P}_1$ as

$$\dot{P}_1 = \frac{1}{C_1}(\omega_{in} - \omega_1 - \omega_2)$$

The fluid capacitance is calculated by using the area of the tank A_1 as

$$C_1 = \frac{A_1}{\rho g}$$

With these last two equations we can get the value of $\dot{l_1}$ as

$$\dot{l_1} = \frac{1}{A_1}(\omega_{in} - \omega_1 - \omega_2)$$

Note that the level of the water in the tank can either be obtained by $l_1 = \frac{P_1 - P_0}{\rho g}$ or by integrating $\dot{l_1}$.

B. Modelling of Subsystem 2

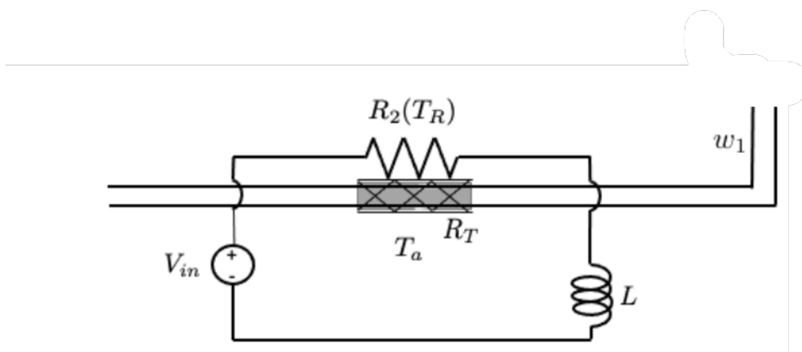


Fig. 3. Subsystem 2.

We consider now the situation where ω_2 is closed, and ω_1 is open. In addition, the voltage source is dependent on the water level height l_1 on Tank 1 as:

$$V_{in}(l_1) = V_o + \gamma l_1$$

with $\gamma > 0$. The physical equations of the system with output the resistor temperature T_R are derived as follows:

$$V_{in} = L\frac{di_2}{dt} + i_2 R_2(T_R) \Leftrightarrow V_o + \gamma l_1 = L\frac{di_2}{dt} + i_2 \alpha_2 T_R \Leftrightarrow \dot{i_2} = \frac{V_o + \gamma l_1 - i_2 \alpha_2 T_R}{L}$$

$$P_e = V_{R_2(T_R)} i_2 = i_2^2 R_2(T_R) = i_2^2 \alpha_2 T_R$$

Assuming that all the electric power that is dissipated through the resistance is transferred to the heat-sink as heat, we have that the heat W is the same as the electric power P_e.

$$W = P_e$$

The heat-sink dynamics can be represented as follows

$$W = \frac{1}{R_T(\omega_1)}(T_R - T_a) = \frac{1}{R_{T_n} - \omega_1}(T_R - T_a)$$

And combining the equation of the electric power with the one of the heat-sink we obtain the following relation

$$W = P_e \Leftrightarrow i_2^2 \alpha_2 T_R = \frac{1}{R_{T_n} - \omega_1}(T_R - T_a) \Leftrightarrow T_R = \frac{T_a}{(1 - (R_{Tn} - \omega_1) i_2^2 \alpha_2)}$$

C. Bond graph of the system

Figure 4 shows the bond graph of the entire system including the pendulum, DC motor and respective circuit, valve and tank, and RL circuit with thermal resistance.

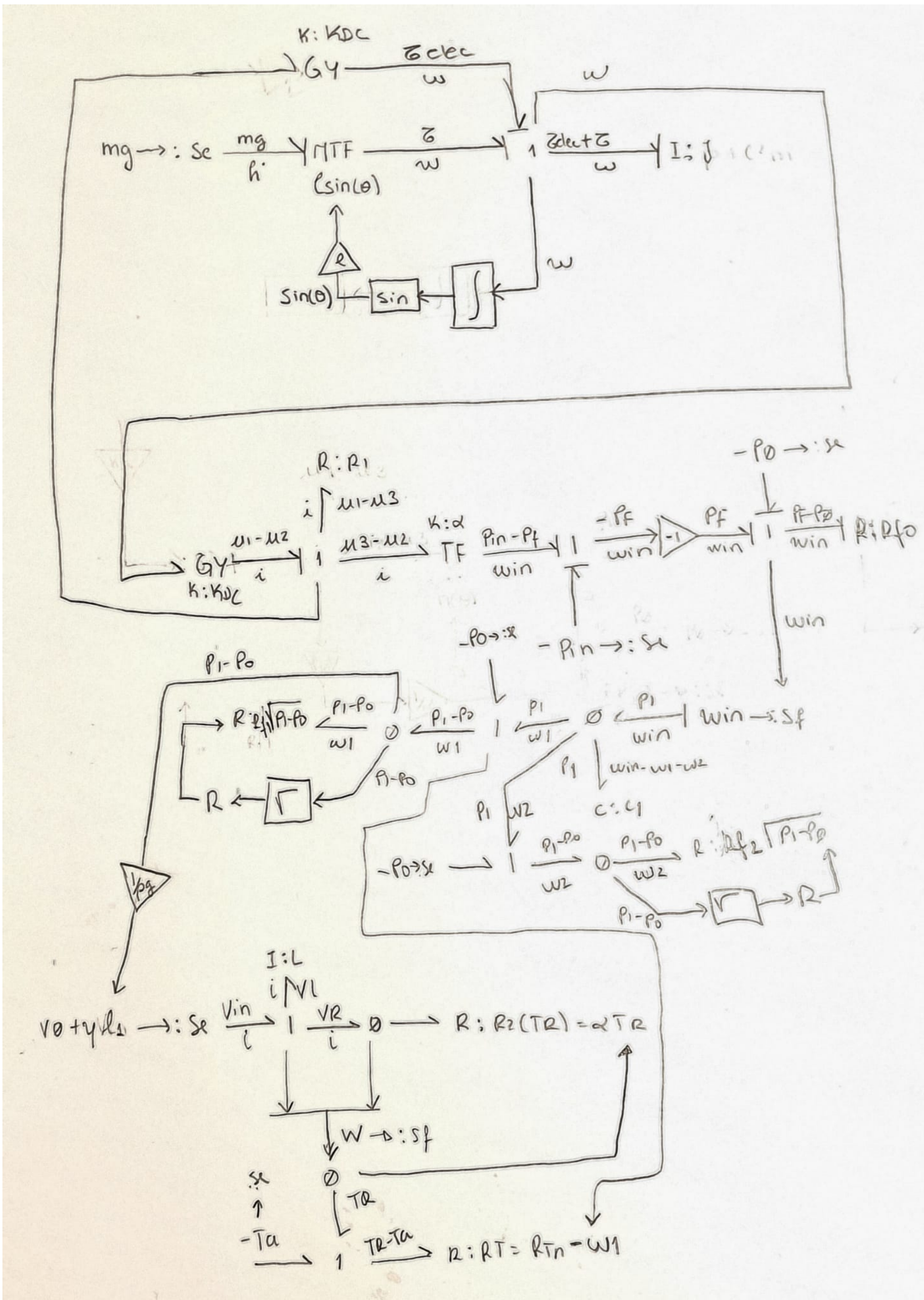


Fig. 4. Bond graph of the system

D. State-Space Model of the system

By choosing as states the vector $x(t) = [\theta(t) \quad \dot{\theta}(t) \quad P_1(t) \quad i_2(t)]^T$ and as input the vector $u(t) = [\omega_{in}(t) \quad \omega_1(t) \quad \omega_2(t)]$, a state-space model of the system can be defined as follows where f and h are nonlinear mappings:

$$\dot{x} = f(x,u) \Leftrightarrow \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{P}_1 \\ \dot{i}_2 \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ \frac{-mlg\sin(\theta)(\alpha R_1 - \frac{R_{f0}}{\alpha}) - K_{DC}\cdot(K_{DC}\cdot\dot{\theta}\alpha - (P_{in}-P_0))}{ml^2(\alpha R_1 - \frac{R_{f0}}{\alpha})} \\ \frac{1}{C_1}(\omega_{in} - \omega_1 - \omega_2) \\ \frac{V_o + \gamma\frac{P_1 - P_0}{\rho g}}{L} - \frac{T_a i_2 \alpha_2}{L(1-(R_{Tn}-\omega_1)i_2^2\alpha_2)} \end{bmatrix}$$

The output was chosen to be the resistor's temperature T_R and it can be represented in the state-space model as

$$y = h(x,u) \Leftrightarrow y = \frac{T_a}{(1-(R_{Tn}-\omega_1)i_2^2\alpha_2)}$$

Note that we could have also chosen the level of the water in the tank as a state, but since that value can be directly obtained from P_1 as $l_1 = \frac{P_1 - P_0}{\rho g}$, and as P_1 is already a state, we do not consider the level of the water as part of the state-space model.

E. Simulation and results

The dimensions and parameter values were selected bearing in mind reasonable values close to reality. To achieve this task, some research was done to find what is, for instance, the normal water pressure in water supply systems in the Netherlands [1]. Note that the parameter α, together with P_{in}, needs to guarantee that P_f is never smaller than the atmospheric pressure so that water does not go into the source. In table I one can find all the values attributed to all the different parameters used in the model.

P_{in}	250 kPa
P_0	101.325 kPa
α	5
g	9.8 m/s^2
m	10 kg
l	1 m
R_1	15 ohm
R_{f0}	5 kPa/(m/s)
K_{DC}	2.5
C_1	0.0158 F
R_{f1}	25 kPa/(m/s)
R_{f2}	30 kPa/(m/s)
ρ	997 kg/m^3
L	100 H
R_{Tn}	2.5 °C/W
T_a	21 °C
γ	0.5
α_2	15
V_0	10 V

TABLE I
VALUES CHOSEN FOR THE PARAMETERS

The pendulum was modeled in Simukink as shown in figure 5. As discussed before, the pendulum does not swing infinitely since the motor produces a counter-torque that slows down the pendulum. Therefore, the parameters had to be chosen so that we could notice a decay of the angular velocity of the pendulum until eventually it stops - figure 6. For this case, the initial position of the pendulum was chosen as $\pi/6$. However, no matter which initial condition we choose, the pendulum will present the same behaviour (unless we choose it to be 0, for which the pendulum will obviously stand still).

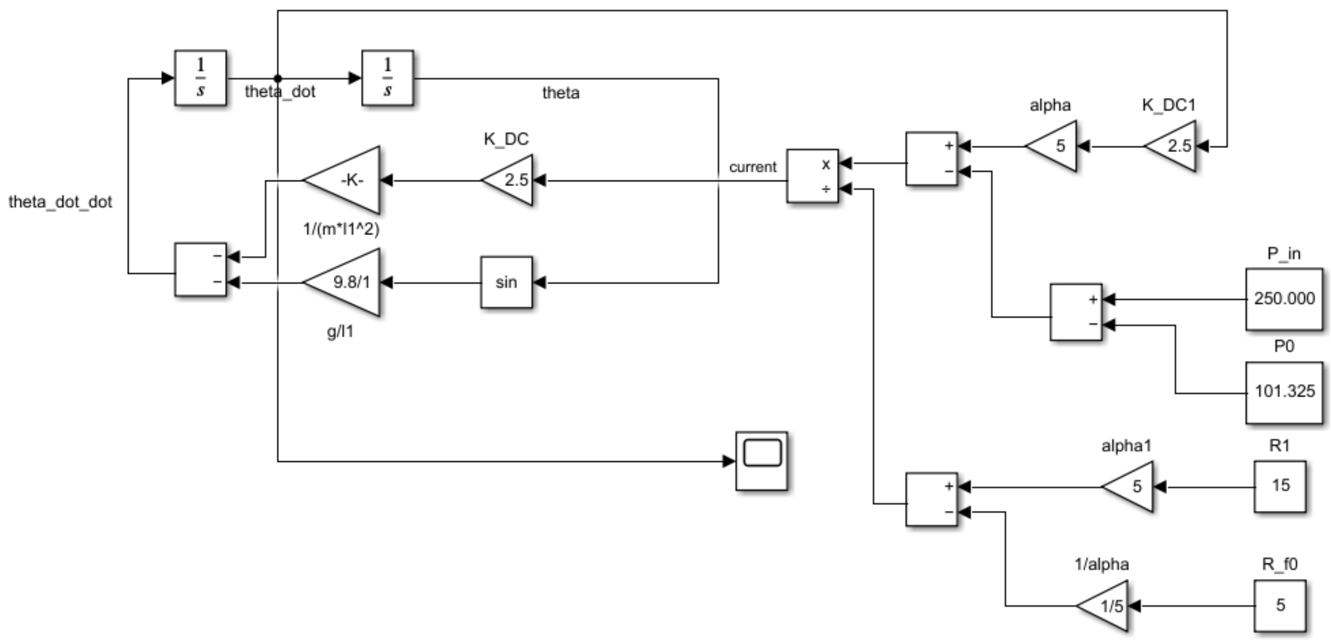


Fig. 5. Pendulum modeled in Simulink

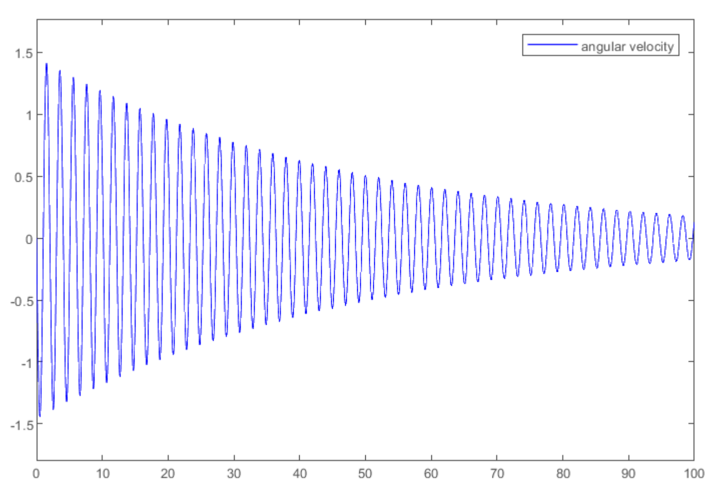


Fig. 6. Angular velocity

The pendulum is connected to an ideal DC motor/-generator attached to an ideal pressure valve V_f. As seen before, the current from the electrical circuit that is attached to the valve is used to obtain the water flow that is going into the tank ω_{in} by looking at the valve as a transformer. The model of the tank together with the pendulum - subsystem 1 from subsection A. - is shown in figure 7. Two switches were added to the model so that it would be possible to change the states of the two valves ω_1 and ω_2 to either *'open'* (value 1) or *'close'* (value 0).

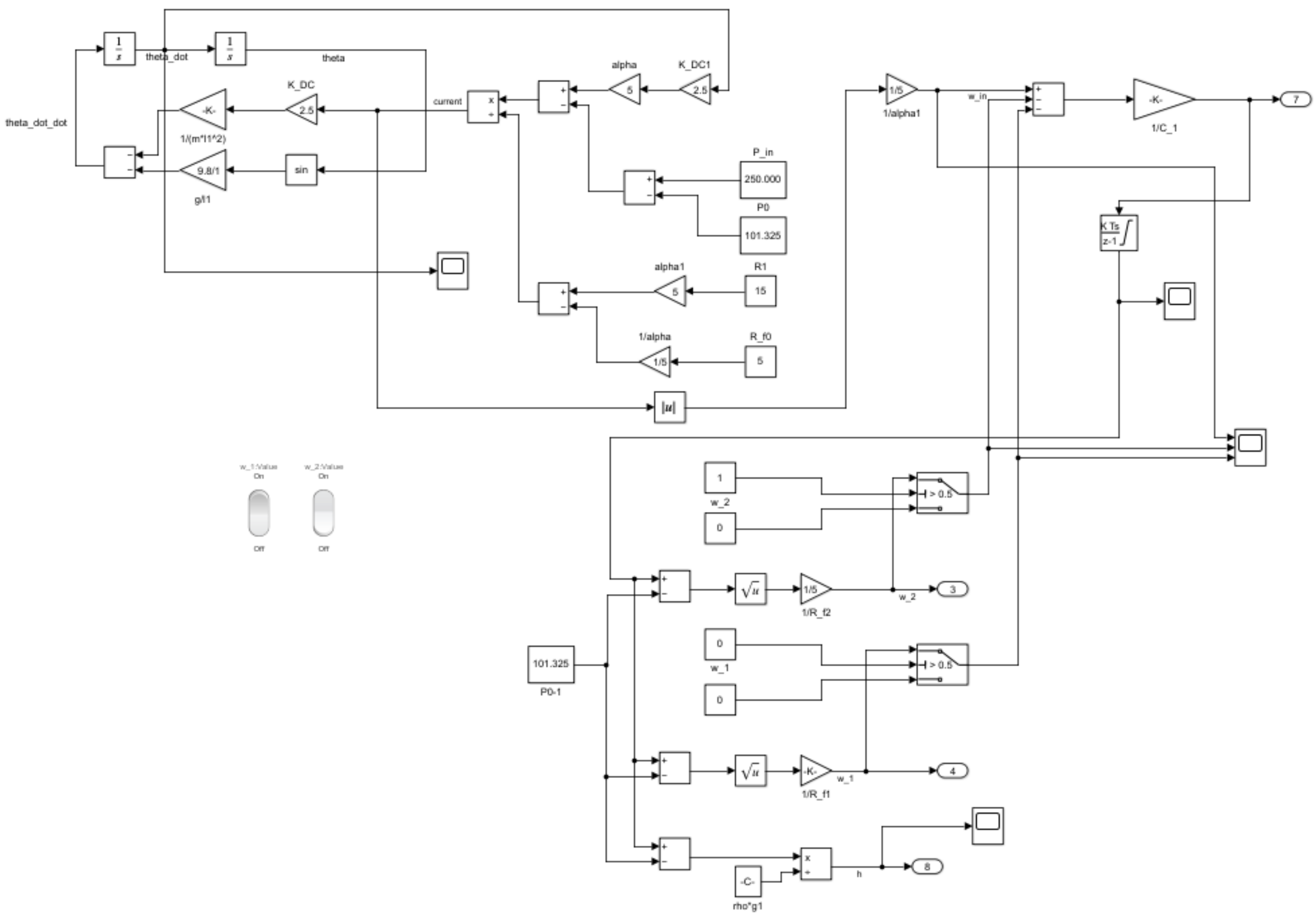


Fig. 7. Subsystem 1 modeled in Simulink

The pressure in the bottom of the tank P_1 is set in the beginning of the simulation as equal to the atmospheric pressure P_0, assuming there is no water in the tank yet. Note that the pressure in the tank can never be less than P_0. Several simulations were run for different inputs: one of the valves opened and both valves opened. The analysis for these situations is done bellow.

The simulation results for when the valve ω_1 is opened and valve ω_2 is kept closed can be seen in figures 8, 9 and 10. One can notice that the pressure and water level slowly increase experiencing an exponential behaviour due to the fluid capacitance C_1. Eventually, the system inside the tank reaches its steady-state with a maximum pressure of around 240 kPa.

In figure 10 one can see that when the switch for valve ω_1 is on ('1') the flow in the valve increases meaning that the water is leaving the tank. This flow also reaches its steady-state since it is dependent on the pressure inside the tank. The flow for the other valve is kept at 0 since the valve is closed and no water is going through it. It is also worth explaining that since the angular velocity of the pendulum decreases with time, the flow that goes from the water supply to the tank also decreases, which indicates that the valve is closing. When the pendulum stops, the current inside the upper circuit remains constant and thus the flow is also constant.

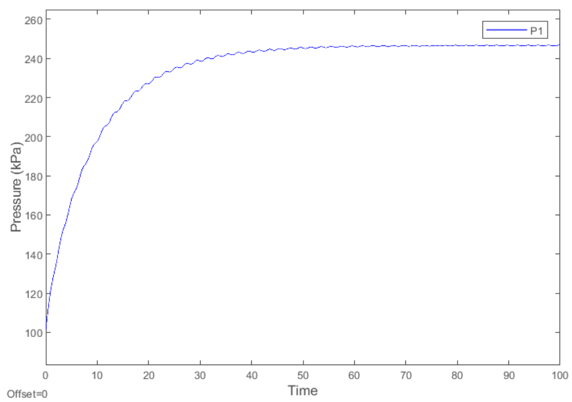


Fig. 8. Pressure in the bottom of the tank when valve w_1 is opened

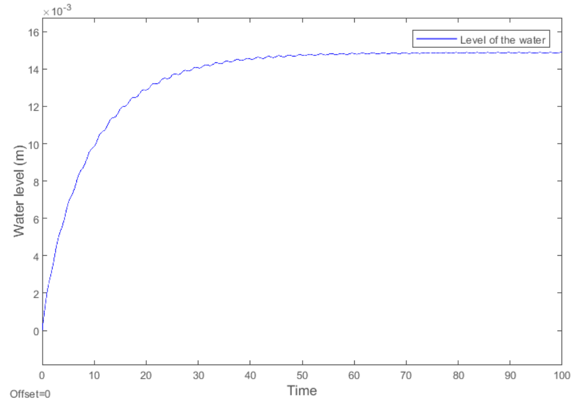


Fig. 9. Water level when valve w_1 is opened

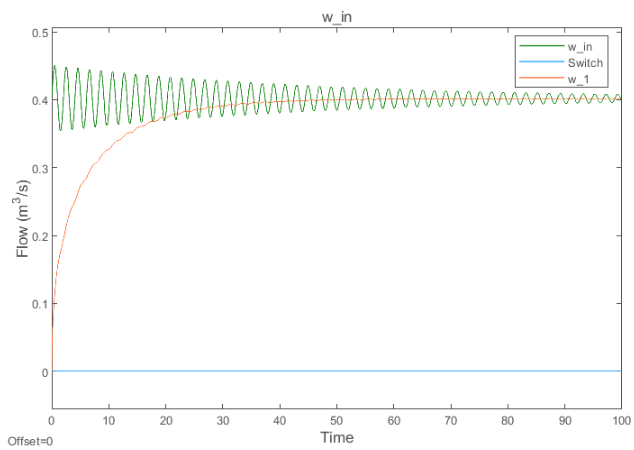


Fig. 10. Flow in the valves for when valve w_1 is opened

The simulation results for when the valve ω_2 is opened and valve ω_1 is kept closed can be seen in figures 11, 12 and 13. One can notice that the pressure and water level have again a very similar behaviour to the one seen in the previous situation. However, this time the maximum pressure reached is lower since the resistance for valve ω_2 is lower than the one for valve ω_1, which means the flow going out is a bit higher than in the previous case leading to a smaller pressure inside the tank and, thus, also lower level of the water.

In figure 13 one can see that when the switch for valve ω_2 is on ('1') the flow in the valve increases as it would in the previous case. Again, the flow reaches its steady-state together with the rest of the system.

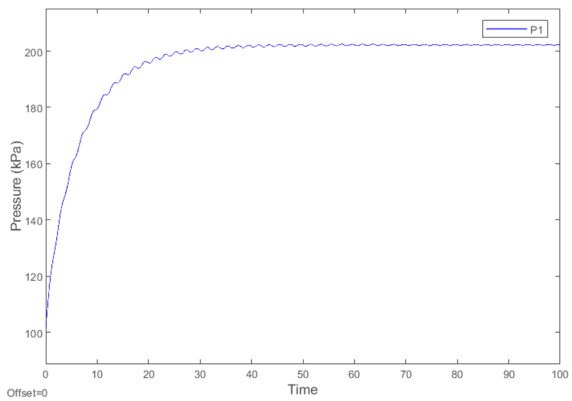


Fig. 11. Pressure in the bottom of the tank when valve w_2 is opened

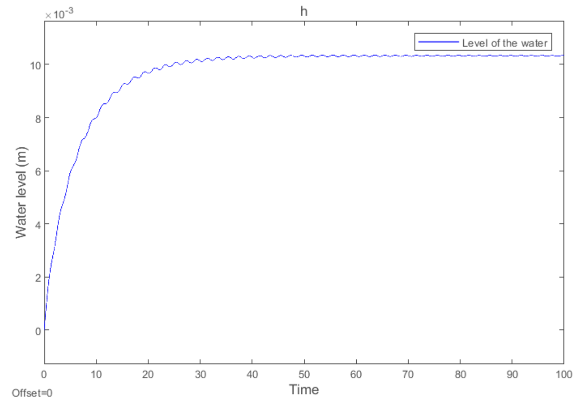


Fig. 12. Water level when valve w_2 is opened

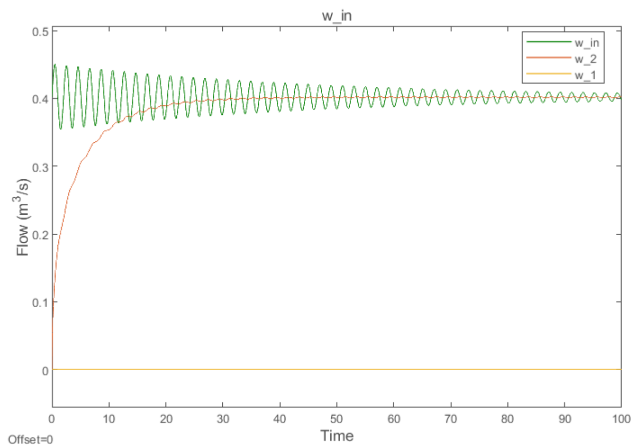
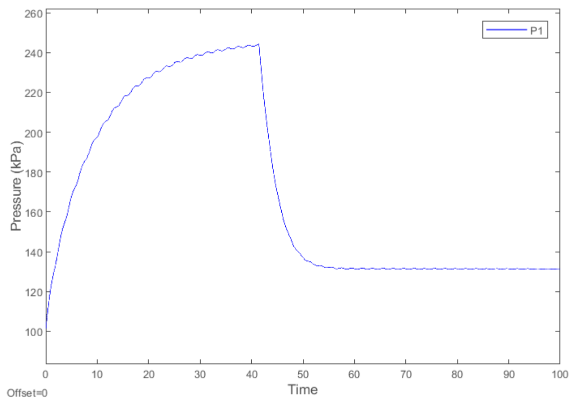


Fig. 13. Flow in the valves for when valve w_2 is opened

A simulation was run for a case where in the beginning only valve ω_1 is opened and then after some time the other valve also opens. It can be seen from figures 14 and 15 that in the beginning we have the same behaviour as for the first case - figures 8 and 9 - but when the second valve is opened the pressure in the tank drops exponentially reaching a steady-state around 140 kPa. This happens because the rate of change of the pressure inside the tank is dependent on the flows, and, therefore, if we have a larger flow going out of the tank, then the pressure will drop.



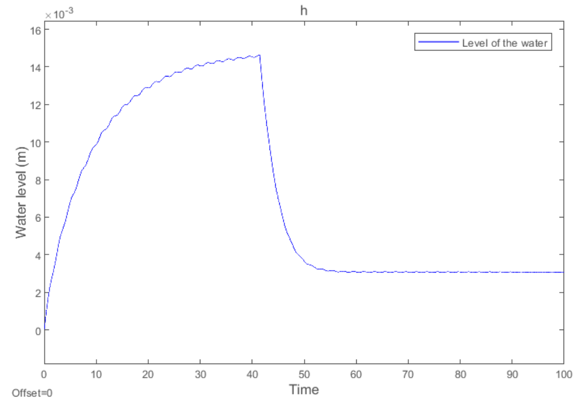


Fig. 14. Pressure in the bottom of the tank when both valves are opened

Fig. 15. Water level when both valves are opened

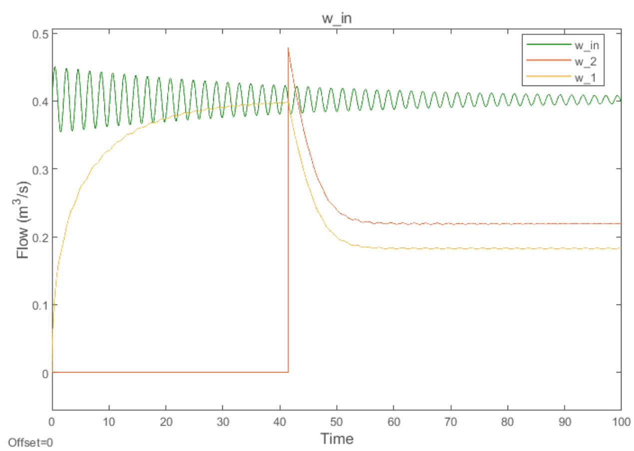


Fig. 16. Flow in the valves for when both valves are open

The Simulink model of the subsystem 2 from subsection B., representing the electrical circuit with the thermal resistance, is shown in figure 17.

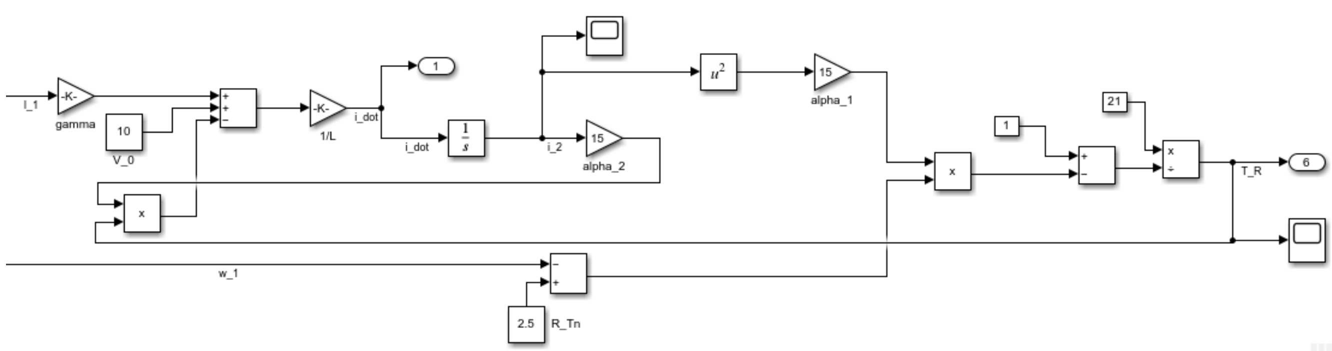


Fig. 17. Subsystem 2 modeled in Simulink

The output of the system was considered to be the temperature of the thermal resistance. Its simulation result is shown in figure 18. We can notice that when the water flows inside the circuit the temperature increases up to 24.5 °C where it stabilises. This can be explained by the fact that when water flows in the system, the thermal resistor's value decreases since it is dependent of the speed of the circulating water. On one hand, as the value of the thermal resistance decreases, the heat flow W as well as the power dissipated by the electrical resistor increase since when the level of the water in the tank increases the voltage in the circuit also increases, leading to the increase of the temperature T_R as well. On the other hand, when the thermal resistance decreases the cold water temperature becomes easier to transfer to the resistor. Thus, we have a balance between two situations that make both the temperature increase/decrease which leads to a steady state.

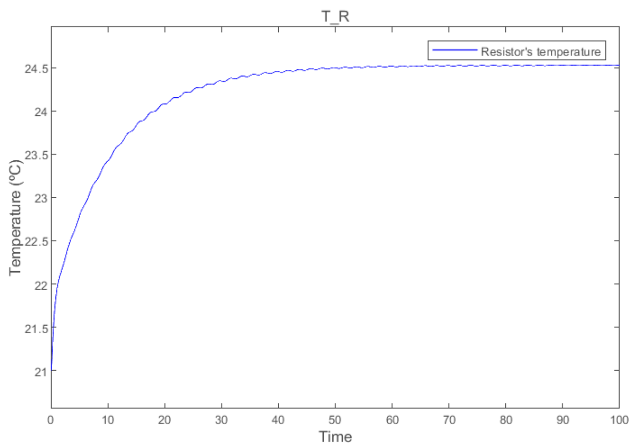


Fig. 18. Temperature T_R

II. Computing Modeling

A. Send-on-data protocol including encoder and decoder

Considering the case in which the water level l_1 of Tank 1 is measured through a quantized sensor only providing 4 level measurements (Tank empty, Level above 10 cm, Level above 20 cm, Level above 30 cm), a send-on-delta communication protocol was implemented with the aim of saving in communication bandwidth. When initialising, the sensor sends a 0 if the tank is empty, or a 1 if the tank is over 30 cm (full). We assume no other message is transmitted, otherwise.

A four bit sequence is used as an input for the encoder which represents the four possible measurements of the level of the water in the tank. The lowest possible measurement point is when the tank is empty - $l_1 = 0$ cm. When in this state, the sensor provides a sequence '0000'. When the water starts filling the tank and reaches a certain level (above 0), the bits in the sequence will change from '0' to '1', from the lowest bit (MP1) to the highest one (MP4). All bits are '1' in the situation where the tank is full. To better understand the behaviour of the output of the water level sensor, it is specified on table II the 4-bit sequence for each water level and its respective decimal value.

Water level	MP1	MP2	MP3	MP4	4-bit output	decimal value
$l_1 = 0cm$	0	0	0	0	0000	0
$0cm < l_1 <= 10cm$	1	0	0	0	0001	1
$10cm < l_1 <= 20cm$	1	1	0	0	0011	3
$20cm < l_1 <= 30cm$	1	1	1	0	0111	7
$l_1cm > 30cm$	1	1	1	1	1111	15

TABLE II
SEQUENCE OF 4-BIT SENT BY THE WATER LEVEL SENSOR

The encoder outputs a sequence of bits indicating if the level of the water increases ('1') or decreases ('0'). Note that the system is initialised either when the water level is $l_1 = 0cm$ or $l_1 > 30cm$, since, as stated before, during initialisation we assume no other message other then these two are transmitted. A Mealy machine - Figure 19 - is built to better represent the behaviour of the encoder. The following is defined using the alphabet $\Sigma = \{0, 1\}$:

* Set of states $Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$
* Input Alphabet Σ^4
* Output Alphabet Σ
* Initial states $Q_0 = q_0$

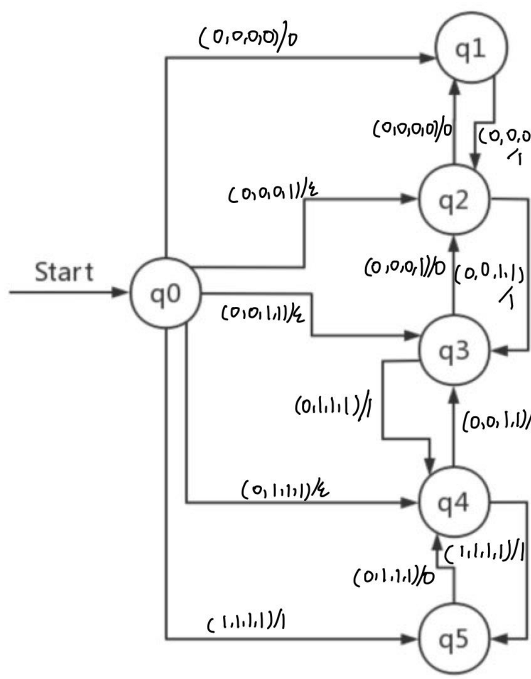


Fig. 19. Mealy Machine used to model the encoder

A model was designed in Simulink to simulate the encoder - Figure 20. The input sequence of the encoder can any possible 4-bit combinations. Thus, to avoid sending wrong combinations, the state-flow represented in Figure 21 was modeled so that only the five sequences provided by the water level sensor (table II) would be available as an input for the encoder.

A *Bits to integer Converter* block is used to get the decimal value of the 4-bit sequence (e.g. '0010' produces decimal value 2). The *Detect Increase* and *Detect Decrease* blocks are used to capture any changes on the decimal value. The *detect Rise Positive* block together with the Sample and Hold block can locate the rising edge time of the increases or decreases of the decimal value. In order to define the initial state of water level status, whether q_1 or q_5, we use the digital clock to generate the time information which helps us to find the initial state.

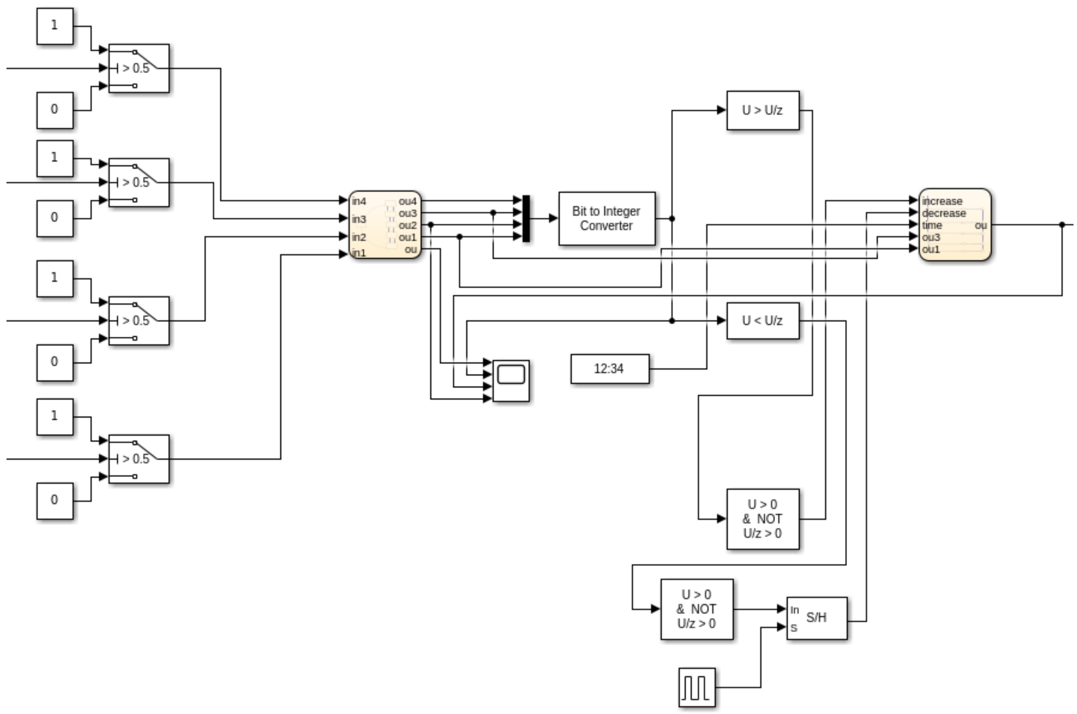


Fig. 20. Model of the encoder in Simulink

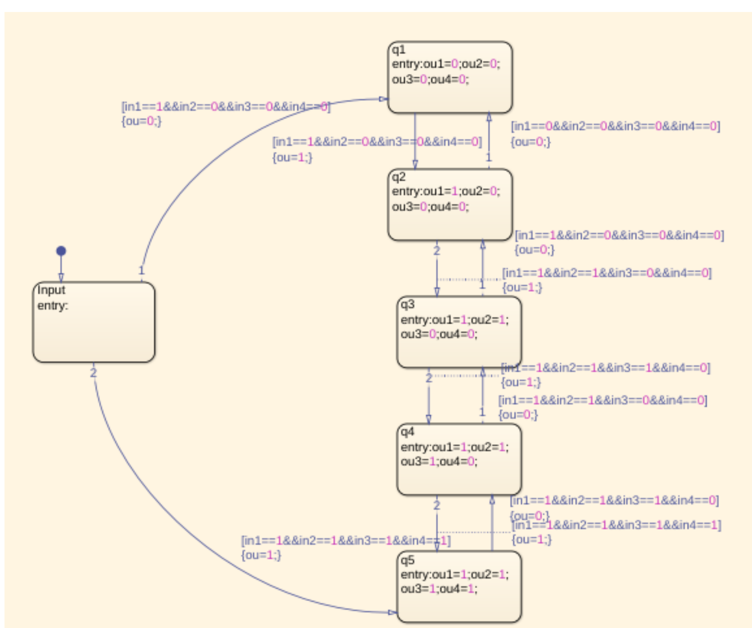


Fig. 21. State-flow chart for the encoder

Unfortunately in Simulink, it is quite difficult to model an analog signal transmission process, more specifically, if we have output '0', it will lead to misunderstanding. In the cases of initial water level not achieving state q_1 or q_4 and water level having increase or decrease with a amount of value less than $10cm$, the encoder transmits a bit '0' to decoder part. If the encoder does not transmit any messages to the decoder, the value of the output of the encoder is still '0'. In order to solve this problem, we use '-1', '1' and '0' as three different states of the encoder's output to represent three modes: increasing, decreasing and not transmitting anything.

The decoder receives a sequence of bits from the encoder: if the encoder outputs a '1', then a '1' is transmitted to the decoder; if the encoder outputs a '-1', a '0' is transmitted to the decoder; if the encoder outputs a '0' then nothing is transmitted. It can be seen in the simulation results in Figure 22, that the encoder outputs a '1' when the water level increases to the above state, and a '-1' when the water level decreases to the state below. Note that, in the figure, the blue line represents the decimal value of the water level state provided to the encoder, and the red impulses with magnitude 1 and -1 represent the output values of the encoder.

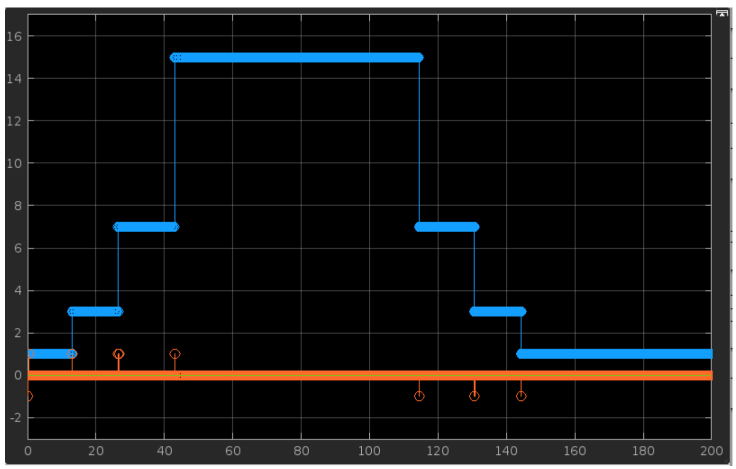


Fig. 22. Simulation of the output of the encoder

A Moore Machine was designed to represent how the decoder works - Figure 23. Let us define $\Sigma = \{0, 1\}$ and

* Set of states $Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$
* Input Alphabet Σ
* Output Alphabet $\{l_1 = 0cm, 0cm < l_1 <= 10cm, 10cm < l_1 <= 20cm, 20cm < l_1 <= 30cm, l_1cm > 30cm\}$
* Initial states $Q_0 = q_0$

A model of the decoder was built in Simulink - Figure 24. For the decoder, a counter capable of resetting was designed to calculate the number of '1' and '-1' received from the encoder. Excluding the bits '1' and '-1' received at time zero point which define the initial state, the upcoming bits '1' and '-1' represent moving to the next water level state, and the number of '1' and '-1' represent how many steps we take to the new state. Note that, we need to reset the counter because when the water level state transfers from q_4 to q_5 or from q_2 to q_1, there are no more steps to move forward and also if we want to step back, we still need to set the counter back to zero as we have accumulated values from historical steps. For example, starting from initial state, we always transfer from q_2 to q_4, and when we first enter q_5, we already accumulate a high value of the number of '1' and '-1', if we want to transfer from q_5 to q_4, first we need to eliminate historical number of -1. Based on this analysis, we design the counter for

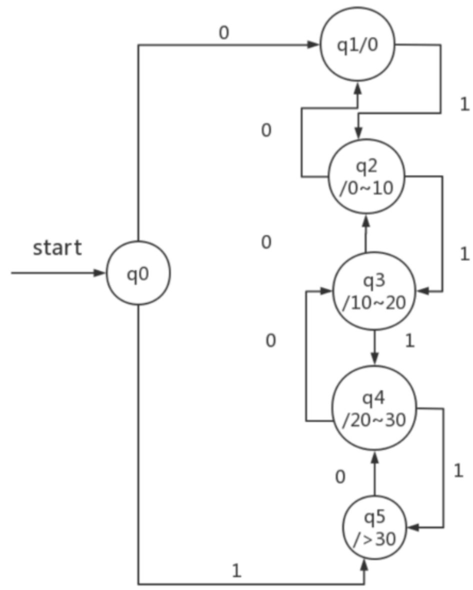


Fig. 23. Moore Machine used to model the decoder('$q2/0$ 10' means state $q1$ with the output 0 10)

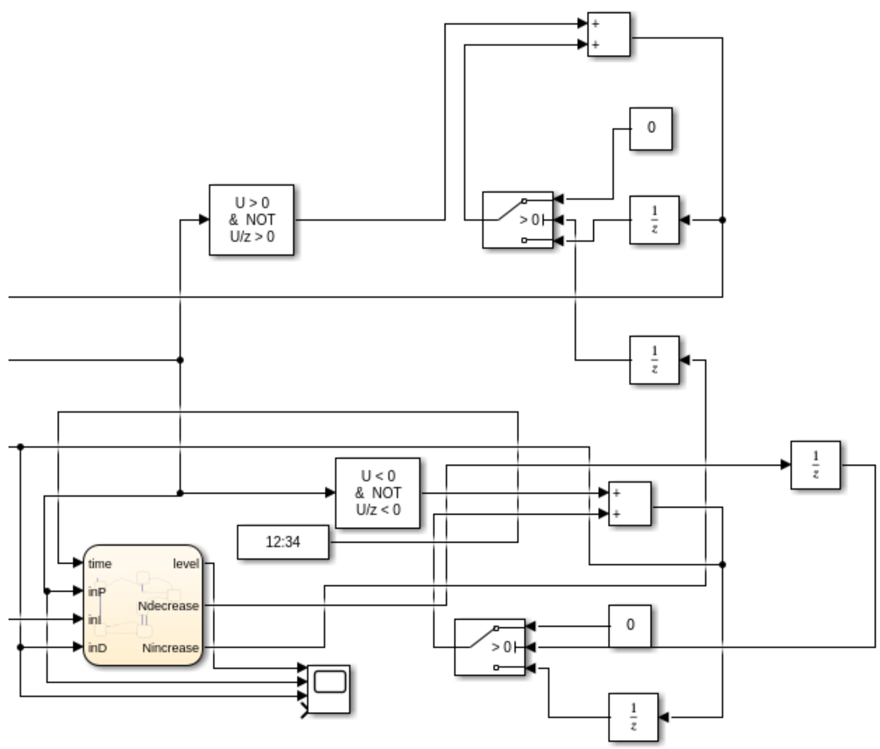


Fig. 24. Model of the decoder in Simulink

the number of '1' and '-1' the decoder received, with the exterior zero control input which is triggered when system arrives at q_1 and q_5 (besides the initial state). This transition relation is showed in Figure 25. The output of the decoder is the water level state and, in this case, the values '0', '1', '2', '3' and '4' were chosen to represent the water level states q_1, q_2, q_3, q_4 and q_5, respectively.

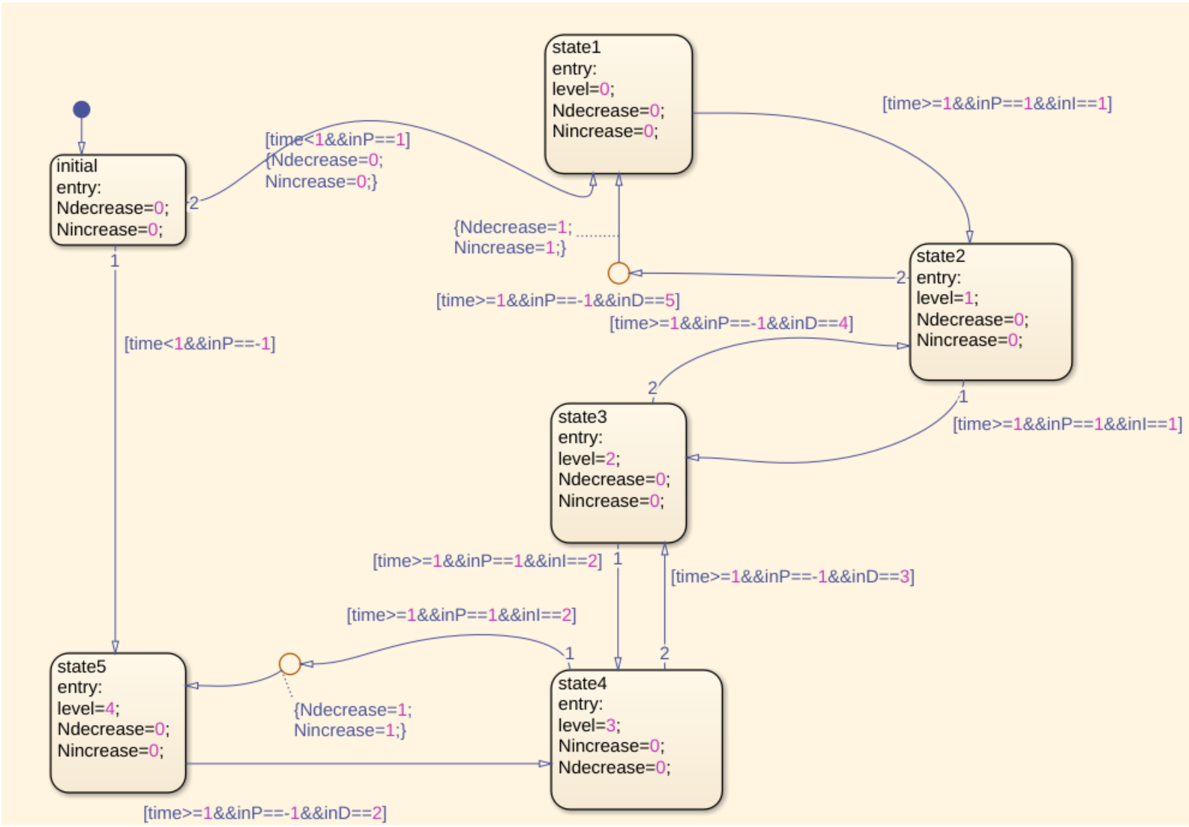


Fig. 25. State-flow chart for the decoder

In Figure 26, the yellow line represents the output of water level states and the blue impulses represent the input of the decoder, with a magnitude of 1 representing the time in which we receive '1' (red) and '-1' (green) respectively. From this graph we can observe that the decoder part can identify the input sequence and show the states of water level.

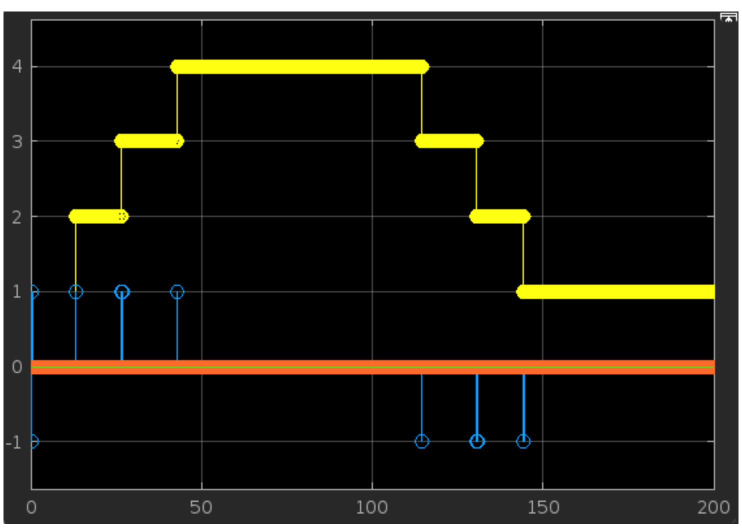


Fig. 26. Simulation of the output of the decoder

B. Minimum Quantization Error

In the send-on-data protocol, we assume that the decoder can only estimate the water level with an error no less than $10cm$ - Quantization Error. According to what we analyzed before, we can only identify four water level states: $\{l_1 = 0cm, 0cm < l_1 <= 10cm, 10cm < l_1 <= 20cm, 20cm < l_1 <= 30cm, l_1cm > 30cm\}$. The minimum quantization error for this case is $10cm$. For example, if we have the water level of 0.0001cm and 10cm we will get the same output sequence '0001' for decoder, even if the actual difference of the value 0.0001cm and 10cm is almost 10cm. If we want to decrease the water level error from the decoder, we can choose more than 4 measurement points to represent more states of water level. For example, if we add one more measurement point in the range of $(0, 30)$cm, and arranging all the measurement points with the same height interval, the minimum quantization error becomes $30/4 = 7.5$cm. However, the initialization status is still a problem, as we can only achieve q_1 and q_5, if the initial water level drops in the height interval $(0, 30)$, the current state of water level cannot be determined without the correct initial state, with the maximum water level error of almost $30cm$, changing to a proper initial state may help to decrease the maximum water level error, but may not decrease the minimum quantization error, which means water level cannot be estimated with an error less than $10cm$. To minimize the maximum water level from $30cm$ to $10cm$, the protocol can be modified about initialisation. It transmits a four-bit length of sequence out from $\{0000, 0001, 0011, 0111, 1111\}$ to the decoder which defines the initial state. After the sequence, the encoder keeps the same mechanism of generating a bit '1' when the water level increases to the next water level state and generating a bit '0' when the water level decreases to the next water level state. By using this protocol, the encoder can be enhanced by decreasing the maximum water level error to $10cm$.

C. A system capable of detecting two Alarm Events

In this section, we consider adding two alarm events to the system. The system will now be capable of detecting water flooding and malfunction of the sensors. The encoder now has a 5-bit sequence as its input. In order to differentiate the two alarm events and five water level states $q_0, q_1, q_2, q_3, q_4, q_5$, we define the sequence $\sigma_o = 11111$ and $\sigma_a = 11110$ as the events of water flooding and malfunction of the sensors, as shown in table III.

Event	MP1	MP2	MP3	MP4	MP5	5-bit output	decimal value	Alarm status
$l_1 = 0cm$	0	0	0	0	0	00000	0	Normal
$0cm < l_1 <= 10cm$	1	0	0	0	0	00001	1	Normal
$10cm < l_1 <= 20cm$	1	1	0	0	0	00011	3	Normal
$20cm < l_1 <= 30cm$	1	1	1	0	0	00111	7	Normal
$l_1 > 30cm$	1	1	1	1	0	01111	15	Normal
water flooding	1	1	1	1	1	11111	31	alarm
malfunction of the sensors	0	1	1	1	1	11110	30	alarm

TABLE III

SEQUENCE OF 5-BIT SENT BY THE WATER LEVEL SENSOR

It is worth mentioning that, we can easily differentiate the two alarm events and five measurement status, because after using the Bits to integer converter block, the decimal values of '11111' and '11110' as 31 and 30 are both much higher than decimal values of '00000', '00001', '00011', '00111' and '01111', with the maximum decimal value of 15.

In Simulink, we create two events *alarm1* and *alarm2*, shown in Figures 27 and 28. *alarm1* activates every 100s with a duration of 0.2s, while *alarm2* activates every 70s also with a duration of 0.2s.

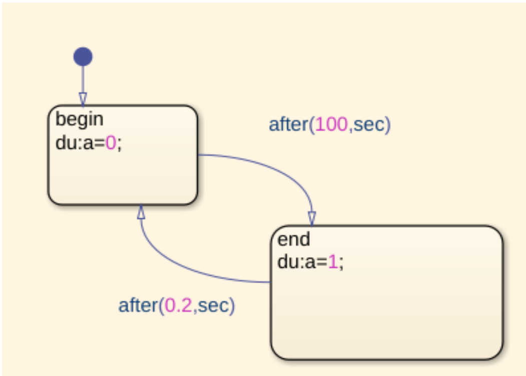


Fig. 27. Alarm1: Water flooding

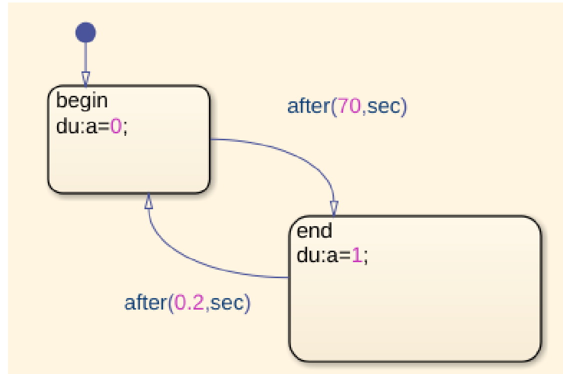


Fig. 28. Alarm2: Malfunction of sensors

In Figure 29 we have the structure of the system capable of detecting water flooding and malfunctioning sensors with the state-flow chart in Figure 30 that represents the process of the system detecting the two alarm cases and water level states. When there is an alarm, the measurement from the sensor raises to either decimal value of 30 or 31 as can be seen in Figure 31.

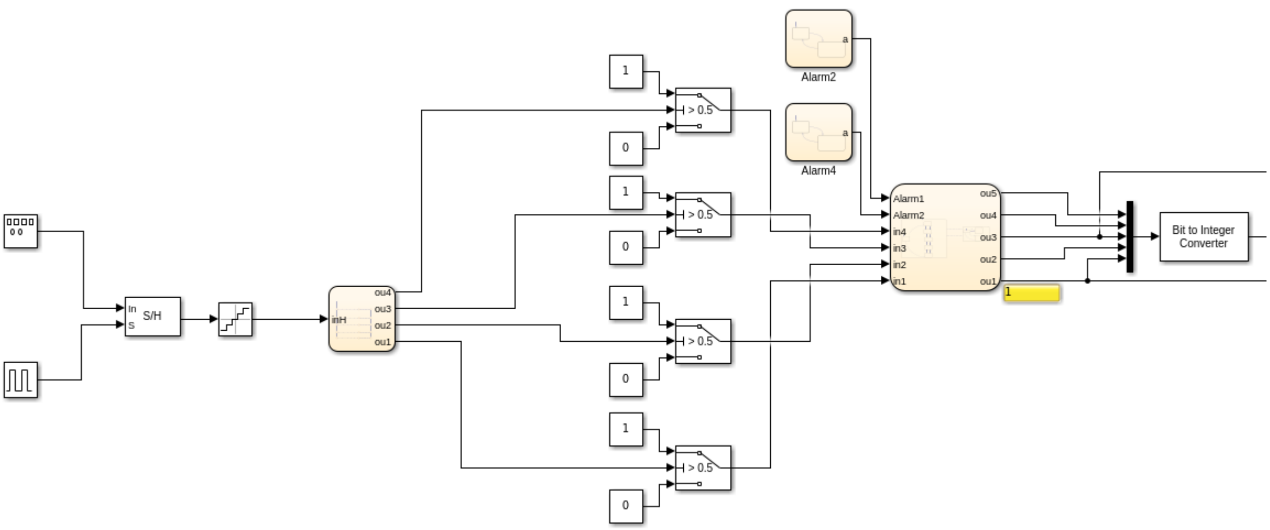


Fig. 29. System capable of detecting two alarms cases

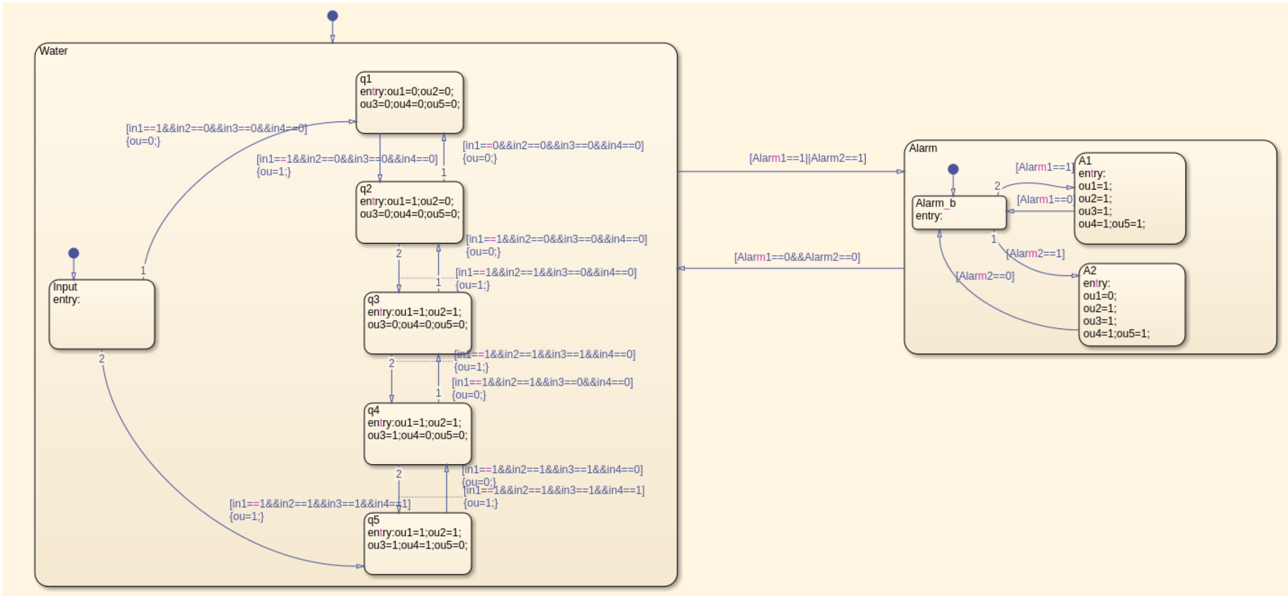


Fig. 30. State flow for the system

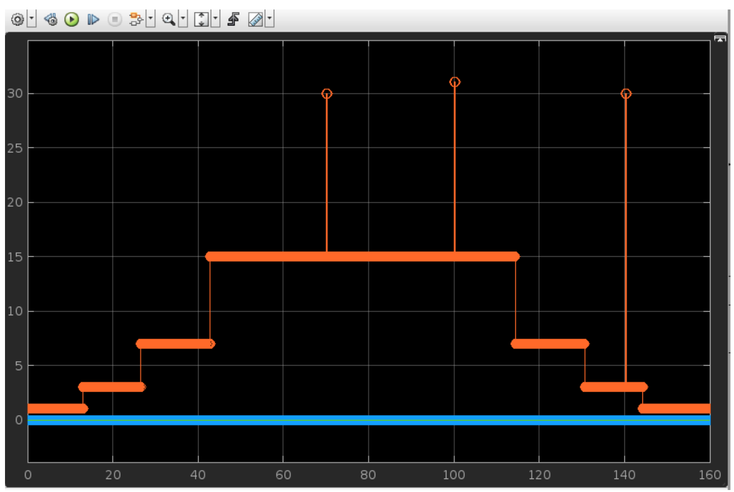


Fig. 31. Measurements from the sensor and alarm cases

D. *Enhanced send-on-data protocol incorporating two alarm events*

In this section, the two alarm transmissions are added to the send-on-delta encoder and decoder systems. First, another Mealy Machine - Figure 32 - is designed to represent the encoder behaviour. Once more, the alphabet employed is defined as $\Sigma = \{0, 1\}$,

* Set of states $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$
* Input Alphabet Σ^5
* Output Alphabet Σ
* Initial states $Q_0 = q_0$

where q_6 and q_7 represent the water flooding case and malfunction of sensors, respectively. The Moore Machine in Figure 33 is designed to represent the decoder from the send-on-data protocol. The following is also defined using the alphabet $\Sigma = \{0, 1\}$

* Set of states $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$
* Input Alphabet Σ
* Output Alphabet $\{l_1 = 0cm, 0cm < l_1 <= 10cm, 10cm < l_1 <= 20cm, 20cm < l_1 <= 30cm, l_1 cm > 30cm, flooding, malfunction\}$
* Initial states $Q_0 = q_0$

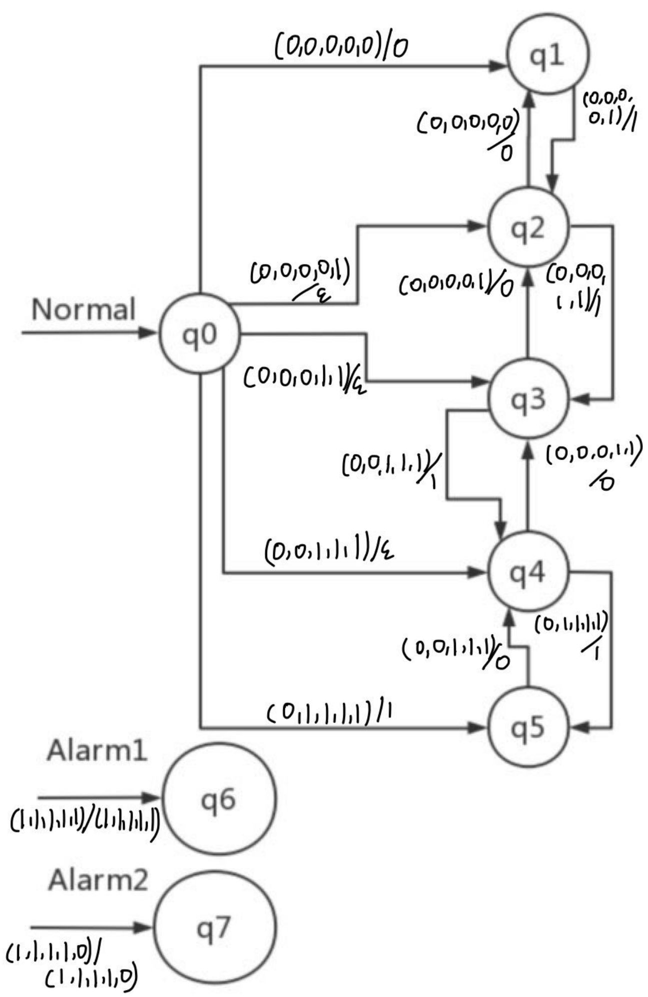


Fig. 32. Mealy Machine with alarms.

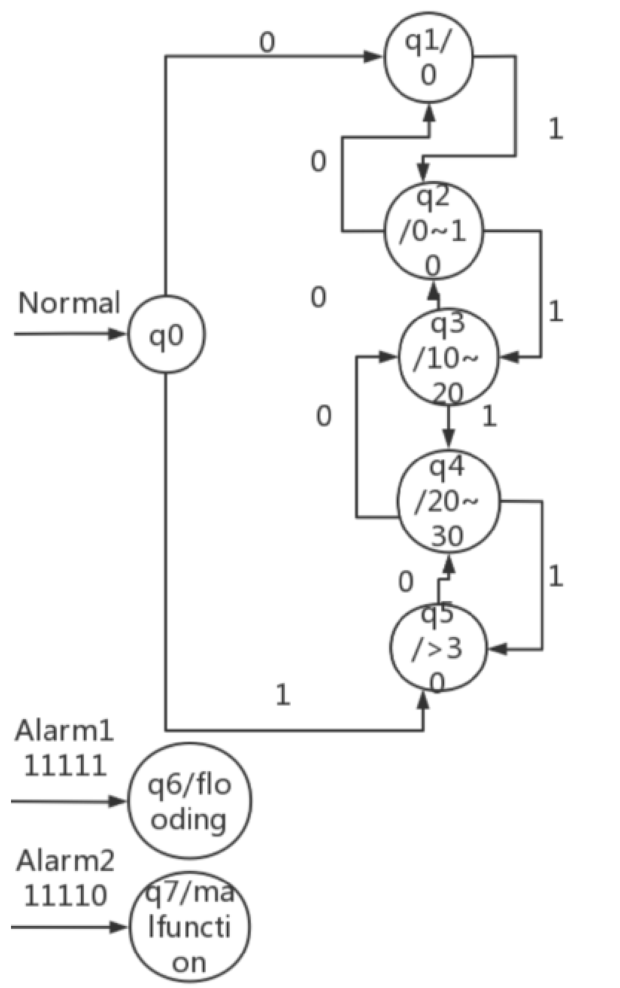


Fig. 33. Moore Machine with alarms (the notation '$q2/0 \sim 10$' means state $q1$ with the output '$0 \sim 10$').

Realistically, the alarm cases can happen randomly and unpredictably. As stated before, the sensor produces a higher decimal value in case of alarm. So, when using the send-on-data protocol introduced in subsection A., there will be a sequence of unwanted '1' (bearing in mind that '1' represents an increase to the next level), which leads to the wrong representation of water level. In order to solve this problem, a mechanism is designed to select the alarm-triggering signal out of the sequence produced by the *Detect increase* block. The decimal value from the sequences is compared with a threshold value T. When higher than the threshold, the system is in the presence of an alarm case, so an alarm triggering signal is produced and the encoder outputs either the sequence $\sigma_o = 11111$ for *alarm1* or $\sigma_a = 11110$ for *alarm2*.

When the decoder receives a sequence sent by the encoder, we should analyse whether or not it is in the presence of either sequences σ_o or σ_a. For this case, it was assumed that the water level does not change abruptly. In other words, it takes more to transmit a signal from any intermediate state q_1, q_2 and

q_3 to the initial q_1 or final q_5, when comparing to the time needed to transmit the specific sequences σ_o or σ_a. Thus, it becomes almost impossible to have a circumstance where the signals interfere with each other. For instance, in the case the sequence '011110' is received during a small time interval, it would be hard to define whether it only includes the normal sequence '01111' or the alarm one '11110'. Therefore, this situations are not assumed to happen to simplify the simulation.

E. Simulation and discussion of the results

In this section, the whole send-on-data protocol with the mechanism of raising alarms is modeled in Simulink. The model and state-flow chart of the encoder are shown in Figures 34 and 35, respectively.

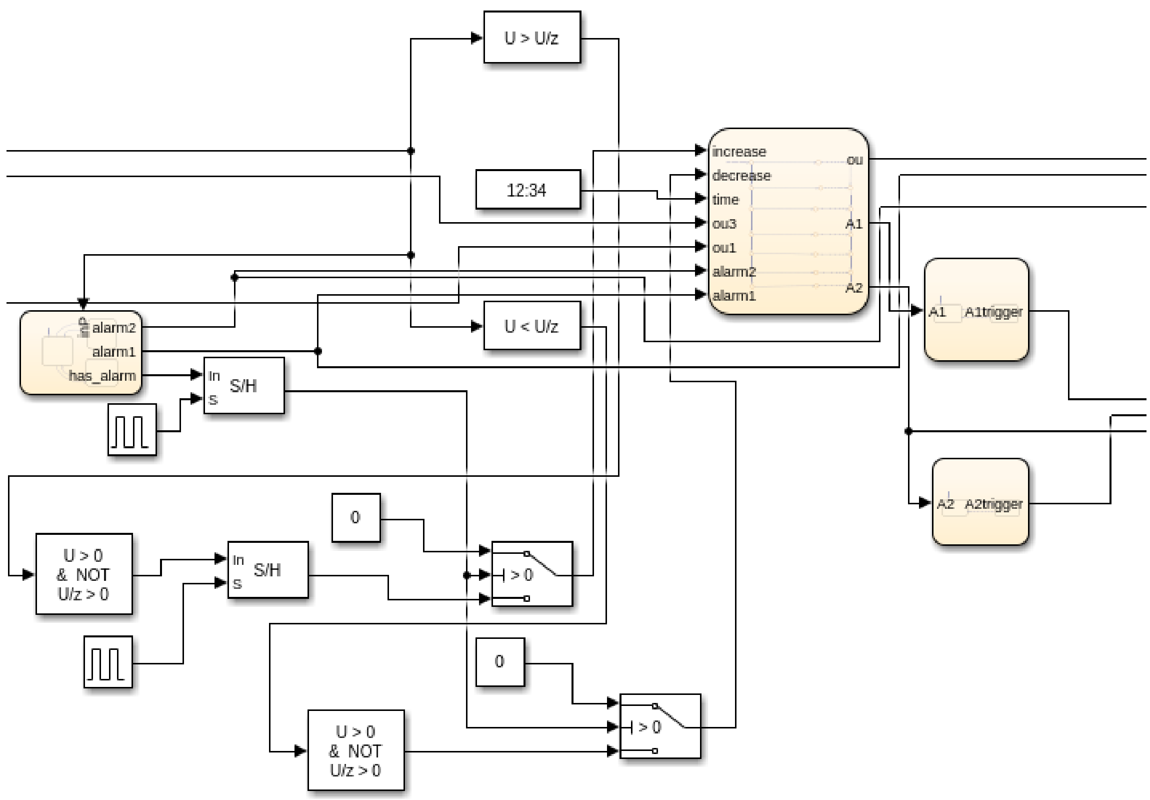


Fig. 34. Encoder from the send-on-data protocol

For the encoder, by decomposing the sequence produced by the *Detect increase* block, we can separate the encoder part into two different modes: first mode uses the protocol mentioned in subsection A., which is only used to detect the transmissions of water level states; the other mode is used to detect the two alarm states - Figure 36.

In Figure 37 the all the increases can be detected even for alarm-triggering time. This will lead to an incorrect instruction 'increase to next water level' to the decoder at the time of alarm events. In Figure 38, we can observe that after removing all the alarm-triggering signals from the sequence produced by the *Detect increase* block, the encoder only has output '1', '-1' or '0' to transmit to the decoder a bit '1' when the water level increases to the next state ('1'), and a bit '0' when the water level decreases to the next state ('0'). For the alarm mechanism, the events *alarm1* and *alarm2* are triggered by transmitting either the sequence σ_o or σ_a to the decoder in a small time interval.

For the decoder, the model in Figure 39 was built, which adds 'A1trigger' and 'A2trigger' as extra inputs of the decoder. This is how we imitate the process of the decoder which receives a special sequence

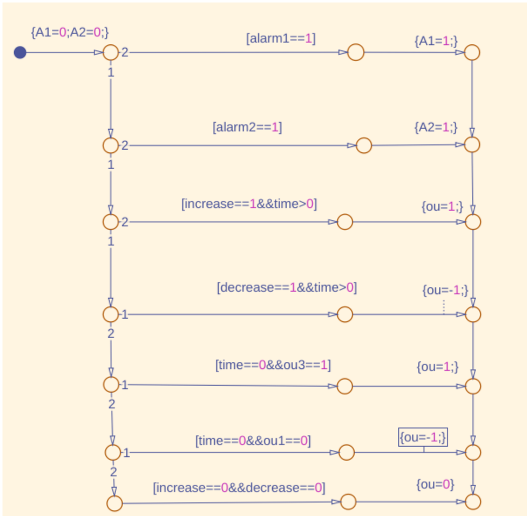


Fig. 35. State-flow chart used for the encoder

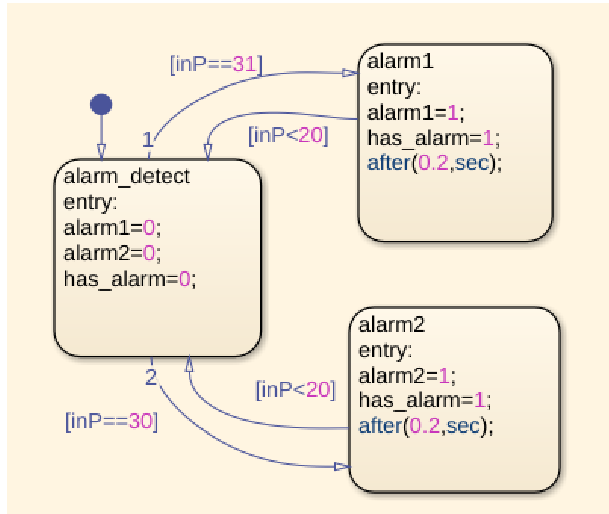


Fig. 36. State flow chart for alarm detection

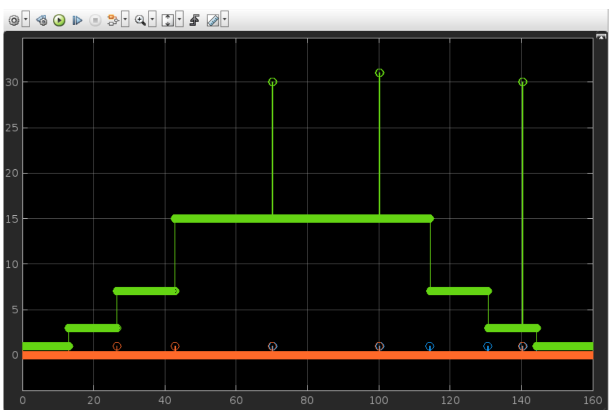


Fig. 37. Encoder: Before elimination of alarm-triggering signal

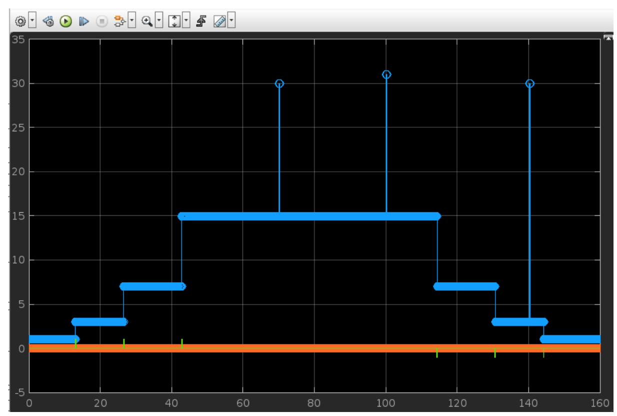


Fig. 38. Encoder: Elimination of alarm-triggering signal

σ_o and σ_a and stops the counting function at the small time interval. After transmitting the alarm events successfully, the encoder restarts counting the function and comes back to the mechanism to detect the water level states. In this way we can transmit alarms which doesn't affect the normal mechanism of decoder. The state flow chart is shown in Fig.40, in which we built the hierarchy of the state flow. The state block 'Normal water level measure' is the normal mechanism of detecting water level changes under the condition that the decoder doesn't receive σ_o and σ_a in a small time interval. If the encoder receives the special sequences, the state block 'Alarm triggering' works and the count function closes. After the decoder produces the output of alarm events, the count function restarts.

In Figure 41, the yellow line shows the water level, the red line shows the alarm1 and the blue line shows the alarm2 (for lines of alarm1 and alarm2, value '1' means having alarm events and '0' means not having alarm events). It can be observed that changes of the water level and changes of the states are independent of alarm events. However there are some challenges we cannot solve. For instance, the alarm events last much longer than the input alarm triggering holding time (more than $50s$) which destroys the water level output of decoder. In figure 40, we put the state block 'Normal water level measure' and the state block 'Alarm triggering' as a relation of mutually exclusive. As mentioned before the decoder stuck in 'Alarm triggering' state block for so long that we lost some shapes of decreases transition of water level, and the water level output of the decoder did not get back to level 0 anymore.

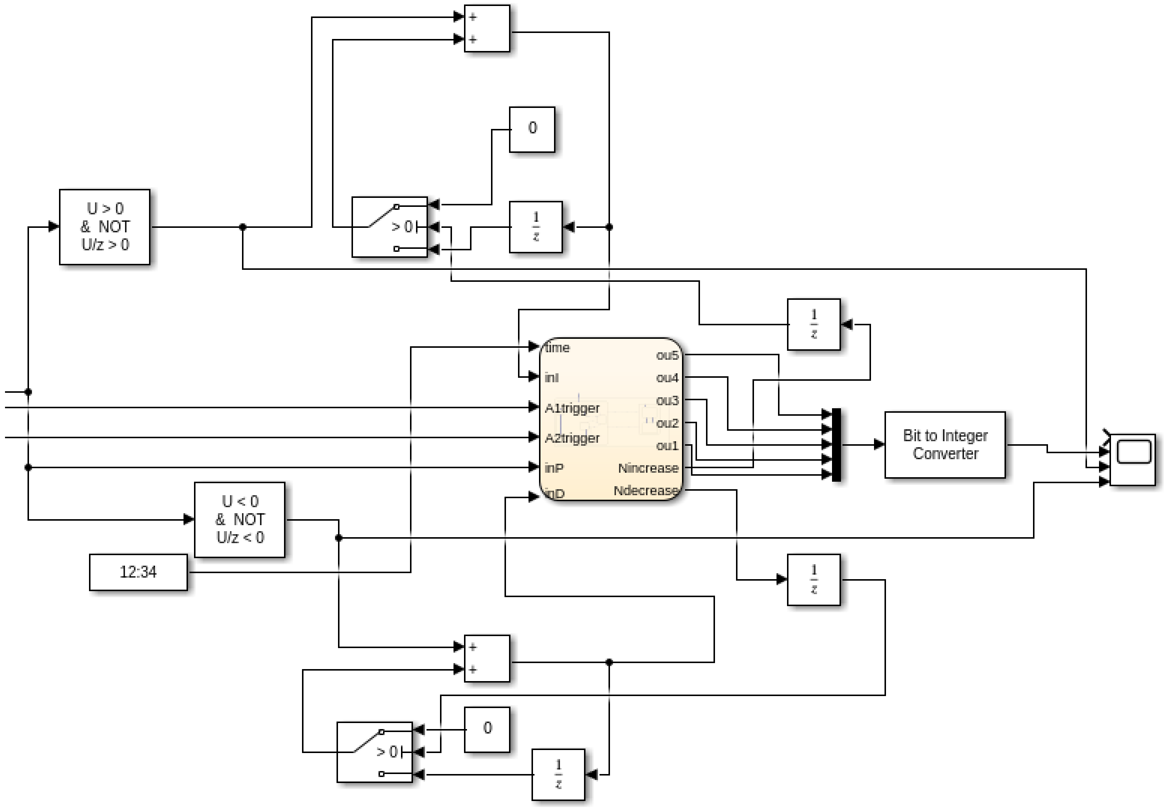


Fig. 39. The Decoder part of send-on-data protocol.

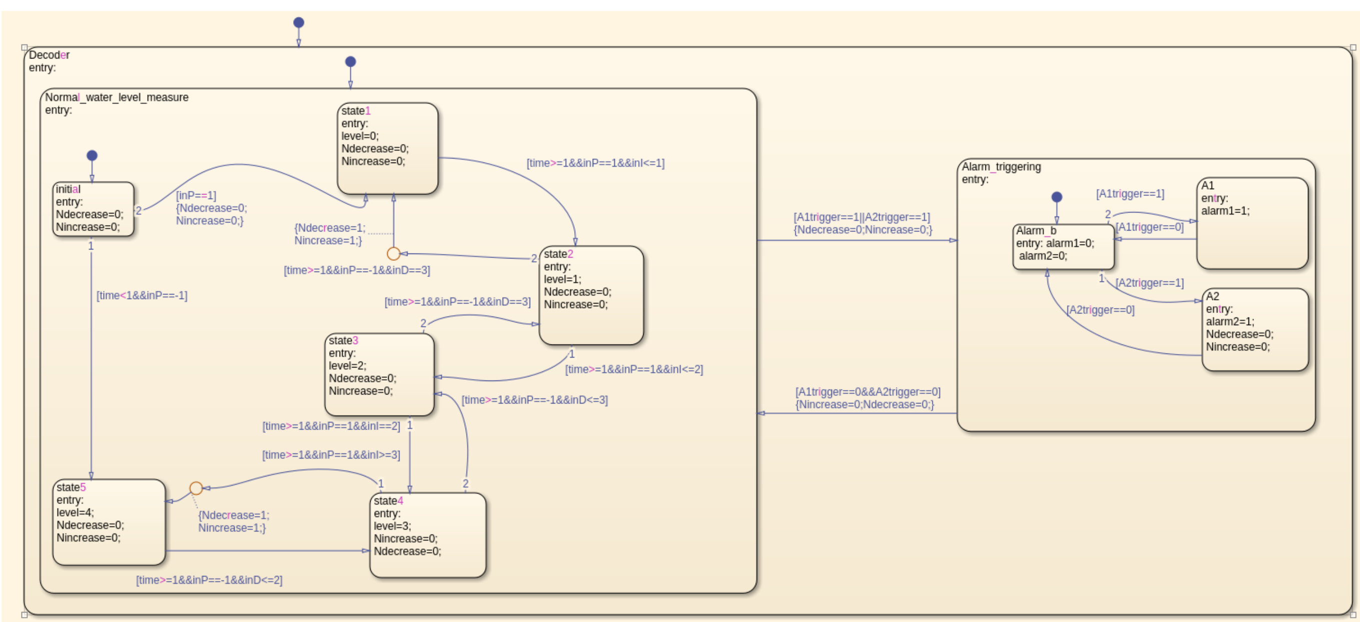


Fig. 40. The state flow chart of decoder.

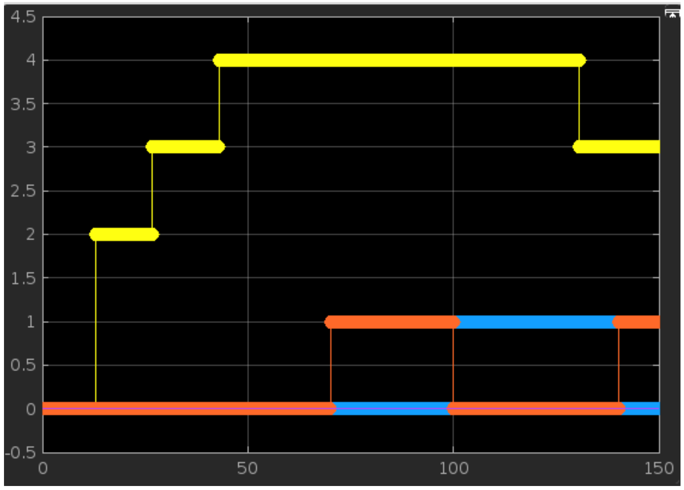


Fig. 41. The simulation result of Decoder part.

III. Hybrid System

Considering now the situation in which the voltage $V(l_1)$ is obtained from the decoder of the received signals, we define the system's behaviour as follows:

- When the tank is empty or any alarm is raised $V(0) = 0$;
- When the level is between $h = 10$ and $h = 20$ cm $V(l_1) = 5V$;
- When the level is above $h = 20$ cm $V(l_1) = 10$;
- The valve ω_1 stays open while $l_1 > 10$ cm and the temperature of the resistor is above some ρT_a, $\rho > 1$, and otherwise the valve closes;
- The valve ω_2 remains closed unless an alarm is raised or $l_1 > 30$ cm, in both cases the valve opens.

We can now construct the following hybrid automata - figure 42. For this part we have as inputs the water level, the temperature of the resistor and the alarm situation that comes from the decoder. The system produces as outputs the values of the voltage and the state of the valves (opened - '1', or closed - '0'). Note that in this exercise we defined s_1 and s_2 as the value of the switch for the valves ω_1 and ω_2, respectively.

* Q = {q0, q1, q2, q3, q4, q5, q6, q7}
* X = R
* Inv(q0) = $\{x \| l_1 \leq 10,\ A_1=0,\ A_2=0\}$
* Inv(q1) = $\{x \| 10 < l_1 \leq 20,\ A_1=0,\ A_2=0,\ T_R \leq \rho T_a\}$
* Inv(q2) = $\{x \| 20 < l_1 \leq 30,\ A_1=0,\ A_2=0,\ T_R \leq \rho T_a\}$
* Inv(q3) = $\{x \| l_1 > 30,\ A_1=0,\ A_2=0,\ T_R \leq \rho T_a\}$
* Inv(q4) = $\{x \| 10 < l_1 \leq 20,\ A_1=0,\ A_2=0,\ T_R > \rho T_a\}$
* Inv(q5) = $\{x \| l_1 > 30,\ A_1=0,\ A_2=0,\ T_R > \rho T_a\}$
* Inv(q6) = $\{x \| 20 < l_1 \leq 30,\ A_1=0,\ A_2=0,\ T_R > \rho T_a\}$
* Inv(q7) = $\{x \| A_1=1 \text{ v } A_2=1\}$

The hybrid automata for the entire system can now be defined as presented in figure 43. The final system is composed of the physical model built in section I, which is controlled by the protocols from the send on delta encoder and decoder systems and the hybrid system from figure 42. The encoder receives the signal from the water level sensor which provides information to the decoder about the level of the water. This information is used in the hybrid system from figure 42 to set the values of the voltage and the state of the valves ω_1 and ω_2. In case an alarm is risen, then the system stops, the voltage is set to

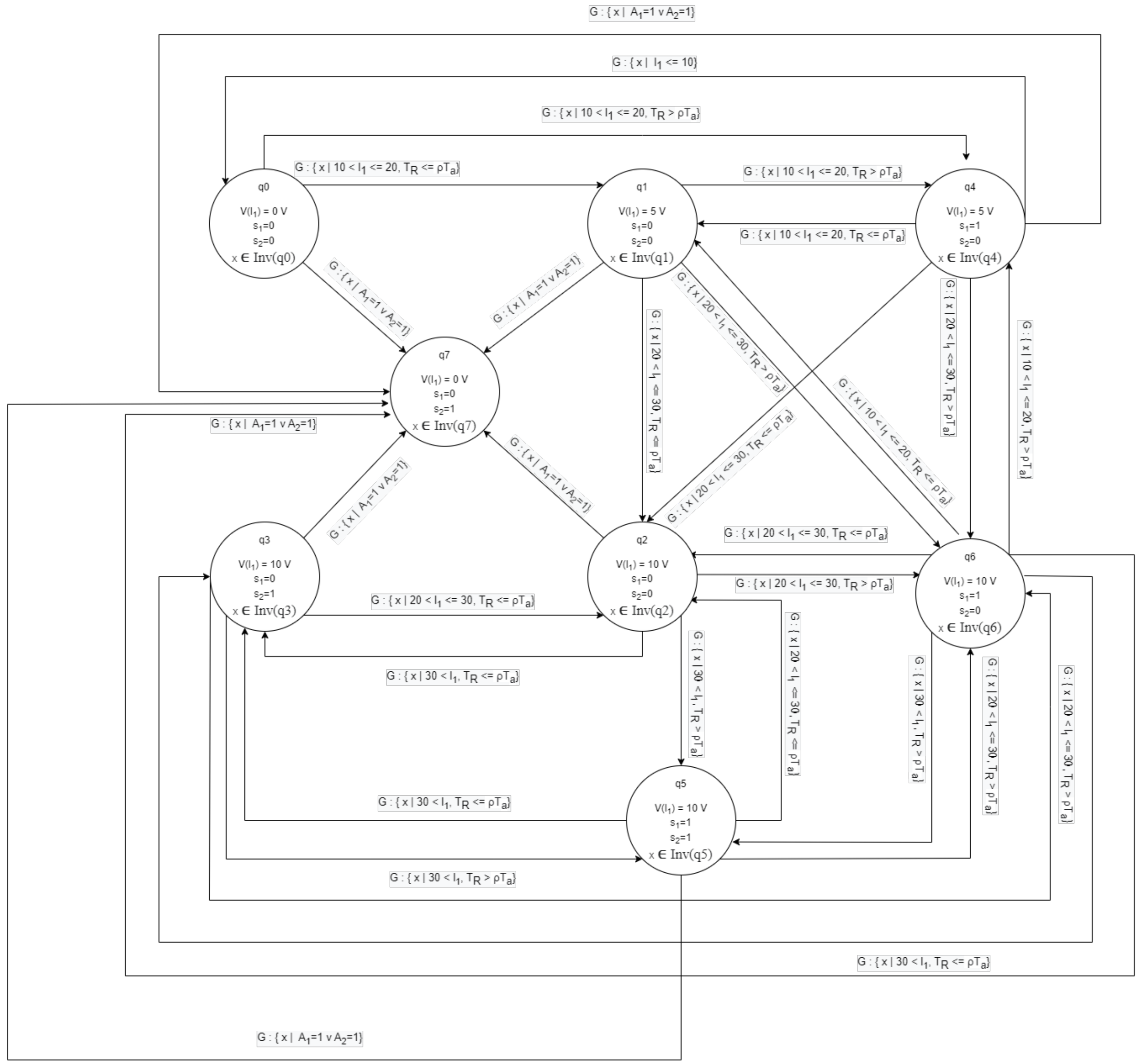


Fig. 42. Hybrid automata for the change in the valves' states and voltage

0 V, and the valve ω_2 is opened so it can drain the water from the tank. It is worth noting that when the tank is full at a water level higher than 30 cm, the valve ω_2 opens, letting the water get out of the tank. If the resistor's temperature is still less than ρT_a, once the water reaches the level between 20 and 30 cm, the valves are closed to let the level of the water rise again. For the case where the temperature of the resistor is always less than the specified constant, the system will only change between those two states (q_3 and q_2). Also note that in some situations the rate of water leaving the tank can be lower than the rate of water entering the tank and that's why even if a valve is opened we can get to higher states of the level of the water in the tank, and not necessarily only go to lower levels.

The complete hybrid system was modeled in Simulink as shown in figure 44. Unfortunately, it was not possible to run simulations with the entire hybrid system.

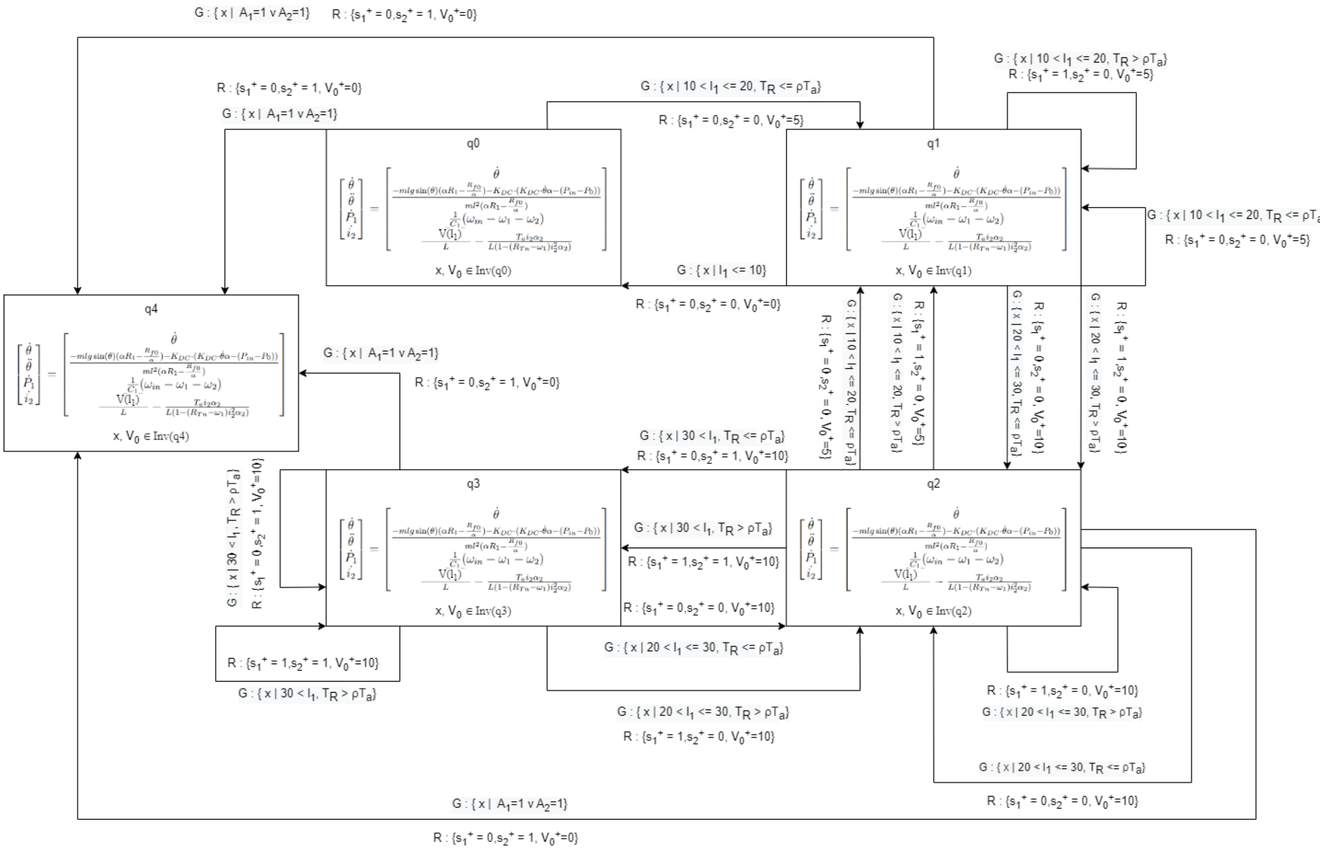


Fig. 43. Hybrid automata for the whole system

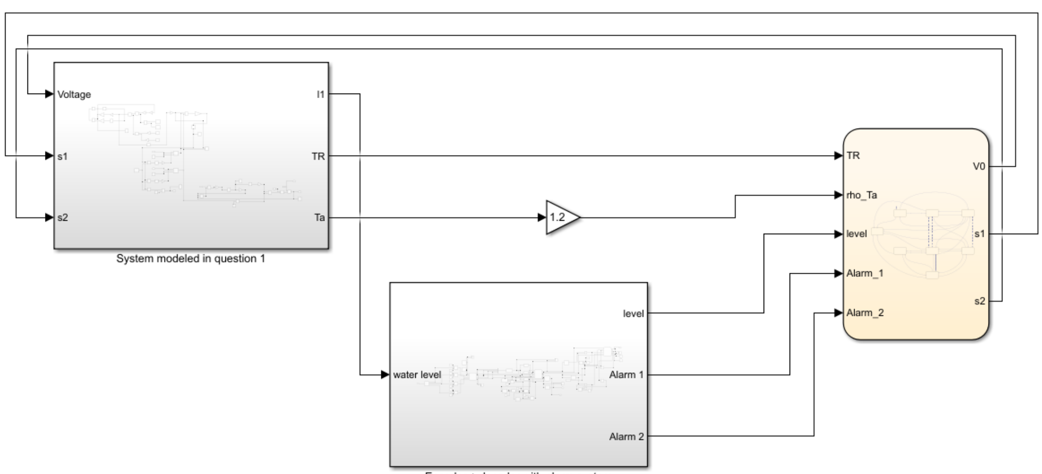


Fig. 44. Hybrid automata for the whole system modeled in Simulink

References

[1] Waterdruk, informatie... - Aquacombi. (2022, November 02). Retrieved from https://www.aquacombi.nl/hydrofoor/waterdruk