

System Dependability: *Lab exercise on MBSA*

Lab reporting

Each question clarifies what are the expected report inputs. Concise answers are welcome.

! Complete the report step by step: some initial results are no more observable at the end of the exercises. Send your report in pdf format before 17/05/2024 to christel.sequin@onera.fr and christophe.frazza@satodev.fr.

Introduction

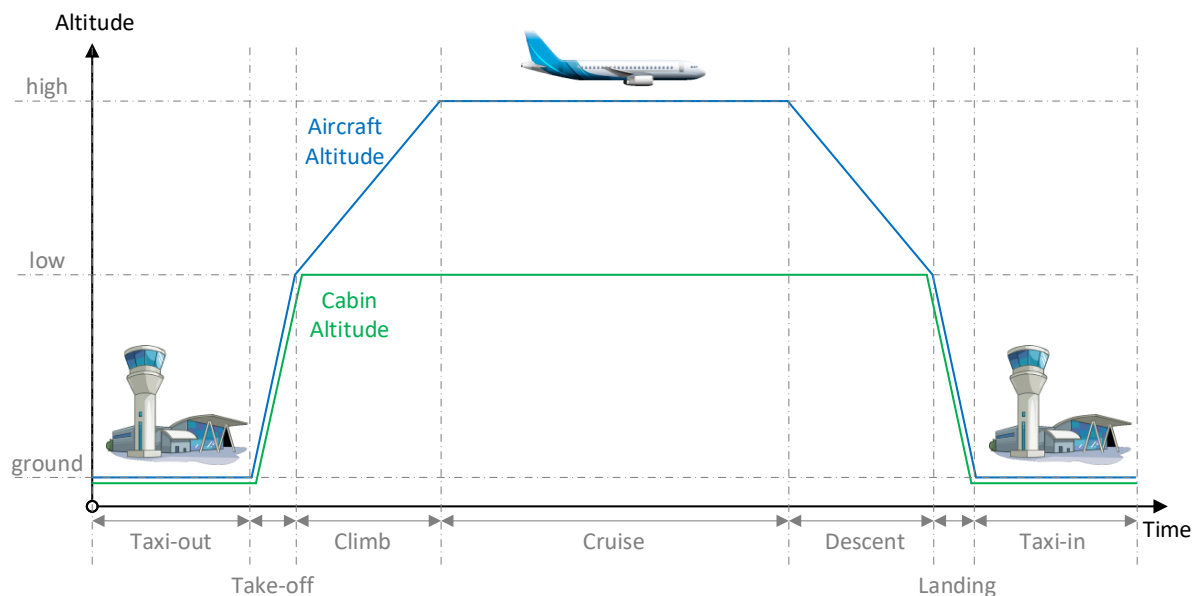
The function of a cabin pressurization system is to control the Cabin Altitude that is felt by the passengers inside the cabin. The system controls airflow valves in order to avoid that Cabin Altitude becomes too high. For this lab exercise, three Aircraft Altitude levels are considered:

- ground (during taxi phases),
- low (end of take-off and beginning of landing phases),
- high (end of climb, cruise and beginning of descent phases).

Valves are open when the Aircraft Altitude is equal to ground or low. In that case, the Cabin Altitude is equal to the Aircraft Altitude.

Valves are closed when the Aircraft Altitude is equal to high. In that case the Cabin Altitude remains equal to low.

In a Normal Flight, Cabin Altitude and Aircraft Altitude have the following profiles:



Cabin and Aircraft Altitudes - Normal Flight

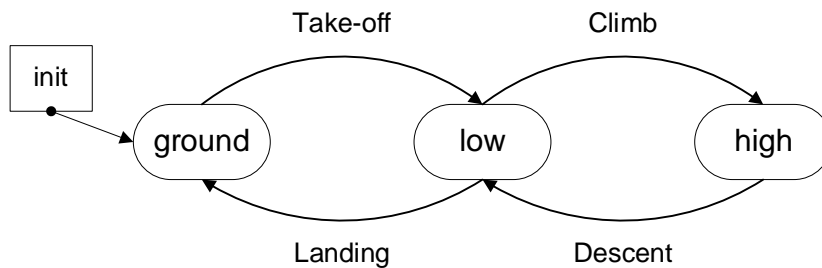
We are interested in the following Failure Conditions:

- FC1: “Cabin Altitude is high”: that could lead to the death of passengers due to the lack of oxygen,
- FC2: “Wrong Cabin Altitude at high altitude”: that occurs when the Aircraft is in cruise and if Cabin Altitudes is different from what is required in a normal flight. This could lead to damage the fuselage and lose the aircraft,
- FC3: “Wrong Cabin Altitude on ground” that occurs when the Aircraft is on ground and the Cabin Altitude is different from what is required in a normal flight. This could lead to damage the doors when opening them, due to sudden pressurization.


Nota : the Failure Condition “*Wrong Cabin Altitude at low altitude*” is not taken into account as “Cabin Altitude is high” is covered by FC1 and as “Cabin Altitude is ground” has only a minor safety impact.

1. Aircraft Altitude Model

In the [Projects](#) tab, [Aircraft](#) family, [Pressurization](#) sub-family, open the [P_1](#) model. The [Aircraft](#) node should model the evolution of Aircraft Altitude during the flight as follows:



Question 1:

1) Complete the transitions of the [Aircraft](#) node in order to model the previous state diagram. To edit the **Aircraft** node, double-click on it and go in the AltaRica code tab. Once you have modified the node, perform [Syntax](#) and [Consistency](#) checks, then test the node using the simulation tool: 

2) Copy the AltaRica code into your Lab report. **Do not forget to comment the code!**

2. Cabin Altitude

Open the [P_2](#) model ([Projects](#) tab, [Aircraft](#) family, [Pressurization](#) sub-family). The AltaRica model contains several nodes:

- The [Aircraft](#) node that has been studied in question 1,
- The [Cabin](#) node that models the evolution of Cabin Altitude,
- The [Controller](#) node that computes the command to open or close the pressurization valve,
- [FC1_Observer](#) (resp. [FC2_Observer](#) and [FC3_Observer](#)) whose output is true when the Failure Condition FC1 (resp. FC2 and FC3) is reached.

Question 2.1:

- 1) Complete the assertions of [FC1_Observer](#), [FC2_Observer](#) and [FC3_Observer](#) nodes.
- 2) Once you have modified the nodes, use the [Sequence generation](#) tool to check if the Failure Conditions can occur (cf “How to” at the beginning of the next page).
- 3) Copy the AltaRica code and the generated sequences into your Lab report.

Question 2.2:

- 1) Modify the assertions of the [Controller](#) node so that the Failure Conditions are never true. You have to perform a computation again to be sure of your modification.
- 2) Copy the AltaRica code into your Lab report.

How to: *Sequence generation*

You have to keep the *P_2* model open. *MBSA* menu, *Sequence generation* functionality:


You can add several targets with .

For each target, you have to specify the file in which results will be stored.

For example:


- create a folder *MBSA_results*
- use files *FCn.seq* (with n=1,2,3)

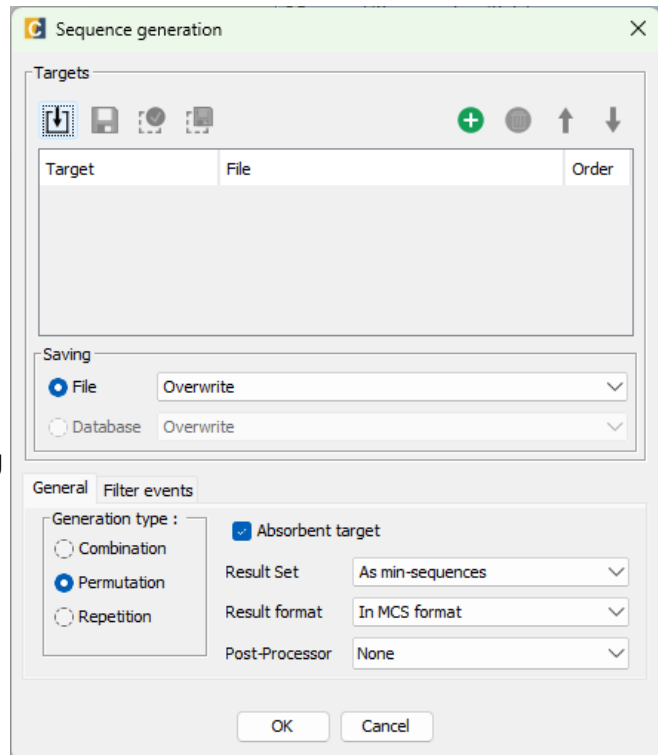
Choose order 9 instead of 3.

Once all the targets have been filled out, you can store this configuration by clicking on the floppy disk .

For example:

Seq_gen_FC1_FC2_FC3.xml

The next time you need to perform the same computation, just load it with: .



3. Faulty Transmitter

Open the *P_3* model. It contains a new node: the *Transmitter* node models a transmission link between the *Controller* and the pressurization valve.

The *Transmitter* node has two failure states:

- *stuck_open* the link permanently sends an opening command to the valve,
- *stuck_closed* the link permanently sends a closing command to the valve.

Question 3:

- 1) Complete the transitions and assertions of the *Transmitter* node.
- 2) Once you have modified the node, use the *Sequence generation* tool to check if the Failure Conditions can occur.
- 3) Copy the AltaRica code and the generated sequences into your Lab report.

4. Failure Condition Avoidance

Open the *P_4* model. The *Aircraft* node has a new input which represents the status of the Transmission link. The pilot can use this new information in order to avoid some flight events that would lead to a Failure Condition.

Question 4:

- 1) Modify the transitions of the *Aircraft* node in order to prevent, as much as possible, the transitions that would lead to the Failure Conditions. Justify the proposed modifications.
- 2) Copy the AltaRica code and the generated sequences into your Lab report.

5. Failure Condition Tolerance

Open the *P_5* model. The system contains:

- 3 *Transmitter* components: *T1*, *T2* and *T3*,
- a *CommandVoter* component,
- a *TransmissionStatusVoter* component.

The *CommandVoter* component receives the commands transmitted by the three links. When received commands are different, the voter computes the majority command (2 over 3) and sends it to the valve.

The behavior of the *TransmissionStatusVoter* component is similar to the *CommandVoter* component, but it works with status values instead of command values.

Question 5:

- 1) Complete the assertions of the *CommandVoter* and *TransmissionStatusVoter* nodes.
- 2) Once you have modified the nodes, use the *Sequence generation* tool to check if the Failure Conditions can occur.
- 3) Copy the AltaRica code and the generated sequences into your Lab report. Comment the results.