# Modelling and Coding Guidelines

The following guidelines shall be applied to MATLAB/Simulink development. They should serve as a best practice approach.

# Documentation

- Add as much comments as you can when implementing new functions. This facilitates future improvements/debugging either by you or other colleagues who will be working in the same model.

## Simulator Files

- File names must not contain any blank spaces, use underscore `_` instead.
- File names may only contain one dot `.` to separate the file name from the file extension.
- File names shall have meaningful names to quickly recognize its function as an entity of the simulator.

## Variable Naming

- The name of a variable shall start with a lower case letter.
  - Exceptions:
    - Rotation matrices shall be name `DCM_<FRAME>_from_<FRAME>`.
- Multiple words shall be separated by underscore `_`.
- The name of a variable shall display its logical inclusion hierarchy, starting with the top level, i.e. `mass_fuel_initial`.
- Variables shall be expressed in the International Unit System.
- The following physical variables have predefined names:
  - Forces shall be named `for_<name>_in_<frame>`.
  - Torques shall be named `torq_<name>_in_<frame>`.
  - Positions shall be named: `pos_<object>_wrt_<FRAME>_in_<FRAME>`.
  - Velocities shall be named: `vel_<object>_wrt_<FRAME>_in_<FRAME>`.
  - Accelerations shall be named: `acc_<object>_wrt_<FRAME>_in_<FRAME>`.
  - Euler angles vectors shall be name: `eul_<FRAME>_from_<FRAME>`.
  - Quaternions shall be named: `quat_<FRAME>_from_<FRAME>`.
  - Angular rates shall be named: `omg_<object>_wrt_<FRAME>_in_<FRAME>`.
  - Angular accelerations shall be named: `alp_<object>_wrt_<FRAME>_in_<FRAME>`.
  - Use `lat_geoc` for geocentric latitude ($\varphi_{\text{geoc}}$).
  - Use `lat_geod` for geodetic latitude ($\varphi_{\text{geod}}$).
  - Longitude shall be named `lon` ($\lambda$).
  - Radial distance shall be named `r`.
  - Altitude shall be named
    - `alt` if expressed with respect to the reference sphere.
    - `alt_geod` if expressed with respect to the reference ellipsoid.
  - 1st time derivative: `<variable name>_dot`,
  - 2st time derivative: `<variable name>_dot2`,
  - 3rd time derivative: `<variable name>_dot3`, etc.
  - The norm of a vector shall remove `_in_<frame>` and instead have the suffix `_norm`
  - Euler angles
    - The default rotation sequence is $Z \to Y \to X$. Otherwise, it will be clearly indicated.
    - Roll ($\phi$), pitch ($\theta$) and yaw ($\psi$) angles are called `roll`, `pitch` and `yaw`, respectively.
    - The angles can be expressed with regards to any Geographic frame `G`, i.e., North-East-Down adds `_NED`, East-North-Up adds `_ENU`, Up-East-North adds `_UEN`, etc.
    - By default the value of `G`-frame is `NED`. Otherwise, it will be clearly indicated.
  - Aerodynamic angles:
    - The angle of attack ($\alpha$) shall be named `aoa`.

    - The angle of sideslip ($\beta$) shall be named aos.
    - The total angle of attack ($\alpha_{\text{tot}}$) shall be named aoa_tot.
    - The windward meridian angle ($\phi_\alpha$) shall be named wma.
    - The projections of $\alpha$ and $\beta$ onto $XY_\mathtt{B}$ and $XZ_\mathtt{B}$ planes, respectively, are called pitch and yaw angles of attack ($\alpha_1$ and $\alpha_2$), they shall be named aoa1 and aoa2.
    - The aerodynamic angles computed wrt. the aerodynamic (air-based) velocity do not have any index.
    - The aerodynamic angles computed wrt. the trajectory (ground-based) velocity have the sub-index _T (e.g., $\alpha_\mathtt{T}$ will be aoa_T).
    - The aerodynamic angles between the trajectory and aerodynamic velocity are called wind aerodynamic angles, they will have the sub-index _W (e.g., $\alpha_\mathtt{W}$ will be aoa_W).
  - Flight-path angles:
    - The flight-path angle ($\gamma$) shall be named fpa.
    - The flight-path azimuth ($\chi$) shall be named faz.

    - The trajectory flight-path angles (computed wrt. the trajectory velocity) do not have any sub-index .
    - The aerodynamic flight-path angles (computed wrt. the aerodynamic velocity) have the sub-index _A (e.g., $\gamma_\mathtt{A}$ will be fpa_A).
    - The angles between the trajectory flight-path and aerodynamic flight-path are called wind flight-path angles, they will have the sub-index _W (e.g., $\gamma_\mathtt{W}$ will be fpa_W).
    - The flight-path angles can computed wrt. any of the G frames (NED, UEN, ENU, etc.).
    - By default, the value of G-frame is NED. Otherwise, it will be clearly indicated.
  - Bank angle:
    - The bank angle ($\sigma$) shall be named aob.

## MATLAB

- The usage of global variables is forbidden.
- Vectors shall be always implemented as column vectors, series/arrays of vectors shall be provided as an horizontal array of N vectors.
- Variables should be organized within meaning full structures.
- For variable organization follow an object oriented naming approach.
- In general, all functions should be implemented as Embedded MATLAB function if possible.
- Functions that are part the on-board software shall be implemented in Embedded MATLAB.
- Algebraic manipulations, transformations, and other mathematical operations shall be implemented in Embedded MATLAB.
- Embedded MATLAB functions shall contain the directive `%#codegen` as the first line of code below the function definition.
- Embedded MATLAB functions that do fork the control flow shall initialize all output arguments after defining the `%#codegen` directive.
- Embedded MATLAB functions shall not use the `varargin` language feature.
- Function specific to the must start with the prefix `CLT_Xyz`.

## Simulink

### Naming Conventions

- Simulink libraries shall have the extension `*.slx`
- Simulink simulation models shall have the extension `*.slx`
- Block names shall be always activated and placed below the block.

### Inputs Outputs

- Input ports shall be placed on the outermost left side.

- Outputs ports shall be placed on the outermost right side.
- Inputs and outputs shall have the same names as the incoming/outgoing signal.

### Color Coding

- The foreground color is always black.
- The background color of a Simulink block shall follow the following color coding to make the model more readable. The same color coding can be found for labeling Git issues.
  - `#61bdfc` On board computer blocks (Light Blue)
  - `#00d1d1` Actuator and sensor blocks (Cyan)
  - `#00d100` Visualization and data logging blocks (Green)
  - `#e8d152` Documentation and constants (Yellow)
  - `#ffffff` Plant and Environment blocks (White)
  - `#00d100` Inputs ports (Green)
  - `#e60000` Output ports (Red)

### Signal Routing, Model Layout, Vector and Buses

- The control flow shall be routed from left to right.
- Parallel blocks or sub-systems shall be arranged from the top to the bottom.
- Use subsystems to structure the model upon a functional decomposition of the algorithm. Do not use them to save space.
- Physically valid vectors shall be implemented as Simulink vectors. Do not use Simulink vectors for grouping elements of different logical origin.
- Vectors shall always be column vectors. Make sure this option is also active using the `constant` block.
- Scalars (or vectors) of different logical origin shall be grouped using Simulink buses.
- Buses shall be used hierarchical to display the origin of each signal.
- Signals shall be named where they are created.
- Signals shall be grouped where it is useful to reduce visual clutter.
- Signals shall be routed using right angles.
- Signals shall be routed such that cross sections are prevented as much as possible.
- The option “Show propagated signals” within “Signal properties” shall be activated.
- The “GoTo” block shall not be used. The only exception is for passing data outside the control flow, i.e. for debugging and visualization.
- Use Ground and Termination blocks for unconnected signal lines.

### MATLAB Functions

- The dimensions of input and output arguments shall be explicitly defined using the Ports and Data Manager. In particular, vectors shall always be enforced as column vectors.

### Simulink Library Blocks

- The top level block in a library model must be a subsystem or a MATLAB function.
- The block mask shall allow to fully parametrize the library block. In particular no workspace variables may be accessed from within the block. All input must be passed through the mask.
- In the mask avoid using select boxes or radio buttons.
- A default value shall be given for each mask parameter to allow the user to run a model using the library block without further configuration. A larger data amount (e.g. model coefficients, look-up tables) shall be provided via a file.