# µXAV

# Communication and Fault Tolerance Specification

(Draft E.L 02/2018)

# Introduction

The specification traits are aggregated like a layer 0 specification, for convenience;

But they are as detailed as a layer 1 specification.

This interim document is intended to be split into parts to be inserted in the three layer1-functional specifications.

The draft addresses three functions at increment 4, i.e as fully fledged specification:

* F\_CM: Communications,
* F\_FT: Fault Isolation and Recovery. Fault detection is ensured by F\_HM. See the list of failure modes to be detected in the spreadsheet RESSAC\_muXAV\_FailureModes\_FTconfigurations.xlx. The function inhibits (isolation) and activates the redundancies (recovery – fail operational or fail safe design).
* F\_AL: **A**ctivation **L**ogic function. This function was omitted in the first versions of system functional analysis. Specifying the functional aspects of task scheduling made the need for this additional function emerge. It is in charge of computing the activation Booleans of all the functions and sub-functions.

This draft primarily presents the design rationale (intentional specification), and strives to be as comprehensive as possible on the designated elements (extensional specification).

# MMS

## Channel 1

### F\_AL

**Inputs**

* CH1\_BOOT\_OK(Boolean) from the executive layer and F\_HM
* CH1\_BOOT\_FAIL(Boolean) from the executive layer and F\_HM
* FT\_configuration({AllOK, M1, …..,M12P12B12})

**Outputs**

* An activation Boolean per function and sub-function F\_XY\_Activation(Boolean).

**Contract**

* After power-on at CP, boot of CH1 succeeds or fails. There are two input signals for distinguishing boot in progress from failure (not failed = (booting or booted)).
* Ensures that only the necessary functions are activated, and that all the necessary ones actually are, in conformance with the FT configuration table.

**NOTE:** All failures in the case study are assumed to be possibly temporary: from transients (a few seconds) to lasting a few minutes or hours.

**Behaviour**

An application-level automaton, event-driven or periodically activated, enforces MMS.CH1’s part of the global table (Excel table).

The activation Booleans are used by the executive layer to schedule accordingly. Then, a second level of scheduling and software architecting can split some function-tasks to meet the deadline constraints on limited computing resources.

Within a given rate, or a given event flow, the data dependencies between the functions active on a CPU must be converted into scheduling constraints. See the functional specifications for the data flow dependencies.

The use case does not feature distributed scheduling to meet the end-to-end timing constraints. Design of the local schedulers must meet the global timing objectives without any distributed scheduling policy. The only distributed scheduling aspect is indirect: F\_AL is partly driven by F\_FT, which applies a global function activation logic that drives the sets of schedulable tasks.

### F\_CM

**Inputs**

* **Analog signals**
  + List to be given (see Layer 0 Architectural Specification)
* **RF Receiver signals**
  + List to be given (see Layer 0 Architectural Specification)
* **Digital signals**
  + List to be given (see Layer 0 Architectural Specification)

**Outputs**

* **Analog signals**
  + List to be given (see Layer 0 Architectural Specification)
* **RF Receiver signals**
  + List to be given (see Layer 0 Architectural Specification)
* **Digital signals**
  + List to be given (see Layer 0 Architectural Specification)

**Contract**

Formalization of the contract needs:

* MMS computer’s architecture (the chosen technological solution), especially the I/O devices and their associated drivers,
* The technology of the sensors and of all the physical devices connected to the analog links (for signal modulation/demodulation specification),
* The technology of the data bus, and of its associated emitter/receiver hardware and software.

These off-the-shelf components lead to a contract-based approach restricted to:

* Making explicit the assumptions to be satisfied by the application for proper functioning of the COTS (usage model, fault model),
* Making explicit the communication QoS expected by the application (Guarantees)

The contract may address:

* Data type and format issues,
* Message ordering issues (causal ordering on event-carrying messages that are induced by the application),
* Losslessness issues,
* Timeliness issues.

**Behaviour**

Technology dependent, so not specified.

Can be structured into six modules (Read , Write) x (RF, Analog, Digital).

Each of these modules can been split further to give more scheduling flexibility.

Layer 0 architectural specification specifies the rates and types of information to be transferred.

Management of the sensor and actuator analog links is the most critical part of the I/O services.

### F\_FT

**Inputs**

* CH1 FAIL
* CH2 FAIL
* InterCom FAIL
* EPS.ECU1 FAIL
* EPS.ECU2 FAIL
* HBS.ECU1 FAIL
* HBS.ECU2 FAIL

**Outputs**

* FT\_configuration({AllOK, M1, M2, M12,…..,M12P12B12}

**Contract**

It consists in MMS.CH1’s contribution to the AV-level regulatory safety objective: “no single failure”.

The FT-contract compositionality rationale is the following *(uniformly applied to all the systems and to all the ECUs therein)*:

* Local logics:
  + The ECU is both self-tested and externally monitored (functionally),
  + The ECU receives the FAIL statuses from the resources its FT-configuration depends on,
  + The ECU applies its local part of the global FT-activation logic (see the Excel sheet: 64 configurations),
* Global logics:
  + The 6 communication contracts ensure that the FT-events are lossless and timely transmitted:
    - Message losses would lead to mode inconsistencies across systems (consensus problem).
    - Untimely transmissions would potentially lead to flight control instabilities, depending on the perturbations that occur during the system wide transition. A deterministic approach to substantiation is mandatory (worst case analysis): the regulatory objective (see CS25.1309.b) is not in the probabilistic group (CS25.1309.a).
  + Within a time window all the ECUs transition “simultaneously” (some sort of multi-system transaction). Each of them activates and deactivates the FT-specified functions consistently with its received FAIL events. Global FT-mode consistency is ensured by a correctness property of the set of ECU-supported FT-tables against the global FT-table (Excel spreadsheet).
  + A fault separation assumption is made at AV-level to discard the case of resource failure during a system-wide transition. No failure event is lost locally (they are buffered). But it is assumed that there is no overlap of system-wide FT-transitions. The multi-system may not be robust to such overlaps (such robustness is not required but it is better if it ensured!) This assumption is validated by reliability and probabilistic arguments. They are highly sensitive to independence assumptions (see CMA analyses, including CCA, CEA, PRA end ZSA of the Safety assessment process). Validity coverage of the assumption should be addressed by development. Some overlap protection mechanism should be added if the assumption failure rate is not apportioned to the quantitative safety objectives.
  + The perturbation separation hypothesis does not apply to wind gusts. In case of bad weather conditions overlap of system wide FT-transitions and of mechanical bursts on the AV body are almost certain. It is one of the main objectives of the use case to explore how a continuous-time, synchronous[[1]](#footnote-1), closed-loop control correctness argument (increment 1 and 2) can be adapted to a formal argument applying to the hybrid-time, distributed, synchronous[[2]](#footnote-2), fault-tolerant version of the closed loop (increment 3 and 4).

**Behaviour**

* Mission Abortion:
  + P12 case
  + M12 case
* Shutdown:
  + Boot FAIL
  + CH1 FAIL
* Recovery:
  + CH1 FAIL disappears (auto-test case only – supposed to be a hardware-based mechanism)

## Channel 2

Identical to Channel 1, save the fact that when CH1 is up CH 1 is the master and CH2 is in slave mode.

In slave mode CH2 only executes minimal communications and computations on flight control (active redundancy for continuous control in case of CH1failure).

### F\_AL

### Identical to Channel 1 when Channel 2 is the master (CH1 failed).

In slave mode:

* F\_CM is active
* F\_FC is active (limited to the monitoring part of the flight control laws - safety envelope).
* F\_HM is active

### F\_CM

* Limited to exchange of the sensor values, the AV state vector and the failure events

### F\_FT

* Switches between master/slave mode, or shutdown/recovery

# EPS

## ECU1

### F\_AL

### F\_CM

### F\_FT

## ECU2

### F\_AL

### F\_CM

### F\_FT

# HBS

## ECU1

### F\_AL

### F\_CM

### F\_FT

## ECU2

### F\_AL

### F\_CM

### F\_FT

1. ‘Synchronous programming languages’ interpretation [↑](#footnote-ref-1)
2. ‘Distributed algorithms’ interpretation, i.e asynchrony with bounded delays (e.g Nancy Lynch « Distributed Algorithms »). [↑](#footnote-ref-2)