4x Software Manual

Release 6.8

Embention

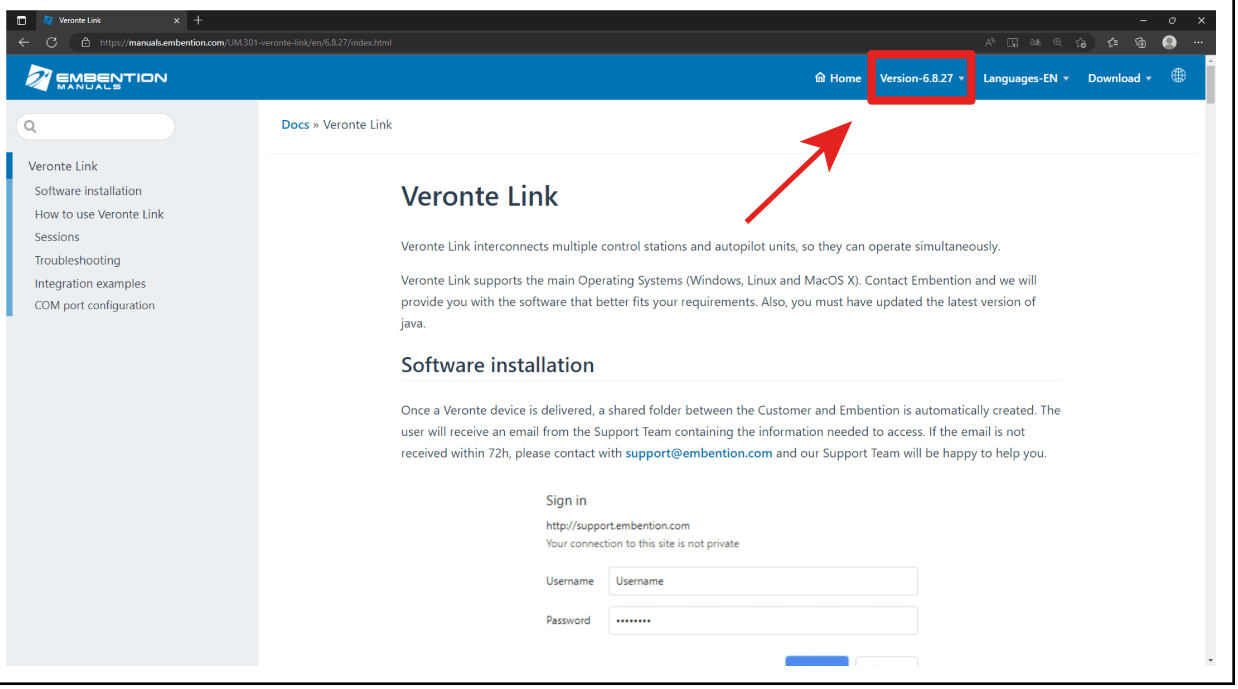
2023-08-09

CONTENTS

In this manual the user can consult a brief description of all the applications created and designed to work together with the **Veronte Autopilot 4x**.

In addition, links are available to access the manuals for each application.

Warning: Select your version before reading any user manual for software. The following image shows where to select a version from any Embention user manual.



CHAPTER

ONE

SOFTWARE APPLICATIONS

1.1 Veronte Link

Each inner **Autopilot 1x** and the **Arbiter** must be configured individually. First of all, a connection to a computer is required employing **Veronte Link**. All devices can be connected one by one or simultaneously, since **Veronte Link** is able to manage multiple connections.

For more information, read the user manual for Veronte Link.

1.2 1x Software Manual

To operate individually any **Veronte Autopilot 1x**, read the 1x Software Manual, since all software applications are applicable to **Autopilot 4x**.

1.3 1x PDI Builder

After that, each **Autopilot 1x** must be configured. For more information, read the user manual for 1x PDI Builder.

1.4 4x PDI Builder

4x PDI Builder allows to configure the **Arbiter** communications, including CAN buses and input/output signals. It also configures the criterion to select the autopilot which controls the aircraft.

For more information, read the user manual for 4x PDI Builder.

CHAPTER

TWO

LIST OF PDI ERRORS

This section shows the errors that can be displayed by **Veronte Autopilot 4x**. The rest of errors can be read in the List of PDI errors section of 1x Software Manual.

Code	Nº	Explanation
pdi_arbitration	10000	Error ID for Arbitration cfg.
pdi_arbitration_can	10001	Error ID for Arbitration_can cfg.
pdi_arbitration_can1	10002	Error ID for Arbitration_can cfg.
pdi_arb_cfg0	10003	Error ID for Arb cfg preferred ap out of range.
pdi_arb_cfg1	10004	Error ID for Arb cfg method out of range.
pdi_arb_cfg2	10005	Error ID for Arb cfg tmin out of range.
pdi_arb_cfg3	10006	Error ID for Arb cfg hysteresis out of range.
pdi_arb_init_time	10011	Error ID for Arbiter Power Init Time less than 0.
pdi_arb_varcfg	10013	Incorrect arbiter variable configuration.

CHAPTER

THREE

LISTS OF VARIABLES

This section shows the variables employed exclusively by **Veronte Autopilot 4x**. The rest of variables can be read in the Lists of variables section of 1x Software Manual.

3.1 Activation System Error bits

The **System Error** variable is indicated by bit number 7. This bit checks whether the system is running properly. If one of certain malfunctions occur, the **System Error** will be set as 0 and the FTS will be activated. Othwerwise, if everything is OK, it will remain as 1.

Warning: This bit works different for **Autopilot 1x** and **Arbiter**. This explanation is for **Arbiter**.

The **System Error** will be triggered and remain as 0 if one of the following unwanted events happens:

- An error occurred with **System power up** according to bit 12.
- RAM allocation is in error state due to try using more memory than available, this is indicated with a 0 on bit 8.
- CAN A bus is not working, hence bit bit 73 is set as 0.
- CAN B bus is not working, hence bit bit 74 is set as 0.
- One of the internal voltages is not in range.
- There is not any autopilot alive.
- Task frequency is not correct.
- Acquisition task frequency is not correct.

3.2 Bit Variables

Note: Variables marked with "*" are stored in **Autopilot 1x**, but they represent information transmitted from the **Arbiter**.

ID	Name	Description
20*	4XV System	System Error bit (bit 7) from the Arbiter
21*	4XV System Power up BIT	Autopilot 4x power up - 0 for fail, 1 for running OK.

continues on

Table 1 – continued from previous page

ID	Name	Description
22*	4XV PDI	PDI files for Autopilot 4x - 0 for wrong PDI configuration, 1 for OK.
23*	4XV Memory allocation	RAM allocation - 0 for trying to use more than available memory, 1 for OK.
24*	4XV CAN-A BUS OFF	Autopilot 4x CAN A bus - 0 for error, 1 for running OK.
25*	4XV CAN-B BUS OFF	Autopilot 4x CAN B bus - 0 for error, 1 for running OK.
26*	4XV C1 arbiter	Main Task of CP1 in Autopilot 4x - 0 for error, 1 for running OK.
27*	4XV Acquisition arbiter	Autopilot 4x acquisition task in real time - 0 for error, 1 for OK.
28*	4XV Power A	State of power supply for Autopilot 4x - 0 for error, 1 for OK.
29*	4XV not in maintenance mode	1 for NOT in mantenance mode - 0 for maintenance mode.
30	4XV Alive 1	Indicates whether Autopilot 1x number 1 is sending status messages or not - 0 for dead
31	4XV Alive 2	Indicates whether Autopilot 1x number 2 is sending status messages or not - 0 for dead
32	4XV Alive 3	Indicates whether Autopilot 1x number 2 is sending status messages or not - 0 for dead
33	4XV Alive 4 external	Indicates whether external Autopilot is sending status messages or not - 0 for dead, 1 fo
34	4XV Ready 1	Inner Autopilot 1x number 1 state - 0 for not ready, 1 for ready.
35	4XV Ready 2	Inner Autopilot 1x number 2 state - 0 for not ready, 1 for ready.
36	4XV Ready 3	Inner Autopilot 1x number 3 state - 0 for not ready, 1 for ready.
37	4XV Ready 4 external	External Autopilot 1x state - 0 for not ready, 1 for ready.
38	4XV Arbitrating	Arbiter state - 0 for not ready, 1 for ready.
39	4XV File Open Error	System file manager state - 0 for error, 1 for running OK.
40	4XV PDI version not compatible	PDI files state - 0 for not compatible with current version, 1 for compatible.
41	4XV Stack usage FAIL	0 for memory overflow allocated for local variables, 1 for OK.
42	4XV PWM1 GPIO Off	GPIO/PWM 1 Value to read - 0 for OFF, 1 for ON.
43	4XV PWM2 GPIO Off	GPIO/PWM 2 Value to read - 0 for OFF, 1 for ON.
44	4XV PWM3 GPIO Off	GPIO/PWM 3 Value to read - 0 for OFF, 1 for ON.
45	4XV PWM4 GPIO Off	GPIO/PWM 4 Value to read - 0 for OFF, 1 for ON.
46	4XV PWM5 GPIO Off	GPIO/PWM 5 Value to read - 0 for OFF, 1 for ON.
47	4XV Watchdog Error	*For version 4.7 or higher* - 0 for watchdog signal is not read correctly, 1 for OK
124	4XV Vcc for Arbiter CPU Error	Power state of CPU Arbiter - 0 for error, 1 for OK.
125	4XV Vcc-A Error	State of redundant power supply A - 0 for error, 1 for OK.
126	4XV Vcc-B Error	State of redundant power supply B - 0 for error, 1 for OK.
127	4XV Vcc-1 Error	Power supply for inner Autopilot 1x number 1 - 0 for error, 1 for OK.
128	4XV Vcc-2 Error	Power supply for inner Autopilot 1x number 2 - 0 for error, 1 for OK.
129	4XV Vcc-3 Error	Power supply for inner Autopilot 1x number 3 - 0 for error, 1 for OK.
230-293	4XV Custom msg 1-64 Rx Error	Custom message of Arbiter - 0 for timeout, 1 for OK.

3.3 Real Variables (RVar) - 32 Bits

ID	Name	Units/Values	Description
1350	4XV ADC0 Converted Value	customType	Reserved variables for future versions
1351	4XV ADC1 Converted Value		
1352	4XV ADC2 Converted Value		
1353	4XV ADC3 Converted Value		
1354	4XV ADC4 Converted Value		
1355	4XV ADC5 Converted Value		
1356	4XV ADC6 Converted Value		
1357	4XV ADC7 Converted Value		
1358	4XV ADC8 Converted Value		
1359	4XV ADC9 Converted Value		
1360	4XV ADC10 Converted Value		
1361	4XV ADC11 Converted Value		
1362	4XV ADC12 Converted Value		
1363	4XV ADC13 Converted Value		
1364	4XV ADC14 Converted Value		
1365	4XV ADC15 Converted Value		
1366	4XV Autopilot 1 Score	customType	Score of Autopilot 1x number 1
1367	4XV Autopilot 2 Score	customType	Score of Autopilot 1x number 1
1368	4XV Autopilot 3 Score	customType	Score of Autopilot 1x number 1
1369	4XV Autopilot External Score	Decimal	Score of external Autopilot 1x

3.4 Integer Variables (UVar) - 16 Bits

ID	Name	Description
53	4XV Veronte ID	ID of the Autopilot 1x for the redundant configuration (0 - 3).
54	4XV Veronte CAP	Current Autopilot 1x selected. If the Autopilot has version 4.7 or higher, it is obtained from MUX readings, otherwise it is copied from VAR 55 (in this table).
55	4XV Veronte selected	Autopilot 1x selected.
56	4XV Config manager status	Configuration manager state (flash, sd or safe).
57	4XV File system status	State error for DFS2 file system.
58	4XV CAN-Serial 0 frames dropped	Lost messages during transformations CAN to Serial 0.
59	4XV CAN-Serial 1 frames dropped	Lost messages during transformations CAN to Serial 1.
60	4XV ADC channel 0	Reserved variables for future versions.
61	4XV ADC channel 1	
62	4XV ADC channel 2	
63	4XV ADC channel 3	
64	4XV ADC channel 4	
65	4XV ADC channel 5	
66	4XV ADC channel 6	
67	4XV ADC channel 7	
68	4XV ADC channel 8	
69	4XV ADC channel 9	
70	4XV ADC channel 10	
71	4XV ADC channel 11	
72	4XV ADC channel 12	
73	4XV ADC channel 13	
74	4XV ADC channel 14	
75	4XV ADC channel 15	

CHAPTER

FOUR

CAN BUS PROTOCOL

CAN message structure is defined by two main parts: **cmd** and **data**.

1. **cmd (8 bits - 1 byte)**: First byte refers to the **Message Type**.

   Messages Type are defined as follows:

Type	Value	Description
t_arbitration	0	Arbitration message
t_version	1	Version request / response
t_pwm_0_3_set	2	PWMs 0 to 3
t_pwm_4_7_set	3	PWMs 4 to 7
	4	Reserved
t_esc_tm	5	Scorpion Tribunus ESC telemetry data
t_esc_tm2	6	Jeti ESC telemetry data
t_bec_tm1	7	Jeti BEC telemetry data 1
t_bec_tm2	8	Jeti BEC telemetry data 2
t_temp_tm	9	Jeti Temperature sensor telemetry data
t_mcu_cmd	10	Lift MCU battery command
t_pwm_8_11_set	11	PWMs 8 to 11
t_pwm_12_15_set	12	PWMs 12 to 15
t_pwm_16_19_set	13	PWMs 16 to 19
	14	Reserved
	15	Reserved
t_cmd_maint	16	Command to go to Maintenance Mode
t_stick_sel	17	Command for Stick selection
t_mcu_tm1	18	Lift MCU telemetry data 1
t_mcu_tm2	19	Lift MCU telemetry data 2

Note: All these *Message Type* are defined as a “Matcher” in the CAN custom messages configuration. For example, for PWMs 0-3, the *Message Type* will be configured as follows:

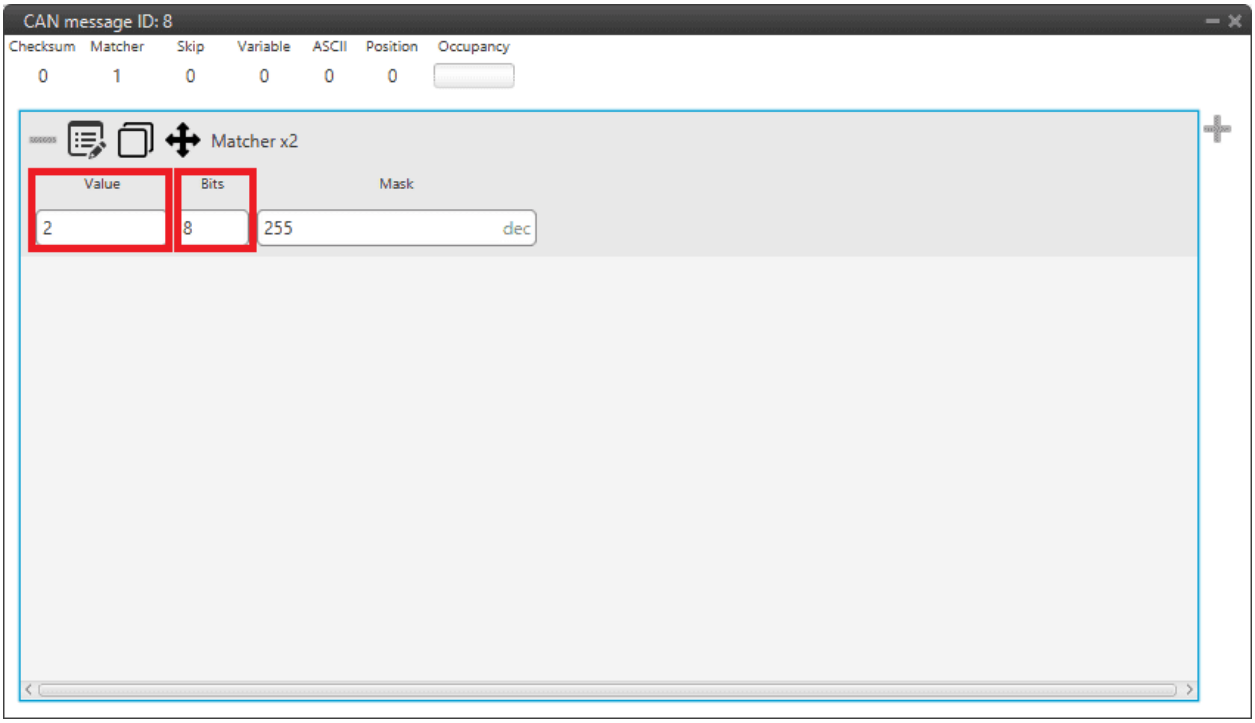


Fig. 1: **Message Type example**

- **Value**: **2**. This is because it is the value for the message for PWMs 0 to 3 (it is **indifferent to the PWM number**).
- **Bits**: **8**. This is because the *Message Type* is an 8-bit message.

---

2. **data (up to 56 bits - 8 bytes)**: The following bytes refer to the **Message data**.

The following examples include complete messages, so each beginning corresponds to *Message Type*.

4.1 Arbitration messages

4.1.1 Status Message

Status message summarizes the status of all autopilots. This message is a **producer** in the **4x PDI Builder** configuration, since it is created from the CAN messages of all three or four autopilots.

Byte	Position	Value	Description
0	0 - 7	0x00	Header
1	0 - 7	0xFF	Status message Header
2	0 - 6	0 - 3	Selected AP (0 = AP1, 1 = AP2, 2 = AP3, 3 = External AP)
	7	0 - 1	Arbitration OFF/ON
3	0	0 - 1	AP1 Alive
	1	0 - 1	AP2 Alive
	2	0 - 1	AP3 Alive
	3	0 - 1	External AP Alive
	4	0 - 1	AP1 Ready
	5	0 - 1	AP2 Ready
	6	0 - 1	AP3 Ready
	7	0 - 1	External AP Ready
4	0	0 - 1	CBIT. System error. Will fail if any of the below items fail.
	1	0 - 1	PBIT. System Boot Ok.
	2	0 - 1	PDI Ok. Will fail if there is an error in configuration files.
	3	0 - 1	Memory Allocation OK
	4	0 - 1	CAN A Ok
	5	0 - 1	CAN B Ok
	6	0 - 1	CIO Low Task Ok
	7	0 - 1	CIO High Task Ok
5	0	0 - 1	Power OK. All power indicators are OK.
	1	0 - 1	A Bus Voltage Ok
	2	0 - 1	B Bus Voltage Ok
	3	0 - 1	Arbiter Voltage Ok
	4	0 - 1	AP1 Voltage Ok
	5	0 - 1	AP2 Voltage Ok
	6	0 - 1	AP3 Voltage Ok
	7	0 - 1	Arbiter Mode. 1 = Normal, 0 = Maintenance

4.1.2 Score Message

Score message contains the final score of an specific autopilot. This message is a **producer** in the **4x PDI Builder** configuration, so it is created from the CAN messages of all three or four autopilots.

Byte	Position	Value	Description
0	0 - 7	0x00	Header
1	0 - 7	0 - 3	Autopilot ID (0 = AP1, 1 = AP2, 2 = AP3, 3 = External AP)
2 - 5	0 - 31	0 - 0xFFFFFFFF	Autopilot Arbitration Score

4.1.3 Ready Message

Ready message is sent from **Autopilot 1x** to **Arbiter**. It tells whether an **Autopilot 1x** is ready to fly or not. It is as **consumer** in the **4x PDI Builder** configuration, so it is stored in memory.

Byte	Position	Value	Description
0	0 - 7	0x00	Header
1	0 - 7	0xFF	Ready Message Header
2	0 (1 bit)	0 - 1	Ready/Not Ready

4.1.4 Arbitration Message

Arbitration message tells the arbitration value of a specific variable. It is as **consumer** in the **4x PDI Builder** configuration, since it is stored in memory.

Byte	Position	Value	Description
0	0 - 7	0x00	Header
1	0 - 7	0 - 31	Arbitration Variable Number
2 - 5	0 - 31	0 - 0xFFFFFFFF	Arbitration Variable Value

CHAPTER

FIVE

STATUS MANAGEMENT

5.1 Arbiter boot

When the **Arbiter** is booted, it enters an **Iddle** state, trying to verify the status of the system.

While **Iddle**, the **Status Message** is not sent.

If all checks success, the **Arbiter** will enter **Normal mode**.

If, after 30 seconds, system errors are still present, **Arbiter** will enter **Maintenance mode**.

5.2 Status message

Once in **Normal mode**, the **Arbiter** will start sending the **Status** and **Scores** telemetry messages.

The frequency of these messages is configurable. They can also be disabled.

The status message contains information such as:

- Arbitration ON/OFF
- Selected autopilot
- System BITs
- etc.

While in **Maintenance Mode**, the **Arbiter** will send the **Status** message, even if it is disabled, but will not send the **Scores** messages.

5.3 Ready status

After the **Arbiter** enters **Normal mode**, it will wait until a Ready Message from each autopilot is received. Only then the Arbitration will start.

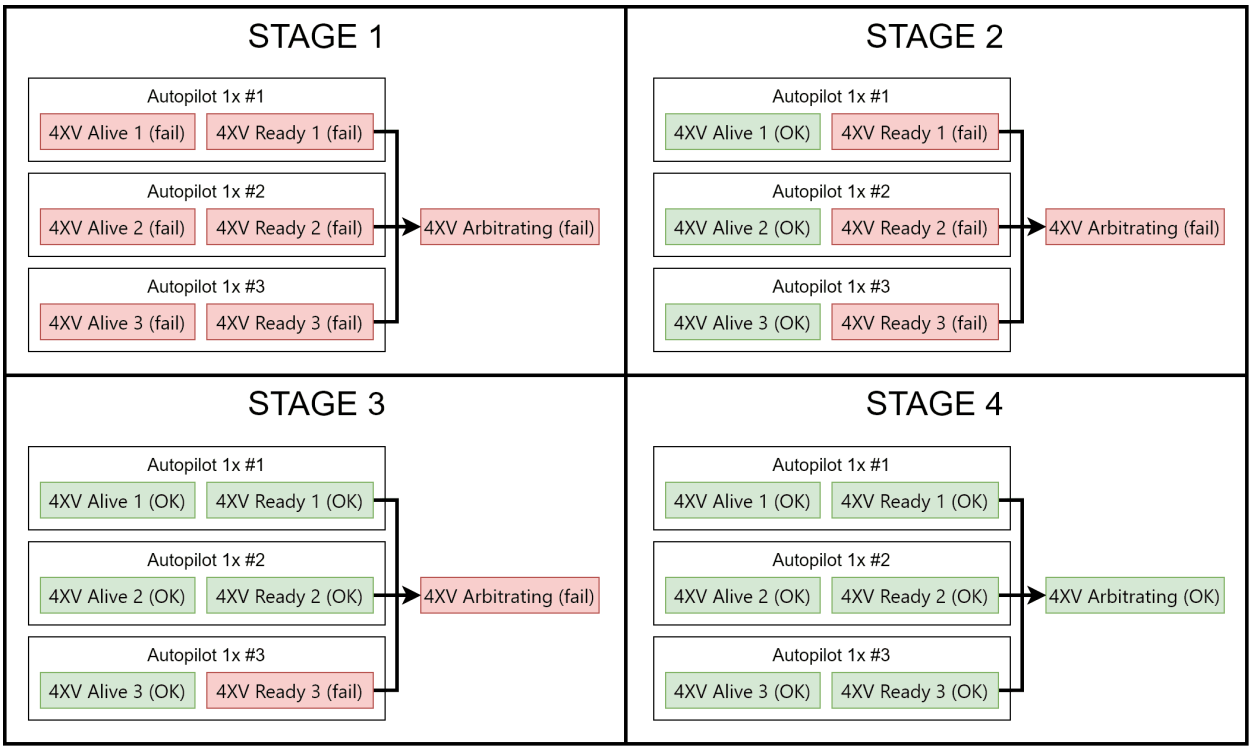


Fig. 1: **Example of normal arbitration start**

If the Arbiter is in maintenance mode, the arbitration will not start. Even if the **Ready messages** are received.

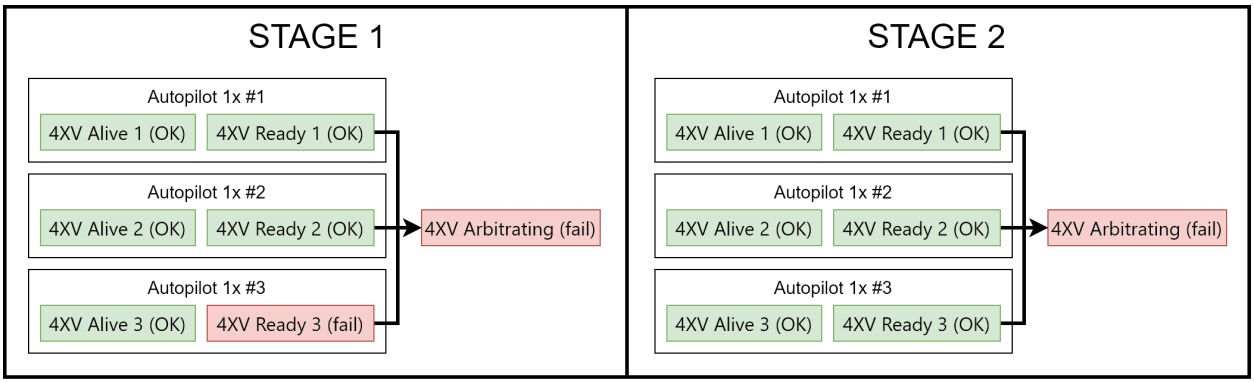


Fig. 2: **Example of failed arbitration start**

5.4 Alive status

Once arbitration starts, all autopilots are declared as alive by default, but it is possible that they are declared as **Dead** if a critical error is found.

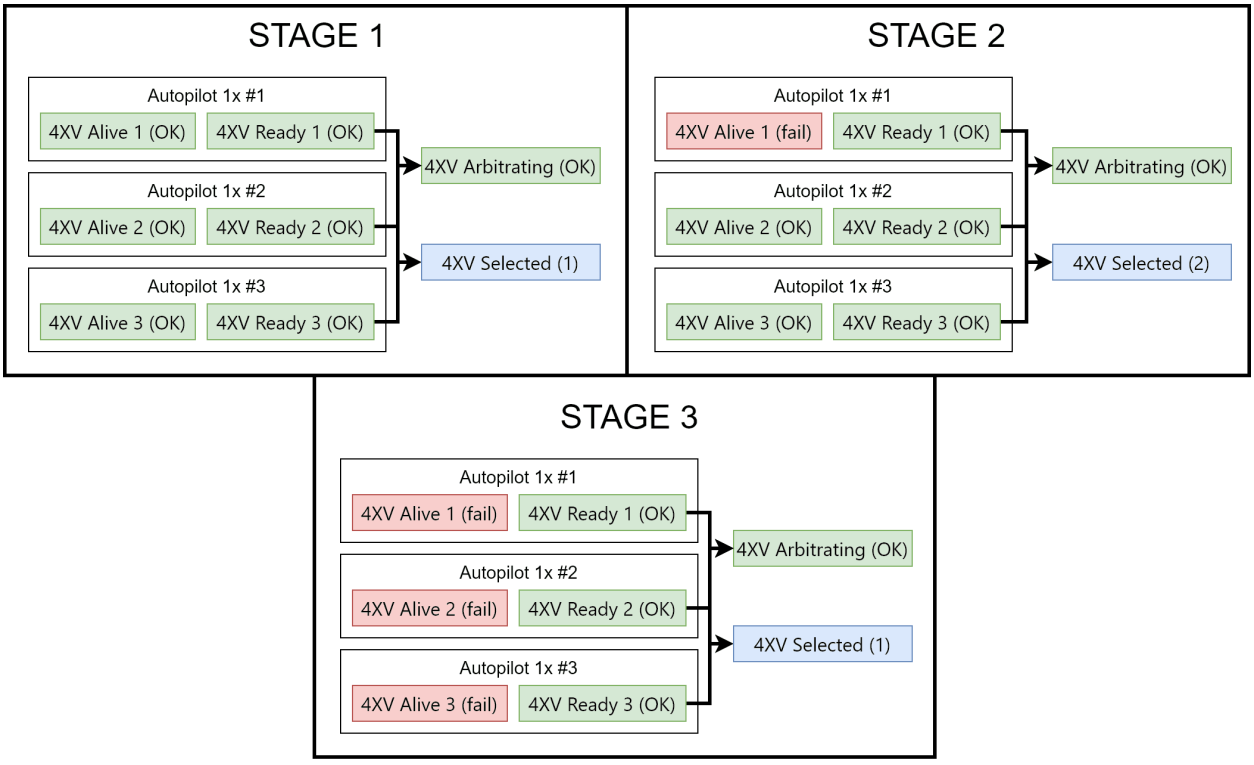


Fig. 3: **Example of APs being declared Dead**

The **Arbiter** will declare an autopilot dead if one of the following incidences is found:

- One of the arbitration messages (including the Ready message) is not received for 0.1 seconds.
- A **Not Ready** message is received.
- A **System Not OK** error is raised on any of the autopilots, activating the **FTS signal**.
- The **watchdog** signal for any autopilot is not ok.

Warning: Make sure to configure the sending of arbitration messages as **High priority**. Otherwise, the sending of messages could be shortly interrupted by a higher priority task and the autopilot will be declared **Dead**.

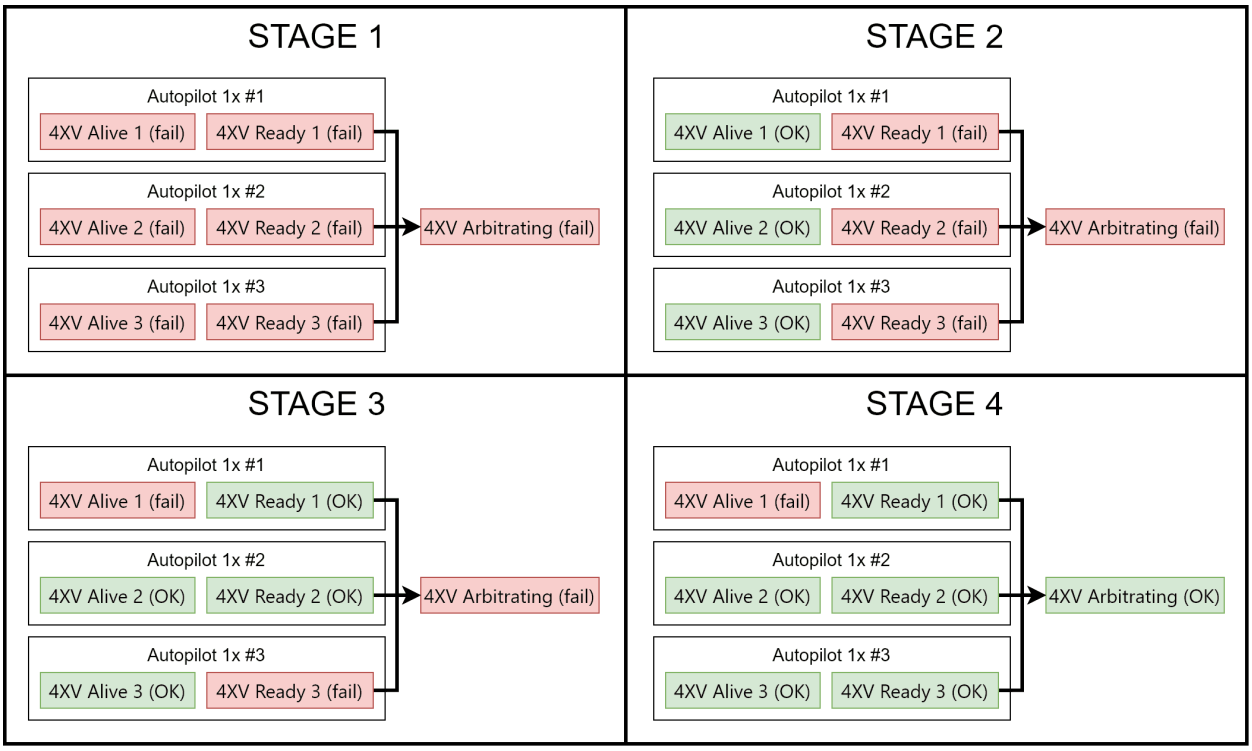


Fig. 4: **APs being declared Dead when arbitration starts due to missed messages**

A **Dead** autopilot can never be selected again as long as arbitration persists.

The **Dead** status is not reversible, it is necessary to reboot the whole system in order to recover a **Dead** autopilot.

If the number of **Alive** autopilots is 2 or less, relative arbitration variables are disabled (since at least 3 autopilots are needed).

If **only one** autopilot is **Alive**, it will be selected no matter the score.

If all autopilots are **Dead**, the **Preferred** autopilot will be selected.

5.5 Maintenance Mode

Maintenance mode is used for changing the **Arbiter** configuration.

Arbitration is disabled while in maintenance mode.

Arbiter will also enter maintenance Mode after a failed boot.

The reason of the failed boot can be checked in the **Status message**.