|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Document type: | **Module Specification** | | | | | |
| Document status: | **Approved** | | | | | |
| Title: | **MACS Evo** | | | | | |
| Document no.: | **SP000012530** | Revision: | **A** | Previous document no.: | |  |
| Authors: | **A.Battaglia** | | Date of release: | | **2020-08-04** | |
| Contributors: | **E.Marson, L.Fabrici, R.Stec, M.Johar** | | | | | |
| Revised by: |  | | | | | |
| This document contains confidential information and is the property of Electrolux Company without whose permission it may not be copied, shown or handed to a third party or otherwise used, and it is to be returned promptly upon request to the Electrolux Company which is responsible for the document. | | | | | | |

**Revision History:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Authors** | **Date** | **Description** | **Version** |
| A.Battaglia | 2020-08-04 | First Release | A.001 |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1 Introduction 4](#_Toc43720408)

[1.1 Purpose 4](#_Toc43720409)

[1.2 Scope 4](#_Toc43720410)

[1.2.1 Product, Brand and Model Definition 4](#_Toc43720411)

[1.3 Documents Referenced 4](#_Toc43720412)

[1.3.1 Codes (Norms) Agency Standards 4](#_Toc43720413)

[1.3.2 Electrolux Standards 4](#_Toc43720414)

[1.3.3 Third Party Standards 4](#_Toc43720415)

[1.3.4 Other Documents 4](#_Toc43720416)

[2 Module Overview 5](#_Toc43720417)

[2.1 General Description 5](#_Toc43720418)

[2.2 Block Diagram and Architecture 5](#_Toc43720419)

[2.2.1 Basic MACS Version 5](#_Toc43720420)

[2.2.2 Enhanced MACS Version 5](#_Toc43720421)

[3 Module Requirements 6](#_Toc43720422)

[3.1 Bandwidth 6](#_Toc43720423)

[3.2 Reaction Time 6](#_Toc43720424)

[3.3 Number of Nodes 6](#_Toc43720425)

[3.4 Cable Length between Nodes 7](#_Toc43720426)

[3.5 Efficiency 7](#_Toc43720427)

[3.6 Scalability 7](#_Toc43720428)

[3.7 Flexibility 7](#_Toc43720429)

[3.8 Diagnostics and Debug 8](#_Toc43720430)

[3.8.1 Hot-Plug 8](#_Toc43720431)

[3.8.2 Matching query/reply 8](#_Toc43720432)

[3.9 Reprogramming 8](#_Toc43720433)

[3.10 Safety 8](#_Toc43720434)

[3.11 Robustness 9](#_Toc43720435)

[3.12 Power Management 10](#_Toc43720436)

[3.13 Synchronization 10](#_Toc43720437)

[3.14 Cost 10](#_Toc43720438)

[4 MACS Use Cases 10](#_Toc43720439)

[5 MACS Messages 12](#_Toc43720440)

[5.1 Food Preparation 12](#_Toc43720441)

[5.1.1 Requirement of MACS Messages 12](#_Toc43720442)

[5.1.2 Purpose of Message 13](#_Toc43720443)

[5.1.3 Types of Messages 13](#_Toc43720444)

[5.1.4 MACS Message priority 15](#_Toc43720445)

[5.2 Dish Washer 15](#_Toc43720446)

[5.2.1 Types of Messages 15](#_Toc43720447)

[5.3 Fabric Care 16](#_Toc43720448)

[5.3.1 Types of Messages (Reference Documents) 16](#_Toc43720449)

[5.3.2 Tumble Dryer (Training Information) 17](#_Toc43720450)

[5.4 Refrigerator 18](#_Toc43720451)

[6 Notes 19](#_Toc43720452)

[6.1 Terms, Definitions, Abbreviations and Acronyms 19](#_Toc43720453)

[6.1.1 Abbreviations for Electrical Units 19](#_Toc43720454)

[6.1.2 Other Abbreviations 19](#_Toc43720455)

# Introduction

## Purpose

The goal of this Module Specification is to define the characteristics, improvements and modifications required for the MACS proprietary protocol in order to cope with the increasing complexity of Electrolux appliances. Although current MACS represents a brilliant solution to connect the electronic boards inside an appliance, it reached its technological limits. All these reasons, together with the huge knowledge accumulated in several projects and the need to support a unified electronic architecture named Horizon, gives the chance to bring the protocol to next level – MACS Evo - in order to fulfill more demanding requirements.

## Scope

### Product, Brand and Model Definition

MACS Evo shall be applicable to any Electrolux appliance without any distinction of product, Brand or Model.

## Documents Referenced

### Codes (Norms) Agency Standards

|  |  |  |
| --- | --- | --- |
| **Standard** | **Revision** | **Description** |
| To be defined | To be defined | To be defined |
| To be defined | To be defined | To be defined |
| To be defined | To be defined | To be defined |

### Electrolux Standards

|  |  |  |
| --- | --- | --- |
| **Standard** | **Revision** | **Description** |
|  |  |  |

### Third Party Standards

|  |  |  |
| --- | --- | --- |
| **Standard** | **Revision and Date** | **Description** |
|  |  |  |

### Other Documents

* [Ref 1] SP000012529-MS-GLOBAL-Horizon
* ETONASW-0017-UI\_HMI\_CDD v15
* SP000001962\_C Communications Interface Specification
* SP000003393 EU Cooking MACS Overview
* SP000005713 EU Cooking MACS hob communication specification
* SP000005715 EU Cooking MACS oven communication specification
* FCV-MB\_MACS\_Msg\_Interface\_EWX14
* MACS application protocol
* MB-FCV\_VSC MACS dryer
* UI-MB\_MACS\_Msg\_Interface
* WD-MB\_MACS\_Msg\_Interface
* EDR16 MB-MCB MACS
* UI-MB\_MACS\_Msg\_Interface\_EWX1493A
* WS-MB\_MACS\_Msg\_Interface
* WSD-MB\_MACS\_Msg\_Interface
* UI-MB\_MACS\_Msg\_Interface\_USA
* DIVSNet documentations

# Module Overview

## General Description

Pending…

## Block Diagram and Architecture

### Basic MACS Version

Current MACS basic version utilizes a 3-wire connection. Specifically, one wire is used as communication channel (transmission and reception, derived from uC UART TX and RX) and two wires for power supply.

As per current specification bit rate is allowed in the interval 300bit/s and 9600 bit/s, for a maximum of 6 nodes connected to the bus, and maximum length of the cable between two nodes less than 3 meters.

The MACS circuit has been developed to guarantee electrical insulation among MACS nodes, and it is typically used today in Motor Control boards. Iit has Opto-insulators and 3.3V reference.

The circuit utilizes opto-insulators, a 3.3V reference and has a limited baud rate of 9600, limitation given by the particular components used to provide insulation

### Enhanced MACS Version

A MACS enhanced version of the physical layer has been also defined although not very diffused and implemented in current projects. It utilizes a 4-wire connection. Specifically, two wires are used as differential communication channel (transmission and reception) and two wires for power supply.

As per current specification bit rate is allowed in the interval 300bit/s and 9600 bit/s, for a maximum of 16 nodes connected to the bus, and maximum length of the cable between two nodes less than 10 meters.

Although the enhanced version shows better performaces, the maximum speed is still limited.

The MACS circuit has been developed to guarantee electrical insulation among MACS nodes, and it is typically used today in Motor Control boards. Iit has Opto-insulators and 3.3V reference.

The circuit utilizes opto-insulators, a 3.3V reference and has a limited baud rate of 9600, limitation given by the particular components used to provide insulation

.

To improve speed, an enhanched version with insultaion has been developed and validated, in order to reach 115200 baud.

# Module Requirements

## Bandwidth

The increasing complexity of our products, due to the introduction of new features and technologies, brings often inevitably to an increase of the amount of data to be transmitted between MACS nodes. An example can be the configuration data transmitted from a User Interface to a Power Control or vice versa after plug-in of the appliance.

Considering also that MACS could be used to reprogram serially the uC(s) inside a MACS node, which nowadays can reach a memory size of several MB, the *bandwidth* must be increased sensibly.

Taking as reference the GPU electronic board used in PUX User Interface, reprogramming the uC thorugh SWD takes about 45 seconds. In order to achieve the same performances through MACS Evo, the bus should reach 921600 baud rate.

Start <SP000012530-301>

**MACS Evo shall support a maximum bandwidth of 1 Mbits/s.**

End <SP000012530-301>

Obviously not all the MACS nodes can achieve this speed, for instance due to cost considerations, usage of microcontrollers with limited performances, etc. In this case it could be possible to group the high-speed MACS nodes and low-speed MACS nodes in two separate MACS busses, using therefore specific system topologies and architectures, which are better explained in the document [Ref 1].

## Reaction Time

Another important parameter is the *reaction time*. A significant example is given by Food Preservation with the water dispensing system integrated into the refrigerator. When the User Interface senses the paddle is released, it communicates immediately the Power Control to close the water valve. If a delay above 80 ms occurs, it may happen the water still flows outside the container (glass, bottle, etc.) the user wants to fill.

A severe reaction time is required by Food Preparation, too. When the users press a key of an Oven HMI, in other words, a key located in a MACS node different from the User Interface, the response time shall be less than 50 ms.

Start <SP000012530-302>

**MACS Evo shall support a reaction time of less than 50 ms.**

End <SP000012530-302>

The possibility to assign dynamically priorities to messages should be properly evaluated to improve the reaction time as well as any other technic and communication strategy.

## Number of Nodes

Similarly to what happened and is happening to all markets, the electronics is becoming more and more the core of the product. Consequently, the number of boards and microcontrollers inside an appliance increased and will keep increasing. We have already today up to 8 – 10 MACS nodes in the system.

Start <SP000012530-303>

**MACS Evo shall support a maximum of 16 nodes connected to the same bus.**

End <SP000012530-303>

## Cable Length between Nodes

Togheter with bandwith and number of MACS nodes, the maximum length of the cable between nodes is an important parameter to be specified in order to design properly the communication system. The worst case is given by some multi-door fridges, where the MACS cable length reaches 5100 mm.

Start <SP000012530-304>

**MACS Evo shall support a maximum length of the cable between two nodes of less than 6 meters.**

End <SP000012530-304>

## Efficiency

Current MACS has been designed and developed considering the microcontroller capabilities of decades ago. Microcontrollers technology and speed today is such that it is possible to work on timing components to reduce the time the bus is *apparently* free.

Start <SP000012530-305>

**MACS Evo shall improve the efficiency in channel utilization compared to current MACS at the same baud rate.**

End <SP000012530-305>

Although it is possible already today to send messages to all nodes of the same time (User interfaces, Power Boards, etc), in order to improve efficiency and reduce the number of massages in the bus, it is worth to consider the possibility to group and ungroup dynamically MACS nodes. In other words, it could make sense to consider the possible to build and destroy groups at run-time and address a group of nodes with a single message.

## Scalability

MACS Evo can be deployed into systems with slightly different requirements and constraints, due to uCs performances limitations, cost reasons or simply because some advanced MACS Evo features are not needed in the specific system. Therefore, the protocol must be scalable, having a core functionality and extensions added according to the needs. It allows to define the right trade-off between complexity and simplicity, code size and speed.

Start <SP000012530-306>

**MACS Evo shall be scalable through the separation of core functionalities and extended functionalities.**

End <SP000012530-306>

## Flexibility

At application level it happens often that existing MACS messages are required to be extended with new fields. Although this issue can be often avoided with the usage of more atomic messages, there are cases where extending the message in the most clean and logical solution. Therefore, dynamic length of the message can often overcome limitations given by fixed length of messages.

Start <SP000012530-307>

**MACS Evo shall support dynamic length of messages.**

End <SP000012530-307>

Another interesting case is given by MACS nodes which might not require the acknowledge of their messages, like for example temperature sensors. The acknowledge mechanism may be not needed in this case, therefore it should be possible to define messages which do not require it.

Start <SP000012530-308>

**MACS Evo shall support the capability to define if a message requires or not acknowledge.**

End <SP000012530-308>

## Diagnostics and Debug

Although the primary target of MACS protocol is the internal communication of the electronic modules of an appliance, it would be very beneficial if a certain portion of the bandwidth could be allocated for diagnostic and debug messages used for instance during development to investigate bugs or by service to discover malfunctions.

Start <SP000012530-309>

**MACS Evo shall use a percentage of the bandwidth for diagnostic and debug messages.**

End <SP000012530-309>

With the purpose of simplify diagnostic and debug also the following improvements are required.

### Hot-Plug

It happens often to have the need to connect a MACS logger tool to inspect the messages flowing from and to the nodes of the system, while a bug, defect or problem occurs. Turning off and on the system to connect a MACS logger and try to replicate again the issue is not optimal. Therefore, MACS Evo shall support the capability to connect a MACS logger without reboot.

Start <SP000012530-310>

**MACS Evo shall support the capability to connect/disconnect a MACS node in the system without reboot.**

End <SP000012530-310>

### Matching query/reply

Often MACS tools used to inject messages on the bus or emulate a MACS node are Windows-based, and they cannot guarantee the requested reaction times. Therefore, often messages are nor acknowledged in time and resent multiple times, until the acknowledge and reply is provided. It is then not possible to understand what reply is triggered from what specific query.

Start <SP000012530-311>

**MACS Evo shall support the capability to relax the timeout to response to a given MACS node destination to avoid resends of messages**

End <SP000012530-311>

Start <SP000012530-312>

**MACS Evo shall support a mechanism of session counting in order to be able to match a reply message to the proper query message.**

End <SP000012530-312>

## Reprogramming

As already mentioned in the paragraph 3.6, although the primary target of MACS protocol is the internal communication of the electronic modules of an appliance, it could be very used to reprogram the uC(s) connected to the bus. This would open interesting scenarios for supplier, factory and service reprogramming.

Start <SP000012530-313>

**MACS Evo shall support the capability to program all the MACS nodes connected to the bus and all the MACS nodes potentially connected to MACS sub-networks of a MACS node.**

End <SP000012530-313>

## Safety

In current MACS implementation a 8-bit XOR of all bytes is used inside a MACS message for error detection. European and Nord American normative (IEC-60730 and UL-60730) and approval agencies (VDE and UL) require a more robust error detection mechanism. Therefore, a 16-bit XOR and a CRC16-CCITT are used in Food Preparation EU and Food Preparation NA respectively for safety related MACS messages.

For standardization reasons and considering the higher capabilities of today microcontroller, it makes sense to embed a CRC16-CCITT in all MACS messages.

Start <SP000012530-314>

**MACS Evo shall use CRC16-CCITT for error detection in all messages.**

End <SP000012530-314>

A specific version of MACS Evo is also needed where electrical insulation is required, for instance in MACS node implementing Motor Control functionality. As a reference and as mentioned in Par. 2.2.2, an insulated MACS version with a maximum speed of 115200 baud has been already developed and validated. Obviuosly, following scalability priciples defined in Par.3.2, a trade-off among all other requirements must be found, in order, for instance, to find the best compromise between costs and performances.

Start <SP000012530-315>

**MACS Evo shall implement one or more variants which provides electrical insulation.**

End <SP000012530-315>

## Robustness

MACS Evo must be designed and implemented keeping into account the noise and abnormal conditions created during EMC/EMI tests, Electrolux internal quality standards tests, or factory programming. Objectives criteria should be identified to prove the robustness of the technical solution identified for MACS Evo.

Start <SP000012530-316>

**MACS Evo shall provide the proper level of robustness and noise immunity.**

End <SP000012530-316>

Another improvement which may impact EMC/EMI results is related to a possible new mechanism of how the power board communicates to all MACS nodes the happening of a power failure. Currently when the power board detects it, the MACS data line is pulled low, this is recognized from the other MACS node as a sign of deactivate all possible peripherals and move to sleep mode in order to save as much energy as possible to prevent a reboot. If bandwidth and reaction time are improved significantly it might be possible to create a dedicated MACS message to inform all MACS nodes that a power failure is occurring.

Start <SP000012530-317>

**MACS Evo shall implement a way to communicate a Power Fail event over the MACS bus with a reaction time of XX ms.**

End <SP000012530-317>

This new approach would leave the mechanism of pulling the MACS data line low available for another purposes. It may be used, for instance, to force a reset of all MACS nodes to start the reprogramming of one or more of them.

Assuming the appliance provides a MACS connection to the external world, it would be possible to pull the MACS data line for a certain time to force a reboot. The microcontrollers’ bootloader would then start and allow to an external tool, connected to the MACS bus, the reprogramming of the flash memory.

Anyhow, more in general, it is needed a way to force the reboot of MACS node. Pulling the MACS data line low could not be the only and best solution. Therefore it is more appropriate to state:

Start <SP000012530-318>

**MACS Evo shall implement a way to force an HW reboot of the MACS node and give the possibility to reprogram of one or more MACS nodes.**

End <SP000012530-318>

Finally, it could be needed for a MACS node to request to all other nodes for a short communication break, comparable with a message length, to perform operations which require the code to be halted. The break could be from all the MACS nodes to the MACS node requesting the break, while all the others can still communicate among themselves. This may be useful when a node performs some write operation in ROM/Flash, where it is common to stop the execution of the application code, and to prevent messages are lost.

Start <SP000012530-319>

**MACS Evo shall support the possibility for a MACS node to inform to the other MACS nodes to stop/resume temporarily the communication with itself.**

End <SP000012530-319>

## Power Management

Power consumption of an appliance, above all in standby, is a very relevant topic, regulated in many countries by specific normative and regulations. It is important to allow MACS nodes to go in stand-by when not needed and be waked up by specific MACS messages on demand.

Start <SP000012530-320>

**MACS Evo shall allow one or more nodes to move into sleep mode and not participate to the network activity and communication. It shall also support a mechanism to wake up the specific node on demand.**

End <SP000012530-320>

It may be even required to unpower one or more MACS nodes to fullfill demanding scenario of standby power consumption of the whole appliance. In this case the possibility depends on the way power supply is distributed for MACS nodes or groups of MACS nodes. In any case, if a MACS node is powered off, the rest of the bus must keep on working.

Start <SP000012530-321>

**In the situation one or more MACS nodes are powered off, MACS Evo shall allow the remaining MACS nodes to communicate properly among them. When then powered back on (some of the them or all of them), those MACS nodes shall restore the communication in the bus.**

End <SP000012530-321>

## Synchronization

Often synchronization among MACS nodes and time generation are made through the 50/60 Hz main supply through the mechanism called Zero Crossing. In this case the time information must be transmitted to other MACS nodes with the proper level of time precision.

Start <SP000012530-322>

**MACS Evo shall support synchronization information with the precision of less than 1 ms.**

End <SP000012530-322>

## Cost

Although the higher performances required to MACS Evo compared to current MACS might inevitably increase the direct material cost, the opportunities given by standardization of components and harnesses, the combination of programming, debugging and communication into a unique protocol, the higher programming speed at supplier, factory and service, might compensate and even overcome that cost.

Start <SP000012530-323>

**MACS Evo shall be cost effective compared to current solution, taking into account the cost of ownership.**

End <SP000012530-323>

# MACS Use Cases

In order to contestualize the usage of MACS Evo communication protocol in real applications, some possible use cases are presented for each appliance:

UI = User Interface HMI = Human Machine Interface

PC = Powe Control EC = Expansion Control MC = Motor Control

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Oven** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** | Low End, Single Cavity | UI + PC | 2 | ≤ 1 m | No |  |
| **Normal Case** | Mid End, Single Cavity, external HMI | UI + HMI + PC | 3 | ≤ 1 m | No |  |
| **Worst Case** | High End, Double Cavity,  external HMI,  Automatic Door Opener | UI(FE) + UI(BE) + HMI  + 2 x PC + EC | 6 | ≤ 1 m | No |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Hob** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** | Low End, 4 Zones | UI + 2 x PC | 3 | ≤ 1 m | Functional |  |
| **Normal Case** | Mid End, 4 Zones | UI + 2 x PC | 3 | ≤ 1 m | Functional |  |
| **Worst Case** | High End, 5 Zones, Illumination,  Wireless Probe | UI(FE) + UI(BE) +  + 3 x PC + 2 x EC | 7 | ≤ 1 m | Functional |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Hood** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** |  |  |  |  |  |  |
| **Normal Case** |  |  |  |  |  |  |
| **Worst Case** |  |  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Oven + Hob** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** |  |  |  |  |  |  |
| **Normal Case** |  |  |  |  |  |  |
| **Worst Case** |  |  |  |  |  |  |

**Note:** MACS Evo shall support a specific microcontroller in UI of HOB comprises of at least 2 UART ports for linking of two different MACS buses of two appliances (for e.g. Oven +Hob) to form an electronic system, one UART for HOB MACS data bus and other UART for Oven MACS data bus.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Washing Machine** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** | Low End,  Universal Motor | UI + PC | 2 | ≤ 1.5 m | No |  |
| **Normal Case** | Mid End, Inverter | UI + PC +  MC | 3 | ≤ 1.5 m | Si (MC) |  |
| **Worst Case** | Hign End, Inverter, Weight Sensor,  Water Softner,  Turbidity Sensor  Autodosing | UI + HMI + PC  + MC + 4 x EC | 8 | ≤ 1.5 m | Si (MC) |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dryer** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** | Low End,  Asyncronous Motor | UI + PC | 2 | ≤ 1.5 m | Si (EC?) |  |
| **Normal Case** | Mid End, Inverter | UI + PC+ MC | 3 | ≤ 1.5 m | Si (MC) |  |
| **Worst Case** | High End, Inverter,  Knob, Capacitive Sensor | UI + 2 x HMI + PC  + MC | 5 | ≤ 1.5 m | Si (MC) |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Washing + Dryer** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** | Low End,  Universal Motor | UI + PC + EC | 3 | ≤ 1.5 m | No |  |
| **Normal Case** | Mid End, Inverter | UI + PC+ MC | 3 | ≤ 1.5 m | Si (MC) |  |
| **Worst Case** | Hign End, Inverter, Heat Pump, Weight Sensor, Water Softner,  Turbidity Sensor  Autodosing | UI + HMI + PC + 2 x MC + 4 x EC | 9 | ≤ 1.5 m | Si (MC) |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dish Washer** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** | Low End,  Washing Pump | UI + PC + MC | 3 | ≤ 1.5 m | Yes |  |
| **Normal Case** | Low End,  Washing Pump,  Drain Pump | UI + PC + 2 x MC | 4 | ≤ 1.5 m | Yes |  |
| **Worst Case** | High End,  Washing Pump,  Drain Pump,  Autodosing | UI + PC + 2 x MC + EC | 5 | ≤ 1.5 m | Yes |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Fridge** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** |  |  |  |  |  |  |
| **Normal Case** |  |  |  |  |  |  |
| **Worst Case** |  |  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Freezer** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** |  |  |  |  |  |  |
| **Normal Case** |  |  |  |  |  |  |
| **Worst Case** |  |  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Fridge + Freezer** | **System** | **Nodes** | **Num of Nodes** | **Cable Length** | **Insulation** | Speed |
| **Simple Case** | Simple System | UI + PC | 2 | ≤ 0.5 m | Yes |  |
| **Normal Case** | Mid System | UI + PC | 2 | ≤ 2.5 m | Yes |  |
| **Worst Case** | High End, MultiDoor  Inverter | 3 x UI + PC + MC | 5 | ≤ 3 m | Yes |  |

Table 4: MACS Use Cases

# MACS Messages

## Food Preparation

### Requirement of MACS Messages

A message is usually sent out

* if the status of a unit has changed
* if it is demanded by another unit
* if it is required to send it periodically
* if another unit has to be forced to do some actions

### Purpose of Message

The CTRL(control) byte contains information about the purpose of the message. In current hob appliances it is sufficient to decode only the operation mode ,These bits have following meaning:

|  |  |  |
| --- | --- | --- |
| Operation mode | Definition | Meaning |
| 00 | SET / DO | This operation mode is an instruction for the addressed unit. |
| 01 | RESULT | If a message is send with a confirmation option ( then the addressed unit must answer with a RESULT message).  This feature is mainly used for debug or test operation. Therefore it can be decided by each project if it has to be implemented! |
| 10 | READ\_REQUEST | With this operation mode a certain message can be requested from another unit. |
| 11 | INFO / READ\_REPLY | The content of this message can be used as information about the current status of the unit. Answers according to a READ\_REQUEST are send with READ\_REPLY too. |

Table 5-1: Purpose of Messages

### Types of Messages

|  |  |
| --- | --- |
| **Main Category** | **Subgroup Messages** |
| Configuration Messages | Class A configuration, Class B configuration, Configuration read request, Configuration response, Source version, Communication key. |
| Regulation Parameters Messages | Temperature algorithms, Heating element manager parameters, Read request, Response messages. |
| Functional Messages | Auxiliary load, Read request, Response message, Factory test load, Execution commands. |
| Status Messages | Binary sensor, Numerical Sensor, Temperature sensor, Read request, Reply message, UI control status, Triac control status. |
| Periodic Messages | Minute tick. |
| Error/Alarm Messages | Alarm info message, Read request, Response message. |
| Communication Alive Messages | Heartbeat (control status), State initialization message. |
| Info Request/Response Messages | Read Request, Reply message |

Table 5-2: Types of Messages

* **General Messages:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | | **Purpose** | **Boards Involved** |
| Factory Test | LED/LCD | set | TEST->ALL(SET)  (test machine (e.g. a PC) to the board under test) |
| Key | set | TEST->ALL(SET) |
| Load | set | TEST->ALL(SET) |
| Sensor data | request/reply | TEST-> ALL(READ\_REQUEST)  ALL -> TEST(INFO) |
| Alarm info | request/reply | TEST-> ALL(READ\_REQUEST)  ALL -> TEST(INFO) |
| Operating hours | Operating hours of dedicated function, or total time device connected to mains unit | Request/Reply | ALL->board with operation hours (READ\_REQUEST)  Board with operation hours-> ALL (READ\_REPLY) |
| DAAS Messages | S/W ID 1 | Request/Reply | ALL |
| S/W ID 2 | Request/Reply | ALL |
| Memory read/write | Set/Request/Reply | ALL |
| RAM/Boot loader | Memory write | set | ALL |
| ANC\_Read | F/W, config data | Request/Reply | ALL |
| Test Messages | Virtual Key | Set/Request/Reply | ALL -> TEST |
| Debug Messages | Key Signal | Set/Request/ Read\_Reply | ALL  PC -> HUIx HUIx -> PC |
| Diagnostic Messages | Memory read/write | Set/Request/ Read\_Reply | ALL |
| Memory\_Explicit | Memory read/write | Set/Request/ Read\_Reply | ALL |

Table 5-3: General Messages

* **Product Specific Messages:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | | **Purpose** | **Boards Involved** |
| Configuration Messages |  | Request/ Read\_Reply | OC -> OUI (request) OUI -> OC (READ\_REPLY) |
| Regulation Parameters | Temperature control parameters | Set/Request/ Read\_Reply | OUI -> OC (SET) OC -> OUI (READ\_REQUEST) OUI -> OC (READ\_REPLY) |
| Functional Messages | new cooking phase  & Loads activation | Set/Request/ Read\_Reply | OUI -> OC (SET) OC -> OUI (READ\_REQUEST) OUI -> OC (READ\_REPLY) |
| Execution Commands: Sensor Enable/Disable, Demo ON/OFF, System ON/OFF, child Lock ON/OFF, Factory Test ON, Smell Filter ON/OFF | SET | OUI -> OC (SET) |
| Change Parameters Messages | Wet Steam Temperature Threshold | Set/Request/ Read\_Reply | OUI-> OC(SET,READ\_REQUEST)   OC-> OUI(READ\_REPLY/INFO) |
| Cooling Fan Low Speed |
| Microwave Cooling Fan Low Speed |
| Steamer Tank Parameters |
| Humidity Sensor Parameters |
| Hot Steamer Cavity Temperature Threshold |
| Water Level Parameters (reed switch sensor) |
| Periodic Message | Minute Tick | Read\_Reply | OC -> ALL (INFO) |
| Alarm Messages | MACS Alarm | Request/ Read\_Reply | OC -> OUI (INFO, READ REPLY) OUI -> OC (READ REQUEST) |
| Status Messages | Demo, OFF, Standby, ON, Test, Remote, Switch, lamp, Sensor data, Alarm | Request/ Read\_Reply | OC -> All (INFO) All -> OC (READ REQUEST) OC -> All (READ REPLY) |

|  |  |  |  |
| --- | --- | --- | --- |
| Power Management Messages |  | Set/Request/ Read\_Reply |  |
| Self Test | Self test power board | REQUEST, READ\_REPLY/INFO, SET | HC-> ALL (INFO/READ\_REPLY) HUI/PC -> HC (SET, READ\_REQUEST) |

Table 5-4: Product Specific Messages

### MACS Message priority

* Configuration messages shall be high priority
* Heartbeat messages between the UIs and the Power Boards shall be high priority
* Heartbeat messages / status change messages between UIs shall be low priority
* All cooking related messages shall be medium priority

## Dish Washer

### Types of Messages

* The message types going from UI to MB are basically commands/actions to perform inputted by knobs and push buttons.
* While the message types going from MB to UI are mainly the machine status to be displayed by means of LEDs or LCD.

|  |  |
| --- | --- |
| **Main Category** | **Subgroup Messages** |
| **Status Message** | Power fail, Program resume, Clean, Actuator test active, Sanitation successful, Door open, Rinse empty, Salt empty, Machine Care Alert, AdoActive, TestModeActive |
| **Communication Alive Message** | Heartbeat |
| **Operational/ Functional Message** | Start program, cancel program, Auto Power Off, UI Command ON/OFF, Dispenser Activate, Dispenser Rinse Aid Activate, Dispenser Deactivate, Pause Program, Resume Program, Turn off BOF / Button ID, UI reboot on Macs |
| **Info Request/Response Message** | Service Info Request, Service Info, Extended Service Info, Read Memory Request, Dispenser Done, UI info Request, UI info Response |
| **Error/Alarm Message** | Error Occurred, Error Acknowledgment, Hidden Alarm |
| **Test Message** | Actuator Test, DOF LED Test |
| **Wake up Message** | Base Board Awake / Cancel Auto Shut off |
| **Debug/Diagnostic Message** | UI Debug Message, Diagnostics Info Request, Diagnostics Info Response |
| **Configuration Message** | UI Version Request, UI Version Response, UI Prepare Download Data, UI Prepare Download OK |

Table 5-5: Types of Messages

## Fabric Care

### Types of Messages (Reference Documents)

|  |  |  |
| --- | --- | --- |
| **Name** | **Meaning** | **Direction** |
| **WAKEUP** | UI notifies each start-up it is ready to communicate | UI  MB |
| MB replies/notifies UI each start-up it is ready to communicate | UI  MB |
| **UI\_CFG** | UI configuration data request/validity checking request | UI  MB |
| UI configuration data transferring or validity checking answer | UI  MB |
| **USER\_SEL** | User selections parameters when setting up program/options | UI  MB |
| User selections recovered from Power Failure backup | UI  MB |
| **USER\_CMD** | User commands to start/stop machine | UI  MB |
| **MACHINE\_STATUS** | Machine/cycle status info to display or to enable UI selections | UI  MB |
| **UI\_TEST** | UI Input status (knobs, keys) remotely monitored via MB | UI  MB |
| UI Output status (LEDs, LCD, buzzer) remotely driven via MB | UI  MB |
| **PROG\_PAR** | Program parameters, selectable options enabled by cycle | UI  MB |
| **DISPLAY\_INFO** | Machine/cycle info (TTE, …) to be displayed on LCD | UI  MB |
| **ALARMS** | UI local alarms status to be stored on MB statistical memory | UI  MB |
| Alarms status to be displayed | UI  MB |
| **LANG\_CFG** | Language configuration data transferring | UI  MB |
| **UI\_IO\_STATUS** | Remote management of UI Output mounted on MB | UI  MB |
| Remote management of UI Input mounted on MB | UI  MB |
| **WATCH\_DOG** | Watch-dog for communication presence checking | UI  MB |
| **SW\_ID1** | SW identification string reading (first 4 chars) | UI  MB |
| **SW\_ID2** | SW identification string (last 4 chars) | UI  MB |
| **MEMORY** | Read/Write the contents of a memory area | UI  MB |
| **FORCE\_BOOT** | Board restart to activate boot | UI  MB |
| **MACS\_ATEST** | MACS connector automatic test loop (UI board standalone) | UI  UI |

Table 5-6: Types of Messages

### Types of Messages (Reference DocumentsEWX1493A)

|  |  |  |
| --- | --- | --- |
| **Name** | **Meaning** | **Direction** |
| **WAKEUP** | UI notifies each start-up it is ready to communicate | UI  MB |
| MB replies/notifies UI each start-up it is ready to communicate | UI  MB |
| **UI\_CFG** | UI configuration data request/validity checking request | UI  MB |
| UI configuration data transferring or validity checking answer | UI  MB |
| **USER\_SEL** | User selections parameters when setting up program/options | UI  MB |
| User selections recovered from Power Failure backup | UI  MB |
| **USER\_CMD** | User commands to start/stop machine | UI  MB |
| **MACHINE\_STATUS** | Machine/cycle status info to display or to enable UI selections | UI  MB |
| **UI\_TEST** | UI Input status (knobs, keys) remotely monitored via MB | UI  MB |
| UI Output status (LEDs, LCD, buzzer) remotely driven via MB | UI  MB |
| **PROG\_PAR** | Program parameters, selectable options enabled by cycle | UI  MB |
| **DISPLAY\_INFO** | Machine/cycle info (TTE, …) to be displayed on digits | UI  MB |
| **ALARMS** | UI local alarms status to be stored on MB statistical memory | UI  MB |
| Alarms status to be displayed | UI  MB |
| **LANG\_CFG** | Language configuration data transferring | UI  MB |
| **UI\_IO\_STATUS** | Remote management of UI Output mounted on MB | UI  MB |
| Remote management of UI Input mounted on MB | UI  MB |
| **REMOTE\_STATUS** | Network remote status | UI  MB |
| **ADOSE\_PAR** | Autodose parameters changed in setting mode | UI  MB |
| Autodose parameters default/last values stored by MB | UI  MB |
| **ADOSE\_STATUS** | Autodose status info (warnings, detergent dose, ...) | UI  MB |
| **WATCH\_DOG** | Watch-dog for communication presence checking | UI  MB |
| **MACH\_ID** | PNC ELC SN for Serialization | UI  MB |
|  | UI  MB |
| **BOARD\_ID** | Boards traceability data | UI  MB |
| **GCF\_HASH** | GCF hash for traceability data | UI  MB |
| **CTF\_HASH** | CTF hash for traceability data | UI  MB |
| **MAC\_ADDR** | NIU MAC address to PB | UI  MB |
| **NIU\_CMD** | NIU commands from PB | UI  MB |
| **SAT\_LIST** | Sat list for traceability data | UI  MB |
| **MAINTENANCE** |  | UI  MB |
|  | UI  MB |
| **SW\_ID1** | SW identification string reading (first 4 chars) | UI  MB |
| **SW\_ID2** | SW identification string (last 4 chars) | UI  MB |
| **MEMORY** | Read/Write the contents of a memory area | UI  MB |
| **FORCE\_BOOT** | Board restart to activate boot | UI  MB |
| **MACS\_ATEST** | MACS connector automatic test loop (UI board standalone) | UI  UI |

1. power failure condition notified to all connected units via MACS data line low state for about 20 milliseconds.
2. every system start-up, all units connected to MB notify with a broadcast message that they are ready to communicate and wait for MB directives before starting to send functional messages.

|  |  |  |
| --- | --- | --- |
| **Main Category** | **Subgroup Messages** | **Direction** |
| Power Up synchronization Message | Wakeup Messages | UI<>MB |
| USER INTERFACE CONFIGURATION MESSAGE | UI>MB  MB>UI |
| USER SELECTIONS MESSAGE | after a power failure restart to recover the program parameters saved on MB, and UI request data  (Temp, Speed, Dry Time, Dry options, wash options, delay time) | UI>MB (READ\_REQUEST)  MB>UI (READ\_REPLY) |
| Button Press | UI>MB (INFO) |
| PROGRAM PARAMETERS MESSAGE | each time Program Parameters change during cycle execution  (Temp, Speed, Dry Time etc.) | MB>UI |
| USER COMMAND MESSAGE | Start/Pause/Resume/Reset/ special commands for mode or cycle | UI>MB |
| UI\_STATUS command(UI standy , UI demo, UI stand off) | UI>MB |
| MACHINE STATUS MESSAGE | Cycle State info (Init, idle, Pause, delay etc.)/  Current phase info (wash, spin, dry etc.)/  Door State info/  Mode Info.  Machine status info to be displayed on UI (at the time of M/C startup and during cycle execution) | MB>UI |
| DISPLAY INFO MESSAGE | Time to end, delay start time, laundry weight, M/C working hours no. etc. | MB>UI |
| AUTODOSE PARAMETERS MESSAGE | Tank A/B detergent default/last configured value  Tank A/B detergent dose value set by user, etc.  Autodose parameters to configure detergent dose. | MB>UI |
| AUTODOSE STATUS MESSAGE | Status of tanks, drawer .  Tank A/B detergent quantity used in the cycle.  Estimated remaining cycles with Tank A/B detergent in reserve, etc.  Autodose status info to be displayed on UI | MB>UI |
| ALARMS MESSAGE |  | MB>UI (to display)  UI>MB (to save alarm data) |
| WATCH DOG MESSAGE | to check the presence of communication between MB and UI | MB>UI |
| USER INTERFACE TEST MESSAGE | UI to MB in remote control mode each time the knob(s) is moved or a button is pressed or released.  MB>UI to display diagnostic info such as current values of temperature, drum speed and water level.  (This message is used to perform a remote UI test managed by an external Tool) | UI>MB  MB>UI |

### Purpose of Message

|  |  |  |
| --- | --- | --- |
| **OPERATION**  **Value** | **Name** | **Description** |
| 0:0 | SET / DO | **Only For Originator messages:** The unit must execute some action according to the OPCODE and data fields. |
| 0:1 | RESULT | **Only For Confirmation messages:** The result of execution/answer to the Originator message with the above SET/DO operation. |
| 1:0 | READ\_REQUEST | **Only For Originator messages:** The unit must reply to the Originator with the values of some internal parameters according to the OPCODE and data fields. The CONFIRM\_TYPE should be different than NO\_CONFIRM value. |
| 1:1 | READ\_REPLY / INFO | **For Confirmation messages:** The result of processing the Originator message with the above READ\_REQUEST operation. The message contains the values of requested parameters.  **For Info messages:** The unit sends the Info message with this value when some event occurs (e.g. machine status changes). |

### Types of Messages between MCB and MB ( Env06 DR MB/MCB MACS APPLICATION PROTOCOL)

|  |  |  |
| --- | --- | --- |
| **Main Category** | **Subgroup Messages** | **Direction** |
| Synchronization Messages | Configuration Messages | MCB> MB (Info), MB> MCB (Set, Reply) |
| Wakeup Message | MB> MCB (Request), MCB> MB (Reply) |
| Alarm Messages (CGF Timeout) |  |
| Operative Mode | Status Messages (Current speed, temp., alarm) | MCB> MB (Info), MB>MCB (Set) |
| Alarm Messages | MB> MCB (Request to clear msg), MCB>MB(Reply) |
| Service Messages | Diagnostic/Debug/Testing  (SW String ID, Memory Reading/Memory Writing) | Diagnostic tool / MB >MCB (Request)  Diagnostic tool > MB (Request)  MCB>MB/Diagnostic tool (Reply)  MCB > MB (Reply)  Diagnostic tool / MB >MCB (Set)  Diagnostic tool > MB (set) |

### Types of Messages between Weight Sensor Board(WS) and PB ( Env06 DR MB/MCB MACS APPLICATION PROTOCOL)

|  |  |  |
| --- | --- | --- |
| **Main Category** | **Subgroup Messages** | **Direction** |
| Synchronization Messages | Wakeup Message | WS> PB (Broadcast) PB>WS(Broadcast or point to point), PB> WS (Reply) |
| Configuration Messages | WS> PB (set), PB> WS (Reply) |
| Reporting of measures | Low power mode message (Start/Stop reporting of measures)  Receiving stop reporting measure it will go to low power mode | PB>WS (Set) |
| Current Status | Temperature, Sensor related info etc. | WS>PB (Info) |
| Alarm clearing | To clear alarms | PB>WS(Set) |
| Diagnostic Messages | Board FW identification (SW1\_ID/ SW2\_ID) | PB / diagn. tool  WS (Read\_Request)  PB / diagn. tool  WS (READ\_REPLY) |
| Remote access to uC memory | PB / diagn. tool  WS ( READ\_REQUEST)  PB / diagn. tool  WS (READ\_REPLY) |
| Memory write | PB / diagn. tool  WS (Set)  PB / diagn. tool  WS (Result) |

### Tumble Dryer (Training Information)

|  |  |  |
| --- | --- | --- |
| **Name** | **Meaning** | **Direction** |
| **WAKEUP** | UI notifies each start-up it is ready to communicate | UI  MB |
| MB replies/notifies UI each start-up it is ready to communicate | UI  MB |
| **UI\_CFG** | UI configuration data request/validity checking request | UI  MB |
| UI configuration data transferring or validity checking answer | UI  MB |
| **USER\_SEL** | User selections parameters when setting up program/options | UI  MB |
| User selections recovered from Power Failure backup | UI  MB |
| **USER\_CMD** | User commands to start/stop machine | UI  MB |
| **MACHINE\_STATUS** | Machine/cycle status info to display or to enable UI selections | UI  MB |
| **UI\_TEST** | UI Input status (knobs, keys) remotely monitored via MB | UI  MB |
| UI Output status (LEDs, LCD, buzzer) remotely driven via MB | UI  MB |
| **PROG\_PAR** | Program parameters, selectable options enabled by cycle | UI  MB |
| **DISPLAY\_INFO** | Machine/cycle info (TTE, …) to be displayed on LCD | UI  MB |
| **ALARMS** | UI local alarms status to be stored on MB statistical memory | UI  MB |
| Alarms status to be displayed | UI  MB |
| **LANG\_CFG** | Language configuration data transferring | UI  MB |
| **UI\_IO\_STATUS** | Remote management of UI Output mounted on MB | UI  MB |
| Remote management of UI Input mounted on MB | UI  MB |
| **Remote STATUS** | Network Remote Status | UI  MB |
| **ADOSE\_PAR** | Autodose parameters changed in setting mode | UI  MB |
| Autodose parameters default/last values stored by MB | UI  MB |
| **ADOSE\_STATUS** | Autodose status info(warning, detergent dose, ..) | UI  MB |
| **WATCH\_DOG** | Watch-dog for communication presence checking | UI  MB |
| **MACH\_ID** | PNC ELC SN for Serialization | UI  MB |
|  | UI  MB |
| **BOARD\_ID** | Board traceability data | UI  MB |
| **GCF\_HASH** | GCF has for traceability data | UI  MB |
| **CTF\_HASH** | CTF hash for traceability data | UI  MB |
| **MAC\_ADDR** | NIU MAC address to PB | UI  MB |
| **~~SW\_ID1~~** | ~~SW identification string reading (first 4 chars)~~ | ~~UI  MB~~ |
| **~~SW\_ID2~~** | ~~SW identification string (last 4 chars)~~ | ~~UI  MB~~ |
| **~~MEMORY~~** | ~~Read/Write the contents of a memory area~~ | ~~UI  MB~~ |
| **~~FORCE\_BOOT~~** | ~~Board restart to activate boot~~ | ~~UI  MB~~ |
| **~~MACS\_ATEST~~** | ~~MACS connector automatic test loop (UI board standalone)~~ | ~~UI  UI~~ |

Table 5-7: Product Specific Messages

|  |  |
| --- | --- |
| **Main Category** | **Subgroup Messages** |
| **Machine Status Message** | Door Status, current phase, cycle execution state, progress of phases, specific sub phase information like heating, drum positioner status, water level. |
| **Operational/ Functional Message** | Start, pause, reset |
| **Display Info Message** | Time to end cycle, delay time, sesor data |
| **Wake up Message** |  |

Table 5-8: Types of Messages

## Refrigerator

### Priority

* No Priority messages are being used at the moment.

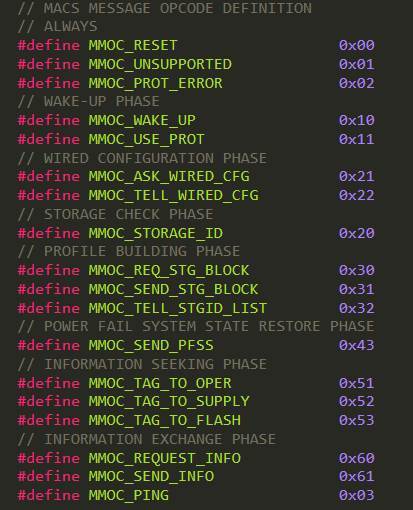
### Application layer Approach

* A system Called DIVSNet, is Master/Slave (or Requester/Responder) system, where the main board (usually) send a request to each node and wait for the answer, in polling.
* DIVSNet is the communication application layer protocol- it takes care of distributing needed information to each board connected to MACS bus.
* Information shared within a system is dubbed with a unique name called TAG.
* Master works as a NA(Network Administrator) which can only open a communication session and the passive clients can only participate in answering.

#### Special Requirements

* Approach a multi-master system, but keeping in mind of dispatch information (send parameters to each node), so during some step a node can be dominant than others (for some kind of messages, not for all)
* Wake up method, easy to understand and manage
* To have system fast and reactive, when we exchange information, using as less as possible the bus, so using the right messages in optimized way (short packet).

### Types of Messages



#### ON System Startup

Following types of communication takes place through the MACS Bus

|  |  |  |
| --- | --- | --- |
| Synchronization/ initialization messages | Device enrolling (for 1st start up) | Built the list of discovered devices on the bus |
| Wired configuration | Check the list of enrolled devices against the registered one |
| Storage check | Check the presence of the configuration |
| Profile building | Load configuration from the storage |
| Power fail system state (PFSS) restore | Recover the last saved power fail state |
| Information seeking | Check if all needed information objects are available on the system |
|  | Information exchange | Start the information exchange making the system ready to go |

|  |  |  |
| --- | --- | --- |
| Wake up message  (for device enrolling and wired configuration) | Broadcast message | NA> Passive clients (broadcast)  Passive>NA (reply) |
| Information/query Message  (for storage check) |  | ALL boards (Request and Reply) |

Note: here 4 passive devices are configured with address 0x01, 0x02, 0x03 and 0x04. (may be in different variants some of these passive devices are used instead of all)

For detailed information please find the link below:



# Notes

## Terms, Definitions, Abbreviations and Acronyms

### Abbreviations for Electrical Units

NA

### Other Abbreviations

|  |  |
| --- | --- |
| **Term** | **Defination** |
| API | Application Interface |
| EC | Expension Control |
| EDM | Electronic Design Module |
| EOL | End Of Line Test |
| FVT | Functional Verification Test​ |
| HIL | Hardware in the Loop​ |
| HMI | Human Machine Interface |
| MACS | Major Appliance Communication System​ |
| MB | Main Board |
| MC | Motor Control |
| OTA | Over The Air​ |
| PC | Powe Control |
| PCBA | Printed Circuit Board Assembly​ |
| TBD | To Be Determined |
| TOD | Time of Day |
| UI | User Interface |