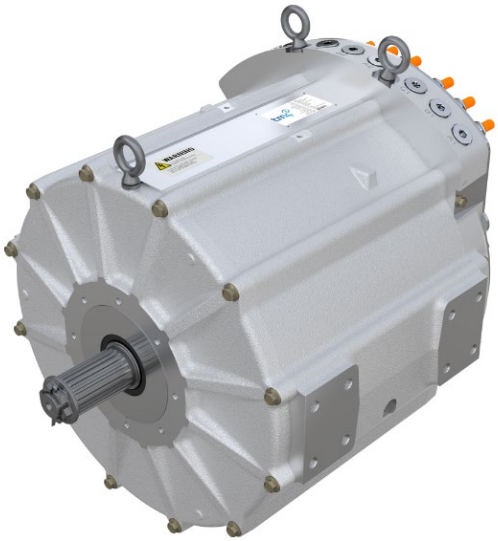
DANA TM4

Operations and Maintenance Guide

SUMO HD



Motor: LSM280A-HV-A1
Motor Control Unit (MCU): CO300-HV-A1

Product information

Product series:	TM4 SUMO™ HD
System types:	SUMO HD HV2700-9P-L SUMO HD HV3400-9P-L SUMO HD HV3500-9P-L
Product version:	ALPHA1
Product software version:	MCU_INVHP2HV_0309_TM4_MO450_240_A_v2_4_x_x MCU_INVHP2HV_0309_TM4_MO450_240_B_v2_4_x_x MCU_INVHP2HV_0309_TM4_MO450_240_C_v2_4_x_x MCU_INVHP2HV_0309_TM4_MO450_240_A_v2_6_x_x MCU_INVHP2HV_0309_TM4_MO450_240_C_v3_2_x_x
Bootloader version:	PC0045 v1.2.5.xxxxx (CAN1 & CAN 2 ports) PC0045 v1.4.2.xxxxx (CAN1 & CAN 2 ports) PC0053 v1.2.1.xxxxx (CAN1 & CAN 2 ports) PC0053 v1.4.1.xxxxx (CAN1 & CAN 2 ports)
CAN protocol version:	TM4 CAN protocol v4.0/v4.1/v4.4/v4.5/v8.0 TM4 CAN protocol J1939 v2.0/v2.1

Document information

Reference:	TG-0059 TM4 SUMO HD Operations and Maintenance Guide
Release date:	2021-05-10
Version number:	17.0

Template information

Reference:	IN-8033_3

Table of contents

List of tables

List of figures

1 Introduction

From the TM4 SUMO™ HD series, TM4 introduces this system composed of the TM4 LSM280 high-density motor and CO300-HV Motor Control Unit (MCU). The system targets electric and hybrid buses, trucks and other heavy-duty applications and offers superior powertrain performance.

This motor and MCU combination offers a cost-effective solution to vehicle electrification. The system is designed and optimized for components to operate efficiently together while getting the best performance out of the package.

The TM4 LSM280 motor technology uses permanent magnets to offer high efficiency. The motor topology is based on an external rotor technology that maximizes the use of the magnets and reduces the amount of material. The high torque/low speed of the system is designed to directly interface with standard axle differentials without the need for an intermediate gearbox.

The TM4 CO300-HV MCU utilizes the latest technology of automotive grade insulated-gate bipolar transistors (IGBT) coupled with Reflex™ driver technology to deliver the industry's highest specific power and current densities.

WARNING **Prototype X and ALPHA version components cannot be combined.**

Do not pair –X and -A versions of components in your integration.

1.1 Purpose

This technical guide describes how to use the TM4 LSM280A-HV-A1 motor in combination with the TM4 CO300-HV-A1 MCU; the main functions of the system, the location and operation of controls and the maintenance required.

1.2 Scope and target audience

This technical guide contains information useful to all personnel involved in installing and operating the TM4 SUMO™ HD system.

This guide is organised as follows:

- **Introduction** – general information about the product, definitions and document reference information.
- **Operation** – how to safely operate the product and descriptions of interfaces.
- **System behaviour** – how the system behaves during operation.
- **Usage constraints** – information on system limitations.
- **System diagnosis** – how to use TM4 ODIN event logs to diagnose system issues.
- **Maintenance and inspection** – visual checks and maintenance routine.
- **Customer service** – TM4 customer service contact details.
- **Running the system in a test environment** – how to safely set up and operate the system in a test environment.

1.3 What's new

The following update has been made to this version:

- Removed information on verifying the torque on certain HV and Phase connectors during regular maintenance as this is no longer considered necessary, see Table 28.

1.4 Disclaimer

All installation instructions, limits and warnings given in the technical documentation supplied by TM4 must be respected in order to ensure that the system runs optimally and is not at risk of damage by misuse. Operating the system outside of the established limits constitutes misuse and may invalidate any warranty.

WARNING	**Operation of the system outside the specified limitations could permanently damage the system.** The user is required to limit the usage of the system within the specifications defined in this guide.

1.5 Safety instructions

This product must be installed and manipulated by qualified personnel who are fully aware of the types of hazards involved in working with electrical circuitry and are familiar with standard practices for preventing accidents. The vehicle integrator is responsible for ensuring that proper training is given to all those who use this system in order to avoid physical, electrical and operational hazards.

1.5.1 Format and location of safety warnings in this guide

Each warning in this guide follows the same format and includes the reason for the potential hazard and how to avoid it:

WARNING	**Reason for the warning – explanation of the potential hazard.** How to avoid the hazard.

Note: When more than one safety warning applies to the same procedure, they are grouped together in one box and identified with the appropriate safety symbol:

General/Irritant/Operational: This warning symbol indicates that you are in a potentially hazardous situation that could result in damage to the product or in some situations lead to bodily harm or death.

Electrical: This warning symbol indicates that you are in a potentially hazardous situation that is electrical in nature and could result in damage to the product or in some situations lead to bodily harm or death.

To remind you of the potential hazards involved, appropriate safety warnings are located throughout this guide in procedures that if performed incorrectly may harm you or damage the product.

1.6 Definitions, acronyms and abbreviations

Auxiliary battery	Standard 12 V or 24 V vehicle battery
BMS	Battery Management System
CAN	Controller Area Network
DC	Direct Current
EEPROM	Electrically-Erasable Programmable Read-Only Memory
EMF	ElectroMotive Force
GD	Graceful Degradation
GUI	Graphical User Interface
HVIL	Hazardous Voltage Interlock Loop
MCU	Motor Control Unit (drive)
PDU	Power Distribution Unit
TM4 ODIN	TM4 diagnostic software
VMU	Vehicle Management Unit

1.7 References

All system and product documentation is available in the Documentation section of the TM4 Extranet. Requests to open an account can be made online via https://extranet.tm4.com.

Table 1 Related document references

	Reference	Title	Note
[1]	IN-8013e	MCU CAN Protocol v4.0 Specifications MCU CAN Protocol v4.1 Specifications MCU CAN Protocol v8.0 Specifications	1
[2]	IN-8013e	MCU J1939 CAN Protocol v2.0 Specifications	1
[3]	TG-0001	TM4 ODIN v4 Technical Guide	1
[4]	TG-0126 TG-0146	TM4 Communication Graceful Degradation Module Guide TM4 Position Sensor Graceful Degradation Module Guide	1
[5]	TG-0058	TM4 LSM280-A1 Installation Guide	1
[6]	TG-0057	TM4 CO300-HV-A1 Installation Guide	1
[7]	TG-xxxx	TM4 SUMO HD System Specifications	1, 2
[8]	SC-6000E-036	Kit-0076 VMU connector (wire version) assembly instructions	1
[9]	SC-6000E-025	Adjusting damping parameters in the MCU	1
[10]	SC-6000E-045	Connector lubricant application	1
[11]	IN-8103_001	Application Note: High-Speed Fault Management	1
[12]	xxxx.html	TM4 Error Codes and Corrective Action	3
[13]	INT3-xxxx	Interface drawings	1, 2
[14]	TM4 Extranet site	https://extranet.tm4.com	-

Notes:
1. Refer to the latest published version of documentation and/or software package on the TM4 Extranet site.
2. Documentation specific to your system is available on the TM4 Extranet.
3. Delivered as part of software package.

2 Operation

The TM4 SUMO™ HD system is operated via CAN message exchange between the Motor Control Unit (MCU) and the Vehicle Management Unit (VMU). The VMU is in charge of the user interface and also interfaces with the Battery Management System (BMS)/Power Distribution Unit (PDU) and all other components included in the vehicle architecture.

Typical VMU and BMS/PDU operations involved in TM4 SUMO™ HD include:

- Applying power to the system from the auxiliary battery.
- Asserting the enable signal (VMU or vehicle ignition).
- Performing high-voltage battery pre-charge (BMS/PDU).
- Applying high-voltage battery voltage to the system (BMS/PDU).
- Transmitting CAN messages (VMU) with the MCU:
  - Starting and stopping the system.
  - Applying a mechanical torque.
  - Safely shutting down the system.
  - Verifying operational status.

2.1 Safety warnings related to operating the system

Read these general safety warnings before operating the system.

WARNING **Mishandling of this product may damage the product and/or cause injury or death.**

- Do not attempt to open or repair this product. In case of damaged casing or suspected product malfunction, contact TM4.
- Use only recommended points to lift and secure the system.

When manipulating and/or installing this product, you must **NOT**:

- Modify any part of the MCU.
- Apply any external load to the casing of the MCU.

Applying excessive torque or speed when the motor is cold may negatively impact the durability of the component.

When the motor is used in an environment with an ambient temperature of below 0 °C, it is recommended that you avoid using excessive speed or torque and follow a normal driving cycle for the first few minutes until the motor has had a chance to warm up.
Note: This is a best practice commonly applicable to any mechanical equipment. The TM4 motor does not limit performance; full speed and torque are available, but not recommended, after a cold start.

The product can reach very high temperatures that can cause serious burns and/or other injuries.

Avoid any contact with surfaces during and directly after use.

2.2 Pre-requisites

2.2.1 CAN-enabled vehicle controller

The traction system is mainly controlled through the exchange of messages over the CAN bus. A CAN-enabled vehicle controller is required to operate the traction system.

Refer to the Standard CAN Protocol Specifications [1] and Section 2.6.3.1 for detailed information about the standard CAN frame protocol.

Refer to the J1939 CAN Protocol Specifications [2] and Section 2.6.5.1 for detailed information about the J1939/extended CAN frame protocol.

2.2.2 High-voltage and auxiliary batteries

The traction system is not involved in the management of either the high-voltage or the auxiliary battery.

The system will draw power and recharge the high-voltage battery based on the torque request and will limit the maximum charge/discharge currents as specified using the associated CAN protocol message. However, in case of hazardous behaviour or maintenance of the traction system, two hardware signals should be connected for safety purposes: **HVIL** and **EmergencyStop**.

When the system is fully installed and connected, the **HVIL** internal loop is closed therefore resulting in a short circuit between its two input pins. The **HVIL** signal is used to open the high-voltage battery contactor when its internal loop is opened during product maintenance or repair (e.g. removal of MCU cover or disconnection of motor sensor cable) thereby protecting the user.

The **Emergency Stop** signal is driven by a software algorithm.

Refer to the suggested **HVIL** and **EmergencyStop** safety circuit diagram in the VMU interface harness section of the MCU Installation Guide [6] that shows how to utilise these safety features.

Disclaimer: Note that if you choose not to implement this circuit, TM4 is not responsible for any effects of hazardous behaviour or system malfunction during maintenance or due to a situation requiring an emergency stop.

2.2.3 Vehicle management unit

A vehicle management unit (VMU) is required to interface with the different user interface peripherals (pedals, drive selector, etc.) and handle the traction system through the exchange of the required CAN messages with the MCU.

2.3 System calibration

For each of the following sub-sections, use the following procedure to configure the parameters.

1. Start TM4 ODIN and connect to the system.
2. Open the TM4 ODIN file UserInterface.odn4.
3. From the **Parameters** tab, expand the **Drive** folder.
4. Set the parameters to the desired values.
5. Set variable **DrvParameters.Save** to 1 and wait for it to come back to – to save the parameters to non-volatile memory.
6. Switch OFF the auxiliary power (12 V/24 V) and wait 5 seconds and switch it ON again.

2.3.1 CAN baud rate configuration

The baud rate of each CAN bus can be configured using the ODIN software. There is a choice of 4 available CAN baud rates: 125 kbps, 250 kbps, 500 kbps and 1 mbps. The default CAN baud rate of each CAN bus is 500 kbps.

Table 2 CAN baud rate parameter descriptions

Parameter	Description
Can1BaudRate	CAN baud rate value associated with MCU CAN bus 1. 0: 125 kbps 1: 250 kbps 2: 500 kbps 3: 1 mbps
Can2BaudRate	CAN baud rate value associated with MCU CAN bus 2. 0: 125 kbps 1: 250 kbps 2: 500 kbps 3: 1 mbps

2.3.2 CAN protocol bus configuration

It is possible to configure which CAN bus is connected to the VMU.

Table 3 Control CAN bus parameter descriptions

Parameter	Description
ControlCanPortNumber	Specifies the CAN bus on which the VMU will control the MCU using the CAN Protocol (Standard and J1939). 1: The VMU is connected to CAN bus 1. 2: The VMU is connected to CAN bus 2.

2.3.3 Control protocol selection

The MCU proposes two types of CAN protocol:

- CAN standard frame
- CAN J1939 (extended frame).

Table 4 Control CAN bus parameter descriptions

Parameter	Description
J1939.UseJ1939 ControlProtocol	***Available in version C of software only***. Specifies which control protocol to use to control the MCU. 0: Standard – MCU Standard CAN Protocol Specifications [1]. 1: J1939 – MCU J1939 CAN Protocol Specifications [2].
J1939AndStandard ProtocolMixAllow	***Available in version C of software only***. Activates communication using CAN standard protocol v4.0 and v4.1 that is also able to receive and transmit diagnostic messages in extended J1939 format. **Note**: Parameter **J1939.UseJ1939ControlProtocol** must be disabled before enabling this parameter.

2.3.4 Client CAN protocol selection

A specific client CAN protocol can be selected; the selection of protocol varies depending on the software version but is clearly indicated in the drop-down list associated with the **CommClientSelectProtocol** parameter.

Note: This parameter may be used to replace the CAN communication initialization process described in the TM4 CAN Protocol Specifications [1].

To select a specific CAN protocol version, choose an item from the drop-down list (items available in list may vary):

Table 5 Client CAN protocol selection

Parameter	Item in list	Description
CommClient SelectProtocol	0:VMU_REQUEST_CC_SEL 2:TM4_40_CC_SEL 3:TM4_41_CC_SEL 8:TM4_80_CC_SEL	0: = VMU will select protocol using the CAN communication initialization process [1] 2: = TM4 v4.0 Protocol 3: = TM4 v4.1 Protocol 8: = TM4 v8.0 Protocol

2.3.5 Bus diagnostic address configuration

TM4 uses TM4 ODIN as debug and configuration software. It communicates with the MCU using CAN and has a configurable base address on each CAN bus. TM4 ODIN will only use standard CAN frame, even when the J1939 CAN protocol is being used.

Table 6 Diagnostic address descriptions

Parameter	Description
OdinCan1BaseAddress	CAN base address of the communication with the ODIN software on CAN bus 1. This base address reserves 32 addresses from the configured value. The default value is 0x0660 which reserves the range [0x0660, 0x067F].
OdinCan2BaseAddress	CAN base address of the communication with the ODIN software on CAN bus 2. This base address reserves 32 addresses from the configured value. The default value is 0x0680 which reserves the range [0x0680, 0x069F].

2.3.6 Standard CAN protocol bus address configuration

When the Standard CAN Protocol Specification [1] is selected, the CAN base addresses of the bus can be configured using the ODIN software. Each of these addresses is referenced in the CAN Protocol Specifications [1]. For more information on the CAN base addresses, refer to the CAN Protocol Specifications [1].

Table 7 Standard CAN bus address parameter descriptions

Parameter	Description
HighPrioBaseAddress	CAN base address of the high priority message of the system. Referred to as **Can1BaseAddr1** in the CAN Protocol Specifications [1].
LowPrioBaseAddress	CAN base address of the low priority message of the system. Referred to as **Can1BaseAddr2** in the CAN Protocol Specifications [1].

2.3.7 J1939 CAN protocol bus configuration

When selected, the J1939 parameters can be configured using the ODIN software. For more information on each of these items, refer to the J1939 CAN Protocol Specifications [2].

Table 8 J1939 CAN bus address parameter descriptions

Parameter	Description
J1939. J1939Name.EcuInstance	Identifies the ECU instance inside the function field. An identity number is 3-bits wide. The default value is 0x0 (0).
J1939. J1939Name.IdentityNumber	Identifies the ECU identity number inside the function field. An identity number is 21-bits wide. The default value is 0x0 (0).
J1939.SourceAddress	Represents the address of the MCU. A source address is 8-bits wide. The default value is 0xEF (239).
J1939.VmuSourceAddress[1]	Represents the address of the controlling device (VMU). A source address is 8 bits wide. The default value is 0x27 (39).

2.3.8 HV battery operational range received via CAN messages

The VMU sends the high-voltage battery operational range (minimum and maximum voltage) to the MCU via CAN messages. The MCU reduces system performance around the range boundaries depending on the values of the parameters described in Table 9. This reduction in system performance can be configured to start and end either before or after the maximum and minimum values sent.

Table 9 High-voltage battery range definition parameter descriptions

Parameter	Description
VBatt.BoundaryType	Indicates if minimum and maximum voltage values sent by the VMU represent the point **before** which or **after** which derating should begin. 0: VBATT_INNER_LIMITS: Reduces performance **before** reaching the specified limits. 1: VBATT_OUTER_LIMITS: Allows full system performance up to the specified limits and then reduces performance **after** reaching the specified limits. The default value is 0x1 (1).
VBatt.TractionMargin	Voltage derating margin during traction (around battery minimum voltage boundary). This is a read-only parameter set by the manufacturer. Default value is 50 V.

[1] The MCU uses this address to react to the Vmu Info 1 message.

Parameter	Description
VBatt.RegenerationMargin	Voltage derating margin during regeneration (battery maximum voltage boundary). This is a read-only parameter set by the manufacturer. Default value is 50 V.

Figure 1 Inner/Outer limit configuration around MinBattVoltage

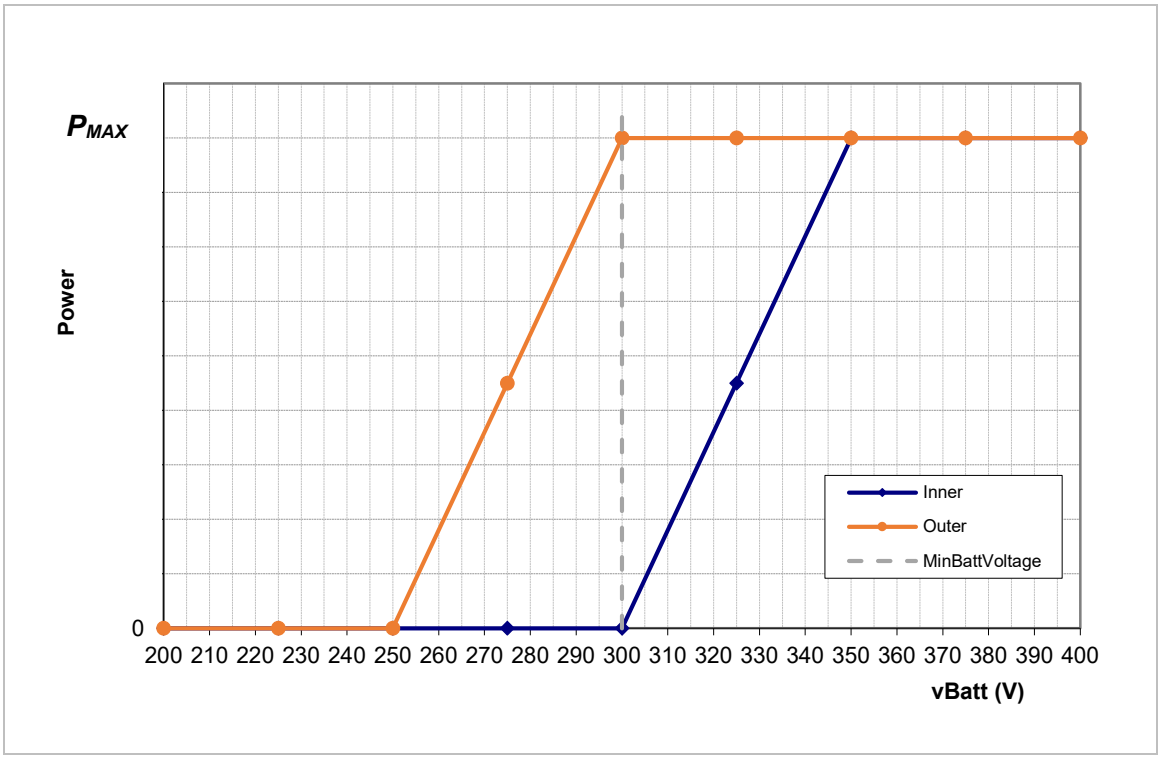
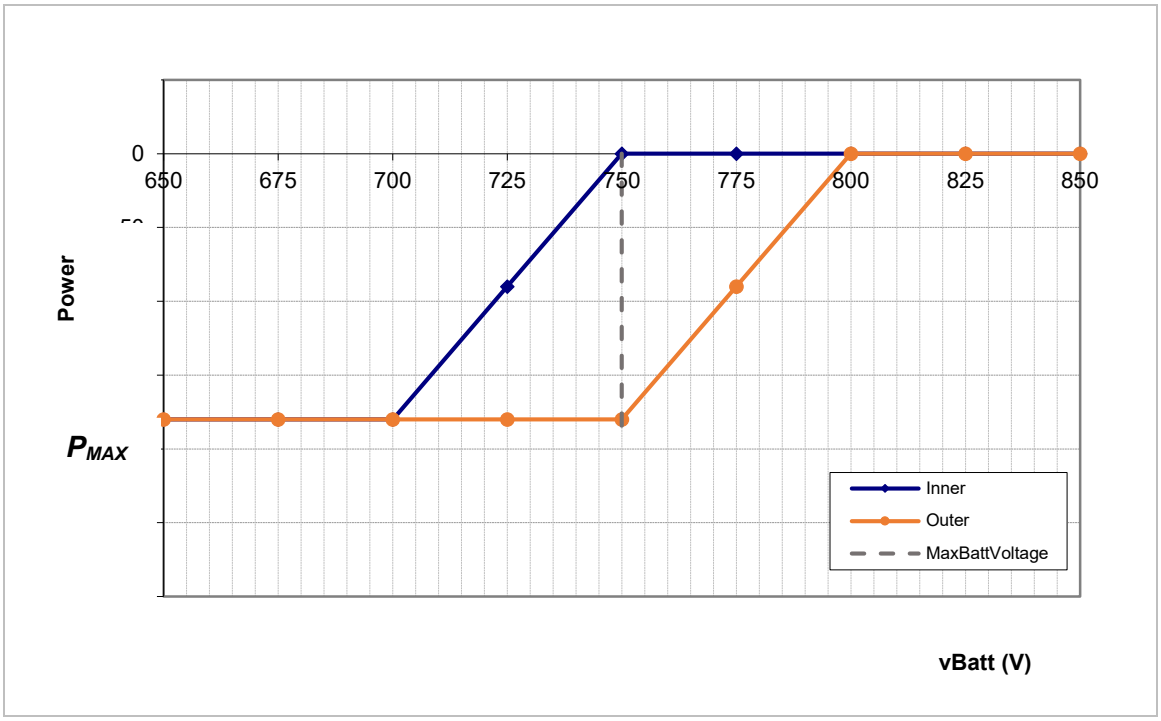


Figure 2 Inner/Outer limit configuration around MaxBattVoltage



2.4 Motor calibration

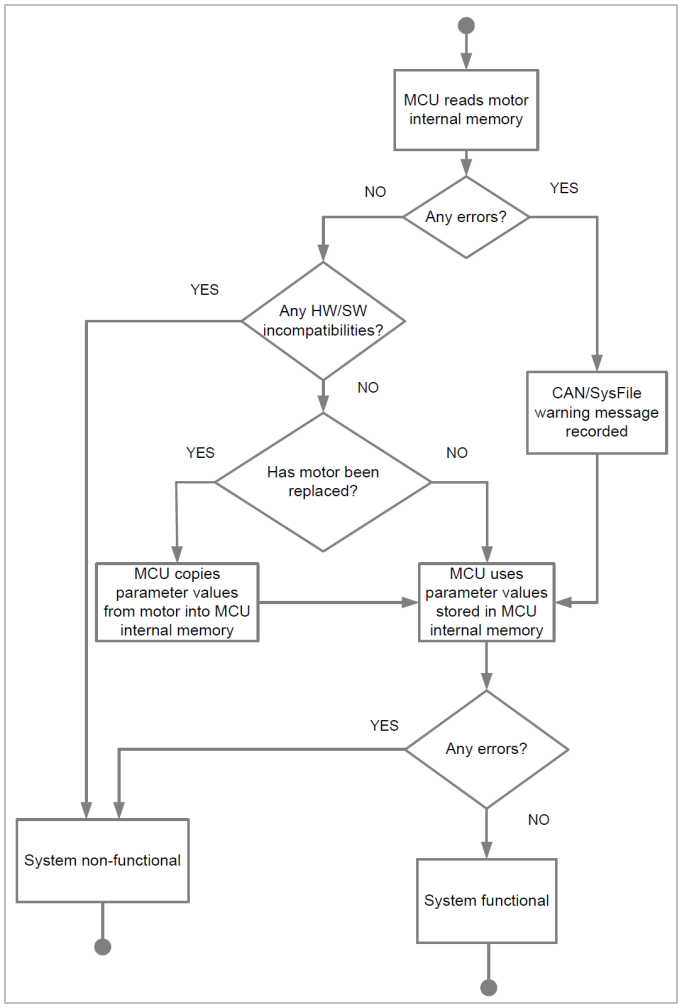
Each motor has its own internal permanent memory which stores different characteristics for calibration purposes (e.g. motor phase, maximum speed, and maximum torque). For this reason, the MCU and motor do not need to be paired.

Because of the information stored in permanent memory, the system does not usually need to be manually recalibrated before use in a test environment or in a vehicle.

However, if you are using a non-TM4 motor, have updated the software, or replaced a component of the original TM4 MCU/motor pairing, we recommend that you recalibrate the motor. Under these circumstances, whether you recalibrate or not, you **must ensure** you do not get any system errors/warnings in the SysFile before use. This is because although the MCU may seem to function correctly, corrupt parameters from the previous pairing might cause damage to the system or injury to the user.

A copy of the motor calibration data will be stored in the internal permanent memory of the TM4 MCU and will be valid for that specific TM4 MCU/motor pairing only.

Figure 3 Managing MCU and motor internal permanent memory



The following procedures are required to complete calibration:

- Configuration of the rotation direction of the motor, see Section 2.4.1.
- Calibration of the motor phase connection sequence, see Section 2.4.2.
- Configuration of system limit parameters, see Section 2.4.3.
- Configuration of speed control parameters (Kp, Ki), see Section 2.4.4.

WARNING **Risk of unexpected behavior and safety issues.**

Motor calibration is required in order to ensure safe usage of the system and maximize performance; failure to complete the calibration procedure when required might result in safety issues, unexpected behavior and/or performance.

2.4.1 Configuring the rotation direction of the motor

The following steps describe how to configure the rotation direction of the motor in order that the application of positive torque will cause the vehicle to move forward.

Note: This procedure is optional as it depends on the position the motor will be installed in the vehicle.

> **WARNING** **Mishandling of this product may cause injury or death.**
>
> In order to avoid personal injury, the motor must be free to turn and the MCU high-voltage bus must be disconnected (or not energized).

1. Remove the high voltage from the MCU by turning OFF the high-voltage power supply or disconnecting the cable.
2. Ensure that the MCU and motor are properly connected and switch ON the auxiliary power (12 V/24 V).
3. Start TM4 ODIN and connect to the system.
4. Open the TM4 ODIN file UserInterface.odn4.
5. From the **Parameters** tab, expand the **Drive** and **Motor** folders.
6. Set the value of parameter **ReverseRotationDirection** to 0.
7. Set variable **DrvParameters.Save** to 1 and wait for the value to come back to 0 meaning that the parameter values have been saved to non-volatile memory.
8. Switch OFF the auxiliary power (12 V/24 V), wait 5 seconds and switch it ON again.
9. Open the **System Status** tab and observe **MotorSpeed**.
10. Turn the motor shaft by hand in the positive direction (as described by the manufacturer) and verify if **MotorSpeed** is positive or negative. If the speed is positive, the calibration is complete. If the speed is negative, set the value of **ReverseRotationDirection** to 1, and save parameters values using **DrvParameters.Save**.
11. Redo steps 8-10 and confirm that **MotorSpeed** is now positive meaning that positive torque will cause the vehicle to move forward.
12. If after repeating this procedure, the value of **MotorSpeed** is still negative, contact TM4 Customer Service, see Section 7 for contact information.

2.4.2 Calibrating the motor phase connection sequence

The following steps describe how to configure the motor phase connection sequence using TM4 ODIN.

Depending on the selected phase cable connection scenario, the **PhaseCableReversed** parameter can be configured via the TM4 ODIN GUI, in one of two ways:

- **Standard** connection (A1-1; B1-2, etc.): **PhaseCableReversed** should be set to 0 (default);
- **Reversed** connection (A1-9; B1-8, etc.): **PhaseCableReversed** should be set to 1.

To save the value of the **PhaseCableReversed** parameter, refer to Section 2.3.

WARNING **Mishandling of this product may cause injury or death.**

- In order to avoid personal injury, the motor must be free to turn.

Danger of system overheating.

Ensure that cooling system is connected and functioning, and that coolant is flowing in the MCU and motor before beginning the motor phase calibration.

Note: If the system is installed in a vehicle, we recommended that the vehicle is raised off the ground or that the shaft is disconnected.

2.4.3 Configuring system limit parameters

Certain parameters (see Table 10) can be modified to accommodate, for example, certain gearboxes. To ensure system safety and maximize performance you must configure system limits based on the actual limitations of the selected system components. At start-up, these configured limits will be taken into consideration to determine the maximum safe working performance factors of the system.

Figure 4 System limit parameters (forward speed)

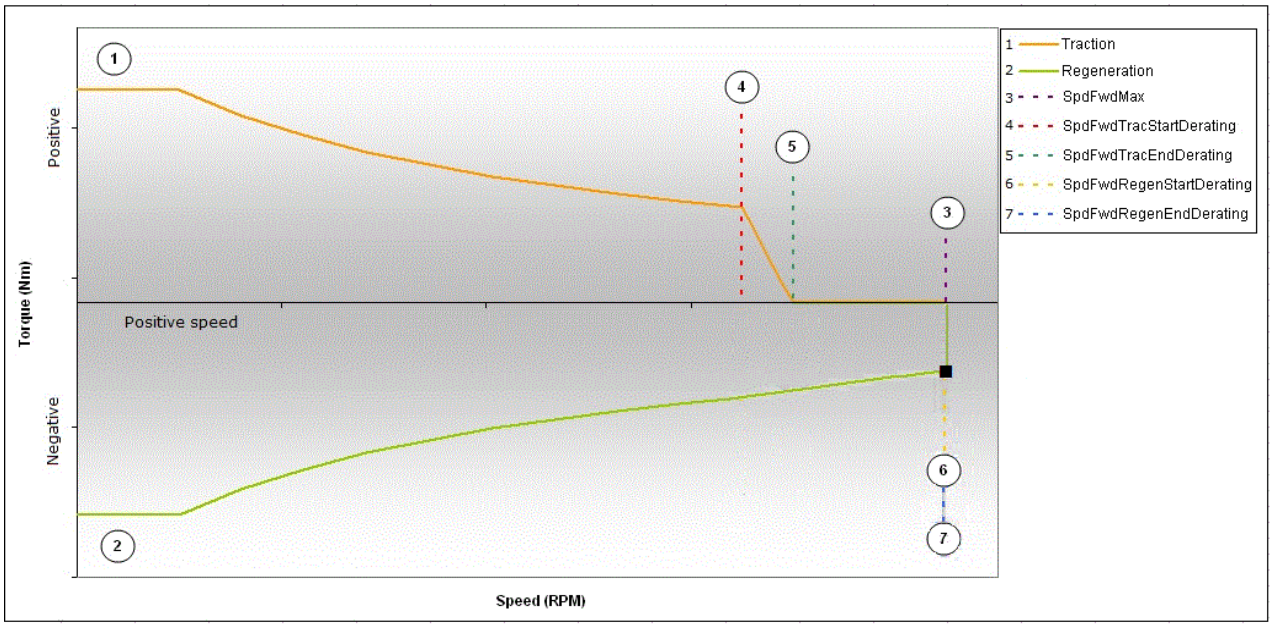
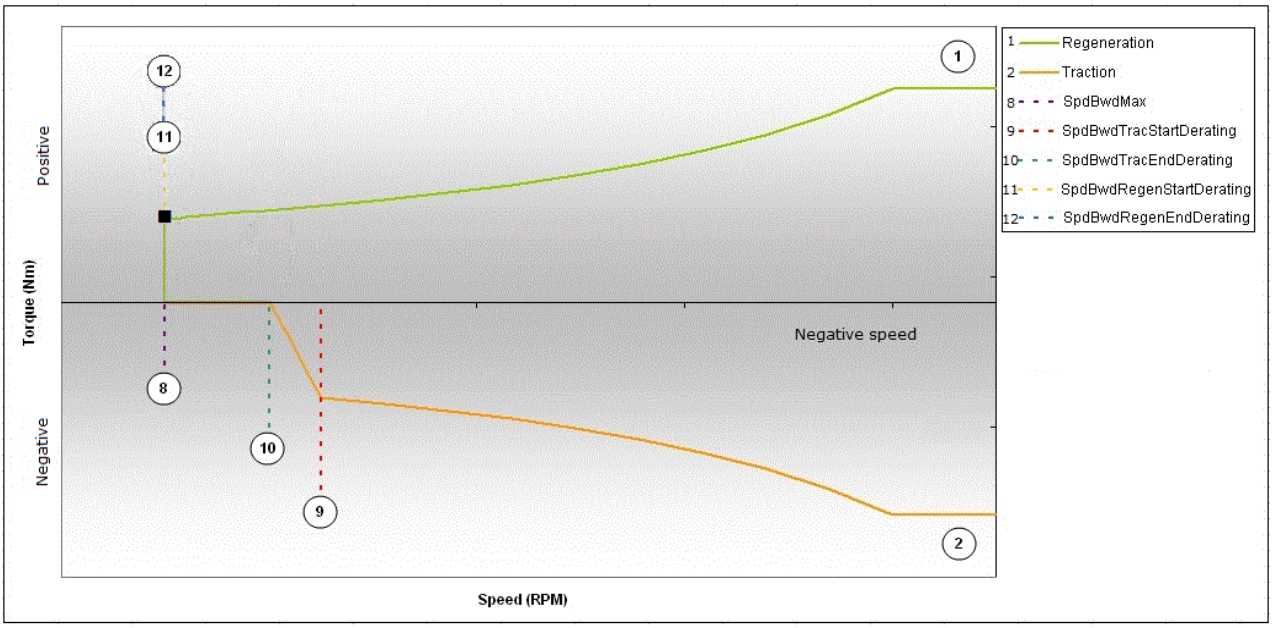


Figure 5 System limit parameters (reverse speed)



Note: For parameters associated with the numbers in Figure 4 and Figure 5, see Table 10.

Table 10 System limit parameter descriptions

#	Parameters	Description
n/a	**Motor.ReverseRotationDirection**	Normal or reverse rotation direction. 0 (False = normal) or 1 (True = reverse) Default value is 0 (normal) See note 1.
1	**Motor.PositiveTorqueMax**	Maximum positive torque. (Nm)
2	**Motor.NegativeTorqueMax**	Maximum negative torque. (Nm)
n/a	**Motor.TorqueRamp**	Increases and decreases torque ramp for the torque control. (Nm/s)
3	**Motor.SpdFwdMax**	Absolute maximum forward speed. (RPM)
4	**Motor.SpdFwdTracStartDerating**	Defines the start point of the traction derating zone in forward direction. (RPM)
5	**Motor.SpdFwdTracEndDerating**	Defines the end point of the traction derating zone in forward direction. (RPM)
6	**Motor.SpdFwdRegenStartDerating**	Defines the start point of the regeneration derating zone in forward direction. (RPM)
7	**Motor.SpdFwdRegenEndDerating**	Defines the end point of the regeneration derating zone in forward direction. (RPM)
8	**Motor.SpdBwdMax**	Absolute maximum reverse speed. (RPM)
9	**Motor.SpdBwdTracStartDerating**	Defines the start point of the traction derating zone in reverse direction. (RPM)
10	**Motor.SpdBwdTracEndDerating**	Defines the end point of the traction derating zone in reverse direction. (RPM)
11	**Motor.SpdBwdRegenStartDerating**	Defines the start point of the regeneration derating zone in reverse direction. (RPM)
12	**Motor.SpdBwdRegenEndDerating**	Defines the end point of the regeneration derating zone in reverse direction. (RPM)

Notes:

1. When the **Motor.ReverseRotationDirection** parameter is set to True (1), it inverses the direction convention so that applying positive torque results in the vehicle moving in reverse and negative torque results in the vehicle moving forward. This is to accommodate a variety of installation positions of the motor during integration into a vehicle.

The following steps describe how to configure the system limits using TM4 ODIN.

1 Ensure that the MCU and the motor are properly connected and powered.
2 Switch ON TM4 ODIN and connect to the system.
3 Open the TM4 ODIN file UserInterface.odn4.
4 In the **Parameters** watch window, based on the range and description of each parameter, modify the current value to the desired system limits.
5 Save the new system limits by changing the **DrvParameters.Save** item to 1 and wait for it to come back to 0 to save the parameters to non-volatile memory.
6 Turn OFF the system and wait 10 seconds.
7 Turn ON the system.
8 In the **Parameters** watch window, verify that the configured limits are equal to the values previously entered.

Note: If the system limits were not properly configured the first time, the procedure should be repeated before contacting TM4 Customer Service; see Section 7 for contact information.

2.4.4 Configuring speed control parameters

The standard MCU control mode is a torque control in which the MCU controls the torque to follow the **TorqueCommand**. However, a speed control mode has also been introduced for the torque to follow a **SpeedCommand**, see Table 16.

The speed ramp parameter is followed by the **SpeedCommand** to smooth the speed transition.

The speed control mode is based on a Proportional and Integral (PI) controller and these factors are configured by two parameters to allow the speed control to adjust the inertia of rotating components and ensure optimum dynamic behaviour (quickly reaching the limit without excessive overshoot).

Table 11 Speed control parameter descriptions

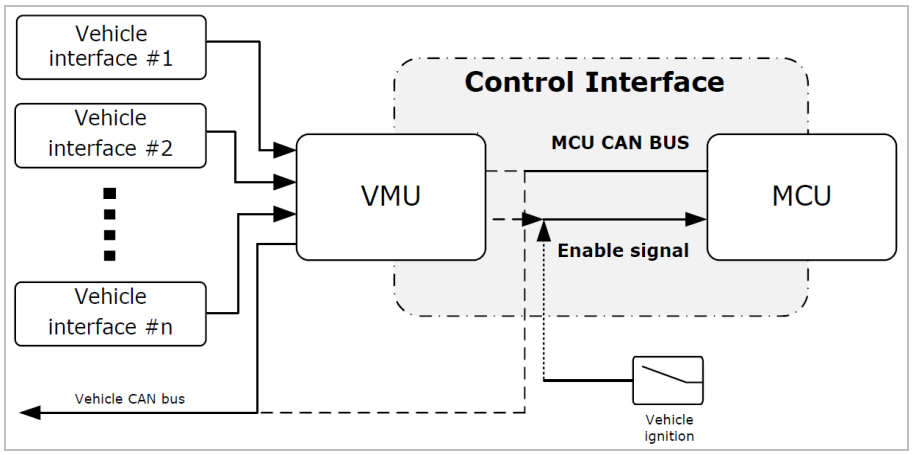
Parameter	Description
Motor.SpeedRamp	Increases and decreases the speed ramp for the speed control. (RPM/s)
Motor.SpeedControlKp	Proportional factor of the speed control PI controller.
Motor.SpeedControlKi	Integral factor of the speed control PI controller

2.5 Control interface

To control the MCU, an Enable line is used to activate the system and a CAN BUS is used through which all messages required to operate the system are carried.

Refer to Figure 6 for the traction system vehicle context when integrated in a typical automotive application (VMU dedicated CAN bus with MCU).

Figure 6 System control interface



Notes:

1. The MCU CAN bus can be controlled through the dedicated CAN bus of the VMU or through the vehicle CAN bus. The dedicated bus structure is preferred to lower the traffic on the vehicle bus; however, both configurations are supported. For more information on the communication structure, refer to the MCU CAN Protocol Specifications [1].
2. The enable signal of the MCU (IGNITION input) can be controlled either from the VMU (Wakeup motor output) or from the vehicle ignition.

2.6 Control operation

2.6.1 TM4 MCU system states

Figure 7 shows how the TM4 SUMO™ HD system reacts to CAN bus control messages in CAN v8.0; for information specific to either CAN 4.0 or 4.1, see the protocol specifications document for each one [1]. Transition operational requests are made through the **VmuCommand1.OperationRequest** CAN message.

Table 12 System state descriptions

Values	Description	State
Initialization	Initializing hardware and software. This is the default state at power on.	Transitional
Standby	The system is stopped, no torque/speed/voltage command is allowed.	Final
Activation	Start up, activation and internal pre-charge.	Transitional
Deactivation	Discharge and disabling.	Transitional
Operational	Operational state (including degraded modes). The system is able to apply torque, speed and voltage command.	Final
Failure	Indicates that the system has failed.	Final
Shutdown	Deactivates the system.	Transitional
WaitForSafeCondition	The system still needs to complete some actions (e.g. cool system, reduce speed, etc.) before being in a safe enough condition to power off. The **WaitForSafeCondition** state is configured to be active until the MCU meets all safety conditions, such as cool enough to be in an acceptable temperature zone (non-derating operation state) or up to a maximum period of 10 minutes (in order to protect the battery from overuse). If the MCU is in this state, this allows the VMU to perform the proper actions to put the system into a safe condition. **Note: WaitForSafeCondition** is available in CAN v8.0; for CAN v4.0 and 4.1 the **WaitForCooling** state is used taking only temperature into account	Transitional
ReadyToPowerOff	The power could be removed at this point. Make sure you switch off only when the state goes into **ReadyToPowerOff**.	Final

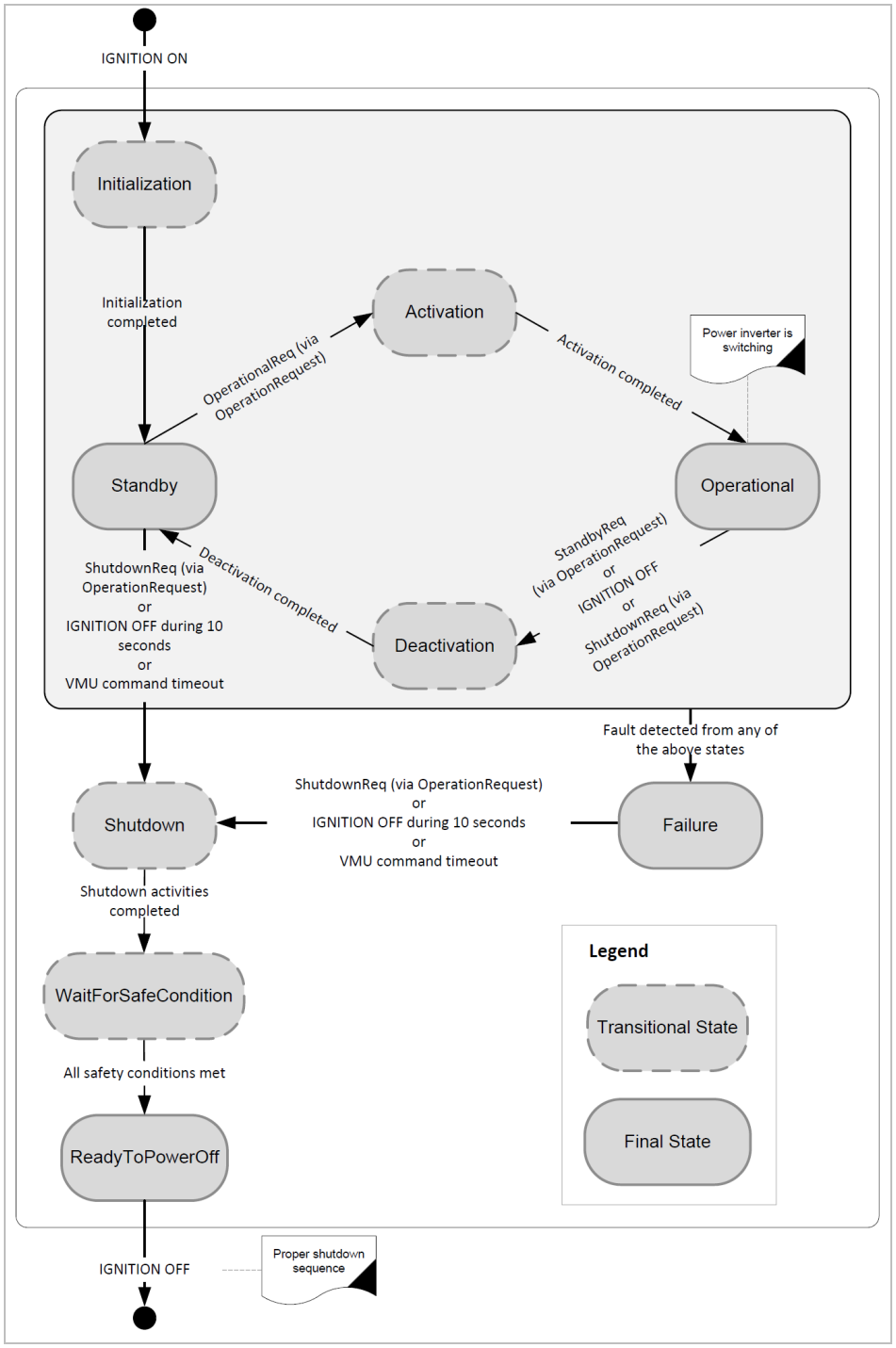
2.6.2 Operating modes

The operating modes are used to control the different state transitions and statuses of the system.

Table 13 Operating mode descriptions

Id	Request name	Description
1	**Standby Request**	Sets the system in a standby state. This is the default state when the system is switched ON.
2	**Operational Request**	Sets the system in an operational state. It sets the MCU electronics to be able to respond to a motor torque request.
3	**Shutdown Request**	Prepares the system to be powered off. It saves system information and BlackBox data in its internal permanent memory.

Figure 7 MCU state transition diagram



2.6.3 CAN system management

The only way to operate the MCU is by communicating using a CAN protocol. The MCU proposes two CAN protocols:

- Standard CAN frame
- Extended CAN frame (J1939) - ***Available in software version C only***.

These protocols provide the ability to control the maximum current of charge/discharge, voltage range, torque command, speed command or voltage command and the system states. Furthermore, the system provides information over the CAN BUS, such as motor speed, applied torque and more.

Both the standard and J1939 CAN protocol specifications are used by multiple TM4 applications. We recommend using the latest version of the CAN Specifications; for more information, refer to the MCU CAN Protocol Specifications [1] or to the J1939 MCU CAN Protocol Specifications [2].

2.6.3.1 Standard CAN protocol specification

This section describes specific usage of the MCU when using the Standard CAN Protocol Specification. It refers to CAN messages from the CAN Protocol Specifications, for more information, refer to the MCU CAN Protocol Specifications [1].

2.6.3.1.1 Message timeout

After start-up, once either a **VmuCommand1** or **VmuCommand2** CAN message has been sent, both will then need to be sent continuously along with the **VmuCommandSafety** message, within the time defined in the periodicity of each message. Refer to the MCU Standard CAN Protocol Specifications [1] for more information on the timeout on each of the messages. If these messages are not sent, a timeout will occur, the MCU will fall into failure mode and the system will need to be restarted.

2.6.3.1.2 Initializing CAN communication

Once the auxiliary power is applied to the system and the enable signal (IGNITION input) is asserted to the MCU, the VMU must request use of a specific protocol version using **VmuSelectProtocolVersion** (starting with the highest version it supports) and be able to recognize that the MCU is ready to communicate. At startup, the MCU will remain silent until it received a **VmuSelectProtocolVersion** requesting a valid protocol version.

2.6.3.1.3 Shutdown sequence

To activate the MCU shutdown sequence from an operational state, the **VmuCommand1.OperationRequest** CAN message can be sent by the VMU with either the **StandbyReq** attribute, then the **ShutdownReq** attribute, or directly with the **ShutdownReq** attribute.

1. If the **StandbyReq** attribute is sent, the VMU will wait until it receives the **McuCommand1Response.State** MCU CAN message with the **Standby** attribute confirming it has reached that state through the completion of the transitional **Deactivation** state. The VMU could then send the **VmuCommand1.OperationRequest** CAN message with the **ShutdownReq** attribute to the MCU. The MCU will then go through the **Shutdown** and the **WaitForSafeCondition** transitional states before reaching the **ReadyToPowerOff** state confirmed by the **McuCommand1Response.State** MCU CAN message with the **ReadyToPowerOff** attribute.

2. If the **ShutdownReq** attribute is sent, the MCU would transit through the **Deactivation** state before reaching the **Standby** state and then it would transit through the **Shutdown** and **WaitForSafeCondition** states before reaching the **ReadyToPowerOff** state confirmed by the **McuCommand1Response.State** CAN message with the **ReadyToPowerOff** attribute.

   Note: In both cases the MCU will shut itself down only when it has reached a **ReadyToPowerOff** state and the MCU enable line (IGNITION input) is low.

2.6.3.1.4 Failure management

The system changes its state to **Failure** if incompatibility errors occur during initialization (see Section 2.7.3) or any other state shown in Figure 7; loss of CAN communication also results in **Failure**.
Once in **Failure**, the system cannot be operated but the MCU is still using power from the auxiliary battery. In order to prevent draining the battery, as soon as the system goes into **Failure**, one of the following operations must be performed:

1. Send a **ShutdownReq** via **OperationRequest** to initiate the **Shutdown** sequence;
2. Turn the IGNITION input to OFF; after 10 seconds the system will initiate the **Shutdown** sequence.

2.6.3.1.5 Performance availability

Information is returned on the current thermal status of the complete system via the **McuThermal1 [ThermalIndicator]** signal. For more information about performance availability, refer to Section 3.2.2.

2.6.3.1.6 Cooling

The cooling requirements of the motor and MCU are handled by two signals **McuThermal1 [DriveCoolingFlowRequest]** and **McuThermal1 [MotorCoolingFlowRequest]**. For more information on cooling management, refer to Section 3.2.1.

2.6.3.1.7 Software and protocol version

The software version can be requested by sending the following 2 messages:

- **VmuPollingMcu** with the **McuSwVersion1** ID as the **McuRequestId**
- **VmuPollingMcu** with the **McuSwVersion2** ID as the **McuRequestId**

The MCU will respond with two **McuPollingMcuResponse** messages; one containing the **McuSwVersion1** as the **McuRequestId** and the other the **McuSwVersion2** as the **McuRequestId**. The full software version is contained within those 2 values.

The protocol version can be requested by sending this message:

- **VmuPollingMcu. McuRequestId** with the **McuProtocolVersion** attribute.

Note: The extension pack value returned will be 0.0 since it is not used by the TM4 SUMO™ HD application.

2.6.4 Specific application of CAN protocol in TM4 SUMO™ HD

The MCU CAN Protocol [1] is used by multiple TM4 applications including the TM4 SUMO™ HD system. This section has information on how TM4 SUMO™ HD uses or does not use certain message attributes that are available in CAN v4.0/CAN v4.1/CAN v8.0. It is recommended to use to latest version of the CAN Protocol.

2.6.4.1 McuEventInfo1

The **McuEventInfo1** message is a notification of major event occurrence in the MCU which may trigger a safety action in the VMU.

2.6.4.1.1 Attributes used by TM4 SUMO™ HD

The **FaultPhaseToChassis** attribute is used by the application. Therefore, a fault value (bit at 1) indicates a fault.

The associated status values (**FaultPhaseToChassisStatus**) is also used. Therefore, if a status value is at 1, the VMU should consider that the associated fault value received is invalid.

2.6.4.1.2 Attributes not used by TM4 SUMO™ HD

The **FaultInterlockBatteryCable**, **FaultInterlockMotorCable** and **FaultDriveCoverWasOpened** attributes are not used by the application. The value sent for those attributes will always be valid (bit at 0).

The associated status values (**FaultInterlockBatteryCableStatus**, **FaultInterlockMotorCableStatus** and **FaultDriveCoverWasOpenedStatus**) will always be sent with a "not used" status (bit at 1). Therefore, the VMU should consider that the functionality is not used and not consider the associated fault attribute.

2.6.5 Operating modes

VmuCommand1, **VmuCommand2**, **VmuCommandSafety**, and **McuCommand1Reponse** messages are used to control the different state transitions and statuses of the system.

Table 14 VmuCommand1.OperationRequest value descriptions

Values	Description
StandbyReq	Sets the system in a standby state. This is the default state when the system is switched ON.
OperationalReq	Sets the system in an operational state. It sets the MCU electronics to be able to respond to a motor torque request.
ShutdownReq	Prepares the system to be powered off. It saves system information and BlackBox data in its internal permanent memory.

Table 15 VmuCommand2.OperationalMode/McuCommand1Response.OperationalMode value descriptions

Values	Description
Neutral	Sets the system in a neutral mode where no torque/speed command is applied to the motor output shaft.
Ev (Purely electric)	Sets the system in an operational state where the torque/speed command is enabled.

Table 16 VmuCommand2.CommandMode/VmuCommandSafety.ReferenceCommandMode value descriptions

Values	Description
TorqueMode	The MCU is controlled by torque using the **TorqueCommand** signal.
SpeedMode	The MCU is controlled by speed using the **SpeedCommand** signal.

Table 17 McuCommand1Response.TractionDerating/McuCommand1Response.RegenDerating value descriptions

Values	Description
None	System can provide the maximum torque.
Battery current	The torque is limited by the available current of the traction battery.
Internal	Internal limitation of the current/power/other.
Overspeed	The torque is limited because the motor speed has reached the design working system speed limits.
Overheat system	Limitation caused by overheating of some components of the system. (CAN v4.0 only).
Defect sensor drive	Limitation caused by defective sensors. (CAN v4.1 only).
Max power	Limitation of the available torque caused by the maximum system performance power.
DC voltage	Limitation of the available torque caused by the level of the DC voltage outside the high-voltage battery voltage range.
Overheat drive	Limitation caused by overheating of some components of the drive. (CAN v4.1 only)
Overheat machine	Limitation caused by overheating of some components of the machine. (CAN v4.1 only)
Defect sensor machine	Limitation caused by defective sensors. (CAN v4.1 only)
Limp Home	Limitation of available torque caused by the Graceful Degradation module.

Refer to Table 24 for details about derating causes.

Table 18 MCUCommand1Response.State value descriptions

Values	Description	State
Initialization	Initializing	Transitional
Standby	Stopped. This is the default state at start-up.	Final
Activation	Start up, activation and pre-charge.	Transitional
Deactivation	Discharge and disabling.	Transitional
Operational	Operational state (including degraded modes).	Final
Failure	Indicates that the system has failed.	Final
Shutdown	Deactivated.	Transitional

Values	Description	State
WaitForSafeCondition	The system still needs to complete some actions (e.g. cool system, reduce speed, etc.) before being in a safe enough condition to power off. The **WaitForSafeCondition** state is configured to be active until the MCU meets all safety conditions, such as cool enough to be in an acceptable temperature zone (non-derating operation state) or up to a maximum period of 10 minutes (in order to protect the battery from overuse). If the MCU is in this state, this allows the VMU to perform the proper actions to put the system into a safe condition. **Note**: CAN v4.0 and 4.1 use the **WaitForCooling** state that takes only temperature into account.	Transitional
ReadyToPowerOff	Before turning off power, a **VmuCommand1 ShutdownReq** (shutdown request) should be sent to the MCU and power should be switched off only when the state goes into the **ReadyToPowerOff**.	Final

Refer to Figure 7 for a graphical view of each of the MCU states.

2.6.5.1 J1939 CAN Protocol Specification

2.6.5.1.1 Message timeout

After start-up, once either a **VmuFastCommand** or **VmuSlowCommand** CAN message has been sent, both will then need to be sent continuously within the time defined in the periodicity of each message. Refer to the MCU J1939 CAN Protocol Specifications [2] for more information on the timeout on each of the messages. If these messages are not sent, a timeout will occur, the MCU will fall into failure mode and the system will need to be restarted.

2.6.5.1.2 Initializing CAN communication

The MCU has pre-assigned addresses and uses them once the auxiliary power is applied to the system and the enable signal (IGNITION input) is asserted to the MCU.

However, the MCU integrates the J1939 identification feature (address claim); this feature consists of two options:

- Send an Address Claim message to claim an address
- Send a request for Address Claim

Once the claiming is done, the MCU starts to communicate as defined in the J1939 CAN protocol specification [2].

2.6.5.1.3 Shutdown sequence

To activate the MCU shutdown sequence from an operational state, the **VmuFastCommand.OperationRequest** CAN message can be sent by the VMU with the **Standby** attribute and then the **Shutdown** attribute, or directly with the **Shutdown** attribute.

1 If the **Standby** attribute is sent, the VMU will wait until it receives the **McuSlowCommandResponse.StateMcu** MCU CAN message with the **Standby** attribute confirming it has reached that state through the completion of the transitional **Deactivation** state. The VMU could

then send the **VmuFastCommand.OperationRequest** CAN message with the **Shutdown** attribute to the MCU. The MCU will then go through the **Shutdown** and the **WaitForSafeCondition** transitional states before reaching the **ReadyToPowerOff** state confirmed by the **McuSlowCommandResponse.StateMcu** MCU CAN message with the **ReadyToPowerOff** attribute.

2 If the **Shutdown** attribute is sent, the MCU would transit through the **Deactivation** state before reaching the **Standby** state and then it would transit through the **Shutdown** and **WaitForSafeCondition** states before reaching the **ReadyToPowerOff** state confirmed by the **McuSlowCommandResponse.StateMcU** CAN message with the **ReadyToPowerOff** attribute.

Note: In both cases the MCU will shut itself down only when it has reached a **ReadyToPowerOff** state and the MCU enable line (IGNITION input) is low.

2.6.5.1.4 Failure management

The system changes its state to **Failure** if incompatibility errors occur during initialization (see Section 2.7.3) or any other state shown in Figure 7; loss of CAN communication also results in **Failure**.
Once in **Failure**, the system cannot be operated but the MCU is still using power from the auxiliary battery. In order to prevent draining the battery, as soon as the system goes into **Failure**, one of the following operations must be performed:

1 Send a **Shutdown** request via **VmuFastCommand.OperationRequest** to initiate the Shutdown sequence.
2 Turn the IGNITION input to OFF; after 10 seconds the system will initiate the **Shutdown** sequence.

2.6.5.1.5 Performance availability

Information is returned on the current thermal status of the complete system via the **McuThermal1.ThermalIndicator** signal. For more information about performance availability, refer to Section 3.2.2.

2.6.5.1.6 Cooling

The cooling requirements of the motor and MCU are handled by two signals **McuThermal1**.**DriveCoolingFlowRequest** and **McuThermal1.MotorCoolingFlowRequest**. For more information on cooling management, refer to Section 3.2.1.

2.6.5.1.7 Software and protocol version

The software version can be requested by sending this message: **McuSoftwareVersion.**

The protocol version can be requested by sending this message: **McuProtocolVersion**.

Note: The extension pack value returned will be 0.0 since it is not used by the application.

2.6.5.1.8 McuInfo1

The **McuInfo1** contains notification of major event occurrences in the MCU which may trigger a safety action in the VMU.

2.6.5.1.8.1 Attributes used by TM4 SUMO™ HD

The **FaultPhaseToChassis** attribute is used by the application. Therefore, a fault value (bit at 1) indicates a problem with the chassis and the high-power voltage and an invalid value (bit at 2) indicates that the system cannot detect this event.

2.6.5.1.8.2 Attributes not used by TM4 SUMO™ HD

The **FaultInterlockBatteryCable**, **FaultInterlockMotorCable** and **FaultDriveCoverWasOpened** attributes are not used by the application. The value sent for these attributes will always be invalid (bit at 1). Therefore, the VMU should consider that the functionality is not used and not consider the associated fault attribute.

2.6.5.1.9 General CAN signal specifications

The tables in this section contain information about how to use some of the CAN signal attributes.

Table 19 McuFastCommand.OperationalMode descriptions

Values	Description
Neutral	Sets the system in a neutral mode where no torque/speed/voltage command is applied resulting in no torque generated to the motor output shaft.
Generator (Voltage regulation)	Sets the system in an operational state where the DC voltage command is enabled.
Purely Electric (EV)	Sets the system in an operational state where the torque/speed command is enabled.

Table 20 McuFastCommand..CommandMode value descriptions

Values	Description
Torque	The MCU is controlled by torque using the **Command Value** as a torque command.
Speed	The MCU is controlled by speed using the **Command Value** as a speed command.
Voltage	The MCU regulates a DC voltage using the **Command Value** as a voltage command.

Table 21 McuSlowCommandResponse.TractionDerating/ McuSlowCommandResponse.RegenDerating value descriptions

Values	Description
None	System can provide the maximum torque.
Battery current	The torque is limited by the available current of the traction battery.
Internal	Internal limitation of the current/power/other.
Overspeed	The torque is limited because the motor speed has reached the design working system speed limits.
Overheat system	Limitation caused by overheating of some components of the system.
Defect sensor drive	Limitation caused by defective sensors.
Maximum power	Limitation of the available torque caused by the maximum system performance power.
DC voltage	Limitation of the available torque caused by the level of the DC voltage outside the high-voltage battery voltage range.
Overheat drive	Limitation caused by overheating of some components of the MCU.
Overheat machine	Limitation caused by overheating of some components of the motor.
Defect sensor machine	Limitation caused by defective sensors.
Limp Home	Limitation of the available torque caused by the Graceful Degradation module.

Refer to Table 24 for details about derating causes.

2.6.6 CAN protocol vs. system limits

This section gives a summary of the CAN commands used by this system along with the range of values permitted.

For more information, refer to the MCU CAN communication protocol specifications [1].

Table 22 Summary of VMU to MCU communication protocol

Command	Field	Supported values	
		From	to
VmuCommand1 or **VmuSlowCommand**	MaxChargeCurrent	0 A_{DC}	**Maximum current**
	MaxDischargeCurrent	0 A_{DC}	**Maximum current**
	MaxBatteryVoltage	0 V_{DC}	upper value of **Operating voltage**
	MinBatteryVoltage	0 V_{DC}	upper value of **Operating voltage**
VmuCommand2 or **VmuFastCommand**	TorqueCommand	– **Maximum output torque**	+ **Maximum output torque**
	SpeedCommand	– end of **Derating range**	+ end of **Derating range**
VmuCommandSafety [2]	Reference TorqueCommand	– **Maximum output torque**	+ **Maximum output torque**
	Reference SpeedCommand	– end of **Derating range**	+ end of **Derating range**

Refer to the System Specification [7] for precise values of parameters entered in the Supported values column.

[2] Not applicable to J1939.

2.7 Troubleshooting

2.7.1 Fault management

Certain fault tolerance mechanisms are available allowing the product to continue to operate but at the same time alerting the user to the presence of the faults and slowly shutting down the system using graceful degradation strategies.
Note: Error codes indicating faults with the product or system are sent over CAN carried in the **McuOnEventInfo1** message.

Disclaimer: TM4 does not take responsibility for prolonged use with one or more defective sensors as this could cause permanent damage to the product and injury to the user. As soon as one of the sensors fails, the vehicle must be returned to the garage for inspection and repair as soon as possible.

2.7.1.1 Temperature sensor faults

WARNING

Overheating could cause irreparable damage to the system leading to severe injury to the user. If one of the temperature sensors fails, the vehicle will still be operational, but you must immediately return to the garage to inspect the vehicle and take the necessary corrective actions to reactivate the sensor. If all of the temperature sensors fail, it will no longer be possible to monitor system temperature; there will be no temperature derating and/or protection which will lead to overheating.

Temperature sensors are continually monitored with errors/warnings logged in the SysFile; you must monitor the SysFile. If one of the sensors ceases to function, there will be a warning logged in the SysFile identifying which sensor is defective and must be reactivated before continuing operation.

Note: This type of failure is often caused by excessive vibration causing the sensors to come loose and weaken the contact; to prevent this from happening; we recommend that you protect the connector pins on the interface harness connector head with a layer of conductive grease. See [10] for the correct procedure.

Table 23 Motor temperature sensor faults (sent over CAN in **McuOnEventInfo1**)

Error code	Description	Warning
0x4B00	Warning - CoilTempSensorMgr 0x00	**WARNING** The integrator must immediately inform the driver that there is a risk of overheating the motor at low speeds.
0x4B01	Warning - CoilTempSensorMgr 0x01	
0x4B02	Warning - CoilTempSensorMgr 0x02	
0x4B03	Warning - CoilTempSensorMgr 0x03	**WARNING** The integrator must immediately inform the driver that there is a high risk of overheating which may cause irreparable damage to the motor.

2.7.1.2 Current sensor faults

Each phase has its own current sensor that is monitored by the system. When one of the sensors in a group of three is defective, the system continues to be operational, but a warning is logged to the SysFile at start-up. If more than one sensor per phase is deactivated, the system should be inspected and repaired to prevent damage to the product or injury to the user.

2.7.1.3 Position sensor faults

During operation, the system continuously returns information related to rotor position; in case of an error with the position sensors, the system will continue to function, but will immediately begin to ramp down the available torque. This form of graceful degradation is in place to prevent damage to the system.

Once a position sensor related error is logged in the SysFile, it is the responsibility of the integrator to find and address the cause of the error before restarting and operating the system; refer to the TM4 Position Sensor Graceful Degradation Module Guide [4].

2.7.1.4 CAN communication faults

During operation, the system continuously communicates via CAN; if this communication is compromised, graceful degradation is triggered, and the system begins to slow down.

Once one or more of these errors is logged in the SysFile, it is the responsibility of the integrator to find and address the related cause of the error before restarting and operating the system; refer to the TM4 Communication Graceful Degradation Module Guide [4] for more information.

2.7.2 Pre-charge and power turn-on time

Sending an operational request before pre-charge is completed or before the power-on sequence is complete could lead to system failure. The TM4 MCU is not designed to operate while the DC bus is pre-charging since the DC current is limited by a pre-charge resistor during power-up.

Before sending an operational request, it is recommended that you verify that the high voltage seen by the MCU is in an acceptable range and that the main contacts of the high-voltage bus are closed.

The **HighPowerVoltage** field of the **McuInfo1** message can be used to verify the voltage level seen by the MCU at its high-power voltage input.

Note: A typical error occurs when the user neglects high-voltage bus contact turn-on time and sends an operational request before voltage is stabilized.

2.7.3 Initialization error

The initialization error can be confirmed by the **McuOnEventInfo1** CAN message or via the TM4 ODIN diagnostic tool in the SysFile view window.

If at start-up, a corruption of MCU internal permanent memory is detected, as a safety measure, the system stays in **Failure** state. This could also occur if incompatible firmware has been loaded in the MCU. When this happens, at start-up, the system will not be able to go into a **Standby** state and will stay in a **Failure** state but CAN communication will be allowed for diagnostic purposes (TM4 ODIN diagnostic tool).

If this problem occurs after an application software update, check that the right firmware has been loaded. If not, reload the correct firmware. If this does not resolve the problem, contact TM4 Customer Service for help; see Section 7 for contact information.

2.7.4 Boot-up error

At boot-up, if there is a problem with the firmware, the MCU will send one of the following CAN messages:

- CAN ID 0x0661 and Data 0x0060: Firmware not present
- CAN ID 0x0661 and Data 0x0050: Firmware corrupted

Note: If the user has modified the CAN addresses configuration represented by **OdinCan1BaseAddress** or **OdinCan2BaseAddress**, the CAN ID will be: OdinCanXBaseAddress + 1.

2.7.5 Software and protocol version

If a failure occurs just after a firmware update, software and protocol version could be verified by sending a request for the software and protocol versions.

Note: During initialization and before establishing standard communication, the VMU must request use of a specific protocol version (starting with the highest version it supports) and be able to recognize that the MCU is ready to communicate. For more information about CAN communication initialization, refer to Section 2.6.3.

Verify that the versions are compatible with the VMU. It is also possible to verify software and protocol versions using the TM4 ODIN diagnostic tool.

2.7.6 Message timeout

After start-up, once either a **VmuCommand1** or **VmuCommand2** CAN message has been sent, both will then need to be sent continuously along with the **VmuCommandSafety** message, within the time defined in the periodicity of each message. Refer to the MCU CAN Protocol Specifications [1] for more information on the timeout on each of the messages.

2.7.7 Failure mode

Once the system is in **Failure** mode, it is not possible to set it back to **Operational** mode without first shutting down the power and rebooting.

2.7.8 Customer registers (errors and warnings)

Customer registers can be consulted using the TM4 ODIN diagnostic tool. The **ErrorsAndWarnings** section, within the **Controller**, **MotorControl** and **PositionSensor** folders, contains information on possible sources of error.

The list of registers that can be consulted within the **Controller** section is as follows:

- **IsInSafemode**
- **CriticalInitError**
- **FirstErrorCode**
- **StorageMgr.Faults**
- **StorageMgr.GlobalFaults**
- **StorageMgr.ParamFaults**
- **StorageMgr.ProtectedFaults**
- **StorageMgr.SavInfoFaults**
- **StorageMgr.BlackboxFaults**
- **StorageMgr.SysfileFaults**

The list of registers that can be consulted within the **MotorControl** section is as follows:

- **FaultsInverter**
- **FaultsMachine**
- **FaultStatus**

The register that can be consulted within the **PositionSensor** section is as follows:

- **PositionSensor.Faults**

3 System behaviour

This section contains information related to how the system behaves during operation.

3.1 Derating

The system is designed to self-limit the applied torque in order to maintain system integrity. When the system cannot provide the maximum torque, it reports the cause of derating via the CAN message. See the MCU CAN Protocol Specifications [1] for more information.

Notes:

1. The derating cause is reported when the torque available in the system is lower than the maximum torque the system can provide when there is no derating.
2. The requested torque will be provided as long as the requested value is lower than or equal to the available torque the system can provide regardless of if the system is in derating or not.

Table 24 Derating limitations

Cause code	Brief description	Detail
None	System can provide the maximum torque.	There is no derating.
Battery current	The torque is limited by the battery current.	The specified discharge/recharge currents of the traction battery do not allow the system to provide the maximum torque. These limits are communicated to the system by the VMU via CAN messages (message **VmuCommand1**: **MaxChargeCurrent** / **MaxDischargeCurrent**).
Internal	Limits on internal components are reached.	Internal limits are set according to the performance battery voltage range. Outside that range, the power will be limited, thus limiting the torque. Here is how these limits impact performance: **Traction**: When the battery is within the performance battery voltage range, the maximum electrical output power is maintained over the range. When the battery voltage level is lower than the minimum level of the performance battery voltage range, speed derating will be applied according to the motor specifications. **Regeneration**: When the battery is within the performance battery voltage range, the maximum electrical output power is maintained over the range. Available torque is gradually reduced from maximum to 0 Nm in a zone from the maximum level of the performance battery voltage range and 2% over.

<table>
<tr><th>Cause code</th><th>Brief description</th><th>Detail</th></tr>
<tr><td>Overspeed</td><td>The torque is limited because the motor speed has reached the recommended system speed limits and must be reduced within a period of 2 minutes to prevent any damage.</td><td>Torque is reduced to prevent exceeding the maximum speed supported by the system.
The following table shows the zones how the torque is progressively reduced to 0 Nm:

<table>
<tr><th colspan="2">Traction mode</th></tr>
<tr><th>Direction</th><th>Speed zone</th></tr>
<tr><td>Forward</td><td>SpdFwdTracStartDerating
to SpdFwdTracEndDerating</td></tr>
<tr><td>Reverse</td><td>SpdBwdTracStartDerating
to SpdBwdTracEndDerating</td></tr>
<tr><th colspan="2">Regeneration mode</th></tr>
<tr><th>Direction</th><th>Speed zone</th></tr>
<tr><td>Forward</td><td>SpdFwdRegenStartDerating to
SpdFwdRegenEndDerating</td></tr>
<tr><td>Reverse</td><td>SpdBwdRegenStartDerating to
SpdBwdRegenEndDerating</td></tr>
</table>

Refer to Table 10, Figure 4 and Figure 5 for more information.</td></tr>
<tr><td>Overheat system</td><td>Limitation caused by overheating of some components of the system; heat must be reduced below the limit within a period of 2 minutes.
(CAN v4.0 only).</td><td>Some components of the system, such as motor coils and IGBT modules, have reached the design working limits. Torque is reduced to prevent the components from getting hotter.</td></tr>
<tr><td>Defect sensor drive</td><td>Limitation caused by defective sensors in the drive.
(CAN v4.1 only).</td><td>In case a drive sensor (or a group of sensors) is defective, and not critical for the system to function correctly, the torque is reduced to a level where the system can operate safely (not causing damage to the system component connected to the sensor(s)).
Type of sensors that can be defective:
- Temperature</td></tr>
<tr><td>Max power</td><td>Limitation of the available torque caused by the maximum system performance power.</td><td>The system has reached a speed where the allowed torque is reduced to not exceed the maximum power of the system.</td></tr>
</table>

Cause code	Brief description	Detail
DC voltage	Limitation of the available torque caused by the level of the DC voltage outside the high-voltage battery voltage range.	Limitations on the available torque are caused by the level of the DC voltage outside the high-voltage battery range. The received limits from the VMU impact performance as follows: **Traction:** **From VMU (MinBatteryVoltage)** Power is reduced gradually (max power to 0) in a zone of 20 V above the received **MinBatteryVoltage** (minimum battery voltage) CAN message from the VMU. **Regeneration**: **From VMU (MaxBatteryVoltage)** Power is gradually reduced from maximum power to 0 kW in a zone of 20 V below the received **MaxBatteryVoltage** (maximum battery voltage) CAN message from the VMU. Limitations on the available torque can also be caused by the level of DC voltage.
Overheat drive	Limitation caused by overheating of some components of the drive; heat must be reduced below the limit within a period of 2 minutes. (CAN v4.1 only)	Some components of the drive, such as IGBT modules or controller board, have reached the design working limits. Torque is reduced to prevent the components from getting hotter.
Overheat machine	Limitation caused by overheating of some components of the motor; heat must be reduced below the limit within a period of 2 minutes. (CAN v4.1 only)	Some components of the drive, such as motor coils, have reached the design working limits. Torque is reduced to prevent the components from getting hotter.
Defect sensor machine	Limitation caused by defective sensors in motor. (CAN v4.1 only).	In case a motor sensor (or a group of sensors) is defective, and not critical for the system to function correctly, the torque is reduced to a level where the system can operate safely (not causing damage to the system component connected to the sensor(s)). Type of sensors that can be defective: - Temperature - Current - Position

Cause code	Brief description	Detail
Limp Home	Limitation of the available torque caused by the Graceful Degradation module.	The Graceful Degradation module has reduced the allowed torque because of a prolonged fault in one of the system components. A SysFile event is raised to indicate which component is responsible. Refer to the appropriate Graceful Degradation module documentation for more information. Possible system components that can trigger the Limp Home Derating: - Communication module (CAN) - Position Sensor

3.2 Thermal management

TM4 products have sophisticated mechanisms in place to manage temperature changes in a working system in order to optimize performance availability vs. durability while allowing the driver to monitor the thermal status of the system with values returned by CAN Protocol signals.

There are two main aspects to thermal management – cooling and performance availability; in both cases, temperatures are monitored and managed by various signals in the **McuThermal1** parameter. A summary of the message and signals is given in Section 3.2.2; for complete information, see the MCU CAN Protocol Specifications [1].

3.2.1 Cooling

TM4 systems cooling management algorithms are based on complex thermal modeling that optimizes system performance while maintaining system integrity and durability.

The cooling requirements of the motor and MCU are handled by two signals **McuThermal1 [DriveCoolingFlowRequest]** and **McuThermal1 [MotorCoolingFlowRequest]**. Working with the established maximum coolant flow for each component, these signals return information needed to maintain the necessary coolant flow and require no driver input or interaction.

The VMU relies on the cooling request messages to start/stop or modulate the cooling system pump(s) and fan(s); modulation depends on operating conditions as described in Table 25.

Table 25 System coolant flow management rate vs operating conditions

Operating conditions	Coolant flow rate management
Torque > 0 Nm	50%
Speed > 500 RPM	50%
Thermal protection	Up to 100%

Note: For operating conditions when torque is 0 Nm and speed is <500 RPM, coolant flow rate management can be 0%.

3.2.2 Performance availability

Information is returned on the current thermal status of the complete system via the **McuThermal1 [ThermalIndicator]** signal which is designed to keep the vehicle driver informed about the level of performance that can still be obtained from the system or to warn the driver that available power may be limited. This information will help the driver adjust his driving to match the current capabilities of the traction system:

- **0%-40%**: Informs the driver that peak performance is available and that he can accelerate without having to worry that the system could derate before the specified peak duration.
- **40%-60%**: Indicates that instantaneous performance is available but that its duration might be less than the specified peak duration.
- **60%-100%**: Indicates that the available instantaneous power is already limited and that acceleration performance will be less than specified as at this point derating is triggered and maximum torque is no longer available until the temperature lowers enough to return to a value below 60%.

3.3 Damping

The damping feature is an algorithm proprietary to TM4 that slightly changes the torque command sent to the motor to reduce the motor movements on the rubber mounts. This algorithm will not eliminate the initial jerk when the motor compresses the rubber mounts but should prevent the rebounds which result in the oscillations being felt by the driver.

For more information on how to configure the damping features, see [9] Customer Procedure SC-6000E-025.

3.3.1 Accessing the damping parameters

To access the damping parameters, you will need to follow these steps:

1. Connect the PC to the CAN interface. Ensure that the CAN interface is connected to the MCU CAN port.
2. Turn the MCU ON by applying VAUX level to its IGNITION input. If the motorisation system is fully integrated in the vehicle, turn the vehicle ignition key to the ON position.
3. Start the ODIN 4 Server 1 application which is normally configured to communicate with the MCU by default. If the device is still not connected (**Device Disconnected** yellow icon at the bottom left of the ODIN window), adjust the **Device Communication** configuration under the **Configuration ODIN** menu accordingly. Once properly configured, the **Device Connected** green icon should appear at the bottom left of the ODIN window.
4. Ensure that the software package delivered to customer includes the damping parameters and is being used by ODIN. The package name is displayed at the bottom center of the ODIN window; if it is not the appropriate package, you will need to flash the appropriate software package on to the MCU.
5. Open the UserInterface.odn4 file located within the **Package Files** menu under **Workspaces** sub menu.
6. Select the **Parameters** tab watch and open the Motor folder that includes all the damping parameters. The **DrvParameters.Save** parameter, that is used to save all parameters after one or more parameter configuration change, is located at the top of the **Parameters** watch.

Table 26 Damping parameter descriptions

Parameter	Description
DrvParameters.Save	Active (1) to save the parameters. System needs to be rebooted for the parameters to take effect.
DampingEnable	Enable (1) or disable (0) the damping algorithm.
DampingMotorMax Torque	Maximum torque the algorithm can add or substract from the torque command in order to perform its task.
DampingCarMinSpeedFor Damper	Motor speed in RPM below which the damping torque is limited to 0 Nm.
DampingSpeed1	Motor speed reference point in RPM. Between **DampingCarMinSpeedForDamper** and **DampingSpeed1**, **DampingFactor1** is used. Between **DampingSpeed1** and **DampingSpeed2**, the factor used is changed linearly from **DampingFactor1** to **DampingFactor2**.
DampingSpeed2	Motor speed reference point in RPM. Above this speed, **DampingFactor2** is used.
DampingFactor1	Damping algorithm reaction factor at low speed. When this factor increases, the amplitude of the torque reaction to oscillations also increases.

Parameter	Description
DampingFactor2	Damping algorithm reaction factor at high speed. When this factor increases, the amplitude of the torque reaction to oscillations also increases.

3.4 Active discharge of the high-voltage DC bus

The main goal of this function is to remove any hazardous electric potential from the DC bus to ensure that the product can be safely handled after power off.

If enabled, this software function uses the motor phases to discharge the DC bus during the shutdown sequence. This controls the electrical angle of the current to prevent the production of torque on the motor, thereby preventing any unexpected movement.

If enabled, the Active discharge function will only be successful under the following circumstances:

- The vehicle must be at a complete standstill;
- The power to the MCU must be connected;
- The batteries must be disconnected;
- A shutdown request must be sent via the VMU or the “IGNITION” signal.

Table 27 Active discharge parameters

Parameter	Description
ActiveDischarge.SafeVoltage	Sets the value of the voltage at which the DC BUS is considered safe; between 10v and 60v.
ActiveDischarge.Enable	Active discharge function is enabled by default (1) and can be disabled (0).
ActiveDischarge.Current	Sets the set-point for the discharge current.

4 Usage constraints

This system is designed to self-limit the speed and the applied torque in order to maintain system integrity. However, some special usage conditions such as towing, driving down a steep hill or in a test environment could submit the system to conditions outside of its control which could cause permanent damage.

WARNING

Operation of the system outside the specified limitations could permanently damage the system.

The user is required to restrict the usage of the system within the limits of the specifications defined in this guide.

Powering off the system while it is operational may permanently damage the system.

This system may be permanently damaged if it is unpowered while the speed exceeds a value where the motor back-EMF exceeds the maximum operating voltage of the internal power module of the MCU.

4.1 Maximum operating speed

The system speed should be limited to the maximum operating speed. Refer to the system parameters selected by the integrator as well as the Product Specifications [7] for the system maximum speed specifications.

4.2 Towing

In case of any breakdown or system failure, there are two methods of towing the vehicle back to a service center:

- Front wheels raised and rear wheels on the ground.
- Rear wheels raised and front wheels on the ground.

In the first method, when the front wheels are raised and the rear wheels are on the ground, you must limit the motor speed to 2000 RPM throughout the transportation of the vehicle. Calculate the motor speed in km/hr using a combination of the ratio of the differential and the circumference of the vehicle wheels.

There are no speed restrictions when the rear wheels are raised, and the front wheels are on the ground.

4.3 Maximum battery boosting voltage

The system must not be boosted with a battery voltage higher than 30 V for a VAUX of 12 V/ 24 V. Using a voltage higher than the one specified may cause irreparable damage.

4.4 Gravity acceleration

In the event where the vehicle is engaged in a prolonged downhill slope steep to the point that speed increases without any torque being requested, the driver and/or VMU should acknowledge the overspeed warning by using whatever means available to reduce speed in order to prevent damaging the system.

WARNING

Be aware of the overspeed condition.

Operating the system in overspeed can cause damage to the system creating hazardous conditions that could endanger the lives of the occupants of the vehicle.

4.5 Overheating

There is a risk of the engine overheating in the case of a temperature sensor fault. Error codes to this effect are displayed in the SysFile and transmitted over CAN. They must be monitored via CAN so that appropriate action can be taken.

If overlooked, it is possible that the system will overheat causing irreparable damage leading to severe injury to the user.

See Table 23 for the error codes related to temperature sensors to look for in the SysFile and Section 2.7.1.1 for more information on how to correctly handle this issue.

WARNING **Be aware of overheating the system.**

Operating the system with one or more defective sensor can cause damage to the system creating hazardous conditions that could endanger the lives of the occupants of the vehicle.

4.6 Short circuit mode warning

The Short circuit mode exists to provide protection to the system against the damage caused by excessive levels of back-EMF.

If the MCU goes into error and engine speed is greater than the base speed, or if the back-EMF produces a voltage higher than the maximum allowed by the DC-link, the short circuit mode is triggered to short the motor phases and prevent the back-EMF voltage from reaching the DC-Link. There will be a warning via an error code displayed in the SysFile and transmitted over CAN. This warning must be monitored via CAN so that appropriate action can be taken within a very short period of time (typically less than 2 minutes).

When the Short circuit mode is triggered, if the electrical motor is still being driven (due to downhill movement or being towed) the electrical motor will rapidly heat up due to the short circuit currents circulating in the motor. The user must avoid driving the electrical motor when the system is in active Short circuit mode.

If overlooked, it is possible that the system will overheat causing irreparable damage to the motor which may lead to severe injury to the user.

Refer to the Application Note on High-Speed Fault Management [11] to better understand the issue and to put in place the necessary protection mechanisms.

Important note: Failure to have the necessary protections in place in relation to the Short circuit mode will be considered misuse and void the product guarantee in place.

5 System diagnosis

The TM4 ODIN diagnostic interface is used to verify and program the embedded application, configure and retrieve the **BlackBox** information as well as providing access to selected embedded application variables.

For specific operation guidelines, refer to the TM4 ODIN v4 Technical Guide [3].

5.1 Set-up

Connect your TM4 ODIN-equipped PC to the MCU CAN port using a CAN case adapter from Vector as specified in the TM4 ODIN v4 Technical Guide [3].

5.2 Watch and graphics

Using TM4 ODIN tools, you can visualize the variable values of the software application in asynchronous watch lists or in a synchronous graphic form. The variable sample frequency could be 10 Hz, 1 kHz and 16 kHz, see the Product Specifications [7] for the exact frequency.

The following list explains the variable arborescence:

- **Calibration**: Contains command/data related to the motor phase and temperature calibration.
- **Drive**: Contains data related to the MCU.
  - CanCommunication: Contains information about the actual CAN bus speed.
  - CustomerSupportInfo: Contains information monitored throughout the life of the MCU.
  - ModelNumber: Contains the model number of the MCU.
  - PartNumber: Contains the part number of the MCU.
  - Sensors: Contains the value of different sensors the MCU monitors.
  - SerialNumber: Contains the serial number of the MCU.
- **ErrorsAndWarnings**: Contains potential errors and warnings. When the system runs correctly, all variables in ErrorsAndWarnings are set to 0. Otherwise, some values will be set.
  - Controller: Contains potential errors and warnings of the MCU.
  - MacEeprom: Contains potential errors and warnings of the internal permanent memory of the motor.
  - MotorControl: Contains potential errors and warnings of the motor.
  - PositionSensor: Contains potential errors and warnings of the position sensor.
- **Motor**: Contains data related to the motor.
  - ModelNumber: Contains the model number of the motor.
  - PartNumber: Contains the part number of the motor.
  - Sensors: Contains the value of different sensors the motor monitors.
  - SerialNumber: Contains the serial number of the motor.
  - …: Could have other information depending on the motor type.
- **Parameters**: Contains customer configurable parameters for the system. Some can be changed; others are read-only.
  - Drive: Contains parameter values contained in the permanent memory of the MCU.
  - Motor: Contains parameter values contained in the permanent memory of the motor.
- **System**: Contains data related to the system.
  - CanMsg: Contains all the CAN messages related to the MCU.
  - ManualControl: Contains an **EnableManualControl** parameter to emulate the VMU commands manually.
  - Status: Contains system status information.

- **Versions**: Contains the version of:
  - Application: Embedded software application version.
  - Bootloader: Bootloader application version.
  - Protocol: CAN protocol and extension pack protocol versions.

Available variables can be viewed in the TM4 ODIN **Items Explorer** window. A detailed description of each variable is given by the application.

The UserInterface.odn4 file can be loaded from TM4 ODIN to quickly view the most commonly used variables.

5.2.1 GetCSInfo script

Important system information can be collected and exported in .zip format to send to TM4 for troubleshooting purposes. Access this script using the **File/Tools** menu in TM4 ODIN; refer to the TM4 ODIN v4 Technical Guide [3] for more information.

5.2.2 Customer registers (errors and warnings)

Customer registers can be consulted using the TM4 ODIN diagnostic tool. The **ErrorsAndWarnings** section, within the **Controller**, **MotorControl, MacEeprom** and **PositionSensor** folders, contains information on possible sources of error.

5.2.3 Error codes and corrective actions

Each error recorded by the embedded MCU software can be viewed in the SysFile accessible via the TM4 ODIN GUI (refer to TM4 ODIN v4 Technical Guide [3] for more information). The SysFile records the most relevant events with enough detail to allow the system integrator to define if the cause of the event is external or internal to the traction system; the file works as a circular buffer that overwrites the oldest entries with the newest ones.

The TM4 Error Codes and Corrective Action [12] document identifies each error code that might be displayed in the SysFile; gives a short description of the cause of the problem and then, in some cases, lists the numbers of the actions to be performed to resolve the issue.

If the issue remains unresolved, contact TM4 Customer Service; see Section 7 for contact information.

5.3 Configuring the BlackBox

The **BlackBox** is used to diagnose unexpected behaviour with the MCU, it records a list of variables based on the occurrence of predefined trigger events. The MCU comes with a default **BlackBox** configuration designed to ease the first step analysis of most issues that could be encountered with the application software.

The **BlackBox** configuration can be performed in any of the operating modes of the system. TM4 ODIN is used to configure the data that needs to be captured. To launch the capture, the user needs to set up a trigger. The new **BlackBox** configuration is then automatically saved in non-volatile memory.

When the **BlackBox** is started, the trigger condition is set to **FALSE** and then when the condition becomes **TRUE**, the **BlackBox** captures data for the configured period. The capture is then stopped, and the data is saved in internal permanent memory. The time is determined by the space of the **BlackBox** memory and the amount of data that needs to be captured.

Note: The usual acquisition time is about 10 ms.

At start-up, each time the MCU becomes operational, the **BlackBox** is automatically cleared of all data but still allows the system to recover data on the last error. This information, which you can view when you connect with the TM4 ODIN GUI, is stored in internal permanent memory in the **BlackBox**. The error data includes the associated boot-up number generated by the VMU at start-up.

When TM4 ODIN connects to the embedded application, the text “BlackBox info available” is displayed in green in the status bar at the bottom of the window, telling you that there is information to be recovered from the **BlackBox**. You can then view the graph captured to diagnose the problem. Using the TM4 ODIN GUI, you can clear the acquired data and start a new capture session at any time.

For further information, refer to the TM4 ODIN v4 Technical Guide [3].

5.4 Application version

To check which version is currently flashed on the embedded system, you can connect to the embedded application with TM4 ODIN. The name of the embedded application and the version of the current application will be displayed in the status bar.

For further information, refer to the TM4 ODIN v4 Technical Guide [3].

5.5 Package documentation

From v4.16.x of TM4 ODIN links to all customer documentation related to a currently loaded package are available via the ODIN GUI in the Package Files menu item.

For further information, refer to the TM4 ODIN v4 Technical Guide [3].

6 Maintenance and inspection

Read these general safety warnings before maintaining and inspecting this TM4 product.

WARNING

Mishandling of this motor control unit (MCU) may damage the product and/or cause injury or death.

- Do not attempt to open or repair this product. In case of damaged casing or suspected product malfunction, contact TM4 Customer Service; see Section 7 for contact information.
- Use only recommended points to lift and secure the system.

When manipulating this product, you must **NOT**:

- Modify any part of the MCU.
- Apply any external load to the casing of the MCU.

This product can reach very high temperatures that can cause serious burns and/or other injuries.

Avoid any contact with surfaces during and after use.

This MCU generates high voltage that can cause an electric discharge or electrocution resulting in injury or death.

When installing/uninstalling the product, verify that:

- The traction battery (high-voltage battery) is disconnected.
- The auxiliary battery (12 V/24 V battery) is disconnected.

Care must be taken when manipulating electrical equipment.

This product must be installed/uninstalled by qualified and authorized personnel in accordance with applicable vehicle standards and industry practices. Always use appropriate insulation and protection before manipulating the product even when the product is disconnected from a high-voltage source.

This product uses differential mode capacitors between the positive high-voltage DC bus (+) and the negative high-voltage DC bus (-) and common mode capacitors between the high-voltage DC bus and the chassis. Even when the product is disconnected from the high-voltage source, these capacitors can hold a voltage high enough to cause an electric discharge or death.

It is important to ensure that no voltage is present on the high-voltage battery wires between both polarities and from each polarity to chassis before manipulation.

Before opening the access panel, wait 10 minutes to ensure that internal TM4 MCU capacitors are discharged. After that delay, the access panel can be opened and the TM4 MCU high-voltage battery inputs can be measured between both polarities and from each polarity to chassis to ensure that no voltage is present.

6.1 Maintenance schedule

The product has been designed to be used with a limited maintenance and inspection schedule, which involves regular visual and mechanical inspections including verifying for:

- Overall system damage.
- Correct harness and cable connection to the MCU and the motor.
- Coolant leaks.
- Unusual component wear.

For a more detailed maintenance schedule, see Table 28.

IMPORTANT NOTE: To prevent damage to the aluminum casing when cleaning the products, avoid solutions containing sodium hydroxide or those with a high level of alkalinity such as cold agent cleaners.

Table 28 Maintenance schedule

Component	Type of verification	Description	Minimal frequency
HV cables (phase and HV battery – MCU) (phase – motor)	Visual	Verify integrity of cables - look for signs of wear and tear.	Once a year or during regular scheduled vehicle maintenance
HV and phase cables glands MCU and motor)	Visual	Validate that the cable glands are still sufficiently tightened.	During regular scheduled vehicle maintenance or a minimally once a year
Phase cables motor	Visual	Verify integrity of cables - look for signs of wear and tear.	Once a year or during regular scheduled vehicle maintenance
Motor sensor cable and connectors	Visual	Verify integrity of cables - look for signs of wear and tear. Verify that connectors are properly engaged.	Once a year or during regular scheduled vehicle maintenance
VMU interface connector	Visual	Verify integrity of cables - look for signs of wear and tear. Verify that connectors are properly engaged.	Once a year or during regular scheduled vehicle maintenance
Coolant in/out MCU	Visual	Verify tubes for coolant leaks at entry and exit points on the MCU. See Section 6.2.	Once a year or during regular scheduled vehicle maintenance
Coolant in/out motor	Visual	Verify tubes for coolant leaks at entry and exit points on the motor. See Section 6.2.	Once a year or during regular scheduled vehicle maintenance
Ground location MCU		Review section on grounding point in the MCU installation guide [5].	Once a year or during regular scheduled vehicle maintenance
Ground location motor		Review section on grounding point in the motor installation guide [6].	Once a year or during regular scheduled vehicle maintenance
Ground location chassis		Review section on grounding point in the MCU installation guide [5] and motor installation guide [6].	Once a year or during regular scheduled vehicle maintenance
Motor	Visual	Verify for overall component damage.	Once a year or during regular scheduled vehicle maintenance
MCU	Visual	Verify for overall component damage.	Once a year or during regular scheduled vehicle maintenance
Vent MCU	Visual	Verify the level of dust accumulation and remove any obstruction. Clean the vent using the appropriate brush.	Once a year or during regular scheduled vehicle maintenance
Mounting point Motor	Visual	Validate the integrity of the rubber mounts and verify that all the bolts are properly tightened.	Once a year or during regular scheduled vehicle maintenance

Component	Type of verification	Description	Minimal frequency
Mounting point MCU	Visual	Validate the integrity of the rubber mounts and verify that all the bolts are properly tightened.	During regular scheduled vehicle maintenance or a minimally once a year
Coolant liquid	Visual	Check coolant level (add more if required).	During regular scheduled vehicle maintenance
Coolant liquid	N/A	Change the system coolant (see Section 6.2.2 for more detailed information on coolant specifications)	Once every two years or every 50000 km
Motor bearing	N/A	No action.	N/A
Software upgrade	Software	If requested, verify and change product software to the latest available version.	As requested; once a year or during regular scheduled vehicle maintenance
Cleaning	N/A	Carefully clean the surface of the equipment using an air pressure gun.	Once a year or during regular scheduled vehicle maintenance

6.2 Coolant

6.2.1 Coolant – Inspection

Read all safety warnings associated with handling the cooling agent before working with the cooling system.

WARNING

The cooling agent contains ethylene glycol that is a highly flammable product. Ethylene glycol can burn with an invisible flame that can cause serious burns and/or other injuries.

Always handle the cooling agent carefully wearing appropriate safety clothing and eye-glasses.

The cooling agent can irritate the skin, the eyes and the mucous membranes.

- Always work in a well-ventilated area when handling the cooling agent; breathing in high concentrations of ethylene glycol can cause nausea.
- In case of contact with eyes and skin, rinse with water and consult a doctor.
- In case of ingestion, seek medical help immediately.

The cooling agent is under pressure when heated; removing the cap when the coolant is hot can cause serious burns and/or other injuries.

Wait until the coolant reaches an ambient temperature before removing the cap.

All potential dangers of handling cooling agents cannot be listed here.

Consult manufacturer warnings and recommendations for safe handling of the cooling agent.

Coolant inspection should be carried out according to the recommended schedule as shown in Table 28.

Inspect around coolant inlets/outlets of the motor and the MCU carefully to ensure that no leak is present. Also verify the emergency coolant overflow holes on the base of the MCU. Stop operation of the system and contact TM4 Customer Service if any leaks are detected during inspection; see Section 7 for contact information.

Refer to Figure 8 for the location of the coolant inlet and outlet on the motor and Figure 9 for information on the MCU.

Note: The coolant inlet and outlets on both the motor and the MCU are labelled on the casing.

Figure 8 Inspecting for coolant leaks – Motor

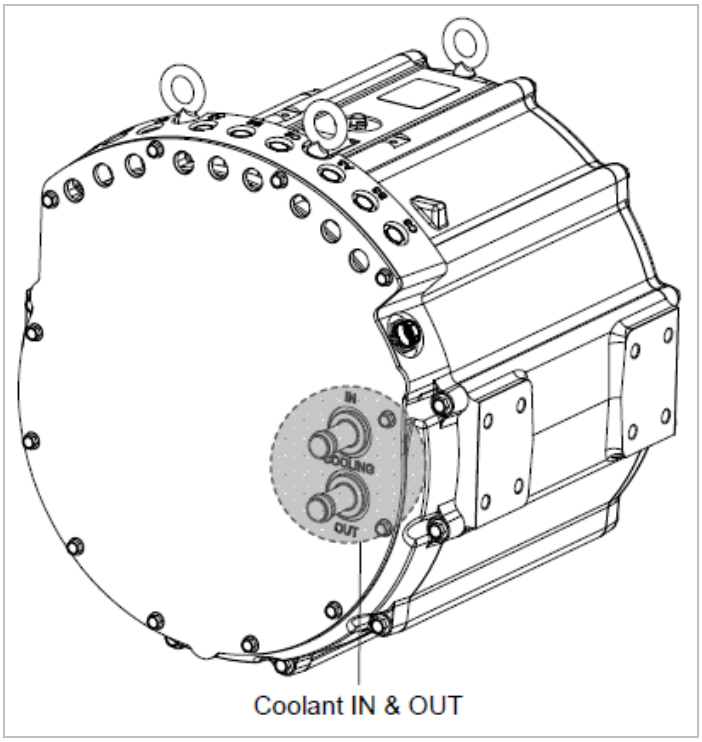
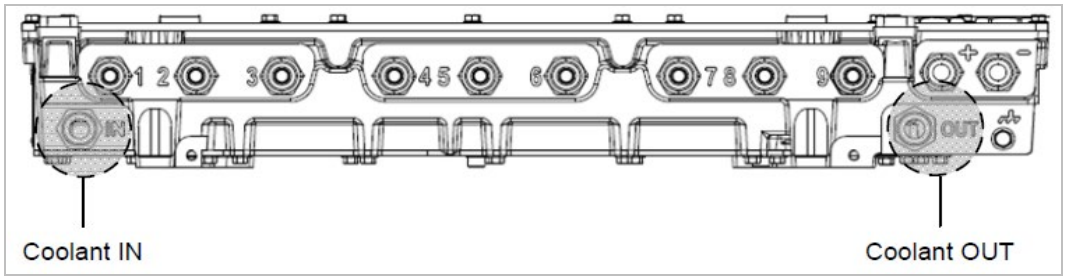
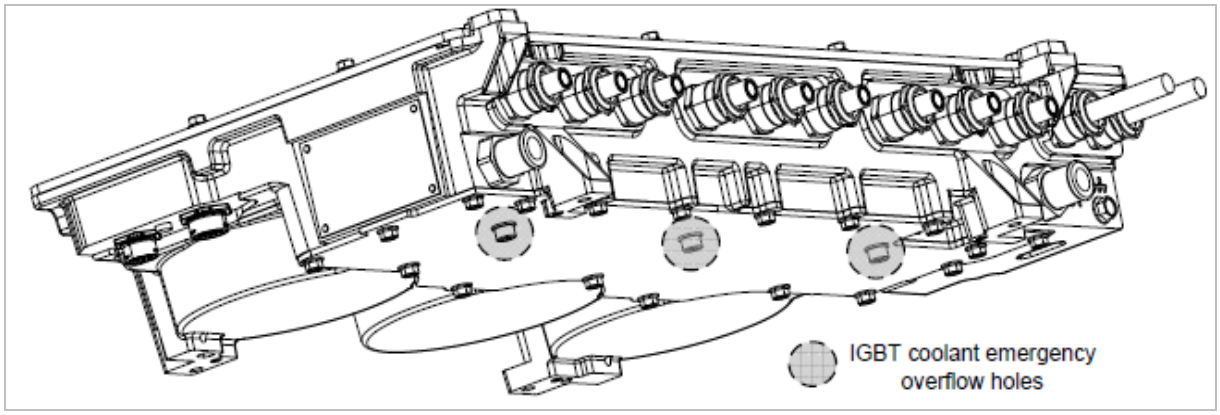


Figure 9 Inspecting for coolant leaks – MCU





6.2.2 Coolant – Specifications

To prevent corrosion problems in cooling systems, here are the recommended best practices.

- The cooling circuit must be rinsed with de-ionized water each time before filling.
- The ethylene glycol must respect the standard corresponding to its application (ASTM D3306 Type III or ASTM D6210 Type III).
- The ethylene glycol must contain some type of active corrosion inhibitors.
- The ethylene glycol must be diluted with de-ionized water (not distilled water).
- The dilution ratio must be 60% ethylene glycol to 40% de-ionized water (minimally 50% ethylene glycol to 50% de-ionized water).
- Two different types of coolant should not be mixed.

7 Customer service

For further technical assistance, please contact TM4 Customer Service:

Email: tm4customersupport@dana.com

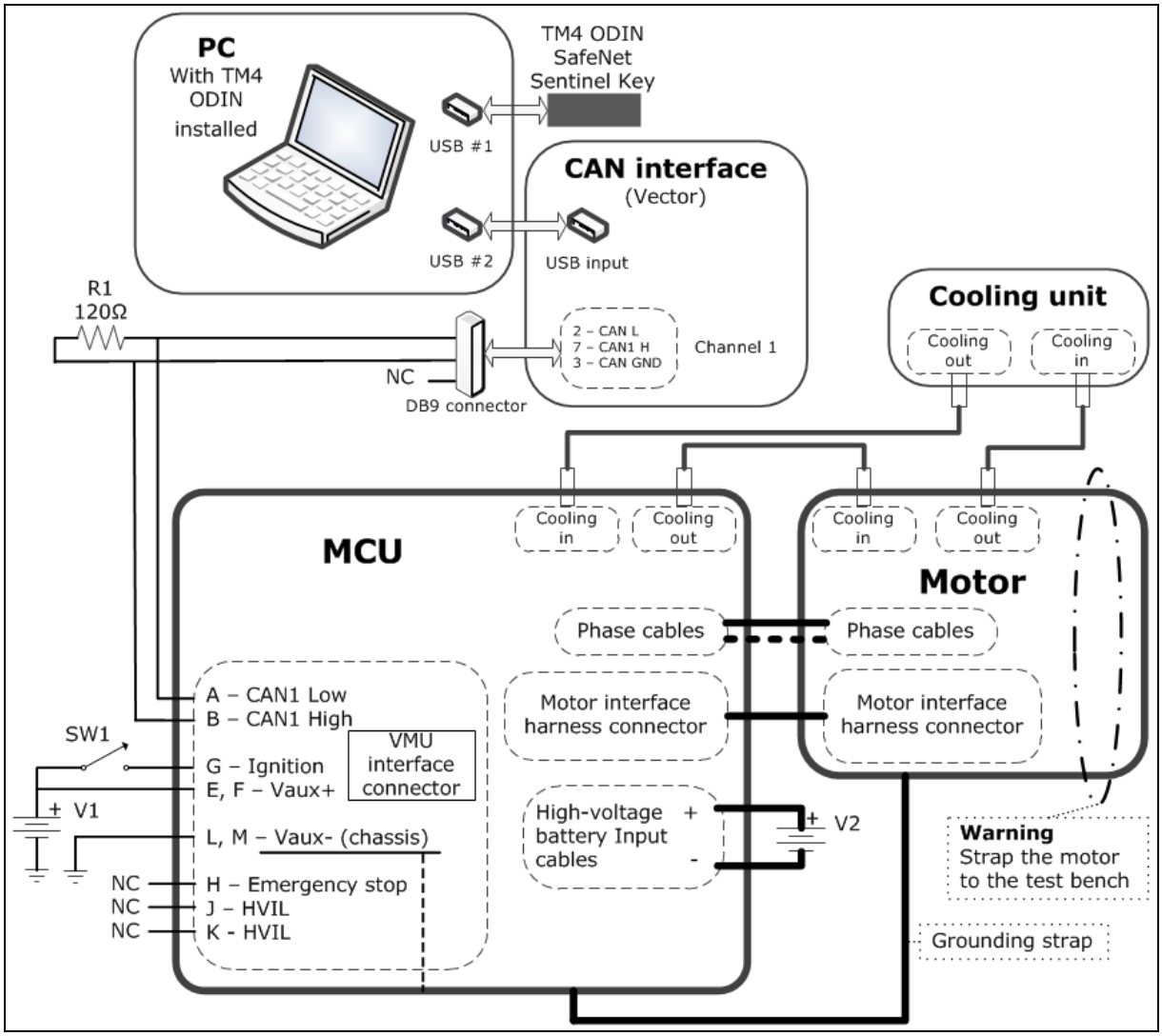
Appendix A Running the system in a test environment

This procedure describes the method to run the MCU and motor in a test environment. It includes details of all required equipment, set-up connections and software operation. This procedure is used to verify basic system functionality such as free wheel or blocked wheel motor operation.

A.1 Setting up the test environment

Refer to Figure 10 for the test environment.

Figure 10 Set–up required to communicate with the TM4 MCU



Notes:

1. The Vaux- input of the MCU (pins L&M) is referenced to the same point as the chassis (casing) of the MCU.
2. The ground V2 source is dedicated to the high-voltage battery minus input of the MCU and is isolated from the chassis (casing) of the MCU.

The cable length of both V2 source polarities should be at least 2.75 meters. This is required to eliminate the possible ringing effect that can happen if wire length is too short.

3. A grounding strap wire is required. One end of the grounding strap wire should be attached to the grounding point location of the MCU and the other end to the grounding point location of the motor. Look for the ground symbol on the motor and MCU, or refer to the installation guides for each for possible grounding point locations and wire size which should be calculated according to the maximum current required from the high-voltage battery source (V2).
4. For safety purposes, the motor should be fixed securely on a test base with the appropriate brackets attached to all motor mounting points in order to prevent the motor from moving while in operation.
5. Connect all the phase cables between the MCU and the motor respecting phase numbering shown in the motor Installation Guide [5].
6. Connect the motor interface cable between the MCU and the motor.
7. Connect the CAN interface to the computer and the set-up using DB9 connectors as shown in Figure 10.
8. The serial cooling system shown in Figure 10 can be used in a test environment.

A.2 Required equipment

Refer to Table 29 for the equipment required for the test set-up shown in Figure 10.

Table 29 Required equipment

Item	Reference (See Figure 10 Note 1)	Notes
PC with TM4 ODIN installed	N/A	2
TM4 ODIN SafeNet Sentinel Key	N/A	3
CAN Bus interface from Vector	N/A	4
DB9 connector	N/A	5
Kit-0076 connector kit	N/A	6
120 Ω resistor	R1	7
Switch	SW1	8
Low-voltage DC power supply	V1	9
High-voltage DC power supply	V2	10
Motor interface cable (between MCU and motor)	N/A	-
MCU	N/A	-
Motor	N/A	-
Cooling unit with appropriate cooling hoses	N/A	11

Notes:

1. References listed here correspond to items in the test set-up schematic shown in Figure 10.
2. Refer to TM4 ODIN v4 Technical Guide [3] for more information on TM4 ODIN.
3. This hardware key is provided by TM4 in order for you to run TM4 ODIN.
4. One CAN interface from Vector. Refer to TM4 ODIN v4 Technical Guide [3] for more information on compatible CAN interfaces to be used for CAN communication.
5. DB9 connector to establish CAN communication between the Vector CAN Interface and the MCU.
6. One connector kit (Kit-0076 assembly instructions [8]) as provided by TM4 with the TM4 MCU product for the VMU interface connector.
7. CAN impedance resistor to be installed between CAN high and CAN low lines to optimize CAN communication between the CAN case and the MCU.
8. One switch (SW1) to turn ON and OFF the system.
9. One DC power supply source (V1) to simulate the auxiliary battery of the vehicle (12 V/6 A or 24 V/3 A) with a current limit protection in case of a short circuit.

10. One DC power supply source (V2) to simulate the high-voltage battery of the vehicle (see note A) with a current limit protection in case of a short circuit. Unlike batteries, most DC sources do not accept recharge currents. In order to prevent current from going back into the source, regeneration parameters of the MCU will need to be set at 0 if the source cannot deal with regenerative current, otherwise a current will go back into the source. It is recommended that you use a DC source instead of a high-voltage battery in a test environment since a pre-charge mechanism, normally taken care of by the vehicle BMS, would need to be installed. A fuse would also be required for protection in case of a short circuit.
    Note A: If full system performance is required, the DC source voltage should be able to supply the voltage and power in line with values given in the Product Specifications document [7].
11. A cooling unit is necessary, but the cooling requirements depend on the type of testing to be performed. Refer to the Product Specifications document [7] for more details.

A.3.1. Getting started

1 Start the ODIN 4 Server 1 application that is, by default, configured to communicate with the TM4 MCU product.
2 From the TM4 ODIN menu, select **Configuration/Device communication** and select the Manual Configuration tab. Make sure that the selected configuration is set to **TM4 Mo4**.
3 Turn the V1 source (12 or 24 V_{DC} power supply) to ON.
4 Turn SW1 to ON.
5 Adjust the V2 source to the system operation voltage as specified in the Product Specifications document [7] and turn the source to ON.
6 Turn the cooling unit to ON.
7 Make sure that the bottom left of the TM4 ODIN window shows “Device connected” (green icon). Make sure that the device software is the latest one published on the TM4 Extranet (https://extranet.tm4.com/). If not, use the TM4 ODIN menu **Configuration/Flash Program Manager** to update the device software to the latest version available.

A.3.2. Setting operational limitations

1 Make sure that both sources (V1 and V2) are ON and that SW1 is also ON.
2 Start the ODIN 4 Server 1 application.
3 Open the **Package Files \ Workspaces \ UserInterface.odn4** file.
4 From the **Parameters** tab, expand the **Drive** folder.
5 Set all motor parameters according to the limitations of the test equipment. Refer to Section 2.4.3 for more information.
    It is recommended that you fix all values to their lower limits to start with in order to validate test set-up functionality before going to higher limits.
6 Save the new system limits by changing the **DrvParameters.Save** item to 1 and wait for it to come back to 0 to save the parameters in non-volatile memory.
7 Turn OFF SW1 and wait 10 seconds.
8 Turn ON SW1.
9 In the **Parameters** tab, verify that the configured limits are equal to the values previously entered.

Note: Depending on the type of vehicle integration, these parameters will need to be modified after the bench test is completed.

A.3.3. Operating the system

WARNING **Extra care should be taken when running the system in a test environment.**

The system will react immediately to the parameters entered. Ensure that you take all necessary safety measures and steps to avoid any unexpected and undesired situations.

1 Ensure that the UserInterface.odn4 TM4 ODIN file is opened.
2 Select the **Manual Control** watch.
3 Set the **EnableManualControl** value to 1.
4 Within the **Battery.Spec** folder, set the **iBatDischargeMax**, **vBatMax** according to V2 source limitations. Since most DC sources do not allow current to go back into them, set the **iBatRechargeMax** either to 1 (minimum working value) or set it to the maximum recharge current the source can handle. Set the **vBatMin** to the value at which you want to have the system in shutdown mode.
5 Set the **OperationalRequest** to **RESET** and then to **START**.
6 Set the **CommandMode** value to **TORQUE_MODE** to test the system using torque commands.
7 Set the **OperationalMode** value to **EV**.
8 At this point, the system is ready to accept **TorqueCommand** values. Positive **TorqueCommand** values will make the motor turn in the positive direction and negative values in a negative direction; note that while the motor is turning in one direction, a reverse direction torque command creates regenerative braking. If the source does not allow regeneration current and reverse motor rotation is desired, first, set the **TorqueCommand** to 0, wait until the motor has stopped turning and then apply a reverse polarity value to the **TorqueCommand** in order to make the motor rotate in a different direction.

WARNING **The TorqueCommand parameter should be set to a low value as high values could cause the motor to move suddenly and violently compromising user safety.**

Initial **TorqueCommand** values should be just enough to engage motor rotation; on a test bench set-up with no load on the shaft, a low torque value is sufficient. If the value is too high, the impact on the movement of the motor on the test table could be violent enough to cause the motor to move suddenly and compromise user safety.
Also to further enhance test bench security during operation on the test table, the initial values of the **Motor.PositiveTorqueMax** and the **Motor.NegativeTorqueMax** parameters in the **Parameters** tab should also be set at low values. See Section A.3.2.

A.3.4. Turning OFF the system after testing

WARNING **Even when the product is disconnected from the high-voltage source, the capacitors can hold a voltage high enough to cause an electric discharge or death.**

Before dismantling the set-up, wait at least 10 minutes to ensure that all inner MCU capacitors are completely discharged to avoid any electrical exposure while manipulating the set-up.

1 Ensure that the UserInterface.odn4 TM4 ODIN file is opened.
2 Select the **Manual Control** tab.
3 Set the **TorqueCommand** value to 0 and wait until the motor stops turning.
4 Set the **OperationalMode** to **NEUTRAL**.
5 Set the **OperationalRequest** to **RESET** and then to **SHUTDOWN**.
6 Verify, in the **System Status** tab that the **SequencerStatus** value goes to **READY_TO_POWER_OFF**.
7 Turn OFF SW1.
8 Turn OFF V2 source.
9 Turn OFF V1 source.