UM11145

NTM88 Firmware Library User Guide

Rev. 7.0 — 9 October 2024 **User guide**

Document information

Information	Content
Keywords	NTM88, Firmware Library
Abstract	This document describes the firmware library provided for all derivatives of the NTM88 family.

1 Introduction

This document describes the functions available in the NTM88 Firmware Library.

The intended audience for this document includes firmware architects, developers, coders, and testers working with the NTM88 device.

This document is divided into three sections: this introduction, a section describing global variables, and standard formats used throughout the functions, and a third section describing each function.

2 Globals and formats

2.1 Global variables

Table 1 summarizes all global variables used by NXP firmware library and their locations. Developers must account for these variables when creating new user firmware.

Table 1. Global variable and their locations

Name	Address	Reference
TPMS_INTERRUPT_FLAG	8Fh	Section 2.1.1

Note: *In the library, the global variable gu8TPMSLPDMFlag is declared at address 8Eh. This variable is present for backward compatibility with previous library versions. The gu8TPMSLPDMFlag variable is not currently used by the NTM88 library functions, so users may use address 8Eh to store application variables.*

2.1.1 TPMS_INTERRUPT_FLAG

This global variable keeps track of interrupts that have occurred. The NTM88 Firmware Library uses it to keep track of expected interrupts. Also, you can use this variable for your own purposes. A number of firmware functions utilize the Stop1 or Stop4 modes. When the MCU enters STOP mode, the watch-dog is halted. In order to provide a back-up recovery, users should utilize either the RTI or PWU which can be programmed for interrupt if a software or firmware routine has consumed too much time. The Watch-dog is automatically restarted when the program goes back in RUN mode.

The TPMS_INTERRUPT_FLAG is not cleared automatically. Users must clear this variable after Power-On-Reset (POR).

Table 2 shows the TPMS_INTERRUPT_FLAG format. The Trigger condition column describes conditions necessary for that flag to be set.

Table 2. TPMS_INTERRUPT_FLAG format and trigger conditions

Flag	Bit	Trigger condition
LVD Interrupt	7	LVD interrupt entered
PWU Interrupt	6	PWU interrupt entered
TOF Interrupt	5	TOF interrupt entered
LFR Error Interrupt	4	LFR interrupt entered and LFERF bit of the LFS register is set
ADC Interrupt	3	ADC interrupt entered
LFR Interrupt	2	LFR interrupt entered and LFERF bit of the LFS register is clear
RTI Interrupt	1	RTI interrupt entered
KBI Interrupt	0	KBI interrupt entered

TPMS_INTERRUPT_FLAG is 1 byte long and is located at address 8Fh. Users must account for this variable when developing for the NTM88.

2.2 Flash memory allocations

In the library, the functions and constant variables are declared under specific sections. In the PRM file of the application project, users can map these sections to the desired addresses in FLASH.

The library functions are declared under the sections FIRMWARE_SECTION_1, APP_LIB_SECTION and USER_ROM. The library constant variables are declared under the DEFAULT section, allocated to ROM_VAR section in the PRM file of the application project.

2.3 Universal uncompensated measurement array (UUMA) format

The NTM88 measurement routines are divided into two subsets:

- Routines that return uncompensated measurements
- Routines that take uncompensated measurements as arguments and return compensated measurements

In order to be consistent and keep the number of CPU cycles down, all uncompensated measurement routines will return data following the array format described in Table 3, and all compensating routines will take data from the same array.

Table 3. Universal uncompensated measurement array

Index	Content
0	Uncompensated voltage
1	Uncompensated temperature
2	Uncompensated pressure
3	Uncompensated X-axis acceleration[1]
4	Uncompensated Z-axis acceleration[2]

[1] Applicable to devices supporting X-axis measurement.
[2] Applicable to devices supporting Z-axis measurement.

This array is referred to as Universal Uncompensated Measurement Array (UUMA). It can be located anywhere the user decides.

Each element must be 16-bits long (2 bytes) regardless of what the actual bit-width of the measurement is.

Each individual uncompensated measurement routine will only update its corresponding item. For example, calling the TPMS_READ_VOLTAGE routine will only modify the voltage element of the array. The rest remains unchanged.

Compensation routines do not modify any elements in the UUMA.

2.4 Simulated SPI interface signal format

The NTM88 includes three routines (TPMS_MSG_INIT, TPMS_MSG_READ and TPMS_MSG_WRITE) that, when used together, allow the user to perform serial communication with the device through a simulated SPI interface.

The following assumptions are made:

- Only two pins are used: PTA0 for data (both incoming and outgoing) and PTA1 for clock. No slave select is included by default, but the user may use any other pin if required.

- The data pin has a pullup resistor enabled.
- The NTM88 will be a master device (the NTM88 will provide the clock).
- Data can be read/written 8 bits at a time.
- Speed of the interface is dependent on bus clock settings.
- Data is transferred MSB first.
- A single line will be used for both sending and receiving data (BIDIROE = SET according to NXP nomenclature).
  - At the clock's rising edge, the master will place data on the pin. It will be valid until the clock's falling edge. The slave must not drive the line during this period.
  - At the clock's falling edge, the master will make the data pin an input and will listen for data. The slave must then place data on the data line until the clock's rising edge.
- Clock Polarity = 0 (Normally low).
- Clock Phase = 1 (First half is high).

Figure 1 shows the details of the simulated SPI interface.

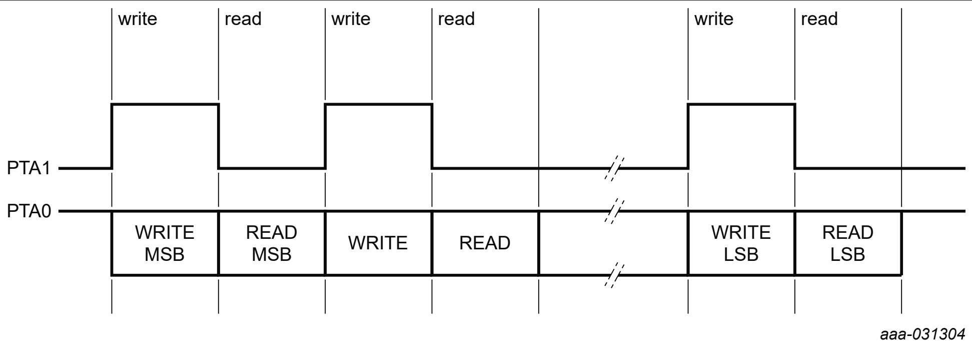

Figure 1. Description of the physical layer on the NTM88 simulated SPI interface

For further information on the use of the Simulated SPI interface routines, see:

- Section 3.2.30 "void TPMS_MSG_INIT (void)"
- Section 3.2.32 "UINT8 TPMS_MSG_READ (void)"
- Section 3.2.31 "UINT8 TPMS_MSG_WRITE (UINT8 u8SendByte)"

2.5 LFR registers initialized by firmware

Some LFR registers are touched by firmware when the function vnfLFRConfigFactoryRegs() provided in the NTM88 Firmware Library is executed. The goal of this action is to configure the LFR module in the best-known configuration for Manchester encoded reception.

LFR registers will be configured differently, depending on the user-selected sensitivity. Table 4 and Table 5 describe these settings.

Table 4. Customer-configurable LF Register with SENS = 1

	Bit name							
Register name	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LFCTL1	LFEN	SRES	CARMOD	—	IDSEL		SENS	
LFCTL2	LFSTM				LFONTM			
LFCTL3	LFDO	TOGMOD	SYNC		LFCDTM			
LFCTL4	LFDRIE	LFERIE	LFCDIE	LFIDIE	DECEN	VALEN	TIMOUT	
LFS	LFDRF	LFERF	LFCDF	LFIDF	LFOVF	LFEOMF	LPSM	LFIAK

Table 4. Customer-configurable LF Register with SENS = 1*...continued*

	Bit name							
Register name	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LFDATA	RXDATA							
LFIDL	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
LFIDH	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
LFCTRLE	—	—	—	—	—	0	0	0
LFCTRLD	1	0	DEQS	1	1	1	0	0
LFCTRLC	0	0	0	1	AZEN	LOWQ		DEQEN
LFCTRLB	1	1	LFFAF	LFCAF	LFPOL	1	1	0
LFCTRLA	—	—	—	—	LFCC			
TRIM1	—	—	—	—	—	—	—	—
TRIM2	—	—	—	—	—	—	—	—

▨	Shaded cells show register touched by firmware; loaded value is displayed.

Table 5. Customer-configurable LF Register with SENS = 2

	Bit name							
Register name	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LFCTL1	LFEN	SRES	CARMOD	—	IDSEL		SENS	
LFCTL2	LFSTM				LFONTM			
LFCTL3	LFDO	TOGMOD	SYNC		LFCDTM			
LFCTL4	LFDRIE	LFERIE	LFCDIE	LFIDIE	DECEN	VALEN	TIMOUT	
LFS	LFDRF	LFERF	LFCDF	LFIDF	LFOVF	LFEOMF	LPSM	LFIAK
LFDATA	RXDATA							
LFIDL	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
LFIDH	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
LFCTRLE	—	—	—	—	—	0	0	0
LFCTRLD	1	0	DEQS	1	1	1	1	1
LFCTRLC	0	0	0	1	AZEN	LOWQ		DEQEN
LFCTRLB	1	1	LFFAF	LFCAF	LFPOL	1	1	0
LFCTRLA	—	—	—	—	LFCC			
TRIM1	—	—	—	—	—	—	—	—
TRIM2	—	—	—	—	—	—	—	—

▨	Shaded cells show register touched by firmware; loaded value is displayed.

3 Firmware functions

3.1 Firmware functions

Table 6 lists the available firmware functions for NTM88 single- and dual-axis accelerometers.

Table 6. NTM88 firmware functions

Return type	Function	Single/Dual axis	Reference
VOID	TPMS_RESET	Single/Dual	Section 3.2.1
UINT8	TPMS_READ_VOLTAGE	Single/Dual	Section 3.2.2
UINT8	TPMS_COMP_VOLTAGE	Single/Dual	Section 3.2.3
UINT8	TPMS_READ_TEMPERATURE	Single/Dual	Section 3.2.4
UINT8	TPMS_COMP_TEMPERATURE	Single/Dual	Section 3.2.5
UINT8	TPMS_READ_PRESSURE	Single/Dual	Section 3.2.6
UINT8	TPMS_COMP_PRESSURE	Single/Dual	Section 3.2.7
UINT8	TPMS_READ_ACCEL_X	Single-X/Dual	Section 3.2.8
UINT8	TPMS_COMP_ACCEL_X	Single-X/Dual	Section 3.2.9
UINT8	TPMS_READ_DYNAMIC_ACCEL_X	Single-X/Dual	Section 3.2.10
UINT8	TPMS_READ_ACCEL_Z	Single-Z/Dual	Section 3.2.11
UINT8	TPMS_COMP_ACCEL_Z	Single-Z/Dual	Section 3.2.12
UINT8	TPMS_READ_DYNAMIC_ACCEL_Z	Single-Z/Dual	Section 3.2.13
UINT8	TPMS_READ_V0	Single/Dual	Section 3.2.14
UINT8	TPMS_READ_V1	Single/Dual	Section 3.2.15
UINT8	TPMS_LFOCAL	Single/Dual	Section 3.2.16
UINT8	TPMS_LFOCAL_BUSCLK	Single/Dual	Section 3.2.17
UINT8	TPMS_MFOCAL	Single/Dual	Section 3.2.18
UINT16	TPMS_WAVG	Single/Dual	Section 3.2.19
VOID	TPMS_RF_ENABLE	Single/Dual	Section 3.2.20
VOID	TPMS_RF_RESET	Single/Dual	Section 3.2.21
VOID	TPMS_RF_READ_DATA	Single/Dual	Section 3.2.22
VOID	TPMS_RF_READ_DATA_REVERSE	Single/Dual	Section 3.2.23
VOID	TPMS_RF_WRITE_DATA	Single/Dual	Section 3.2.24
VOID	TPMS_RF_WRITE_DATA_REVERSE	Single/Dual	Section 3.2.25
VOID	TPMS_RF_CONFIG_DATA	Single/Dual	Section 3.2.26
VOID	TPMS_RF_SET_TX	Single/Dual	Section 3.2.27
VOID	TPMS_READ_ID	Single/Dual	Section 3.2.28
VOID	TPMS_RF_DYNAMIC_POWER	Single/Dual	Section 3.2.29
VOID	TPMS_MSG_INIT	Single/Dual	Section 3.2.30
UINT8	TPMS_MSG_WRITE	Single/Dual	Section 3.2.31
UINT8	TPMS_MSG_READ	Single/Dual	Section 3.2.32

Table 6. NTM88 firmware functions*...continued*

Return type	Function	Single/Dual axis	Reference
UINT8	TPMS_CHECKSUM_XOR	Single/Dual	Section 3.2.33
UINT8	TPMS_CRC8	Single/Dual	Section 3.2.34
UINT16	TPMS_SQUARE_ROOT	Single/Dual	Section 3.2.35
VOID	TPMS_LF_ENABLE	Single/Dual	Section 3.2.36
UINT8	TPMS_LF_READ_DATA	Single/Dual	Section 3.2.37
UINT8	TPMS_WIRE_AND_ADC_CHECK	Single/Dual	Section 3.2.38
VOID	TPMS_FLASH_WRITE	Single/Dual	Section 3.2.39
UINT16	TPMS_FLASH_ERASE	Single/Dual	Section 3.2.40
VOID	TPMS_MULT_SIGN_INT16	Single/Dual	Section 3.2.41
UINT8	TPMS_VREG_CHECK	Single/Dual	Section 3.2.42
UINT8	TPMS_E_READ_ACCEL_X	Single-X/Dual	Section 3.2.43
UINT8	TPMS_E_READ_PRESSURE	Single/Dual	Section 3.2.44
UINT8	TPMS_E_READ_ACCEL_Z	Single-Z/Dual	Section 3.2.45
VOID	TPMS_E_FRC_ENABLE	Single/Dual	Section 3.2.46
VOID	TPMS_E_FRC_DISABLE	Single/Dual	Section 3.2.47
VOID	TPMS_E_FRC_CLEAR	Single/Dual	Section 3.2.48
VOID	TPMS_E_FRC_READ	Single/Dual	Section 3.2.49
UINT8	TPMS_E_FRC_CALIB	Single/Dual	Section 3.2.50
UINT8	TPMS_E_FRC_CALIB_BUSCLK	Single/Dual	Section 3.2.51
UINT8	TPMS_PRECHARGE_EN	Single/Dual	Section 3.2.52
VOID	TPMS_E_ACTIVATE_CHANNEL	Single/Dual	Section 3.2.53
VOID	TPMS_E_DEACTIVATE_CHANNEL	Single/Dual	Section 3.2.54
UINT8	TPMS_E_READ_CONT_ACCEL_X	Single-X/Dual	Section 3.2.55
UINT8	TPMS_E_READ_CONT_ACCEL_Z	Single-Z/Dual	Section 3.2.56
UINT8	TPMS_E_READ_CONT_PRESSURE	Single/Dual	Section 3.2.57
UINT8	TPMS_E_READ_DYNAMIC_ACCEL_Z	Single-Z/Dual	Section 3.2.58
UINT8	TPMS_E_READ_DYNAMIC_ACCEL_X	Single-X/Dual	Section 3.2.59

3.2 Function description

The following function descriptions include stack sizes and approximate duration.

Stack sizes have been calculated by executing each routine and measuring the amount of memory utilized. Unless noted, these sizes represent the maximum stack the function will utilize.

Duration estimates are performed on one part at room temperature. These estimates are intended to serve as a guideline for typical execution time.

Table 7 and Table 8 describe the SMI initial and subsequent delays.

Table 7. SMI initial delay

SMI initial delay	
SAMP_DLY_INIT_104US	0x0h
SAMP_DLY_INIT_128US	0x1h
SAMP_DLY_INIT_160US	0x2h
SAMP_DLY_INIT_200US	0x3h
SAMP_DLY_INIT_256US	0x4h
SAMP_DLY_INIT_320US	0x5h
SAMP_DLY_INIT_408US	0x6h
SAMP_DLY_INIT_512US	0x7h
SAMP_DLY_INIT_648US	0x8h
SAMP_DLY_INIT_816US	0x9h
SAMP_DLY_INIT_1024US	0xAh
SAMP_DLY_INIT_1288US	0xBh
SAMP_DLY_INIT_1624US	0xCh
SAMP_DLY_INIT_2048US	0xDh
SAMP_DLY_INIT_2584US	0xEh
SAMP_DLY_INIT_3248US	0xFh

Table 8. SMI subsequent delay

SMI subsequent delay	
SAMP_DLY_SUBS_64US	0x00h
SAMP_DLY_SUBS_80US	0x10h
SAMP_DLY_SUBS_104US	0x20h
SAMP_DLY_SUBS_128US	0x30h
SAMP_DLY_SUBS_160US	0x40h
SAMP_DLY_SUBS_200US	0x50h
SAMP_DLY_SUBS_256US	0x60h
SAMP_DLY_SUBS_320US	0x70h
SAMP_DLY_SUBS_408US	0x80h
SAMP_DLY_SUBS_512US	0x90h
SAMP_DLY_SUBS_648US	0xA0h
SAMP_DLY_SUBS_816US	0xB0h
SAMP_DLY_SUBS_1024US	0xC0h
SAMP_DLY_SUBS_1288US	0xD0h
SAMP_DLY_SUBS_1624US	0xE0h
SAMP_DLY_SUBS_2048US	0xF0h

3.2.1 void TPMS_RESET (void)

- **Description:** This function will call the function vnfLFRConfigFactoryRegs() desribed in Section 2.5. Next, it will reset the Stack Pointer to the last RAM location and jump to the main() function of the user application project. No further initialization is performed.
- **Stack size:** 2 bytes
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await interrupts. It is not affected by interrupts either.
- **Resources:** Stack
- **Input Parameters:**
  – None
- **Returns:**
  – void

3.2.2 UINT8 TPMS_READ_VOLTAGE (UINT16 *u16UUMA)

- **Description:** Performs a 10-bit uncompensated voltage measurement and places it in the UUMA. While waiting for the ADC to converge, this function goes into STOP4. If the ADC, for an unexpected reason, fails to converge, this function has the following built-in timeout: After five continuous non-ADC interrupts, the function will assume a failed ADC reading, flag it accordingly, and exit.
  – If the ADC value is over or under the normal operating condition, the voltage error status flag will be set. The expected voltage result will be forced to either 0 or 1023.
  – If the ADC times out with no result, the ADC error status flag will be set.
  – Measurements below 2.1 V are not guaranteed for accuracy.
- **Stack size:** 23 bytes
- **Approx. Duration:** 99 µs
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** ADC, band gap
- **Input Parameters:**
  – UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3). Only the 10-bit uncompensated voltage result will be updated.
- **Returns:**
  – UINT8 u8Status: Valid error flags/outputs are described in Table 9.

Table 9. Valid output conditions for TPMS_READ_VOLTAGE

u8Status Value	Measurement value	Condition
20h	03FFh	Uncompensated voltage reading outside the valid range (high)
20h	0000h	Uncompensated voltage reading outside the valid range (low)
80h	Undefined	Uncompensated voltage reading not acquired
00h	0001h to 03FEh	Valid uncompensated voltage reading

Note: *The band gap bit (bit 0 in the SPMSC1 register) must be set prior to calling this function for results to be valid.*

3.2.3 UINT8 TPMS_COMP_VOLTAGE (UINT8 *u8CompVoltage, *UINT16 u16UUMA)

- **Description:** Performs an 8-bit compensated voltage measurement. It is the responsibility of the user to ensure that updated and valid uncompensated voltage reading is available in the UUMA for this routine to return a meaningful value.
  - If V_{out} < 1.7 V, result will be 1 and the over/underflow status flag will be set.
  - Measurements below 2.1 V are not guaranteed for accuracy.
  - If V_{out} ≥ 3.7 V, result will be FFh and the over/underflow status flag will be set.
  - For repeatability data, refer to the NTM88 family data sheets.
- **Stack size:** 31 bytes
- **Approx. Duration:** 204 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await interrupts. It is not affected by interrupts either.
- **Resources:** UUMA
- **Input Parameters:**
  - UINT8 *u8CompVoltage: Updated 8-bit compensated voltage result.
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3). Uncompensated voltage will be utilized from this array.
- **Returns:**
  - UINT8 u8Status: Valid error flags/outputs are described in Table 10.

Table 10. Valid output conditions for TPMS_COMP_VOLTAGE

u8Status Value	Measurement value	Condition
01h	FFh	Compensated voltage reading outside the valid range (high)
01h	01h	Compensated voltage reading outside the valid range (low)
00h	02h to FEh	Valid compensated voltage reading

3.2.4 UINT8 TPMS_READ_TEMPERATURE (UINT16 *u16UUMA)

- **Description:** Performs a 12-bit uncompensated temperature measurement and places the result in the UUMA. While waiting for the ADC to converge, this function goes into STOP4. If the ADC, for an unexpected reason, fails to converge, this function has the following built-in timeout: After five continuous non-ADC interrupts, the function will assume a failed ADC reading, flag it accordingly, and exit. If the LVWF (Low Voltage Warning Flag) hardware bit is set, it will flag it accordingly as well. The LVWF flag is cleared before the function returns.
  - If the ADC value is over or under the normal operating condition, the temperature error status flag will be set. The expected temperature result will be forced to either 0 or 4095.
  - If the ADC times out with no result, the ADC error status flag will be set.
- **Stack size:** 18 bytes
- **Approx. Duration:** 219 µs
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** ADC, band gap
- **Input Parameters:**
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3). Only the 12-bit uncompensated temperature result will be updated.
- **Returns:**
  - UINT8 u8Status:Valid error flags/outputs are described in Table 11.

Table 11. Valid output conditions for TPMS_READ_TEMPERATURE

u8Status Value	Measurement value	Condition
40h	0FFFh	Uncompensated temperature reading outside the valid range (high)
40h	0000h	Uncompensated temperature reading outside the valid range (low)
60h	0FFFh	Uncompensated temperature reading outside the valid range (high) and LVWF set
60h	0000h	Uncompensated temperature reading outside the valid range (low) and LVWF set
80h	Undefined	Uncompensated temperature reading not acquired
A0h	Undefined	Uncompensated temperature reading not acquired and LVWF set
00h	0001h to 0FFEh	Valid uncompensated temperature reading
20h	0001h to 0FFEh	Valid uncompensated temperature reading and LVWF set

Note: *The band gap bit (bit 0 in the SPMSC1 register) must be set prior to calling this function for results to be valid.*

3.2.5 UINT8 TPMS_COMP_TEMPERATURE (UINT8 *u8Temp, UINT16 *u16UUMA)

- **Description:** Performs an 8-bit compensated temperature measurement. It is the responsibility of the user to ensure that an updated and valid uncompensated temperature reading is available in the UUMA for this routine to return a meaningful value.
  - If T_{out} < –50 °C, u8Temp will be 1 and the over/underflow status flag will be set.
  - If T_{out} ≥ 200 °C, u8Temp will be FFh and the over/underflow status flag will be set.
- **Stack size:** 30 bytes
- **Approx. Duration:** 209 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await interrupts. It is not affected by interrupts either.
- **Resources:** UUMA
- **Input Parameters:**
  - UINT8 *u8Temp: Updated 8-bit compensated temperature result.
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3). Uncompensated temperature will be utilized from this array.
- **Returns:** UINT8 u8Status: Valid error flags/outputs are described in Table 12.

Table 12. Valid output conditions for TPMS_COMP_TEMPERATURE

u8Status Value	Measurement value	Condition
01h	FFh	Compensated temperature reading outside the valid range (high)
01h	01h	Compensated temperature reading outside the valid range (low)
00h	02h to FEh	Valid compensated temperature reading

3.2.6 UINT8 TPMS_READ_PRESSURE (UINT16 *u16UUMA, UINT8 u8Avg)

- **Description:** Performs a 12-bit uncompensated pressure measurement and places it in the UUMA. The function configures an initial delay of 2048 µs and a subsequent delay of 512 µs. While waiting for the ADC to converge, this function goes into STOP4. If the ADC, for an unexpected reason, fails to converge, this function has the following built-in timeout: After five continuous non-ADC interrupts, the function will assume a failed ADC reading, flag it accordingly, and exit. If the LVWF (Low Voltage Warning Flag) hardware bit is set, it will flag it accordingly as well. The LVWF flag is cleared before the function returns.
  - If the ADC value is over or under the normal operating condition, the pressure error status flag will be set. The expected pressure result will be forced to either 0 or 4095.
  - If the ADC times out with no result, the ADC error status flag will be set.
- **Stack size:** 32 bytes
- **Approx. Duration:**

Table 13. Approximate duration for TPMS_READ_PRESSURE

Mode	Component	Estimated duration during normal operation	Observed duration [ms]
Average of 1	Bus clock cycles	840	0.21
	Initial delay MFO clock cycles	256	2.04
	ADC conversion time [µs] [1]	20	0.02
	STOP4 exit time [µs] [2]	114	0.014
	Total [ms]	—	2.284
Average of 4	Additional bus clock cycles per additional sample	36	0.009
	Subsequent MFO clock cycles per additional sample	64	0.512
	Additional ADC conversion time per additional sample[1]	20	0.02
	Additional STOP4 exit time per additional sample[2]	14	0.014
	Additional time per additional sample excludes subsequent delay [ms]	0.043	—
	Total for average = 4 [ms]	—	3.949

[1] Typical ADC conversion time with nominal ADC clock at conversion settings.
[2] Typical STOP4 exit time. For exact range, refer to product data sheet.

- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** SMI, ADC, internal bond wires
- **Input Parameters:**
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3). Only the 12-bit uncompensated pressure result will be updated.
  - UINT8 u8Avg: Number of measurements to average into one result. The value can be set to 1, 2, 4, 8, or 16.
- **Returns:** UINT8 u8Status: Valid error flags/outputs are described in Table 14.

Table 14. Valid output conditions for TPMS_READ_PRESSURE

u8Status Value	Measurement value	Condition
04h	0FFFh	Uncompensated pressure reading outside the valid range (high)
04h	0000h	Uncompensated pressure reading outside the valid range (low)
24h	0FFFh	Uncompensated pressure reading outside the valid range (high), and LVWF set
24h	0000h	Uncompensated pressure reading outside the valid range (low), and LVWF set
80h	0000h	Uncompensated pressure reading not acquired
A0h	0000h	Uncompensated pressure reading not acquired, and LVWF set
00h	0001h to 0FFEh	Valid uncompensated pressure reading
20h	0001h to 0FFEh	Valid uncompensated pressure reading, and LVWF set

3.2.7 UINT8 TPMS_COMP_PRESSURE (UINT16 *u16CompPressure, UINT16 *u16UUMA)

- **Description:** Performs a 10-bit compensated pressure measurement. It is the responsibility of the user to ensure that updated and valid uncompensated voltage, temperature, and pressure readings are available in the UUMA for this routine to return a meaningful value.
  - If either the temperature or supply voltage measurements, inherent to this function, result in a fault, the pressure reading will be forced to 0 and the appropriate pressure, temperature, and/or voltage flags will be set in the status flag.
  - If P_{out} < 90 kPa, the over/underflow status flag will be set, and u16CompPressure will be forced to 001h.
  - If P_{out} ≥ maximum pressure for the part number, u16CompPressure will be 3FFh and the over/underflow status flag will be set.
  - If the passed uncompensated voltage measurement is estimated to be under the guaranteed operational region, the routine will set the Voltage status flag. The accuracy of the returned value is not guaranteed.
- **Stack size:** 38 bytes
- **Approx. Duration:** 766.5 μs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await interrupts. It is not affected by interrupts either.
- **Resources:** UUMA
- **Input Parameters:**
  - u16CompPressure: Updated 10-bit compensated pressure result.
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3). Uncompensated voltage, temperature, and pressure will be taken from this array.
- **Returns:** UINT8 u8Status: Valid error flags/outputs are described in Table 15.

Table 15. Valid output conditions for TPMS_COMP_PRESSURE

u8Status Value	Measurement value	Condition
01h	03FFh	Compensated pressure reading outside the valid range (high).
01h	0001h	Compensated pressure reading outside the valid range (low).
21h	03FFh	Compensated pressure reading outside the valid range (high), and uncompensated voltage suspected to be below valid operating range for this function.

Table 15. Valid output conditions for TPMS_COMP_PRESSURE*...continued*

u8Status Value	Measurement value	Condition
21h	0001h	Compensated pressure reading outside the valid range (low), and uncompensated voltage suspected to be under below operating range for this function.
20h	0002h to 03FEh	Uncompensated voltage suspected to be below valid operating range for this function. The compensated reading is not guaranteed for accuracy.
00h	0002h to 03FEh	Valid compensated pressure reading.

3.2.8 UINT8 TPMS_READ_ACCEL_X (UINT16 *u16UUMA, UINT8 u8Avg, UINT8 u8Filter, UINT8 u8OffsetIndex)

- **Description:** Performs an uncompensated 12-bit measurement for the X-axis. The function configures an initial delay of 2048 µs and a subsequent delay of 512 µs. While waiting for the ADC to converge, this function goes into STOP4. If the ADC, for an unexpected reason, fails to converge, this function has the following built-in timeout: After five continuous non-ADC interrupts, the function will assume a failed ADC reading, flag it accordingly, and exit. If the LVWF (Low Voltage Warning Flag) hardware bit is set, it will flag it accordingly as well. The LVWF flag is cleared before the function returns.
  - If the ADC value is over or under the normal operating condition, the acceleration error status flag will be set. The expected acceleration result will be forced to either 0 or 4095.
  - If the ADC times out with no result, the ADC error status flag will be set.
- **Stack size:** 37 bytes
- **Approx. Duration:**

Table 16. Approximate duration for TPMS_READ_ACCEL_X

Mode	Component	Estimated duration during normal operation	Observed duration [ms]
500 Hz, Average of 1	Bus clock cycles	836	0.209
	Initial delay MFO clock cycles	256	2.04
	ADC conversion time [µs] [1]	20	0.02
	STOP4 exit time [µs] [2]	14	0.014
	Total [ms]	—	2.283
500 Hz, Average of 4	Additional bus clock cycles per additional sample	38	0.0096
	Subsequent MFO clock cycles per additional sample	64	0.512
	Additional ADC conversion time per additional sample[1]	20	0.02
	Additional STOP4 exit time per additional sample[2]	14	0.014
	Additional time per additional sample [ms]	0.0436	—
	Total for average = 4 [ms]	—	3.9498

[1] Typical ADC conversion time with nominal ADC clock at conversion settings.
[2] Typical STOP4 exit time. For exact range, refer to product data sheet.

- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.

- **Resources:** SMI, ADC, internal bond wires
- **Input Parameters:**
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3). Only the 12-bit uncompensated acceleration result will be updated.
  - UINT8 u8Avg: Number of measurements to average into one result. The value can be set to 1, 2, 4, 8, or 16.
  - UINT8 u8Filter: Will set up the acceleration measurement based on Table 17.

Table 17. u8Filter options

u8Filter Value	Selected filter
0	250 Hz low-pass filter selected
1	500 Hz low-pass filter selected
2	1000 Hz low-pass filter selected
3	2000 Hz low-pass filter selected

- UINT8 u8OffsetIndex: Selects the offset setting for the appropriate acceleration reading. The default index is 7. Valid range is 0 to 15.
- **Returns:** UINT8 u8Status: Valid error flags/outputs are described in Table 18.

Table 18. Valid output conditions for TPMS_READ_ACCEL_X

u8Status value	Measurement value	Condition
08h	0FFFh	Uncompensated acceleration reading outside the valid range (high).
08h	0000h	Uncompensated acceleration reading outside the valid range (low).
28h	0FFFh	Uncompensated acceleration reading outside the valid range (high), and LVWF set.
28h	0000h	Uncompensated acceleration reading outside the valid range (low), and LVWF set.
80h	0000h	Uncompensated acceleration reading not acquired.
A0h	0000h	Uncompensated acceleration reading not acquired, and LVWF set.
00h	0001h to 0FFEh	Valid uncompensated acceleration reading.
20h	0001h to 0FFEh	Valid uncompensated acceleration reading, but LVWF set.

3.2.9 UINT8 TPMS_COMP_ACCEL_X (UINT16 *u16CompAccel, UINT16* u16UUMA)

- **Description:** Performs a 10-bit compensated acceleration measurement for the X-axis. It is the responsibility of the user to ensure that updated and valid uncompensated voltage, temperature, and acceleration readings are available in the UUMA for this routine to return a meaningful value.
  - If u16CompAccel rails low, u16CompAccel will be forced to 1 and the over/underflow status flag will be set.
  - If u16CompAccel rails high, u16CompAccel will be forced to 3FFh and the over/underflow status flag will be set.
  - If the incoming uncompensated voltage measurement is estimated to be under the guaranteed operational region, the routine will set the Voltage status flag. The accuracy of the returned value is not guaranteed.
  - For repeatability data, refer to the NTM88 family data sheets.
- **Stack size:** 48 bytes

- **Approx. Duration:** 843 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await interrupts. It is not affected by interrupts either.
- **Resources:** UUMA
- **Input Parameters:**
  – UINT16 *u16CompAccel: Updated 10-bit compensated acceleration.
  – UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3). Uncompensated voltage, temperature, and acceleration will be taken from this array.
- **Returns:**
  – UINT8 u8Status: Valid error flags/outputs are described in Table 19.

Table 19. Valid output conditions for TPMS_COMP_ACCEL_X

u8Status Value	Measurement value	Condition
01h	03FFh	Compensated acceleration reading outside the valid range (high)
01h	0001h	Compensated acceleration reading outside the valid range (low)
21h	03FFh	Compensated acceleration reading outside the valid range (high), and uncompensated voltage suspected to be below valid operating range for this function
21h	0001h	Compensated acceleration reading outside the valid range (low), and uncompensated voltage suspected to be below valid operating range for this function
20h	0002h to 03FEh	Uncompensated voltage suspected to be below valid operating range for this function; The compensated reading is not guaranteed for accuracy
00h	0002h to 03FEh	Valid compensated acceleration reading

3.2.10 UINT8 TPMS_READ_DYNAMIC_ACCEL_X (UINT8 u8Filter, UINT8* u8Offset, UINT16* u16UUMA)

- **Description:** This function automatically executes a TPMS_READ_ACCEL_X measurement with a given initial dynamic offset. If the result is too high or too low, it will change the dynamic offset value and re-execute TPMS_READ_ACCEL_X until a) the result is valid or b) the result is railed high or low and there are no more offset steps. Offset and uncompensated acceleration inside the UUMA are updated. The function configures an initial delay of 2048 µs.
- **Stack size:** 48 bytes
- **Approx. Duration:** 2950 µs when starting offset is in target; 29050 µs when the offset is 10 steps away.
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** SMI, ADC, internal bond wires.
- **Input Parameters:**
  – UINT8 u8Filter: If non-zero, 250 Hz filter enabled. Otherwise, 500 Hz filter selected.
  – UINT8* u8Offset: Pointer to initial offset step to load. Valid offset steps range from 0 - 15 and are described in the devices data sheet. An updated offset value is returned at the end of the function. In case the acceleration is too high or too low and function has run out of offset steps, a value of 255 ("0 - 1") or 16 ("15 + 1") shall be returned.

- UINT16 *u16UUMA: Pointer to the Universal Uncompensated Measurement Array. Uncompensated acceleration will be updated accordingly.

- **Returns:**
  - UINT8 u8Status: See Section 3.2.8, TPMS_READ_ACCEL_X for more information on the format of this status byte.

3.2.11 UINT8 TPMS_READ_ACCEL_Z (UINT16 *u16UUMA, UINT8 u8Avg, UINT8 u8Filter, UINT8 u8OffsetIndex)

- **Description:** Performs an uncompensated 12-bit measurement. The function configures an initial delay of 2048 us and a subsequent delay of 512 µs. While waiting for the ADC to converge, this function goes into STOP4. If the ADC, for an unexpected reason, fails to converge, this function has the following built-in timeout: After five continuous non-ADC interrupts, the function will assume a failed ADC reading, flag it accordingly, and exit. If the LVWF (Low Voltage Warning Flag) hardware bit is set, it will flag it accordingly as well. The LVWF flag is cleared before the function returns.
  - If the ADC value is over or under the normal operating condition, the acceleration error status flag will be set. The expected acceleration result will be forced to either 0 (rail high) or 4095 (rail low).
  - If the ADC times out with no result, the ADC error status flag will be set.
- **Stack size:** 37 bytes
- **Approx. Duration:**

Table 20. Approximate duration for TPMS_READ_ACCEL_Z

Mode	Component	Estimated duration during normal operation	Observed duration [ms]
500 Hz, Average of 1	Bus clock cycles	836	0.209
	Initial delay MFO clock cycles	256	2.04
	ADC conversion time [µs] [1]	20	0.02
	STOP4 exit time [µs] [2]	14	0.014
	Total [ms]	—	2.283
500 Hz, Average of 4	Additional bus clock cycles per additional sample	38	0.0096
	Subsequent MFO clock cycles per additional sample	64	0.512
	Additional ADC conversion time per additional sample[1]	20	0.02
	Additional STOP4 exit time per additional sample[2]	14	0.014
	Additional time per additional sample [ms]	0.0436	—
	Total for average = 4 [ms]	—	3.9498

[1] Typical ADC conversion time with nominal ADC clock at conversion settings.
[2] Typical STOP4 exit time. For exact range, refer to product data sheet.

- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** SMI, ADC, internal bond wires
- **Input Parameters:**
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3). Only the 12-bit uncompensated acceleration result will be updated.

- UINT8 u8Avg: Number of measurements to average into one result. The value can be set to 1, 2, 4, 8, or 16.
- UINT8 u8Filter: Will set up the acceleration measurement based on Table 21.
- UINT8 u8OffsetIndex: Selects the offset setting for the appropriate acceleration reading. The default index is 7. Valid range is 0 to 15.
- **Returns:** UINT8 u8Status: Valid error flags/outputs are described in Table 22.

Table 21. u8Filter options

u8Filter Value	Selected filter
0	250 Hz low-pass filter selected
1	500 Hz low-pass filter selected
2	1000 Hz low-pass filter selected
3	2000 Hz low-pass filter selected

Table 22. Valid output conditions for TPMS_READ_ACCEL_Z

u8Status Value	Measurement value	Condition
10h	0FFFh	Uncompensated acceleration reading outside the valid range (high).
10h	0000h	Uncompensated acceleration reading outside the valid range (low).
30h	0FFFh	Uncompensated acceleration reading outside the valid range (high), and LVWF set.
30h	0000h	Uncompensated acceleration reading outside the valid range (low), and LVWF set.
80h	0000h	Uncompensated acceleration reading not acquired.
A0h	0000h	Uncompensated acceleration reading not acquired, and LVWF set.
00h	0001h to 0FFEh	Valid uncompensated acceleration reading.
20h	0001h to 0FFEh	Valid uncompensated acceleration reading, but LVWF set.

3.2.12 UINT8 TPMS_COMP_ACCEL_Z (UINT16 *u16CompAccel, UINT16* u16UUMA)

- **Description:** Performs a 10-bit compensated acceleration measurement. It is the responsibility of the user to ensure that updated and valid uncompensated voltage, temperature, and acceleration readings are available in the UUMA for this routine to return a meaningful value.
  - If u16CompAccel rails low, u16CompAccel will be forced to 1 and the over/underflow status flag will be set.
  - If u16CompAccel rails high, u16CompAccel will be forced to 3FFh and the over/underflow status flag will be set.
  - If the incoming uncompensated voltage measurement is estimated to be under the guaranteed operational region, the routine will set the Voltage status flag. The accuracy of the returned value is not guaranteed.
  - For repeatability data, refer to the NTM88 family data sheets.
- **Stack size:** 48 bytes
- **Approx. Duration:** 843 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await interrupts. It is not affected by interrupts either.

- **Resources:** UUMA
- **Input Parameters:**
  – UINT16 *u16Accel: Updated 10-bit compensated acceleration.
  – UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3). Uncompensated voltage, temperature, and acceleration will be taken from this array.
- **Returns:**
  – UINT8 u8Status: Valid error flags/outputs are described in Table 23.

Table 23. Valid output conditions for TPMS_COMP_ACCEL_Z

u8Status Value	Measurement value	Condition
01h	03FFh	Compensated acceleration reading outside the valid range (high)
01h	0001h	Compensated acceleration reading outside the valid range (low)
21h	03FFh	Compensated acceleration reading outside the valid range (high), and uncompensated voltage suspected to be below valid operating range for this function
21h	0001h	Compensated acceleration reading outside the valid range (low), and uncompensated voltage suspected to be below valid operating range for this function
20h	0002h to 03FEh	Uncompensated voltage suspected to be below valid operating range for this function; The compensated reading is not guaranteed for accuracy
00h	0002h to 03FEh	Valid compensated acceleration reading

3.2.13 UINT8 TPMS_READ_DYNAMIC_ACCEL_Z (UINT8 u8Filter, UINT8* u8Offset, UINT16* u16UUMA)

- **Description:** This function automatically executes a TPMS_READ_ACCEL_Z measurement with a given initial dynamic offset. If the result is too high or too low, it will change the dynamic offset value and re-execute TPMS_READ_ACCEL_Z until a) the result is valid or b) the result is railed high or low and there are no more offset steps. Offset and uncompensated acceleration inside the UUMA are updated. The function configures an initial delay of 2048 μs.
- **Stack size:** 48 bytes
- **Approx. Duration:** 2950 μs when starting offset is in target; 29050 μs when the offset is 10 steps away.
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** SMI, ADC, internal bond wires.
- **Input Parameters:**
  – UINT8 u8Filter: If non-zero, 250 Hz filter enabled. Otherwise, 500 Hz filter selected.
  – UINT8* u8Offset: Pointer to initial offset step to load. Valid offset steps range from 0 - 15 and are described in the devices data sheet. An updated offset value is returned at the end of the function. In case the acceleration is too high or too low and function has run out of offset steps, a value of 255 ("0 - 1") or 16 ("15 + 1") shall be returned.
  – UINT16 *u16UUMA: Pointer to the Universal Uncompensated Measurement Array. Uncompensated acceleration will be updated accordingly.
- **Returns:**
  – UINT8 u8Status: See Section 3.2.11 for more information on the format of this status byte.

3.2.14 UINT8 TPMS_READ_V0 (UINT16 *u16Result, UINT8 u8Avg)

- **Description:** Performs a 10-bit uncompensated measurement at pin PTB0.
- **Stack size:** 24 bytes
- **Approx. Duration:** 102 µs
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** ADC, PTB0.
- **Input Parameters:**
  - UINT16 *u16Result: Updated 10-bit uncompensated measurement.
  - UINT8 u8Avg: Number of measurements to average into one result. The value can be set to 1, 2, 4, 8, or 16.
- **Returns:**
  - UINT8 u8Status: Valid error flags/outputs are described in Table 24.

Table 24. Valid output conditions for TPMS_READ_V0 and TPMS_READ_V1

u8Status Value	Measurement value	Condition
01h	0000h	Reading not acquired
00h	Between 0000h - 03FFh	Valid reading

3.2.15 UINT8 TPMS_READ_V1 (UINT16 *u16Result, UINT8 u8Avg)

- **Description:** Performs a 10-bit uncompensated measurement at pin PTB1.
- **Stack size:** 24 bytes
- **Approx. Duration:** 103 µs
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake-up from Stop mode.
- **Resources:** ADC, PTB1.
- **Input Parameters:**
  - UINT16 *u16Result: Updated 10-bit uncompensated measurement.
  - UINT8 u8Avg: Number of measurements to average into one result. The value can be set to 1, 2, 4, 8, or 16.
- **Returns:**
  - UINT8 u8Status: Valid error flags/outputs are described in Table 24.

3.2.16 UINT8 TPMS_LFOCAL (void)

- **Description:** Performs PWU clock calibration. The wake-up and periodic reset time can be calibrated more accurately by using the TPMS_LFOCAL firmware subroutine. This subroutine turns on the RFM crystal oscillator and feeds a 500 kHz clock via the DX signal to the TPM1 for one cycle of the LFO, but first executes a test to verify the presence of the external XTAL. The measured time is used to calculate the correct value for the WDIV0:5 bits for a WCLK period of 1 second. The resulting value for use in the WDIV0:5 bits is returned by the function. The user can decide whether to load the value to the WDIV0:5 bits or store for future reference. In case the returned value is out-of-range, for example when the LFO is out of spec, the returned value will be truncated to the minimum or the maximum possible (0h or 3Fh). The TPMS_LFOCAL subroutine cannot be used while the RFM is transmitting or the TPM1 is being used for another task. This routine will also consume more power due to the crystal oscillator running. This function accesses and writes data to the

SIMOPT2 register. Because some of the bits in this register are write-once-only, it should be configured prior to calling this routine.
- **Stack size:** 11 bytes
- **Approx. Duration:** 1580 µs
- **Power Management:** This function executes entirely in RUN mode. It requires the MCU to be configured for 4 MHz bus clock, and the RFM to be enabled but not transmitting prior to making the call.
- **Interrupt Management:** This function does not await any interrupts. It WILL be affected by interrupts.
- **Resources:** TPM, SIMOPT2, RFM
- **Input Parameters:**
  – None
- **Returns:**
  – UINT8 u8WDIV: WDIV compensated value, or 80h if the XTAL was not found.

Note: *This routine writes to SIMOPT2. Any configuration involving this register must be performed before calling this routine. Prior to calling this routine, the RFM must be turned on and TPMS_PRECHARGE_EN function called. The execution of this routine will change the contents of RFM registers. Specifically note that RF Direct Mode will be selected after its execution and TIMEOUT[1:0] bits in RFPRECHARGE register set to value 2.*

3.2.17 TPMS_LFOCAL_BUSCLK

- **Description:** This function returns the calculated WDIV value for 1 sec base time PWU period. It uses the internal clock instead of the external RFM oscillator to measure the duration of one LFO cycle.
- **Stack size:** 11 bytes
- **Approx. Duration:** 1919 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts.
- **Resources:** None.
- **Input Parameters:**
  – None.
- **Returns:**
  – Calculated WDIV value

3.2.18 UINT8 TPMS_MFOCAL (void)

- **Description:** Performs MFO cross-check verification. This function will measure the bus clock relative to Dx, but first executes a test to verify the presence of the external XTAL. When error is zero, it returns 128. Any deviation from this value should be considered an error. This result can then be used to estimate the error in the RFBT setting. The TPMS_MFOCAL subroutine cannot be used while the RFM is transmitting or the TPM1 is being used for another task. This function accesses and writes data to the **SIMOPT2** register. Because some of the bits in this register are write-once-only, it should be configured prior to calling this routine.
- **Stack size:** 11 bytes
- **Approx. Duration:** 1803 µs
- **Power Management:** This function executes entirely in RUN mode. It requires the MCU to be configured for 4 MHz bus clock, and the RFM to be enabled, but not transmitting prior to making the call.
- **Interrupt Management:** This function does not await any interrupts. It WILL be affected by interrupts.

- **Resources:** TPM, SIMOPT2, RFM
- **Input Parameters:**
  - None
- **Returns:**
  - UINT8 u8Error: 128 when no error is found. Each LSB away from this value is equal to a 0.78 % error. For example, if u8Error = 125, the bus clock has a –2.34 % error, or is running at 3.9064 MHz. 255 is reserved as an error code for when the external XTAL is not present.

Note: *This routine writes to SIMOPT2. Any configuration involving this register must be performed before calling this routine. Prior to calling this routine, the RFM must be turned on and TPMS_PRECHARGE_EN function called.The execution of this routine will change the contents of the RFM registers. Specifically note that RF Direct Mode will be selected after its execution and TIMEOUT[1:0] bits in RFPRECHARGE register set to value 2.*

3.2.19 UINT16 TPMS_WAVG (UINT8 u8PNew, UINT16 u16POld, UINT8 u8PAvg)

- **Description:** This subroutine calculates a new weighed average value for a given new and old measurement readings by using Equation 1.

$$\mathrm{u16NewAverage} = \left(\left(\mathrm{u16POld} \times \left(\mathrm{u8Avg} - 1\right) + \mathrm{u8PNew}\right) / \left(\mathrm{u8Avg}\right)\right) \tag{1}$$

- **Stack size:** 12 bytes
- **Approx. Duration:** 40.52 μs (average of 2), 44.57 μs (average of 4), 49.5 μs (average of 8), 54.1 μs (average of 16), 58.5 μs (average of 32).
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** N/A
- **Input Parameters:**
  - UINT8 u8Avg: Weight of the average. This value can be 2, 4, 8, 16, 32; any other value will return an incorrect response.
  - UINT16 u16Pold: Old average.
  - UINT8 u8PNew: New value to include in average.
- **Returns:**
  - UINT16 u8NewAverage: resulting weighed average of both old average and the new value (See Equation 1).

Note: *This function is not correctly implemented. The function exists and the user may execute the function but the function will not return the expected data described above. The returned, incorrect, data must not be used by the application and the user should avoid using this function.*

3.2.20 void TPMS_RF_ENABLE (UINT8 u8Switch)

- **Description:** This function enables or disables the RF module in the NTM88 and transfers adequate PLL trim data to the module. It should be called prior to any other RF operation.
- **Stack size:** 4 bytes
- **Approx. Duration:** 364 μs when turning on; 11.2 μs when turning off.
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will be affected by interrupts.

- **Resources:** SIMOPT1, RFM
- **Input Parameters:**
  – UINT8 u8Switch: Enable (non-zero) or disable (zero) RFM.
- **Returns:**
  – void.

Note: *This routine writes to SIMOPT1. Any configuration involving this register must be performed before calling this routine.*

3.2.21 void TPMS_RF_RESET (void)

- **Description:** This function sends a master reset to the RFM and reloads PLL trim values into the module. It requires the RFM to have been enabled previously.
- **Stack size:** 3 bytes
- **Approx. Duration:** 221 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** RFM
- **Input Parameters:**
  – None
- **Returns:**
  – void

3.2.22 void TPMS_RF_READ_DATA (UINT8 u8Size, UINT8 *u8RAMBuffer, UINT8 u8RFMBuffer)

- **Description:** This function reads several consecutive bytes from the dedicated RFM buffer registers and copies them to a given address in RAM. It assumes that BUFF0 is location "0". The data is transferred from the LSB bit of the RFM data registers to the LSB of the target memory address (standard data bit order). This function manages the RFM's buffer paged memory.
  – In case the required buffer address is out of bounds, the routine will return "0" for that location.
- **Stack size:** 9 bytes
- **Approx. Duration:** 196 µs (for 8 bytes, switching pages included).
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** RFM
- **Input Parameters:**
  – UINT8 u8Size: Number of bytes to read.
  – UINT8 *u8RamBuffer: Target memory location.
  – UINT8 u8RFMBuffer: Buffer register (0 to 31) to read.
- **Returns:**
  – void

3.2.23 void TPMS_RF_READ_DATA_REVERSE (UINT8 u8Size, UINT8 *u8RAMBuffer, UINT8 u8RFMBuffer)

- **Description:** This function reads several consecutive bytes from the dedicated RFM buffer registers and copies them to a given address in RAM. It assumes that BUFF0 is location "0". The data is transferred from the LSB bit of each byte of the RFM data registers to the MSB of each of the bytes of the target memory address (reversed data bit order). This function manages the RFM's buffer paged memory.
  – In case the required buffer address is out of bounds, the routine will return "0" for that location.

- **Stack size:** 10 bytes
- **Approx. Duration:** 236 μs (for 8 bytes, switching pages included).
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** RFM
- **Input Parameters:**
  – UINT8 u8Size: Number of bytes to read.
  – UINT8 *u8RamBuffer: Target memory location.
  – UINT8 u8RFMBuffer: Buffer register (0 to 31) to read.
- **Returns:**
  – void

3.2.24 void TPMS_RF_WRITE_DATA (UINT8 u8Size, UINT8 *u8RAMBuffer, UINT8 u8RFMBuffer)

- **Description:** This function copies several consecutive bytes from RAM into the dedicated RFM Output Buffer. It assumes that BUFF0 is location 0. The data is transferred from the LSB bit of RAM to the LSB of the RFM data register (standard data bit order). This function manages the RFM's buffer paged-memory.
  – In case the destination buffer address is out of bounds, the register value will not be written.
- **Stack size:** 12 bytes
- **Approx. Duration:** 151 μs (for 8 bytes, switching pages included).
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** RFM
- **Input Parameters:**
  – UINT8 u8Size: Number of bytes to write.
  – UINT8 u8RAMBuffer: Source memory location.
  – UINT8 u8RFMBuffer: Starting buffer register (0 to 31) to write.
- **Returns:**
  – void

3.2.25 void TPMS_RF_WRITE_DATA_REVERSE (UINT8 u8Size, UINT8 *u8RAMBuffer, UINT8 u8RFMBuffer)

- **Description:** This function copies several consecutive bytes from RAM into the dedicated RFM Output Buffer. It assumes that BUFF0 is location 0. The data is transferred from the LSB bit of each byte in RAM to the MSB of each byte in the RFM data register (reversed data bit order). This function manages the RFM's buffer paged-memory.
  – In case the destination buffer address is out of bounds, the register value will not be written.
- **Stack size:** 11 bytes
- **Approx. Duration:** 204 μs (for 8 bytes, switching pages included).
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** RFM
- **Input Parameters:**
  – UINT8 u8Size: Number of bytes to write.
  – UINT8 u8RAMBuffer: Source memory location.
  – UINT8 8uRFMBuffer: Starting buffer register (0 to 31) to write.
- **Returns:**
  – void

3.2.26 void TPMS_RF_CONFIG_DATA (UINT16 *u16RFParam)

- **Description:** This function configures the RFM for transmission. It does not configure inter-frame wait times, which must be configured manually.
- **Stack size:** 4 bytes
- **Approx. Duration:** 32 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** RFM
- **Input Parameters:**
  – UINT16* u16RFParam Format as described in Table 25.
- **Returns:**
  – void

Table 25. u16RFParam array format

Index	Description
0	See Table 26 for description
1	PLLA value
2	PLLB value

Table 26. Description of element 0 in the u16RFParam array

Bits	Description
15:8	Prescaler value. Described in data sheets as RFCR0.
7	End Of Message- If '1', EOM is set, if '0', it is not set.
6	Polarity Bit - If '1', polarity is inverted, If '0', it is non-inverted.
5:4	Not used.
3:2	Encoding value.
1	Frequency selection - If '1', RFM is configured for 434 MHz, if '0', it is configured for 315 MHz.
0	Modulation - If '1', RFM is configured for FSK, if '0' it is configured for OOK.

3.2.27 void TPMS_RF_SET_TX (UINT8 u8BufferSize)

- **Description:** This function allows the RFM to transmit data previously loaded in the buffer. It should be called after the RF module has been enabled and configured.
- **Stack size:** 4 bytes
- **Approx. Duration:** 13 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** RFM
- **Input Parameters:**
  – UINT8 u8BufferSize: Number of bits in the buffer –1 (for example to transmit 1 bit, u8BufferSize should equal 0).
- **Returns:**
  – void

3.2.28 void TPMS_READ_ID (UINT8 *u8Code)

- **Description:** Copies the device's UniqueID and firmware version stored in firmware flash to RAM.
- **Stack size:** 2 bytes
- **Approx. Duration:** 17 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** N/A
- **Input Parameters:**
  – UINT8 *u8Code: RAM location where data will be copied. Table 27 describes the format of the 6 bytes returned.

Table 27. u8Code format

Index	Description
0	Library version
1	Derivative descriptor
2 to 5	32-bit UniqueID

- **Returns:**
  – void

3.2.29 void TPMS_RF_DYNAMIC_POWER (UINT8 u8CompT, UINT8 u8CompV, UINT8* pu8PowerManagement)

- **Description:** Depending on the passed parameters, this function can:
  – Force the RF power setting (RFCR2_PWR) to a passed value (when BIT5 of u8PowerManagement is clear).
  – Set the RF power setting (RFCR2_PWR) dynamically based on voltage, temperature, and current carrier frequency (when BIT5 of u8PowerManagement is set). The target output level is 3 dBm across all voltages and temperatures, with some small variations. When this option is engaged, the routine limits settings to valid PWR settings - if the resulting value is above maximum allowed setting, the setting is set to maximum; if the resulting value is less than minimum allowed setting, the setting is set to the minimum.
  – When BIT5 of u8PowerManagement is set, find the best RF power setting (RFCFR2_PWR) dynamically based on voltage, temperature, and current carrier frequency in order to target 3 dBm as actual output power. This value of 3 dBm can be increased or decreased in given temperature ranges using the offsets (0.5 dBm/count) in the pu8PowerManagement array.
  – Similar to the case above, the user can specify a target power region with an offset.
- **Stack size:** 21 bytes
- **Approx. Duration:** 140 µs when using voltage, temperature; 22 µs when the power step is passed.
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** RFM
- **Input Parameters:**
  – UINT8 u8CompT: Compensated temperature reading.
  – UINT8 u8CompV: Compensated voltage reading.
  – UINT8* pu8PowerManagement: This is a pointer to an array as described in Table 28 and Table 29:
- **Returns:**
  – void

Note: *The RF module must be turned on prior to calling this routine.*

Table 28. *pu8PowerManagement format

Index value	Description
0	Dynamic compensation switch as described in Table 29.
1	Offset step for power target when temperature is higher than 92 °C. Negative values admitted.
2	Offset step for power target when temperature is lower than 92 °C and higher than 60 °C. Negative values admitted.
3	Offset step for power target when temperature is lower than 60 °C and higher than 43 °C. Negative values admitted.
4	Offset step for power target when temperature is lower than 43 °C and higher than 25 °C. Negative values admitted.
5	Offset step for power target when temperature is lower than 25 °C and higher than 0 °C. Negative values admitted.
6	Offset step for power target when temperature is lower than 0 °C and higher than –20 °C. Negative values admitted.
7	Offset step for power target when temperature is lower than –20 °C. Negative values admitted.

Table 29. pu8PowerManagement format

Bit	Description
MSB	Not used.
BIT6	Not used.
BIT5	Dynamic compensation enable. If set, the function will decide what the optimal power setting is based on voltage and temperature; In this case, values stored in the pu8PowerManagement array, and corresponding to the temperature range will be added to the found target. If clear, BIT4:0 will be used to set the power level directly.
BIT4:0	When BIT5 is clear, the value passed here will be used to set the RF power step in the RFCR2 register directly.

3.2.30 void TPMS_MSG_INIT (void)

- **Description:** This function is to be called before using any MSG routine. It initializes PTA1 and PTA0 to their correct initial state for a simulated SPI.
- **Stack size:** 2 bytes
- **Approx. Duration:** 9 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** Pins PTA1 and PTA0
- **Input Parameters:**
  – None
- **Returns:**
  – void

3.2.31 UINT8 TPMS_MSG_WRITE (UINT8 u8SendByte)

- **Description:** This function is in charge to write a message at a network level via an emulated serial interface on PTA1 and PTA0. As the master, the NTM88 manages the clock on PTA1. On rising edge of the clock, the module puts down a new data bit on PTA0 (programmed as output), MSB first.
- **Stack size:** 2 bytes
- **Approx. Duration:** 78 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** Pins PTA1 and PTA0
- **Input Parameters:**
  - UINT8 u8SendByte: Byte to be outputted through the emulated serial interface
- **Returns:**
  - UINT8 u8ReadByte: Incoming byte from the emulated serial interface

3.2.32 UINT8 TPMS_MSG_READ (void)

- **Description:** This function is in charge to read any incoming message at a network level via an emulated serial interface on PTA1 and PTA0. As the master, the NTM88 manages the clock on PTA1. On falling edge of the clock, the module reads a new data bit on PTA0 (programmed as input), MSB first.
- **Stack size:** 2 bytes
- **Approx. Duration:** 78 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** Pins PTA1 and PTA0.
- **Input Parameters:**
  - None
- **Returns:**
  - UINT8 u8ReadByte: Incoming byte from the emulated serial interface

3.2.33 UINT8 TPMS_CHECKSUM_XOR (UINT8 *u8Buffer, UINT8 u8Size, UINT8 u8Checksum)

- **Description:** Calculates a checksum for the given buffer based on XOR operations.
- **Stack size:** 5 bytes
- **Approx. Duration:** 78 µs for 8 bytes of data.
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** N/A
- **Input Parameters:**
  - UINT8 *u8Buffer: Buffer where data is located.
  - UINT8 u8Size: Size of buffer (in bytes).
  - UINT8 u8Checksum: Previous checksum. This argument is useful when the function is used recursively. It must equal "0" if there is no previous data.
- **Returns:**
  - UINT8 u8NewChecksum: New calculated checksum.

3.2.34 UINT8 TPMS_CRC8 (UINT8 *u8Buffer, UINT8 u8Poly, UINT8 u8MBitSize, UINT8 u8Remainder)

- **Description:** Calculates a CRC8 on a portion of the designated area.

- **Stack size:** 12 bytes
- **Approx. Duration:** 780 µs for 8 bytes (64 bits) of data.
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** N/A
- **Input Parameters:**
  - UINT8 *u8Buffer: Buffer where data is located.
  - UINT8 u8Poly: Polynomial to be used for calculating the CRC8.
  - UINT8 u8MBitSize: Size of the designated buffer (in bits).
  - UINT8 u8Remainder: Initial remainder. This argument is useful when the function is used recursively. It must equal "0" if there is no previous data.
- **Returns:**
  - UINT8 u8NewCRC: New calculated CRC8.

3.2.35 UINT16 TPMS_SQUARE_ROOT (UINT16 u16Process)

- **Description:** Calculates a two-digit remainder of (square root * 10) using a fast algorithm.
- **Stack size:** 49 bytes
- **Approx. Duration:** 362 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** N/A
- **Input Parameters:**
  - UINT16 u16Process: The number from which to get the square root from.
- **Returns:**
  - UINT16 Root of the number * 10.

Note: *The result may include an error due to truncation in the calculation.*

3.2.36 void TPMS_LF_ENABLE (UINT8 u8Switch)

- **Description:** Enables/disables the LFR module; Loads best-case-known LF settings for NXP-only LF registers by calling vnfLFRConfigFactoryRegs() function described in Section 2.5.
- **Stack size:** 5 bytes
- **Approx. Duration:** 30 µs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will not be affected by interrupts.
- **Resources:** LFR
- **Input Parameters:**
  - UINT8 u8Switch: Enable (non-zero) or disable (zero) LFR.
- **Returns:**
  - void

3.2.37 UINT8 TPMS_LF_READ_DATA (UINT8 *u8Buffer, UINT8 u8Count)

- **Description:** Once the user has configured and enabled the LFR, it is customary to go into a low-power state mode and wait for a datagram. After the first byte of an incoming datagram is successfully received, this function should be called immediately. The function will receive the complete datagram and place it in RAM. Be careful to call the function upon reception of the first data byte (LFDRF flag) and not upon detection of the ID (LFIDF flag) in case the LFIDIE is enabled. This function assumes that the LFR module is configured accordingly for a Manchester reception, that the module's interrupts are enabled, and that the first byte has already been received and is waiting in the LFR received buffer. While waiting for the next byte, this function goes into STOP4. If the byte, for an unexpected reason, is not received, this function has the following built-in timeout: After five continuous non-LFR interrupts, the function will assume a failed LFR reception and exit. In order to leave the routine as soon as possible after reception of all the data bytes, NXP recommends enabling the LF error interrupt (LFERIE). In summary, the two necessary interrupts to be enabled are LFDRIE and LFERIE.

- **Description:**
- **Stack size:** 7 bytes
- **Approx. Duration:** Data dependent; ~2 ms per byte received.
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the LFR interrupt to wake up from Stop mode.
- **Resources:** LFR
- **Input Parameters:**
  – UINT8 *u8Buffer: RAM Buffer where data will be placed.
  – UINT8 u8Count: Number of bytes expected.
- **Returns:**
  – UINT8 u8BytesReceived: Actual number of bytes received.

Note: *This function requires ~24 µs from the moment it is called to the moment the first byte is copied into the RAM buffer. The user must consider this time when designing their firmware.*

Note: *This function should be called only after the first byte of the incoming datagram has been received. If it is called before receiving the first byte, the value returned by the function is unpredictable.*

3.2.38 UINT8 TPMS_WIRE_AND_ADC_CHECK (UINT8 u8TestMask)

- **Description:** This function will check if there is any bonding wire failure between the embedded core and the P-cell; or between the core and the g-cell. It will also perform an ADC test, which consists of taking two reference measurements, ground and V_{DD}, using internal channels and comparing them with the expected results. It can only be called when the device is in parking or static mode. When configuring for a P-cell or g-cell wire check, interrupts must be enabled before calling this routine. In case of no issues found, 0 will be returned, else it will set status flags as follows:
  – On P-cell wire-bond error, sets pressure error flag.
  – On g-cell wire-bond error, sets acceleration error flag.
  – On ADC error, sets the ADCERR flag.
- **Stack size:** Up to 37 bytes
- **Approx. Duration:** 9815 µs (all checks), 121 µs (ADC only), 3,296 µs (P-cell only), 3227 µs (X-cell or Z-cell only).
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** ADC, SMI (for g-cell, P-cell checks), internal bond wires
- **Input Parameters:**
  – UINT8 u8TestMask: This variable determines what checks are performed as described by Table 30.

Table 30. u8TestMask format

u8TestMask Bit	Description
1	Reserved
2	If set, P-cell wire-bond check performed

Table 30. u8TestMask format*...continued*

u8TestMask Bit	Description
3	If set, g- Xcell wire-bond check performed
4	If set, g-Zcell wire-bond check performed
5 to 6	Reserved
7	If set, ADC check performed

Note: *Users should not set bits 0, 1, 5, or 6. The results returned, according to Table 31, may be ambiguous including indicating false passing or false failing conditions.*

- **Returns:**
  – UINT8 u8Status: Status flags as described in Table 31.

Table 31. u8Status valid values for TPMS_WIRE_AND_ADC_CHECK

u8TestMask Bit	Description
0	Always clear
1	Always clear
2	If set, P-cell wire-bond error detected
3	if set, g-xcell error is returned
4	if set, g-zcell error is returned
5 to 6	Always clear
7	If set, ADC error detected

3.2.39 void TPMS_FLASH_WRITE (UINT16 u16Address, UINT8* u8Buffer, UINT8 u8Size)

- **Description:** This function writes consecutive bytes from a given address in memory to a specified location in FLASH. Addresses $FD40 - $FDFF are not valid targets in this function
- **Stack size:** 15 bytes
- **Approx. Duration:** 1270 μs for 8 bytes of data.
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It will be affected by interrupts.
- **Resources:** 59 bytes of global RAM locations
- **Input Parameters:**
  – UINT16 u16Address: Flash starting address
  – UINT8 *u8Buffer: Source memory address
  – UINT8 u8Size: Number of data bytes to be written. Must be strictly greater than zero.
- **Returns:**
  – void

Note: *This routine will overwrite the contents of 59 bytes of RAM allocated to FLASHING_ALGO_LOCATION_IN_RAM section. In NXP demo projects, this section is mapped to addresses 0090h to 00CAh.*

3.2.40 UINT8 TPMS_FLASH_ERASE (UINT16 u16Address)

- **Description:** This function erases 1 page (512 bytes) of flash at a time. Addresses $FC00 - $FDFF are not valid targets in this function.
- **Stack size:** 11 bytes

- **Approx. Duration:** 22,750 μs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It may be affected by interrupts.
- **Resources:** 59 bytes of global RAM locations.
- **Input Parameters:**
  – UINT16 u16Address: any given address. The whole page where this address resides will be erased (for example, if u16Address = D234h, the contents of addresses D200h - D3FFh will be erased).
- **Returns:**
  – Zero if the page was erased successfully; else, one.

Note: *This routine will overwrite the contents of 59 bytes of RAM allocated to FLASHING_ALGO_LOCATION_IN_RAM section. In NXP demo projects, this section is mapped to addresses 0090h to 00CAh.*

3.2.41 void TPMS_MULT_SIGN_INT16 (INT16 i16Mult1, INT16 i16Mult2, INT32* pi32Result)

- **Description:** This function will multiply two signed 16-bit numbers together.
- **Stack size:** 17 bytes
- **Approx. Duration:** 68.1 μs
- **Power Management:** This function executes entirely in RUN mode.
- **Interrupt Management:** This function does not await any interrupts. It should not be affected by interrupts.
- **Resources:** N/A
- **Input Parameters:**
  – INT16 i16Mult1: First multiplier
  – INT16 i16Mult2: Second multiplier
  – INT32* pi32Result: Pointer to a 32-bit variable where the result will be stored.
- **Returns:**
  – void.

3.2.42 UINT8 TPMS_VREG_CHECK (UINT8 u8WaitTime, UINT16 u16LimitDelta)

- **Description:** This function will verify if the part has a capacitor properly connected on the VReg pin. This is done by starting an RF transmission in Buffer mode, once the transmission is done, taking a first ADC reading of the VReg pin, awaiting a pre-established amount of time, then taking a second ADC reading of the VReg pin. The difference between the two readings is then calculated. If it stays below a certain limit it means the capacitor is properly connected to the VReg pin.
- **Stack size:** 30 bytes
- **Approx. Duration:** 28,500 μs using default values; time will vary depending on user input.
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes an ADC interrupt to wake-up from Stop mode.
- **Resources:** TPM, RFM, SPMSC2.
- **Input Parameters:**
  – UINT8 u8WaitTime: Amount of time to wait between the two ADC readings (in ms). If zero, it is assumed that a 470 nF capacitor is being used and the default wait time of 25 ms for this capacitor is used.
  – UINT8 u8LimitDelta: This value (in ADC counts) will determine whether the V_{REG} pin passes the test or it does not. It is dependent on the capacitor value and on the value of u8WaitTime, and must be obtained

through characterization. If zero, it is assumed that a 470 nF capacitor is being used and the default limit of $50 is used.

- **Returns:**
  - UINT8 u8Status: If clear, the function has detected good contact with the capacitor; if one, the capacitor has failed the test.

Note: *Write-once register SPMSC2 will be used. Also note that calling this function will start an RF transmission for ~3 ms. Previously set RF settings, such as carrier frequency and PLL dividers, are respected in this short RF burst. Before exiting this function, the RF module will be shut down.*

3.2.43 UINT8 TPMS_E_READ_ACCEL_X (UINT16* pu16UUMA, UINT8 u8Avg, UINT8 u8Filter, UINT8 u8OffsetIndex, UINT8 u8delay)

- **Description:** This function is an extension of TPMS_READ_ACCEL_X which takes sampling delay as application options.
- **Stack size:** 37 bytes
- **Approx. Duration:**

Table 32. Execution time for TPMS_E_READ_ACCEL_X

Initial delay	2048 µs	1024 µs	512 µs
Execution time for u8Avg = 1	2.37 ms	1.35 ms	0.852 ms

u8Avg	1	2	3
Execution time for initial delay = 2048 µs and Subsequent delay = 128 µs	2.37 ms	2.54 ms	2.86 ms

- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** SMI, ADC, Internal bond wires.
- **Input Parameters:**
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3. Only the 12-bit uncompensated acceleration result will be updated.
  - UINT8 u8Avg: Number of measurements to average into one result. The value can be set to 1, 2, 4, 8, or 16.
  - UINT8 u8Filter: Will set up the acceleration measurement based on Table 17.
  - UINT8 u8OffsetIndex: Selects the offset setting normal dynamic offset mode, default index is 7. Valid range is 0 to 15.
  - UINT8 u8Delay: composition of initial and subsequent delay, for example, SAMP_DLY_SUBS_512US| SAMP_DLY_INIT_2048US. See Table 7 and Table 8.
- **Returns:** UINT8 u8Status: Valid error flags/outputs are described in Table 18.

3.2.44 UINT8 TPMS_E_READ_PRESSURE (UINT16* pu16UUMA, UINT8 u8Avg, UINT8 u8Filter, UINT8 u8Delay)

- **Description:** This function is an extension of TPMS_READ_PRESSURE which takes the sampling delay as an application option.
- **Stack size:** 36 bytes
- **Approx. Duration:** 219 µs

Table 33. Execution time for TPMS_E_READ_PRESSURE

Initial delay	2048 µs	1024 µs	512 µs
Execution time for u8Avg = 1	2.37 ms	1.35 ms	0.852 ms

u8Avg	1	2	3
Execution time for initial delay = 2048 µs and Subsequent delay = 128 µs	2.37 ms	2.54 ms	2.86 ms

- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** SMI, ADC, Internal bond wires
- **Input Parameters:**
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3. Only the 12-bit uncompensated acceleration result will be updated.
  - UINT8 u8Avg: Number of measurements to average into one result. The value can be set to 1, 2, 4, 8, or 16.
  - UINT8 u8Filter: Filter setting based on Table 17.
  - UINT8 u8Delay: composition of initial and subsequent delay, for example, SAMP_DLY_SUBS_512US|SAMP_DLY_INIT_2048US. See Table 7 and Table 8.
- **Returns:** UINT8 u8Status: Valid error flags/outputs are described in Table 14.

3.2.45 UINT8 TPMS_E_READ_ACCEL_Z (UINT16 pu16UUMA, UINT8 u8Avg, UINT8 u8Filter, UINT8 u8OffsetIndex, UINT8 delay)

- **Description:** This function is an extension of TPMS_READ_ACCEL_Z which takes sampling delay as application options.
- **Stack size:** 37 bytes
- **Approx. Duration:**

Table 34. Execution time for TPMS_E_READ_ACCEL_Z

Initial delay	2048 µs	1024 µs	512 µs
Execution time for u8Avg = 1	2.37 ms	1.35 ms	0.852 ms

u8Avg	1	2	3
Execution time for initial delay = 2048 µs and Subsequent delay = 128 µs	2.37 ms	2.54 ms	2.86 ms

- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** SMI, ADC, Internal bond wires
- **Input Parameters:**
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3. Only the 12-bit uncompensated acceleration result will be updated.
  - UINT8 u8Avg: Number of measurements to average into one result. The value can be set to 1, 2, 4, 8, or 16.
  - UINT8 u8Filter: Will set up the acceleration measurement based on Table 17.

- UINT8 u8OffsetIndex: Selects the offset setting normal dynamic offset mode, default index is 7. Valid range is 0 to 15.
- UINT8 u8Delay: composition of initial and subsequent delay, for example, SAMP_DLY_SUBS_512US|SAMP_DLY_INIT_2048US. See Table 7 and Table 8.

- **Returns:** UINT8 u8Status: Valid error flags/outputs are described in Table 14.

3.2.46 void TPMS_E_FRC_ENABLE (UINT8 u8ClrRes)

- **Description:** This function enables Free Running Counter.
  - **Stack size:** 3 bytes
  - **Approx. Duration:**1.3 ms
  - **Power Management:** This function executes entirely in RUN.
  - **Interrupt Management:** None
  - **Resources: FRC**
  - **InputParameters:**
    - UINT8 u8ClrRes – clear or not the FRC counter
    - 0 – clear the FRC counter,
    - not 0 – FRC counter is not cleared.
  - **Returns:** void

3.2.47 void TPMS_E_FRC_DISABLE (void)

- **Description:** This function disables the Free Running Counter.
- **Stack size:** 0 bytes
- **Approx. Duration:** 1 ms
- **Power Management:** This function executes entirely in RUN
- **Interrupt Management:** None
- **Resources:** FRC
- **Input Parameters:** void
- **Returns:** void

3.2.48 void TPMS_E_FRC_CLEAR (void)

- **Description:** This function clears the FRC counter register.
- **Stack size:** 0 bytes
- **Approx. Duration:** 9 µs
- **Power Management:** This function executes entirely in RUN.
- **Interrupt Management:** None.
- **Resources:** FRC
- **Input Parameters:** void
- **Returns:** void

3.2.49 void TPMS_E_FRC_READ (UINT16 *pu16Count)

- **Description:** This function reads the FRC counter register
- **Stack size:** 2 bytes
- **Approx. Duration:** 10 µs
- **Power Management:** This function executes entirely in RUN.
- **Interrupt Management:** None

- **Resources:** FRC
- **Input Parameters:**
  – UINT16 *pu16Count - pointer to memory location to save the FRC count
- **Returns:** void

3.2.50 UINT8 TPMS_E_FRC_CALIB (UINT16 *pu16usPerPrd)

- **Description:**
  – This function calculates the number of microseconds per LFO period using the 500kHz signal coming from the external XTAL as reference. On exit the FRC remains enabled. The function takes care of enabling and disabling the RF block, but does not perform a pre- charge. Users should call the pre-charge function before calling the TPMS_FRC_CALIB function.
- **Stack size:** 23 bytes
- **Approx. Duration:** 9.1 µs
- **Power Management:** This function executes entirely in RUN
- **Interrupt Management:** None
- **Resources:** This function uses the FRC block, the timer module TPM1, and the 500 kHz reference clock generated by RF module. On entry, it expects that TPM1 module is in reset state in free-running timer counter mode with modulo counting disabled. Before exiting, function disables TPM1 and RF module, and keeps the FRC enabled.
- **Input Parameters:** UINT16 *pu16usPerPrd - pointer to memory location to save number of microseconds per LFO period.
- **Returns:** UINT8 u8Status - status/error flag:
  – 0 valid value returned in *pu16usPerPrd
  – 0x80 indicates a XTAL error.
  – Other none 0 – timeout, contents of *pu16usPerPrd is not valid.

Note: *This routine writes to SIMOPT2. Any configuration involving this register must be performed before calling this routine. The execution of this routine will change the contents of the RFM registers. Specifically note that RF Direct Mode will be selected after its execution and TIMEOUT[1:0] bits in RFPRECHARGE register set to value 2.*

3.2.51 UINT8 TPMS_E_FRC_CALIB_BUSCLK (UINT16 *pu16usPerPrd)

- **Description:** This function calculates the number of microseconds per LFO period using the Bus Clock as reference. On exit FRC remains enabled.
- **Stack size:** 18 bytes
- **Approx. Duration:** 5.3 ms
- **Power Management:** This function executes entirely in RUN
- **Interrupt Management:** None
- **Resources:** This function uses the FRC block, the timer module TPM1, and the BUSCLK
- **Input Parameters:** UINT16 *pu16usPerPrd - pointer to memory location to save number of microseconds per LFO period.
- **Returns:** UINT8 - status/error flag:
  – 0 if operation is successful,
  – non-0, indicating the function exited on timeout.

3.2.52 UINT8 TPMS_PRECHARGE_EN (void)

- **Description:** Users should call the TPMS_PRECHARGE_EN function every time the RF block is used in order to reduce the current peak due to the start of the analog regulator. Specifically, call the function before the 500 kHz Dx signal is generated and before starting an RF transmission. This function manages the pre-charge feature and assures the pre-charge and completes with an indicated status. It should be called in place of or in addition to setting the AREGPC bit. The pre-charge cannot be performed by setting AREGPC bit only; this function needs to be called. The RF block must be enabled prior to calling this function.

- **Stack size:** 3 bytes
- **Approx. Duration:** 619 µs, the time varies depending on the number of times retry happens
- **Power Management:** This function executes entirely in RUN
- **Interrupt Management:** None
- **Resources:** RF Block
- **Input Parameters:** None
- **Returns:** UINT8 - status/error flag
  - 0 for Success
  - 1 for the pre-charge does not complete and exceed the number of retry.

Note: *The RF registers are reset to their default values inside the function.*

3.2.53 void TPMS_E_ACTIVATE_CHANNEL (UINT8 u8Filter, UINT8 u8OffsetIndex, UINT8 u8Delay, channel_t eChannel)

- **Description:** This function will activate the channel based on channel selected. It enables the 12 bit ADC with proper configuration, sets the appropriate trim information into the trim register and enables the LPDM and acquisition.
- **Stack size:** 18 bytes
- **Approx. Duration:** 140 µs
- **Power Management:** This function executes entirely in RUN
- **Interrupt Management:** None
- **Resources:** SMI and ADC
- **Input Parameters:**
  - UINT8 u8Filter: Will set up the acceleration or pressure measurement based on Table 17.
  - UINT8 u8OffsetIndex: Selects the offset setting normal dynamic offset mode, default index is 7. Valid range is 0 to 15. This parameter is not used when pressure channel is selected.
  - UINT8 u8Delay: composition of initial and subsequent delay, for example, SAMP_DLY_SUBS_512US|SAMP_DLY_INIT_2048US. See Table 7 and Table 8.
  - Channel_t eChannel: Channel selection, for selecting pressure channel, pass CHANNEL_PRESSURE enum, see Table 35
- **Table 35. eChannel values**

eChannel value	Numerical value
CHANNEL_PRESSURE	0
CHANNEL_Z	1
CHANNEL_X	2

- **Returns:** void

Note: *Putting the device into STOP4 mode after activating the particular channel is the responsibility of application code.*

3.2.54 void TPMS_E_DEACTIVATE_CHANNEL (void)

- **Description:** This function de-activates currently active channel
- **Stack size:** 8 bytes
- **Approx. Duration:** 10 µs

- **Power Management:** This function executes entirely in RUN.
- **Interrupt Management:** None
- **Resources:** SMI and ADC
- **Input Parameters:** void
- **Returns:** void

3.2.55 UINT8 TPMS_E_READ_CONT_ACCEL_X (UINT16* u16UUMA)

- **Description:** This function is to be called after the X-axis channel has been activated. It checks if an ADC interrupt occurred, meaning a new X-axis measurement is available. If so, the measurement value is stored in the UUMA array passed as parameter and the function returns 0 otherwise it returns 1 for an error.
- **Stack size:** 8 bytes
- **Approx. Duration:** 18 µs
- **Power Management:** This function executes entirely in RUN
- **Interrupt Management:** None
- **Resources:** SMI and ADC
- **Input Parameters:**
  - • UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3. Only the 12-bit uncompensated acceleration result will be updated.
- **Returns:** UINT8 0 for success and 1 for an error.

Note: *Data ideally to be read under STOP4. Putting the device into STOP4 mode after activating the X channel is the responsibility of application code.*

3.2.56 UINT8 TPMS_E_READ_CONT_ACCEL_Z (UINT16* u16UUMA)

- **Description:** This function is to be called after the Z-axis channel has been activated. It checks if an ADC interrupt occurred, meaning a new Z-axis measurement is available. If so, the measurement value is stored in the UUMA array passed as parameter and the function returns 0 otherwise it returns 1 for an error
- **Stack size:** 8 bytes
- **Approx. Duration:** 18 µs
- **Power Management:** This function executes entirely in RUN.
- **Interrupt Management:** None
- **Resources:** SMI and ADC
- **Input Parameters:**
  - UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3. Only the 12-bit uncompensated acceleration result will be updated.
- **Returns:** UINT8 0 for success and 1 for an error

Note: *Data ideally to be read under STOP4. Putting the device into STOP4 mode after activating the Z channel is the responsibility of application code.*

3.2.57 UINT8 TPMS_E_READ_CONT_PRESSURE (UINT16* u16UUMA)

- **Description:** This function is to be called after the pressure channel has been activated. It checks if an ADC interrupt occurred, meaning a new pressure measurement is available. If so, the measurement value is stored in the UUMA array passed as parameter and the function returns 0 otherwise it returns 1 for an error.
- **Stack size:** 8 bytes
- **Approx. Duration:** 18 µs
- **Power Management:** This function executes entirely in RUN.
- **Interrupt Management:** None

- **Resources:** SMI and ADC
- **Input Parameters:**
  - • UINT16 *u16UUMA: Pointer to Universal Uncompensated Measurement Array (as described in Section 2.3. Only the 12-bit uncompensated acceleration result will be updated.
- **Returns:** UINT8 0 for success and 1 for an error

Note: *Data ideally to be read under STOP4. Putting the device into STOP4 mode after activating the pressure channel is the responsibility of application code.*

3.2.58 UINT8 TPMS_E_READ_DYNAMIC_ACCEL_Z (UINT16* pu16UUMA, UINT8 u8Filter, UINT8* pu8Offset, UINT8 u8Delay)

- **Description:** This function automatically executes a TPMS_E_READ_ACCEL_Z measurement with a given initial dynamic offset. If the result is too high or too low, it will change the dynamic offset value and re-execute TPMS_E_READ_ACCEL_Z until a) the result is valid or b) the result is railed high or low and there are no more offset steps. Offset and uncompensated acceleration inside the UUMA are updated.
- **Stack size:** 57 bytes
- **Approx. Duration:** 2100 µs when starting offset is in target with initial delay of 2048 µs and subsequent delay of 512 µs ; 21000 µs when the offset is 10 steps away.
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.
- **Resources:** SMI, ADC, internal bond wires.
- **Input Parameters:**
  - UINT16* pu16UUMA, Pointer to the Universal Uncompensated Measurement Array. Uncompensated acceleration will be updated accordingly.
  - UINT8 u8Filter: Will set up the acceleration measurement based on Table 17.
  - UINT8* pu8Offset: Pointer to initial offset step to load. Valid offset steps range from 0 - 15 and are described in the devices data sheet. An updated offset value is returned at the end of the function. In case the acceleration is too high or too low and function has run out of offset steps, a value of 255 ("0 - 1") or 16 ("15 + 1") shall be returned.
  - UINT8 u8Delay: composition of initial and subsequent delay, for example, SAMP_DLY_SUBS_512US| SAMP_DLY_INIT_2048US. See Table 7 and Table 8.
- **Returns:** UINT8 return 0 for success and see Section 3.2.11 for more information on the format of this status byte.

3.2.59 UINT8 TPMS_E_READ_DYNAMIC_ACCEL_X (UINT16* pu16UUMA, UINT8 u8Filter, UINT8* pu8Offset, UINT8 u8Delay)

- **Description:** This function automatically executes a TPMS_E_READ_ACCEL_X measurement with a given initial dynamic offset. If the result is too high or too low, it will change the dynamic offset value and re-execute TPMS_E_READ_ACCEL_X until a) the result is valid or b) the result is railed high or low and there are no more offset steps. Offset and uncompensated acceleration inside the UUMA are updated.
- **Stack size:** 57 bytes
- **Approx. Duration:** 2100 µs when starting offset is in target with initial delay of 2048 µs and subsequent delay of 512 µs ; 21000 µs when the offset is 10 steps away.
- **Power Management:** This function requires the core to be configured for STOP4 mode and running at full bus speed.
- **Interrupt Management:** This function utilizes the ADC interrupt to wake up from Stop mode.

- **Resources:** SMI, ADC, internal bond wires.
- **Input Parameters:**
  - UINT16* pu16UUMA, Pointer to the Universal Uncompensated Measurement Array. Uncompensated acceleration will be updated accordingly.
  - UINT8 u8Filter: Will set up the acceleration measurement based on Table 17.
  - UINT8* pu8Offset: Pointer to initial offset step to load. Valid offset steps range from 0 - 15 and are described in the devices data sheet. An updated offset value is returned at the end of the function. In case the acceleration is too high or too low and function has run out of offset steps, a value of 255 ("0 - 1") or 16 ("15 + 1") shall be returned.
  - UINT8 u8Delay: composition of initial and subsequent delay, for example, SAMP_DLY_SUBS_512US| SAMP_DLY_INIT_2048US. See Table 7 and Table 8.
- **Returns:** UINT8 See Section 3.2.8, TPMS_READ_ACCEL_X for more information on the format of this status byte.

4 Revision history

Table 36. Revision history

Revision number	Date	Description
UM11145 v.7.0	9 October 2024	• Section 2.1, inserted a note after Table 1. • Section 2.1.1, revised "These registers disable the hardware Watch-dog block. " to "When the MCU enters STOP mode, the watch-dog is halted." in the first paragraph. • Section 2.2, "Measurement error format", Section 2.2.1, "Definition of signal ranges", and Section 2.2.2, "Error status format" were removed from the document. • Section 2.2, inserted new section titled "Flash memory allocations". • Section 2.5, Table 4 and Table 5, removed "TMPS7 and" and "TPMS and" in the title of each table. • Section 3.1, Table 6, revised as follows: – Function TPMS_FRC_ENABLE: revised the function "TPMS_FRC_ENABLE" to "TPMS_E_FRC_ENABLE". – Function TPMS_FRC_DISABLE: revised the function "TPMS_FRC_DISABLE" to "TPMS_E_FRC_DISABLE". – Function TPMS_FRC_CLEAR: revised the function "TPMS_FRC_CLEAR" to "TPMS__E_FRC_CLEAR". – Function TPMS_FRC_READ: revised the function "TPMS_FRC_READ" to "TPMS_E_FRC_READ". – Function TPMS_FRC_CALIB: revised the function "TPMS_FRC_CALIB" to "TPMS_E_FRC_CALIB". – Function TPMS_FRC_CALIB_BUSCLK: revised the function "TPMS_FRC_ CALIB_BUSCLK" to "TPMS_E_FRC_CALIB_BUSCLK". • Section 3.2.3, revised as follows: – Description: revised the first bullet from "If V_{out} < 2.1 V, u8Voltage will be 1..." to "If V_{out} < 1.7 V, result will be 1...." – Description: revised the third bullet from "If V_{out} ≥ 3.75 V, result..." to "If V_{out} ≥ 3.7 V...." – Input parameters: revised the first bullet from "UINT8 *u8Voltage:…" to "UINT8 *u8CompVoltage:..." – Table 10, revised the measurement value "01h to FEh" to "02h to FEh". • Section 3.2.4, added "The LVWF flag is cleared before the function returns." to the end of the first paragraph of the description. • Section 3.2.5, revised "If T_{out} < –40 °C,…" to "If T_{out} < –50 °C,…." in the first bullet of the description and revised the measurement value "01h to FEh" to "02h to FEh" in Table 12. • Section 3.2.6, revised the description inserting "The function configures an initial delay of 2048 μs and a subsequent delay of 512 μs." after the first sentence and added "The LVWF flag is cleared before the function returns." to the end of the first paragraph. • Section 3.2.7, revised as follows: – Description, second bullet: revised "If P_{out} < 100 kPa, the..." to "If P_{out} < 90 kPa, the..." – Input parameters: Revised the first bullet from "UINT16 *u16Pressure" to "UINT16 u16CompPressure". – Removed the last bullet that read "For repeatability data, refer to the NTM88 family data sheets."

Table 36. Revision history

Revision number	Date	Description
		– Table 15, revised the measurement value of 20h from "0001h to 03FEh" to "0002h to 03FEh" and revised the measurement values of 00h from "0001h to 03FEh" to "0002h to 03FEh". • Section 3.2.8, revised as follows: – Section title: revised the section title from "UINT8 TPMS_READ_ACCEL_X (UINT16 *u16UUMA, UINT8 u8Avg, UINT8 u8ModeSelect, UINT8 u8Dynamic Offset)" to "UINT8 TPMS_READ_ACCEL_X (UINT16 *u16UUMA, UINT8 u8Avg, UINT8 u8Filter, UINT8 u8OffsetIndex)". – Description: revised the description inserting "The function configures an initial delay of 2048 µs and a subsequent delay of 512 µs." after the first sentence and inserted "The LVWF flag is cleared before the function returns." to the end of the first paragraph. – Table 17: revised the table title from "u8ModeSelect options" to "u8Filter options", revised the table headers from "u8ModeSelect Value" to "u8Filter Value" and "Selected mode" to "Selected filter" and removed "..., normal dynamic range" from each row. – UINT8 u8OffsetIndex: Revised "Selects the offset setting for the appropriate acceleration reading depending on what bit 1 of u8ModeSelect is. For normal dynamic offset mode, default index is 6. Valid range is 0 to 15." to "Selects the offset setting for the appropriate acceleration reading. The default index is 7. Valid range is 0 to 15." • Section 3.2.9, revised the first input parameter from "UINT16 *u16Accel:" to "UINT16 *u16CompAccel:" and in Table 19, revised the measurement value of 20h from "0001h to 03FEh" to "0002h to 03FEh" and revised the measurement values of 00h from "0001h to 03FEh" to "0002h to 03FEh". • Section 3.2.10, inserted "The function configures an initial delay of 2048 µs." at the end of the description, revised the first input parameter from "UINT8 u8Filt Select:" to "UINT8 u8Filter:" and revised the third input parameter from "UNIT16*:" to "UINT16 u16UUMA:". • Section 3.2.11, revised as follows: – Section title: revised the section title from "UINT8 TPMS_READ_ACCEL_Z (UINT16 *u16UUMA, UINT8 u8Avg, UINT8 u8ModeSelect, UINT8 u8Dynamic Offset)" to "UINT8 TPMS_READ_ACCEL_Z (UINT16 *u16UUMA, UINT8 u8Avg, UINT8 u8Filter, UINT8 u8OffsetIndex)". – Description: inserted "The function configures an initial delay of 2048 us and a subsequent delay of 512 µs" as the second sentence of the description and inserted "The LVWF flag is cleared before the function returns." to the end of the first paragraph. – Input parameters: revised the third bullet from "UINT8 u8ModeSelect:..." to "UINT8 u8Filter:..." – Input parameters: revised the fourth bullet from "UINT8 u8DynamicOffset: Selects the offset setting for the appropriate acceleration reading depending on what bit 1 of u8ModeSelect is. For normal dynamic offset mode, default index is 7. Valid range is 0 to 15." to "UINT8 u8OffsetIndex: Selects the offset setting for the appropriate acceleration reading. The default index is 7. Valid range is 0 to 15.". – Table 21, revised the table title from "u8ModeSelect options" to "u8Filter options", revised the table headers from "u8ModeSelect Value" to "u8Filter Value" and "Selected mode" to "Selected filter" and removed "..., normal dynamic range" from each row. – Table 22, row 00h, revised the measurement value from "...to 0FFFh" to "...to 0FFEh" and row 20h from "...to 0FFFh" to "...to 0FFEh".

Table 36. Revision history

Revision number	Date	Description
		• Section 3.2.12, revised the input parameter "UINT16 *u16Accel:" to "UINT16 *u16 CompAccel:" and in Table 23, revised the table title from "Valid output conditions for TPMS_COMP_ACCEL_X" to "Valid output conditions for TPMS_COMP_ACCEL_ Z", revised the measurement value of 20h from "0001h to 03FEh" to "0002h to 03FEh" and revised the measurement values of 00h from "0001h to 03FEh" to "0002h to 03FEh". • Section 3.2.13, inserted "The function configures an initial delay of 2048 μs." as the last sentence of the description, revised the first input parameter "UINT8 u8Filt Select:" to "UINT8 u8Filter:" and revised the third input parameter from "UNIT16*" to "UINT16 *u16UUMA". • Section 3.2.14, Table 24 revised the measurement value for 00h from "Between 0000h - 03FEh" to "Between 0000h - 03FFh". • Section 3.2.24, revised "*u8RamBuffer:" to "*u8RAMBuffer:". • Section 3.2.25, revised "*u8RamBuffer:" to "*u8RAMBuffer:". • Section 3.2.28, Table 27, revised the description for Index 0 from "Firmware version" to "Library version". • Section 3.2.29, revised "RFCFR2_PWR" to "RFCR2_PWR" in two locations in the description. • Section 3.2.39, revised the resources text from "Global RAM locations 0090h to 00CAh" to "59 bytes of global RAM locations" and inserted ". Must be strictly greater than zero." on the last Input parameter and updated the note. • Section 3.2.40, revised the resources text from "Global RAM locations 0090h to 00CAh" to "59 bytes of global RAM locations" and updated the note. • Section 3.2.43, revised "Delay" to "u8Delay" in the section title, and revised the input parameter UINT8 u8OffsetIndex from "Selects the offset setting normal dynamic offset mode, default index is 6." to "Selects the offset setting normal dynamic offset mode, default index is 7." • Section 3.2.46 renamed section title from "TPMS_FRC_ENABLE" to "TPMS_E_ FRC_ENABLE". • Section 3.2.47 renamed section title from "TPMS_FRC_DISABLE" to "TPMS_E_ FRC_DISABLE". • Section 3.2.48 renamed section title from "TPMS_FRC_CLEAR" to "TPMS_E_ FRC_CLEAR". • Section 3.2.49 renamed section title from "TPMS_FRC_READ" to "TPMS_E_FRC_ READ". • Section 3.2.50 renamed section title from "TPMS_FRC_CALIB" to "TPMS_E_FRC_ CALIB" and revised the approximate duration from "10.13 μs" to "9.1 μs". • Section 3.2.51 renamed section title from "TPMS_FRC_CALIB_BUSCLK" to "TPMS_E_FRC_CALIB_BUSCLK". • Section 3.2.53, revised Approx. Duration from "140 μs for initial delay of 2048 μs and subsequent delay of 512 μs" to "140 μs" and changed "default index is 6" to "default index is 7" in the second bullet of the input parameters "UINT8u8Offset Index". • Section 3.2.55, revised "(UINT16* pu16UUMA)" to "(UINT16* u16UUMA)" in the title and revised the note from "Data ideally to be read under STOP4 and putting the device into STOP4 mode after activating the X channel is the responsibility of application code." to "Data ideally to be read under STOP4. Putting the device into STOP4 mode after activating the X channel is the responsibility of application code." • Section 3.2.56, revised "(UINT16* pu16UUMA)" to "(UINT16* u16UUMA)" in the title and and revised the note from "Data ideally to be read under STOP4 and putting the device into STOP4 mode after activating the Z channel is the responsibility of application code." to "Data ideally to be read under STOP4. Putting

Table 36. Revision history*...continued*

Revision number	Date	Description
		the device into STOP4 mode after activating the Z channel is the responsibility of application code." • Section 3.2.57, revised "(UINT16* pu16UUMA)" to "(UINT16* u16UUMA)" in the title and and revised the note from "Data ideally to be read under STOP4 and putting the device into STOP4 mode after activating the pressure channel is the responsibility of application code." to "Data ideally to be read under STOP4. Putting the device into STOP4 mode after activating the pressure channel is the responsibility of application code." • Section 3.2.58, revised "TPMS_READ_ACCEL_Z" to "TPMS_E_READ_ACCEL_Z" in two locations in the description. • Section 3.2.59, revised "TMPS_READ_ACCEL_X" to "TPMS_E_READ_ACCEL_X" in two locations in the description.
UM11145 v.6.1	04 March 2022	• Section 3.2.16, revised the note. • Section 3.2.18, revised the note. • Section 3.1, Table 6 and Section 3.2.50, corrected typo in section title changing "CALB" to "CALIB". Also added note at the end of the Section 3.2.50.
UM11145 v.6.0	16 February 2022	• Section 2.2.1, Definition of signal ranges, added second note after Table 3. • Section 2.3, Table 3, added new footnote in the content column for index 3 and revised the footnote for index 4. • Section 2.5, Table 4, LFCTRLD, revised bit 0 from 1 to 0. Table 5, revised LFCTRLD bit 1 from 0 to 1. • Section 3.1, Table 6, revised as follows: – Inserted new table rows to support new firmware functions starting with TPMS_FRC_ENABLE to TPMS_E_READ_DYNAMIC_ACCEL_X in the last row of the table. – Revised "Single/Dual" to "Single-X/Dual" for the following functions: – TPMS_READ_ACCEL_X – TPMS_COMP_ACCEL_X – TPMS_READ_DYNAMIC_ACCEL_X – TPMS_E_READ_ACCEL_X – Revised "Dual" to "Single-Z/Dual" for the following functions: – TPMS_READ_ACCEL_Z – TPMS_COMP_ACCEL_Z – TPMS_READ_DYNAMIC_ACCEL_Z – TPMS_E_READ_ACCEL_Z • Section 3.2.1, revised the description and the note. • Section 3.2.2, revised the first bullet in the description. • Section 3.2.4, revised the first bullet in the description. • Section 3.2.6, revised the first bullet in the description. • Section 3.2.8, revised the first bullet in the description. • Section 3.2.16, revised the note. • Section 3.2.18, revised as follows: – Power management: Removed a duplicate sentence. – Returns: revised "...MFO..." to " the bus clock..." and revised "...122 kHz..." to "3.9064 MHz". – Revised the note. • Section 3.2.26, removed the first sentence regarding backward compatibility. • Section 3.2.36, revised the description. • Section 3.2.38: revised as follows: – Stack size: revised from 38 bytes to 37 bytes.

Table 36. Revision history*...continued*

Revision number	Date	Description
		– Approximate Duration: revised the Approx. Duration bullet. – Table 30, revised u8TestMask Bit 0 to 1, revised the description of bit 3 from "Reserved" to "if set, g- Xcell wire-bond check performed", and revised "g-cell" in the description for bit 4 to "g-Zcell". – Note: Revised the bits from "0, 3, 5, or 6" to "0, 1, 5, or 6". – Table 31, revised the description for bits 3 and 4. • Section 3.2.40, revised the description. • Section 3.2.42, revised as follows: – Description: revised the description. – Stack size: revised the stack size from 29 to 30 bytes. – Approx. Duration: revised 29,300 to 28,500. – Input Parameters: revised the first and second bullets. • Section 3.2.44, revised the description. • Section 3.2.46 through Section 3.2.59, added new firmware functions.
UM11145 v.5.0	22 March 2021	• Global: – Performed minor grammatical, content, and typographic changes throughout. – Revised all "Warnings" to "Notes" to align with NXP definitions of notes, warnings and cautions. • Inserted "Document information" table on the first page. • Relocated the "Revision history" from the end of the document to the beginning to conform to NXP document content guidelines. • Section 3.2.19, added note at the end of the section. • Section 3.2.35, added a note at the end of the section.
UM11145 v.4.0	12 October 2020	• Section 3.2.3, revised "FEh" to "FFh" in the third sub-bullet of "Description" and in the "Measurement value" for 01h in Table 10. • Section 3.2.5, revised "FEh" to "FFh" in the second sub-bullet of "Description" and in the "Measurement value" for 01h in Table 12. • Section 3.2.7, revised as follows: – Description: revised "1FEh" to "3FFh" in the third sub-bullet of "Description" – Table 15, revised as follows: – Revised the "Measurement value" for 01h and 21h from "01FEh" to "03FFh." – Revised the "Measurement value" for 20h and 00h from "0001h to 01FEh" to "0001h to 03FEh." • Section 3.2.9, revised as follows: – Description: revised "1FEh" to "3FFh" in the second sub-bullet of "Description – Table 19, revised as follows: – Revised the "Measurement value" for 01h and 21h from "01FEh" to "03FFh." – Revised the "Measurement value" for 20h and 00h from "0001h to 01FEh" to "0001h to 03FEh." • Section 3.2.12, revised as follows: – Description: Revised "1FEh" to "3FFh" in the second sub-bullet of "Description." Table 23, revised as follows: – Revised the "Measurement value" for 01h and 21h from "01FEh" to "03FFh." – Revised the "Measurement value" for 20h and 00h from "0001h to 01FEh" to "0001h to 03FEh". • Section 3.2.14, revised "PTA0" to "PTB0" in "Description" and "Resources." • Section 3.2.15, revised "PTA1" to "PTB1" in "Description" and "Resources."

Table 36. Revision history*...continued*

Revision number	Date	Description
UM11145 v.3.0	17 September 2020	• Global: – Performed minor grammatical and typographical corrections throughout. – Revised references to register addresses from "$xx" to "xxh". – Revised the document title to "NTM88 Firmware Library User Guide" and changed content references to "Embedded Firmware" to "Firmware Library". • Section 1, revised the first sentence. • Section 2.1, added "library" after "firmware" in the first sentence. • Section 2.1.1, revised the first paragraph. • Section 2, Definition of signal ranges, revised second paragraph from "…in the case of temperature... " to "...in the case of temperature, pressure, and acceleration...." • Section 2.3, revised as follows: – Revised "Uncompensated acceleration" to "Uncompensated X-axis acceleration" in the content description for index number 3. – Added new Index 4 with the "Content" description of "Uncompensated Z-axis acceleration" and associated footnote "Applicable to dual-axis derivatives." • Section 2.5, revised "...when taking the reset vector and before giving control to the user." to "...the function vnfLFRConfigFactoryRegs() provided in the NTM88 Firmware Library is executed." • Section 3.1, revised as follows: – Revised the section title from "Firmware jump table" to "Firmware functions". – Revised the first two paragraphs into a single paragraph. • Section 3.1, Table 6: revised as follows: – Revised the title of Table 6 from "NTM88's firmware function jump table" to "NTM88 firmware functions." – Removed the "Absolute address" column and added "Single/Dual axis" column. – Renamed "TPMS_READ_ACCELERATION" to "TPMS_READ_ACCEL_X" and renamed "TPMS_COMP_ACCELERATION" to "TPMS_COMP_ACCEL_X". – Added the new row entries for sections titled "TPMS_READ_DYNAMIC_ACCEL_X", "TPMS_READ_ACCEL_Z", "TPMS_COMP_ACCEL_Z","TPMS_READ_DYNAMIC_ACCEL_Z", "TPMS_READ_V0", "TPMS_READ_V1", " TPMS_LFOCAL_BUSCLK", "TPMS_WAVG", "TPMS_RF_RESET", "TPMS_RF_READ_DATA", "TPMS_RF_READ_DATA_REVERSE", "TPMS_RF_CONFIG_DATA", "TPMS_RF_DYNAMIC_POWER", TPMS_MSG_INIT", "TPMS_MSG_WRITE", "TPMS_MSG_READ", "TPMS_CHECKSUM_XOR", "TPMS_CRC8", "TPMS_SQUARE_ROOT", "TPMS_MULT_SIGN_INT16", "TPMS_VREG_CHECK", "TPMS_E_READ_ACCEL_X", "TPMS_E_READ_PRESSURE", and "TPMS_E_READ_ACCEL_Z". • Section 3.2, revised as follows: – Added Table 7 and Table 8. – Added new sections: Section 3.2.10, Section 3.2.11, Section 3.2.12, Section 3.2.13, Section 3.2.14, Section 3.2.15, Section 3.2.17, Section 3.2.19, Section 3.2.21, Section 3.2.22,Section 3.2.23, Section 3.2.26, Section 3.2.29, Section 3.2.30, Section 3.2.31, Section 3.2.32, Section 3.2.33, Section 3.2.34, Section 3.2.35, Section 3.2.40, Section 3.2.41, Section 3.2.42, Section 3.2.43, Section 3.2.44, and Section 3.2.45. • Section 3.2.1, removed the first sentence of the description, "This function is called when taking the reset vector." and revised the start of the second sentence from "It will..." to "This function will...." • Section 3.2.6, revised as follows: – Description: revised "...or 1023..." to "...or 4095..." in the first bullet.

Table 36. Revision history*...continued*

Revision number	Date	Description
		– Input parameters: revised "Only the 10-bit..." to "Only the 12-bit..." in the first bullet. – Table 14, revised as follows: – Revised the "Measurement value" from "03FFh" to "0FFFh" for the two "u8 Status values", 04h and 24h. – Revised the "Measurement value" from "0001h to 03FEH" to "0001h to 0FFEh" for "u8Status value' 00h. – Revised the "Measurement value" from "0001h to 03FFH" to "0001h to 0FFEh" for "u8Status value' 20h. • Section 3.2.7, Revised the first bullet under "Input Parameters' from "...9-bit..." to "...10-bit...." • Section 3.2.8, revised as follows: – Section title: Revised the section title from "UINT8 TPMS_READ_ ACCELERATION..." to "UINT8 TPMS_READ_ACCEL_X...." – Description: added "for the X-axis" in the first sentence and revised the first bullet under "Description" from "...or 1023..." to "...or 4095...." – Input parameters, revised as follows: – Revised "Only the 10-bit..." to Only the 12-bit..." in the first bullet. – Revised "u8ModeSelect" to "u8Filter" in the third bullet. – Revised "u8DynamicOffset" to "u8OffsetIndex" in the fourth bullet. – Revised the titles of Table 16 and Table 18 to reflect change in section title. – Table 16, removed the 250 Hz rows from the table. – Table 18, revised as follows: – Revised the first and second "u8Status values" from "10h" to "08h." – Revised the third and fourth "u8Status values" from "30h" to "28h." – Revised the first and third "Measurement values" from "03FFh" to "0FFFh." – Revised the "Measurement value" from "0001h to 03FEh" to "0001h to 0FFEh" for the u8Status value "00h". – Revised the "Measurement value" from "h0001h to 03FEh" to "0001h to 0FFEh" for the u8Status value "20h". • Section 3.2.9, revised as follows: – Revised the title of Section 3.2.9 from "TPMS_COMP_ACCELERATION" to "TPMS_COMP_ACCEL_X". – Description: added "for the X-axis" in the first sentence. – Input parameters: revised "Updated 9-bit..." to "Updated 10-bit..." in the first bullet. – Revised the title Table 19of to reflect change in section title. • Section 3.2.39, added a second sentence "Addresses $FD40 - $FDFF are not valid targets in this function." to the "Description". • Removed section titled "UINT16 TPMS_FLASH_CHECK(void).
UM11145 v.2.1	15 May 2019	• Section 3.2.6, changed description from "Performs a 10-bit uncompensated pressure..." to "Performs a 12-bit uncompensated pressure..." • Section 3.2.7, changed description from "Performs a 9-bit uncompensated pressure measurement." to "Performs a 10-bit uncompensated pressure measurement." • Section 3.2.8, changed description from "Performs an uncompensated 10-bit measurement." to "Performs an uncompensated 12-bit measurement." • Section 3.2.9, changed description from "Performs a 9-bit compensated acceleration measurement." to "Performs a 10-bit compensated acceleration measurement."

Table 36. Revision history*...continued*

Revision number	Date	Description
UM11145 v.2.0	14 December 2018	• Changed part number from PTM88 to NTM88 throughout • Changed Table 6 to match subsections in Section 3.2 • Section 3.2.6 – Changed stack size from 30 bytes to 32 bytes – Deleted approx. duration • Table 13 – Changed Bus clock cycles from 1088 to 840 and 0.272 to 0.21 – Changed Initial delay MFO clock cycles from 323 to 256 and 2.59 to 2.04 – Changed Subsequent MFO clock cycles per additional sample from 203 to 64 and 1.624 to 0.512 – Changed Total for average = 4 [ms] from 7.892 to 3.949 • Section 3.2.7 – Changed stack size from 46 to 38 bytes – Changed approx. duration from 829 to 766.5 µs • Section 3.2.8 – Changed stack size from 35 to 37 bytes • Table 16 – Changed Bus clock cycles from 1128 to 836 and 0.282 to 0.209 – Changed Initial delay MFO clock cycles from 323 to 256 and 2.584 to 2.04 – Changed Subsequent MFO clock cycles per additional sample from 203 to 64 and 1.624 to 0.512 – Changed Total for average = 4 [ms] from 7.9 to 3.9498 – Changed Bus clock cycles from 1112 to 684 and 0.278 to 0.171 – Changed Initial delay MFO clock cycles from 406 to 323 and 3.248 to 2.584 – Changed ADC conversion time [µs] from — to 0.02 – Changed STOP4 exit time [µs] from — to 0.014 – Changed Total [ms] from 3.56 to 2.789 – Changed Subsequent MFO clock cycles per additional sample from 256 to 161 and 2.048 to 1.288 – Changed Additional ADC conversion time per additional sample from — to 0.02 – Changed Additional STOP4 exit time per additional sample from — to 0.014 – Changed Total for average = 9.7 [ms] from 9.833 to 6.782 • Section 3.2.9 – Changed Stack size from 55 to 48 bytes – Changed Approx. Duration from 869 to 843 µs • Deleted section UINT8 TPMS_READ_V0(UINT16 *u16Result, UINT8 u8Avg) • Deleted section UINT8 TPMS_READ_V1(UINT16 *u16Result, UINT8 u8Avg) • Deleted section UINT16 TPMS_WAVG(UINT8 u8PNew, UINT16 u16POld, UINT8 u8PAvg) • Deleted section void TPMS_RF_RESET(void) • Deleted section void TPMS_RF_READ_DATA(UINT8 u8Size, UINT8 *u8 RAMBuffer, UINT8 u8RFMBuffer) • Deleted section void TPMS_RF_READ_DATA_REVERSE(UINT8 u8Size, UINT8 *u8RAMBuffer, UINT8 u8RFMBuffer) • Deleted section void TPMS_RF_CONFIG_DATA(UINT16 *u16RFParam) • Deleted section void TPMS_RF_DYNAMIC_POWER(UINT8 u8CompT, UINT8 u8 CompV, UINT8* pu8PowerManagement) • Deleted section void TPMS_MSG_INIT(void) • Deleted section UINT8 TPMS_MSG_READ(void)

Table 36. Revision history*...continued*

Revision number	Date	Description
		• Deleted section UINT8 TPMS_MSG_WRITE(UINT8 u8SendByte) • Deleted section UINT8 TPMS_CHECKSUM_XOR(UINT8 *u8Buffer, UINT8 u8Size, UINT8 u8Checksum) • Deleted section UINT8 TPMS_CRC8(UINT8 *u8Buffer, UINT8 u8Poly, UINT8 u8 MBitSize, UINT8 u8Remainder) • Deleted section UINT16 TPMS_CRC16(UINT8 *u8Buffer, UINT16 u16MByteSize, UINT16 u16Remainder) • Deleted section UINT16 TPMS_SQUARE_ROOT(UINT16 u16Process) • Deleted section UINT8 TPMS_READ_DYNAMIC_ACCEL(UINT8 u8Filter, UINT8* u8Offset, UINT16* u16UUMA) • Deleted section void TPMS_RF_ENABLE(UINT8 u8Switch) • Deleted section void TPMS_MULT_SIGN_INT16(INT16 i16Mult1, INT16 i16Mult2, INT32* pi32Result) • Deleted section UINT8 TPMS_VREG_CHECK(UINT8 u8WaitTime, UINT16 u16 LimitDelta) • Deleted section UINT8 TPMS_READ_ACCEL_CONT_START(UINT8 u8Filter, UINT8 u8DynamicOffset, UINT8 u8SampleSpeed) • Deleted section UINT8 TPMS_READ_ACCEL_CONT(UINT16* pu16Measurement) • Deleted section UINT8 TPMS_READ_ACCEL_CONT_STOP(void)
UM11145 v.1.0	13 August 2018	Initial release

Legal information

Definitions

Disclaimers

Trademarks

Tables

Figures

Contents