To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

User's Manual

EZ-0010

78K0R/KE3-L IO-Link Starter Kit

78K0R/KE3-L

Document No. U20226EJ1V0UM00 (1st edition)
Date Published February 2010 NS

Windows and Windows NT are registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
Pentium is a trademark of Intel corp. in the United States.

- The information in this document is current as of February, 2010. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific". The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.

  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.
  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).
  "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

  The quality grade of NEC Electronics products is "Standard" unless otherwise expressly specified in NEC Electronics data sheets or data books, etc. If customers wish to use NEC Electronics products in applications not intended by NEC Electronics, they must contact an NEC Electronics sales representative in advance to determine NEC Electronics' willingness to support a given application.

  (Note 1) "NEC Electronics" as used in this statement means NEC Electronics Corporation and also includes its majority-owned subsidiaries.
  (Note 2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

(M8E0909E)

Table of Contents

CHAPTER 1 INTRODUCTION

EZ-0010 for the NEC low power 78K0R 16-bit microcontroller family is designed as a development platform for IO-link sensors, and can also be used as an evaluation tool for the 78K0R microcontroller. It allows the development of IO-Link sensor systems based on the 78K0R/KE3-L (μPD78F1009) device. The board is prepared to be connected to user hardware parts, and contains elements to easily start and demonstrate IO-link sensor applications. This includes I/O-functions, i.e. navigator switch, a Temperature sensor, MR Sensor, analog inputs and outputs, UART serial interface, cables, and more.

In addition, the "sample program for the IO-Link temperature sensor" and "sample program for the IO-Link RPM sensor" can be downloaded from the website and run.

1.1 Package contents

- Link It! board
- USB cable
- Contents list of this package
- Table of Toxic and Hazardous Substance and Elements
- IO-Link Starter Kit for 78K0R/KE3-L EZ-0010 About related documents and softwares

Please verify that you have received all parts listed in the contents list of EZ-0010. If any part is missing or seems to be damaged, please contact the dealer from whom you received your EZ-0010.

1. **The Technologie Management Gruppe, TMG (Technologie und Engineering GmbH) stack is for evaluation purposes only. For mass production end products, a full license must be purchased from TMG.**

2. **The TMG IO-Link Master and TMG IO-Link device Tool are referenced in this manual. However , they are not included in the package**

Notes **Please contact TMG directly for details on obtaining the full license, the TMG IO-Link Master and TMG IO-Link device Tool**

Technologie Management Gruppe-Karlshrue ((Technologie und Engineering GmbH) Internet: www.tmg-karlsruhe.de

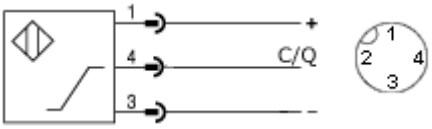
3. **Please also note that the M12 cable shown in this manual is not included in the package.**

1.2 Features of the EZ-0010

- NEC Electronics µPD78F1009 low power consumption general purpose NEC 16-bit 78K0R MCU
- NEC Electronics µPD78F0730 MCU with on-chip USB interface
- E98110A Elmos IO-Link Transceiver
- 4 x 7 segments LEDs driven by serial interface.
- Power Supply by USB interface, external power supply, or M12 cable.
- On-Board debug function (TK-78K0R debugger)
- TMG IO-Link Stack Library (Evaluation version).
- EZ-0010 provides an IO-Link communication channel

Current type:	DC
Wiring:	3-Wire
Output signal:	IO-Link
Rated operational voltage:	DC 24 V
Supply voltage:	DC 18…30 V
Short circuit protection:	yes
Protected against polarity rev:	yes

IO-Link Data:



Physical Layer:	PHY2
Data Transfer Rate:	COM1 (4.8kBaud), COM2 (38.4kBaud), COM3 (230.4kBaud)

- Board size: 135mm x 60mm
- Power supply by USB interface option for Microcontroller.
- Analog to digital signal conversion
- Various input / output signals available, such as
  - I/O ports prepared to be connected to user hardware
  - Timer input / output signals
  - Two or three wires serial I/O
  - Analog input lines
  - Navigation switch
- Virtual UART interface, via the μPD78F0730 78K0 8-bit microcontroller with on-board USB interface

The EZ-0010 is not intended for code development. NEC does not allow and does not support in any way any attempt to use EZ-0010 in a commercial or technical product.

1.3 System requirements

HOST PC	A PC supporting Windows 2000 or Windows XP is required for the C compiler, assembler, and WriteEZ5 GUI software. A Pentium™ processor with at least 1 GHz CPU performance, with at least 256 Mbytes of RAM, allowing you to fully utilize and take advantage of the product features. 500 Mbytes of free disk space, and an additional 10 Mbytes of free disk space on the Windows system drive. A web browser and Adobe Acrobat Reader to be able to access all the product documentation.
Host Interface	USB interface that enables communication based on USB (Ver1.1 or later)

Notes Development tools of 78K0R-Link It! board (programming environment, assembler, C compiler, and debugger), WriteEZ5 GUI software, and the newest documentation can be downloaded from the NEC Electronics website.

1.3.1 Optional system requirements

User may choose to purchase the following additional equipments for the development of IO-Link applications:

- TMG USB IO-Link Master
- PC software- TMG IO-Link Device Tool for configuration and analysis.
- Double Ended 4 pins M12 cable.

Please contact TMG directly for details on the TMG IO-Link Master and TMG IO-Link device Tool

Internet: www.tmg-karlsruhe.de

CHAPTER 2 EZ-0010 SYSTEM CONFIGURATION

The IO-link standard is a point to point based communication, where a master and a slave device exchange information over a communication channel. The master initiates and organizes the data exchange. The slave treats the master's request and replies accordingly.

The *78K0R–Link It!* board (slave device) is based on the NEC's latest 16-bit 78K0R/KE3-L microcontroller family.

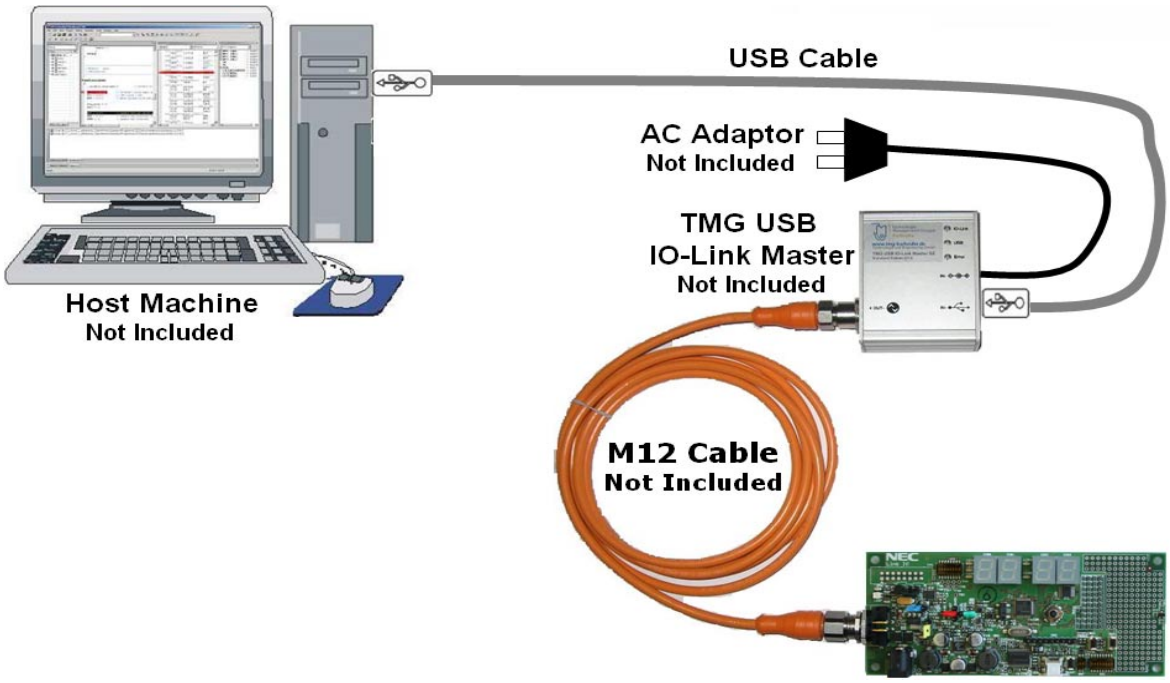


Figure 2-1. IO-Link system configuration

2.1 78K0R–Link It! board

78K0R–Link It! board is a demonstration kit for the 78K0R/KE3-L 16-bit microcontroller of the 78K0R family. The IO-link board can be used in two modes: IO-Link Mode (default mode), and Standalone mode.

2.1.1 IO-Link Mode

When used in IO-Link Mode, the *78K0R–Link It! board* is connected to a host system via an IO-Link Master. Figure 2-1 shows the board being connected to a host computer using the TMG USB IO-Link Master (not included: see section **1.3.1. Optional system requirements** for more details).The Master powers and connects to the *78K0R–Link It! board* via the M12 connector using a M12 cable (not included). The board will be working as an IO-link sensor demonstrating the application stored in the Microcontroller's memory.

2.1.2 Standalone Mode

When the board is not used in an IO-link application, the board is in "standalone mode". In this mode, the user can use the microcontroller for application development and debugging. The *78K0R–Link It! board* is connected to the host system via USB interface with USB cable provided, See. Figure 2-2. The host system may be used for On-Chip debugging by using the ID78K0R debugger and to allow execution of application programs on *78K0R–Link It!* board. It is also used to power the board.

Caution **In "standalone mode", the M12 cable must be disconnected and an AC adapter may be used to power the board if more than 500mA (USB maximum current load) is required**

Remark **The user can optionally use an external AC adapter, the MINICUBE2(not included), or the USB interface, to provide power to the board. See Table 3-1. Power supply selector, JP1 configuration for details**

The microcontroller on the board can run at up to 20 MHz operating speed. The sub-clock is provided with a 32.768 kHz. A 18.432MHz external crystal is also available as main clock.

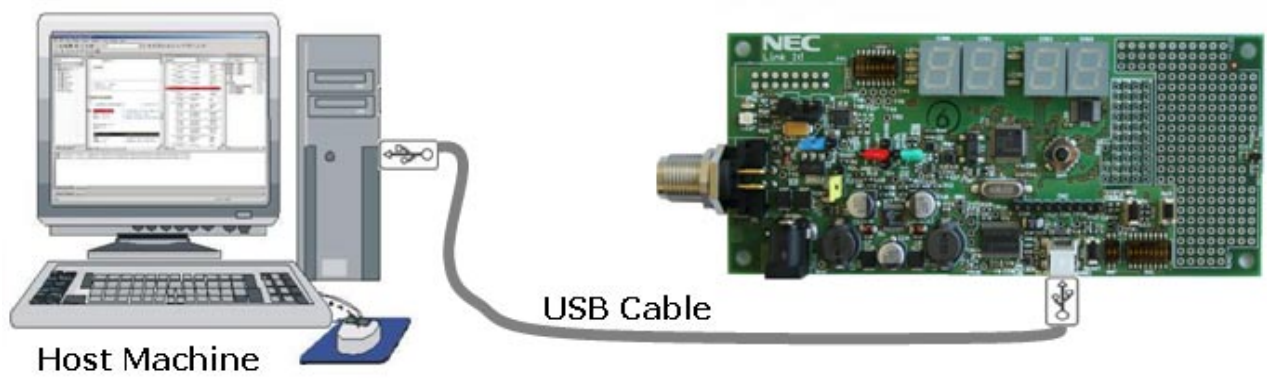


Figure 2-2. USB Powered configuration

2.2 Host computer

The USB host interface enables communication to the *78K0R–Link It!* board. The µPD78F0730, 78K0 8-Bit microcontroller with on-chip USB interface and the NEC virtual UART driver allows application software to access the USB device in the same way as it would access a standard RS232 interface. The NEC virtual UART driver appears to the windows system as an extra Com Port, in addition to any existing hardware Com Ports.

2.3 Power supply via USB interface

The *78K0R-Link it!* board can be powered by the USB interface. Optionally the board can be powered by the QB-MINI2 On-Chip debug emulator as well as an external power supply.

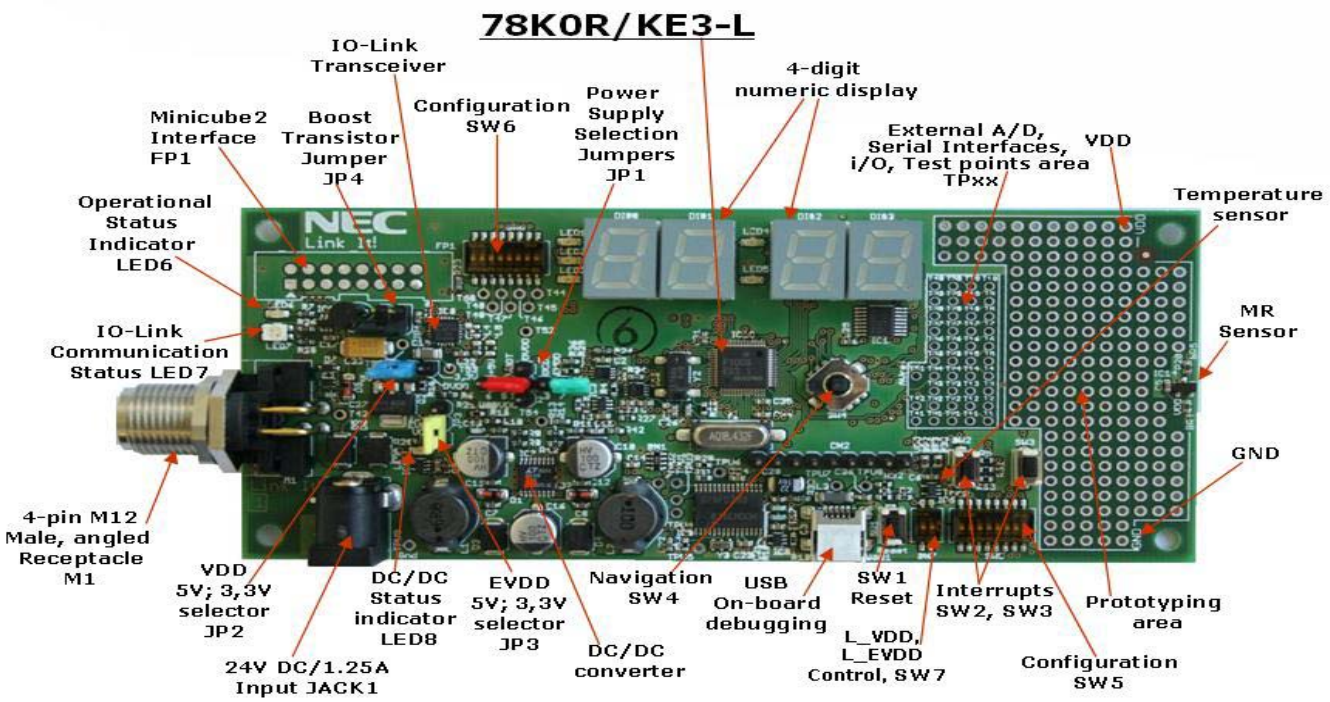
Caution **A different power source must be used to power the board if more than 500mA (USB maximum current load) is required**

CHAPTER 3 78K0R–LINK IT! BOARD COMPONENTS

The *78K0R–Link It!* board is equipped with a variety of components allowing a user to develop sensor applications. These components include: Timer input / output signals, Two or three wires serial I/O, 7 segments numeric displays, LEDs, Power LED, 2 push buttons for external interrupts generation (SW2 and SW3), Dip Switch for control of operating modes of the board, Reset button, wrap area for external user hardware, multi directional Joystick(SW4).

Most I/O ports on the 78K0R/KE3-L microcontroller are prepared to be connected to user hardware.

Several connectors are available in order to be connected to host computers, FLASH programmer or any external target hardware.



3-1. 78K0R-IO-Link board Components

3.1 Power Supply Selector JP1

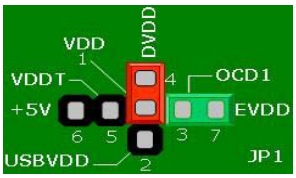


Figure 3-2. Power Supply Selector JP1

Jumper JP1 is the power supply selector of the board. Jumper JP1 controls the power supply of the board.

In default mode (IO-Link mode: JP1 1-4 closed) the board is powered by the IO-link Master (not included), via the M12 connector. The board will be working as an IO-link sensor demonstrating the application stored in the Microcontroller's memory.

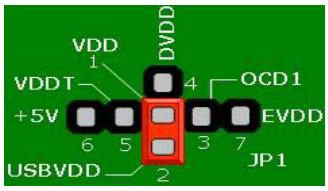
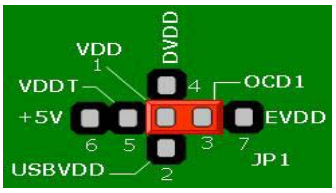
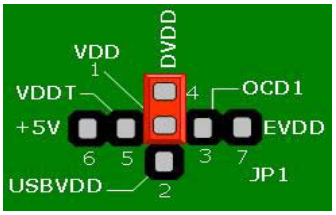
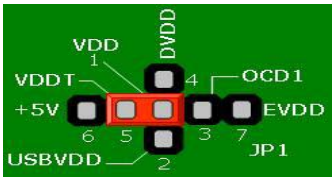
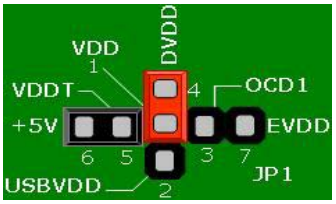
If the board is not used in an IO-link application, the board is in “standalone mode”. The user can use the microcontroller for application development and debugging. In “standalone mode”, the M12 cable (not included) must be disconnected and an AC adapter may be used to power the board.

Additionally, if an AC adapter is not available to be used in standalone mode, the user can optionally use the MINICUBE2, or the USB interface, to provide power to the board.

When powered from the M12 connector or the AC Adapter, the user can then select whether to provide the VDD voltage rail to the microcontroller, from the regulator output from the IO-Link transceiver or the DC/DC Converter available on the board. LED6 lights up when power is apply to VDD, by setting JP1.

The table below summarizes the configuration options for the power supply selector JP1.

Table 3-1. Power supply selector, JP1 configuration

JP1	configuration	Power Supply Source
1-2		Power supply from USB via connector USB1 (5V VDD for 78K0R/KE3-L from USB1) **The IO-Link transceiver transmitter is disabled if VDD_IO is below VDD_IO_UV**
1-3		MINICUBE2 via OCD1 connector (Note2). (5V VDD for 78K0R/KE3-L from OCD1) **IO-Link transceiver transmitter is disabled if VDD_IO is below VDD_IO_UV**
1-4		M12 cable via M12 connector or AC Adaptor from Jack1 (5V VDD for 78K0R/KE3-L from DC/DC converter: Discrete VDD) Internal regulator of IO-Link transceiver used by the transceiver only **JP4 (pins 1 and 2) must be closed.**
1-5		M12 cable via M12 connector or AC adaptor via Jack1 (5V VDD for 78K0R/KE3-L from internal regulator of IO-LINK Transceiver) **1. Q1 has to be mounted on the board (Note1),** **2. JP4 (pins 2 and 3) must be closed.**
1-4, 5-6		Board Power supply from M12 via M12 connector or AC Adaptor via Jack1 (Transceiver’s 5V VDD from DC/DC converter: Internal regulator of transceiver not used) **1. JP4 must be opened.** **2. The Black Jumper must be moved to pins 5 and 6**

Note 1 An additional NPN transistor (not mounted), must be used. Care must be taken to evaluate current consumption when choosing this transistor. If the transistor gets destroyed, other components connected to the 5V rail on the board may also be destroyed.

Note2 **By supplying power from the MINICUBE2, external hardware must not be connected to the board. Within this mode the board can operate without external power supply from USB**

3.2 Discrete Voltage rail selector, JP2

JP2, configures the voltage level (DVDD), provided by the DC/DC converter to power the board. The user has the choice between 3.3V and 5V.

Figure 3-3.DVDD selector, JP2

Table 3-2: DVDD, JP2 Pin configuration

JP2	configuration	Discrete VDD (DVDD)
1-2	Closed (Default)	DVDD = 5V
2-3	Closed	DVDD = 3.3V

3.3 I/O port voltage control, JP3

Use JP3 to apply 3.3V or 5V to the I/O port reference input pin EVDD of the 78K0R/KE3-L. A different reference voltage may also be applied by using an external power source.

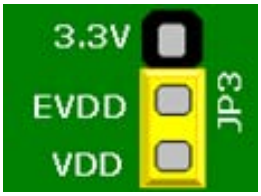


Figure 3-4. EVDD selector, JP3

Table 3-3: EVDD, JP3 Pin configuration

JP3	configuration	Voltage
1-2	Closed (Default)	EVDD = VDD
2-3	Closed	EVDD = 3.3V

3.4 VDD Transceiver selector , JP4

JP4 controls the transceiver's VDD voltage usage.

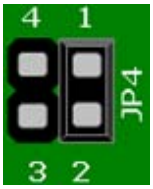


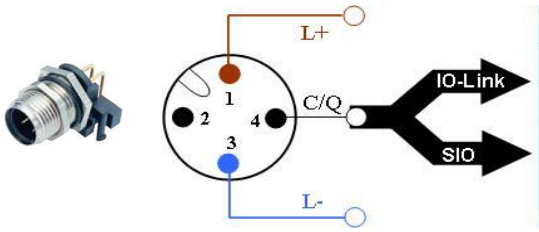
Figure 3-5. VDD Transceiver selector, JP4

Table 3-4: VDD Transceiver, JP4 Pin configuration

JP3	configuration	Voltage
1-2	Closed(Default)	internal regulator active
2-3	Closed	external Transistor Q1 active (not supported)
1-2-3-4	Opened	external supply of VDD: internal regulator inactive

3.5 M12 Connector, M1

The M12 connecter allows the board interfacing with an industrial type master device. Figure 3-6 gives the wiring scheme for the plug. Pin 2 is not currently used for IO-Link. This pin is connected to Test pad T43 for future use.



PIN	Signal	Remark
1	L+	24 Volt
2		
3	L-	Ground
4	C/Q	Communication / switching signal

Figure 3-6.M12 wiring scheme

3.6 USB interface connector, USB1

This interface allows connecting the ID78K0R debugger to the board in order to use the On-Board debug function (TK-78K0R debugging). The TK-78K0R interface supports On-board FLASH erasing / programming and standard debug features like code execution, single stepping, breakpoints, memory manipulation etc.

For standard communication to a host computer - i.e. by using a terminal program - the input/output signals of UART1 of the 78K0R/KE3-L device can be redirected to the USB1 connector via the µPD78F0730 USB microcontroller.

The power supply of the board can also be provided by the USB1 connector.

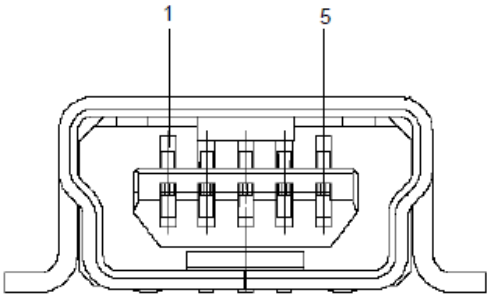


Figure 3-7. USB1, USB Mini-B Type Host Connector Pin Configuration

Table 3-5. Pin Configuration of Connector USB1

Connector USB1	Signal Name
1	VBUS
2	D-
3	D+
4	N.C.
5	GND

3.7 RESET button, SW1

SW1 is the reset button. It activates the power on reset. Switch SW1 controls the reset input signal of the 78K0R/KE3-L microcontroller.

Beside the reset switch, the 78K0R/KE3-L can also be reseted by the on-board debugging interface via the uPD78F0730 microcontroller, and by the on-chip debugging interface, via the FP1 connector.

3.8 (INTP0) Switch, SW2

SW2 is a push button connected to external interrupt input INTP0 of the microcontroller. This is equal to port “P120/INTP0/EXLVI” of the 78K0R/KE3-L device. The port may be programmed to generate the external interrupt INTP0.

3.9 (INTP4) Switch, SW3

SW3 is a push button connected to external interrupt input INTP1 of the microcontroller. This is equal to port “P70/KR0/SO01/INTP4” of the 78K0R/KE3-L device. The port may be programmed to generate the external interrupt INTP4.

3.10 Navigation switch, SW4

Button SW4 is a navigation switch connected to the 78K0R/KE3-L device. It operates in four directions and has a center push function. When the navigation switch is moved to one of the four directions or it is pushed a low-level signal (Vss) is applied to the corresponding pin of the 78K0R/KE3-L device. The connection of SW4 to the microcontroller is shown in the table below.

The switch can be used for sensor application parameterization

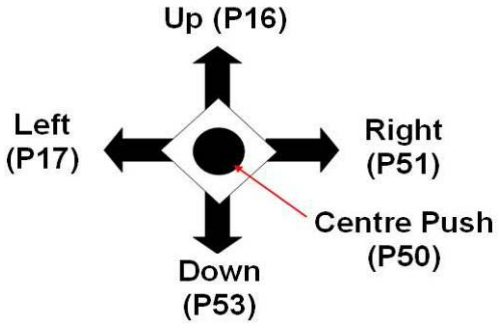


Figure 3-8. SW4, Navigation switch

Table 3-6. SW4, Navigation switch Pin configuration

SW4	Pin Connection to the 78K0R/KE3-L device
Up	P16
Left	P17
Right	P51
Center Push	P50
Down	P53

3.11 Configuration switch, SW5

The different operation modes of the board can be set by switch SW5/bit1-10.

Table 3-7. Configuration switch SW5, default settings

SW5/bit	Factory settings	Mode
1	OFF	Standalone and debug Mode (Note1)
2	OFF	Disable TK-78K0R debugger functions
3	OFF	Disable TK-78K0R debugger functions
4	OFF	TxD1 Disconnected from TK-78K0R debugger
5	OFF	RxD1 Disconnected from TK-78K0R debugger
6	ON	SCL0 Connected to display driver
7	ON	SDA0 Connected to display driver
8	OFF	AVREF Disconnected from VDD(Note2)
9	OFF	Disable TK-78K0R debugger functions
10	OFF	Disable TK-78K0R debugger functions

Notes

1. **USB cable not connected**
2. **The board is shipped from factory with a 2.5V reference voltage rail applied to AVREF, via R40.**

3.11.1 Standalone / debug mode selection, SW5/bit1

Switch SW5/bit1 controls the operation mode of the board, when a USB cable is connected to USB1, and the TK-78K0R Debugger is used.

- Switch SW5/bit1 to ON sets the board to “stand-alone mode”.
  Within this mode, the 78K0R/KE3-L RESET can be controlled by the user via switch SW1 and by the ID78K0R debugger. The program stored in the internal FLASH memory of the 78K0R/KE3-L microcontroller is executed. The usage of the On-Board debug function (TK-78K0R debugging) is also supported within the “stand-alone mode”.

- Switch SW5/bit1 to OFF sets the board to the “debug mode” exclusively.
  Within this mode the 78K0R/KE3-L device is permanently held within RESET state. Only the ID78K0R debugger can control the RESET signal within this mode. The RESET switch SW1 is inactive.

Table 3-8. Operation mode selection SW5/bit1

SW5/bit1	Mode
ON	Standalone and debug mode
OFF	Debug mode only

3.11.2 On-Board debug function via USB selection, SW5/bit2 – bit3 – bit9 – bit10

SW5/bit2 bit3, bit9 and bit10 are controlling the TK-78K0R Debugger interface with the 78K0R/KE3-L

- Switch SW5/bit2 and bit3 ON, to enable On-Board debugging via USB.

  Within this mode a dedicated single-line UART (pin Tool0) of the 78K0R/KE3-L is connected to the µPD78F0730 USB microcontroller. Pin Tool0 of the 78K0R/KE3-L microcontroller is reserved for on-board FLASH programming and debugging purpose.

- Switch SW5/bit2 and bit3 OFF to disconnect the single-line UART from the µPD78F0730 USB microcontroller.

  Within this mode, no debugging is supported. The user can establish standard serial communication via UART1 to a terminal program running on the HOST PC (see **section 3.11.3** for more details).

- Switch SW5/bit9 controls the RESET signal of the TK-78K0R debugger

- Switch SW5/bit10 controls the FLMDO signal of the TK-78K0R debugger

  Switch SW5/bits 9 and 10 ON, only when the TK-78K0R debugger is used

Table 3-9. (TK-78K0R debugging) configuration, SW5/bit2 - bit3 – bit9 – bit10

SW5/bit2 - bit3 - bit9 - bit10	Mode
OFF	Disable TK-78K0R debugger functions
ON	Enable TK-78K0R debugger functions

3.11.3 UART mode selection, SW5/bit4-bit5

The user can establish standard serial communication via UART1 to a terminal program running on the HOST PC.

- Switch bit4 and bit5 of SW5 ON or OFF to respectively connect or disconnect the UART1 signals RxD1 and TxD1 of the 78K0R/KE3-L, to the µPD78F0730 USB microcontroller.

  Within this mode, standard serial communication to a terminal program running on the HOST PC can be established.

Note: when communication with the Host PC is required, the user must ensure the On-board debugging function is disabled (see section 3.11.2 for more details).

Table 3-10. UART mode selection, SW5/bit4-bit5

SW5/bit4	SW5/bit5	Mode
OFF	OFF	Disconnect UART1 signals from TK-78K0R debugger
ON	ON	Connect UART1 signals to TK-78K0R debugger

3.11.4 Display Driver control, SW5/bit6-bit7

Switch SW5/bit6-bit7 control the I²C interface between the 78K0R/KE3-L and the 7-segments displays driver.

- Switch bit6 and bit7 ON or OFF to respectively connect or disconnect the display driver to the 78K0R/KE3-L.

Table 3-11. Display Driver control, SW5/bit6-bit7

SW5/bit6	SW5/bit7	Mode
OFF	OFF	Disconnect 7-segments display driver
ON	ON	Connect 7-segments display driver

Caution The Display driver must be powered when the 7-segment displays are used to display information. Please refer to section 3.13 for more details.

3.11.5 AVREF control, SW5/bit8

Switch SW5/bit8 controls the voltage reference source for the analog input of the 78K0R/KE3-L

- Switch bit8 ON or OFF to respectively apply or disconnect VDD to AVREF.

Table 3-12. AVREF control, SW5/bit8

SW5/bit8	Mode
OFF	AVREF disconnected from VDD (Note1)
ON	AVREF = VDD

Note The board is shipped from factory with a 2.5V reference voltage rail applied to AVREF, via R40.

3.12 Transceiver Configuration Switch SW6

SW6 serve two purposes.

- It allows to isolate the IO-Link transceiver from the 78K0R/KE3-L microcontroller by disconnecting the communication interface
- It put the DC-DC converter in power down mode when it is not needed by the user

3.12.1 DC-DC power down control, SW6/bit1

The IO-Link master provides a 24V voltage rail to the board. A DC/DC converter steps down that voltage to provide the 5V, and 3.3V power supplies for the board.

The board is equipped with dual current mode PWM step-down DC/DC converter with internal power switches capable of generating two 1.4A outputs. Switch SW6/bit1 turns DC-DC converter ON/OFF.

Table 3-13. DC-DC power down control, SW6/bit1 configuration

SW6/Bit1	Setting	DC-DC Converter
1	ON	DC-DC converter active
1	OFF	DC-DC converter inactive

LED8, lights up when power switch SW6/bit1 is ON to indicate that the DC–DC converter is active.

3.12.2 Transceiver bypass control, SW6/bit2-8

Switch SW6/bit2to bit8 allow the user to connected/disconnect the IO-Link transceiver to/from the 78K0R/KE3-L microcontroller.

Table 3-14. Transceiver bypass control, SW6/bit2-bit8 Pin configuration

SW6 Bit	Setting	IO-Link Transceiver	78K0R/KE3-L pin
2	ON/OFF	VDD_IO	EVDD connected / disconnected
3	ON/OFF	TxEN	P76/KR6 connected / disconnected
4	ON/OFF	SPEED	P77/KR7 connected / disconnected
5	ON/OFF	TxD	TxD0 connected / disconnected
6	ON/OFF	RxD	RxD0 connected / disconnected
7	ON/OFF	WAKE	P72/INTPU6 connected / disconnected
8	ON/OFF	ILIM	P71/INTPU5 connected / disconnected

3.12.3 IO-Link Transceiver

The transceiver has two functions. It is used firstly as an IO-Link port Analog front-End. It ensures the analog front end of the IO-Link communication channel by transforming the 3-wire 24V of the M12 IO-link port to a digital interface for the 78K0R/KE3-L device.

Secondly, it can be used as a power supply source:

The communication and the power supply run via 3 separate lines to the IO-Link Master. The integrated voltage regulator is able to provide up to 5mA current for external purposes. With an optional external boost transistor (not mounted on the board), it is possible to supply external devices with a higher current consumption.

Caution **The user must be aware of the current consumption of the application and the maximum current rate of the boost transistor, if using this feature. Since a destruction of the boost transistor, may also cause the destruction of other devices connected to the 5V voltage rail provided by the boot transistor.**

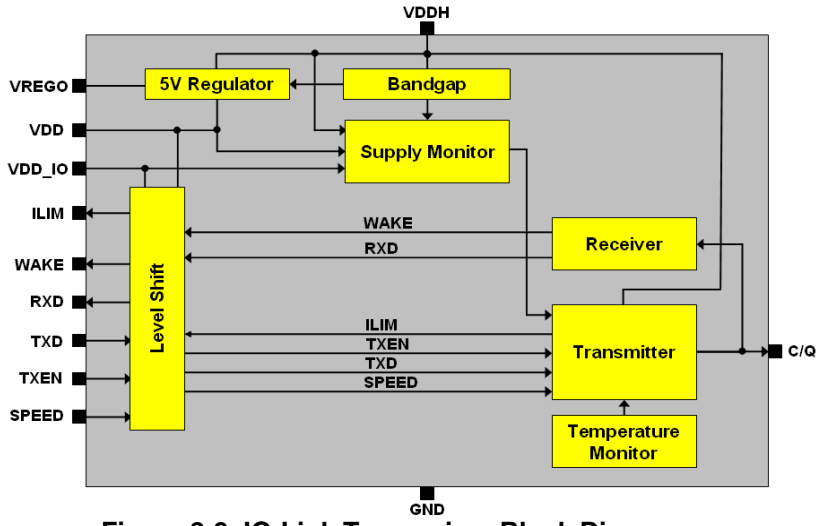


Figure 3-9. IO-Link Transceiver Block Diagram

3.13 Digital logic Power Supply switch, SW7

SW7 is a 2-bits switch used to control respectively, the power supplies (L_VDD, L_EVDD) for the 7 segments displays driver, and the digital logic interface between the USB microcontroller (uPD78F0730) and the 78K0R/KE3-L.

Switch bit1, bit2 to ON or OFF to apply or disconnect VDD and EVDD to L_VDD, and L_EVDD respectively.

Table 3-15. SW7 Digital Logic supply configuration

SW7/Bit1-Bit2	Setting	Power supply
1	ON(Factory setting)	L_VDD = VDD(7-seg display and LED7 active)
	OFF	L_VDD Disconnected (7-seg display and LED7 inactive)
2	ON	L_EVDD = EVDD(USB microcontroller interface active)
	OFF(Factory setting)	L_EVDD Disconnected(USB microcontroller interface inactive)

3.14 4-digit Numeric Display

The numeric display allows hexadecimal fonts displaying of metering information such as, distance, temperature, time. It is made of 4 single 7 segments displays and 5 discrete LEDs.

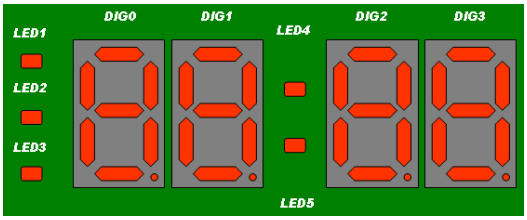


Figure 3-10. 4-digit Numeric Display

LED1 to LED3 can be used to provide common Alarm, AM, PM types of functions, or any user defined function.

The colon segment LED4 and LED5 are connected in parallel and driven together as one LED.

The 7 segments display and discrete LEDs are driven by a display driver. This display driver interfaces with the 78K0R/KE3-L via an I²C interface. see section 3.11.4 for details.

3.15 Power LED, LED6

LED6, lights up when power is supplied to the *78K0R–Link It! board* by setting JP1.See **section 3.1** for details.

3.16 IO-Link Communication indicator, LED7

LED7 is a bi-color LED connected to pins P42, P43, to indicate the activities on the communication channel.

By default the LED is lighted up with an amber color, when the microcontroller is not programmed and power is provided to L_VDD via switch SW7 bit1. The table below shows the defined LED colors and indications for the board, when an IO-Link application is downloaded to the microcontroller.

Table 3-16. LED7 indications for IO-Link port signals

Signal	IO-Link communication
Pin 4 "C/Q line"	● Dark (OFF) : (Device disconnected or Fault on C/Q line: Rx not connected) ● Green (ON) : (Data exchange ok) ● Red (ON) : (Fault on C/Q line TX disconnected

3.17 External main oscillator, Y1

The 78K0R/KE3-L can run at up to 20 MHz operating speed , and the clock generator can be configured to work with a 18.432MHz external oscillator connected between the X1 and X2 pins.

The sub-clock is provided with 32.768 kHz to pins XT1 and XT2 of the 78K0R/KE3-L device.

3.18 Test pads, T1~T42 area and wrap field

Several pins of the 78K0R/KE3-L microcontroller are connected to the test pads T1~T42 area. The corresponding assignment can be found in tables below. Additionally the *78K0R-Link it!* board provides a wire wrap field area allowing the integration of additional application hardware.
VDD, GND, and an MR sensor are also available this area.

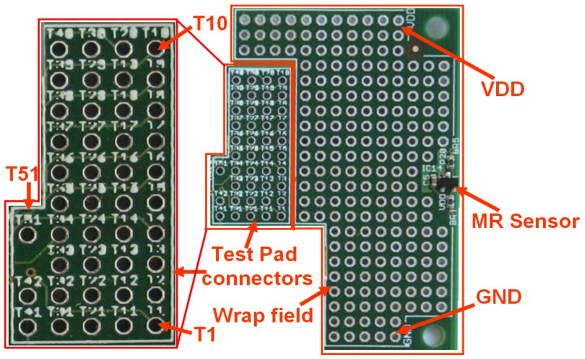


Figure 3-11.Test pad area and wrap field

Table 3-17. Test Pads Tables

Test pads	78K0R/KE3-L I/O pin
T1	P140 / PCLBUZ0
T2	P141 / PCLBUZ1
T3	P00 / TI00
T4	P01 / TO00
T5	P20 / ANI0
T6	P21 / ANI1
T7	P22 / ANI2
T8	P23 / ANI3
T9	P24 / ANI4
T10	P25 / ANI5
T11	P26 / ANI6
T12	P27 / ANI7
T13	P150 / ANI8
T14	P151 / ANI9
T15	P152 / ANI10
T16	P153 / ANI11
T17	P80 / CMP0P / INTP3 / PGAI
T18	P81 / CMP0M

Test pads	78K0R/KE3-L I/O pin
T19	P82 / CMP0M /. INTP7
T20	P83 / CMP1M
T21	P10 / TI02 / TO02
T22	P11 / TI03 / TO03
T23	P12 / TI04 / TO04 / RTCDI / RTCLL
T24	P13 / TI05 / TO05
T25	P14 / TI06 / TO06
T26	P15 / TI07 / TO07
T27	P52 / RTC1HZSL / STLI / SLTO
T28	P71 / KR1 / SIO1 / INTP5
T29	P72 / KR2 / SCK01 / INTP6
T30	P73 / KR3 / SO00 / TxD0
T31	P74 / KR4 / SI00 / RxD0
T32	P77 / KR7
T33	P76 / KR6
T34	P75 / KR5 / SCK00

Test pads	78K0R/KE3-L I/O pin
T35	P33
T36	P32 / SCK10 /SCL10 INTP2
T37	P31 / SI10 / RxD1 / SDA10 / INTP1
T38	P30 / SO10 / TxD1
T39	P61 /SDA0
T40	P60 / SCL0
T41	P42
T42	P43
T43	Pin2 M12 connector
T44	ILIM Transceiver
T45	WAKE Transceiver
T46	RxD Transceiver
T47	TxD Transceiver
T48	SPEED Transceiver
T49	TxEN Transceiver
T50	VDD_IO Transceiver
T51	AVREF

3.19 MR (magneto-resistive) Sensor

A MR sensor is a magneto-resistive sensors integrated circuits in small packages. It can be used in position detection applications, using non-contact type sensor. The MR sensor outputs a digital signal to pin P32/SCK10/SCL10/INTP2 of the 78KR0/KE3-L device

3.20 Temperature sensor

For temperature measurement and primarily as an application example, the *78K0R–Link It!* board is equipped with a temperature sensor IC. The output pin of the temperature sensor is connected to pin P153/ANI11 of the 78KR0/KE3-L device.

3.21 PG-FP4 /PG-FP5 / QB-MINI2 connector, FP1

Connector FP1 (not mounted) allows connection of the PG-FP4/PG-FP5 FLASH programmers to the board in order to program application software into the 78K0R/KE3-L internal flash memory.

Remark , The PG-FP4 /PG-FP5 FLASH programmers are separate product from NEC and are not included in this package.

Additionally, FP1 allows connection of the QB-MINI2 (MINICUBE2 On-Chip debug emulator) to the board in order to use On-Chip debug function of the 78K0R/KE3-L device and program the internal flash memory.

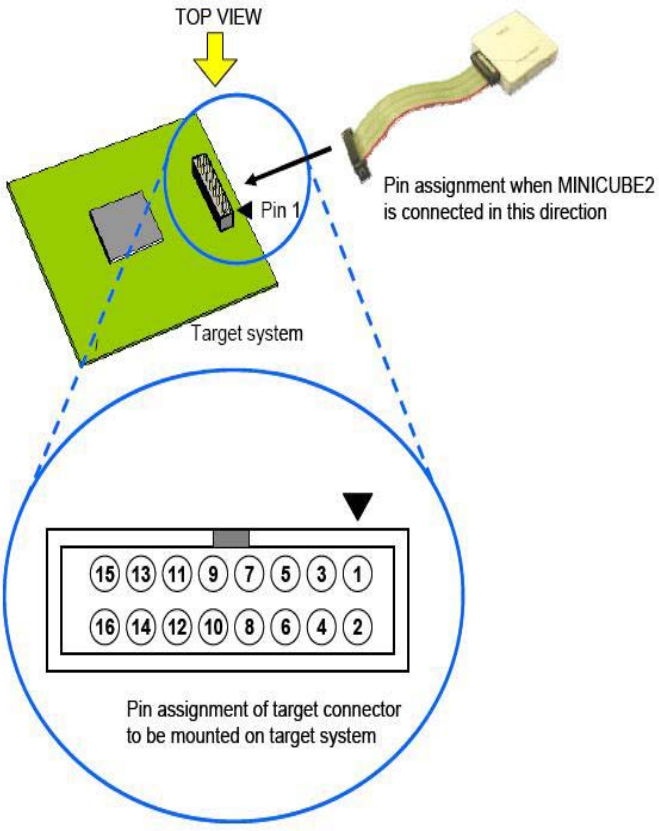


Figure 3-12. FP1, PG-FP4/PG-FP5/Minicube 2 connector

Table 3-18. FP1 pins description

FP1 pin	Function
1	GND
2	RESET
3	SI
4	EVDD
5	SO
6	N.C.
7	N.C.
8	N.C.
9	N.C.
10	N.C.
11	N.C.
12	N.C.
13	N.C.
14	FLMD0
15	RESET_IN
16	CLK_IN

CHAPTER 4 FLASH PROGRAMMING

Flash programming is used to download applications to the internal memory of the 78K0R/KE3-L microcontroller.

This chapter covers the board settings required to perform this operation.

On-chip debugging is documented in the next chapter: Chapter 5. On-Chip debugging

4.1 Flash Programming via MINICUBE2 (QB-MINI2)

When using PG-FP4/PG-FP5 for FLASH programming or QB-MINI2 for FLASH programming /debugging purposes, please configure switch SW7 and SW5 of the *78K0R–Link It!* board as referenced the following tables

Set the Jumpers on the board as shown on **Figure 4-1**: JP1 (VDD=OCD1), JP3 (EVDD=VDD)

JP2 and JP4 are not required if the QB-MINI2 is used to power the board.

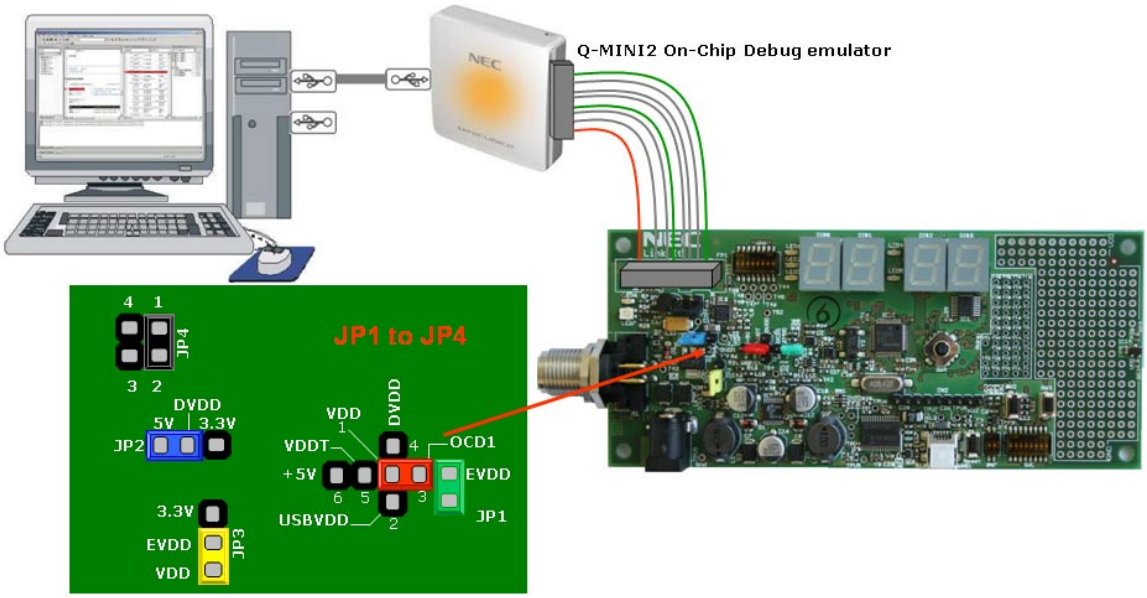


Figure 4-1. Flash Programming via QB-MINI2

Caution The maximum rating current of the MINICUBE2 is 100mA, therefore do not use MINICUBE2 with the target system with the higher current rating.

For applications requiring higher current rate than the maximum rate of the QB-MINI2, please use the power select switch on the Q-MINI2 to set the power supplied to the target system, and use an external power supply to power the board.

Figure 4-2 shows JP1, JP2, JP3, jumper settings used in this configuration:

- The Red Jumper is placed on DVDD as power will be provided by the DC DC converter.
- The Green jumper is placed between EVDD and OCD1 to provide Q-MIN2 with reference voltage
- JP3(EVDD=VDD)
- JP2 can be set to 3.3V or 5V

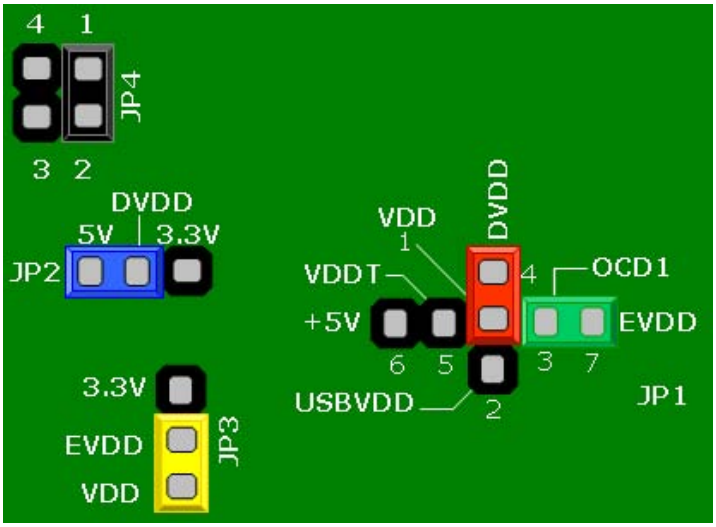


Figure 4-2.Jumper settings for flash programming in high current applications

For more details on the QB-MINI2, please refer to the QB-MINI2 user's manual.

Table 4-1. SW7 configuration for flash programming/On Chip Debugging via QB-MINI2

SW7/Bit1-2	Setting	Power supply
1	ON/OFF (*)	L_VDD = VDD/Disconnected
2	OFF	L_EVDD = Disconnected

(*) = individual selectable by user

Table 4-2. SW5 configuration for FLASH programming via QB-MINI2

SW5 Bit	Setting	Mode
1	ON/OFF (*)	Standalone and debug mode /debug mode only
2	OFF	Disable TK-78K0R debugger functions
3	OFF	Disable TK-78K0R debugger functions
4	ON/OFF (*)	TxD1 connected / disconnected
5	ON/OFF (*)	RxD1 connected / disconnected
6	ON/OFF (*)	SCL0 connected / disconnected
7	ON/OFF (*)	SDA0 connected / disconnected
8	ON/OFF (*)	AVREF applied/not applied
9	OFF	Disable TK-78K0R debugger functions
10	OFF	Disable TK-78K0R debugger functions

(*) = individual selectable by user

4.2 Flash Programming via TK-78K0R debugger

Please refer to Chapter 10. How to use WriteEZ5 FLASH programming software, for information on setting the board for flash programming using the TK-78K0R debugger.

CHAPTER 5 ON-CHIP DEBUGGING

The 78K0R–Link It! board offers two possibilities to use On-Chip debugging (OCD).

The board supports the QB-MINI2 On-Chip debug emulator in order to use the On-Chip debug function of the 78K0R/KE3-L device. The system configuration for On-Chip debugging is shown in figure below.

Alternatively, the TK-78K0R On-Board debugger allows On-Chip debugging without a need of external debug hardware. Within this mode the default USB connection to the Host computer based on the virtual UART driver is used as debug interface. All standard debug functions are available in the On-Board debugging mode like FLASH programming / downloading, code execution, single stepping, breakpoints, memory manipulation etc.

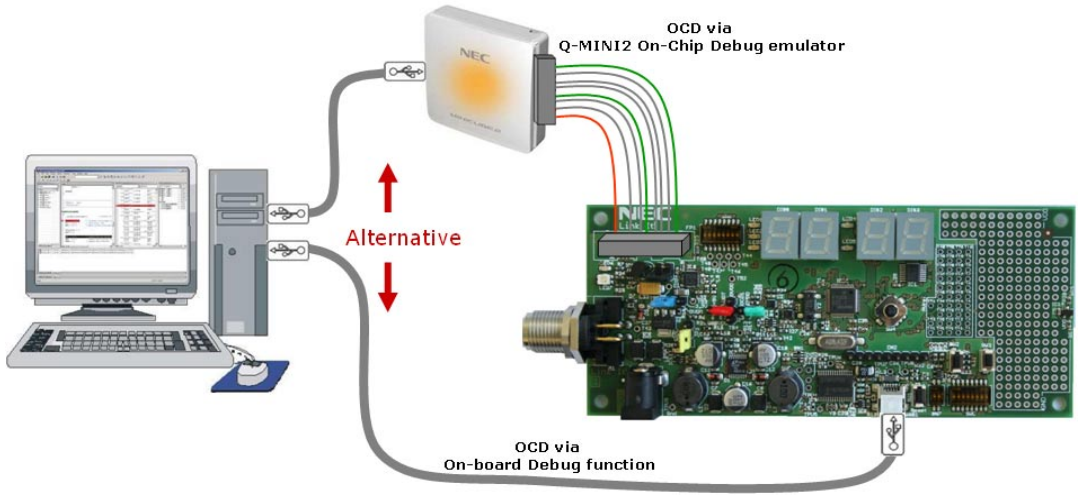


Figure 5-1. On-Chip debugging

5.1 OCD via QB-MINI2 emulator

To operate the board together with the QB-MINI2 On-Chip debug emulator, switch SW5 must be configured as in the following table:

Table 5-1. SW5 configuration for On-chip Debugging via QB-MINI2

SW5 Bit	Setting	Mode
1	ON/OFF (*)	Standalone and debug mode /debug mode only
2	OFF	Disable TK-78K0R debugger functions
3	OFF	Disable TK-78K0R debugger functions
4	ON/OFF (*)	TxD1 connected / disconnected
5	ON/OFF (*)	RxD1 connected / disconnected
6	ON/OFF (*)	SCL0 connected/ disconnected
7	ON/OFF (*)	SDA0 connected/ disconnected
8	ON/OFF (*)	AVREF applied/ not applied
9	OFF	Disable TK-78K0R debugger functions
10	OFF	Disable TK-78K0R debugger functions

(*) = individual selectable by user

5.2 OCD via TK-78K0R On-Board debug function

To operate the board within the On-Board debug mode, switch SW5 and SW7 must be configured as in the following table:

Table 5-2. SW5 Configuration for OCD via TK-78K0R

SW5 Bit	Setting	Mode
1	ON/OFF (*)	Standalone and debug mode /debug mode only
2	ON	Enable On-Board debug function
3	ON	Enable On-Board debug function
4	OFF	TxD1 disconnected
5	OFF	RxD1 disconnected
6	ON/OFF (*)	SCL0 connected/ disconnected
7	ON/OFF (*)	SDA0 connected/ disconnected
8	ON/OFF (*)	AVREF applied/not applied
9	ON	Enable TK-78K0R debugger functions
10	ON	Enable TK-78K0R debugger functions

(*) = individual selectable by user

Table 5-3. SW7 configuration for OCD via TK-78K0R

SW7/Bit1-2	Setting	Power supply
1	ON/OFF (*)	L_VDD = VDD/Disconnected
2	ON	L_EVDD = EVDD

(*) = individual selectable by user

CHAPTER 6 FILES USED FOR EZ-0010 DEVELOPMENT

6.1 Introduction

PM+ and ID78K0R-QB are development tools of 78K0R-Link It! board, and allow building and downloading application programs to the 78K0R-Link It! board.

Additionally the WriteEZ5 can be used for simple FLASH programming of the 78K0R/KE3-L internal FLASH memory.

As communication interface between the PC host system and the *78K0R-Link it!* board a standard USB interface line is needed. Before downloading, debugging, or executing an application program, software must be installed properly.

6.2 Package contents

This package includes following files :

Table 6-1. Contents of the EZ-0010 Package

Package	File name	Contents
Device description file	DeviceDesc.zip	Pictures of IO-Link device description file and board by XML
Device file	df781014_v201.zip	µPD78F1009 device file
Documentation		Manuals include this documentation
Drivers	usbif_v100.zip	USB driver of 78K0R-Link It! board
Sample programs	SamplePrograms.zip	Sample programs for the EZ-0010 • IO-Link RPM sensor demonstration program • IO-Link temperature sensor demonstration program
Software	ra78k0r_w133_j.exe	RA78K0R package
	cc78k0r_w212_j.exe	CC78K0R package
	id78k0r-qb_v360_j.exe	ID78K0R-QB
	WriteEZ5_V100.zip	WriteEZ5
	prm78f1014_v101.zip	µPD78F1009 parameter file

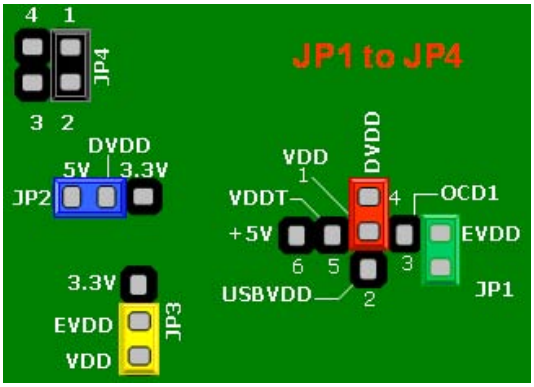
For details of these files, refer to CHAPTER 8 SOFTWARE INSTALLATION.

CHAPTER 7 HARDWARE INSTALLATION

After unpacking the EZ-0010 development kit, connect the board via connector USB1 to your host computer using the provided USB interface cable. When the *78K0R–Link It!* board is connected, the USB driver needs to be installed on the host machine. Please refer to section, **8.8 Installing USB Driver** for more details.

7.1 Factory Settings of jumpers and switches

The picture and tables below show the factory setting for the jumpers and switches on the board.



Mode	SW5									
	1	2	3	4	5	6	7	8	9	10
Stand-alone	ON					ON	ON			
		OFF	OFF	OFF	OFF			OFF	OFF	OFF

Mode	SW6							
	1	2	3	4	5	6	7	8
Transceiver connected	ON	ON	ON	ON	ON	ON	ON	ON

Mode	SW7	
	1	2
Display ON	ON	
USB Interface OFF		OFF

CHAPTER 8 SOFTWARE INSTALLATION

Running the sample programs described in this document requires the following software at a minimum:

(1) WriteEZ5 This is used to write a program to a microcontroller on the 78K0R-Link It! board.

(2) Parameter file This parameter file is for uPD78F1009, which is used with WriteEZ5.

(3) USB driver This is necessary to connect the 78K0R-Link It! board to the host PC.

In addition, the following software is necessary to rebuild or debug the sample programs:

(4) RA78K0R This assembler package for 78K0R includes components such as PM+ and a linker.

(5) CC78K0R This is a C compiler package for 78K0R. Because this is not included in RA78K0R, it must be separately obtained and installed.

(6) ID78K0R-QB This is a debugger package for MINICUBE2. This can also be used to debug the 78K0R-Link It! board. This debugger is used when debugging the sample programs.

When using IO-Link Device Tool, which is manufactured by TMG, the following file is also necessary:

(7) XML description file This is an IO-Link device description file written in XML.

All of the above files can be downloaded from the website below. For details about downloading these files, visit the website.

Website for downloading EZ-0010 IO-Link solution :

http://www.necel.com/micro/en/freesoft/solution/ez10/index.html

This chapter describes how to install the software.

8.1 Installing PM+ and RA78K0R Package

Because PM+ is included in the RA78K0R package, it is installed from this package.

Double-click the installer file for the RA78K0R package (ra78k0r_w*XXX*_e.exe, where *XXX* is the version number), which was downloaded from the website for downloading EZ-0010 IO-Link solution. The following window is displayed.

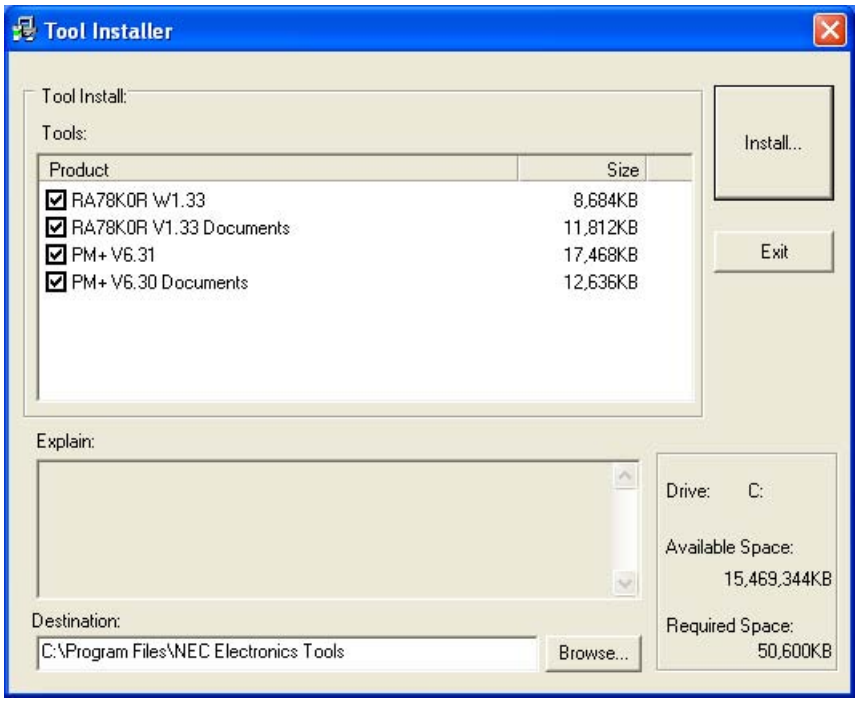


Figure 8-1. RA78K0R Package Installer

In addition to PM+, the RA78K0R package includes LK78K0R (a linker), OC78K0R (an object converter), and other tools necessary to build user-created programs, so select all the check boxes, as shown in Figure 8-1. Next, click the [Install...] button to display the following dialog box, and then click [OK] to start the installation.

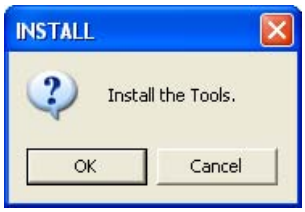


Figure 8-2. Confirmation Dialog Box for Starting Installation

The RA78K0R end user license agreement is displayed in the dialog box below. Read the contents carefully, and then click [Yes] to agree.

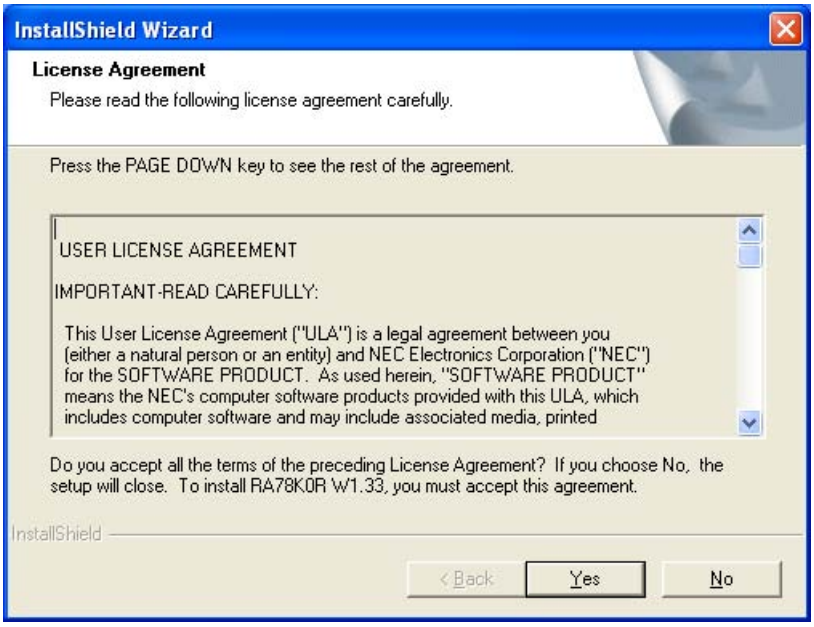


Figure 8-3. RA78K0R Package End User License Agreement

Next, a dialog box for entering the package product ID is displayed. Enter the RA78K0R package product ID provided by NEC Electronics, and then click [Next].

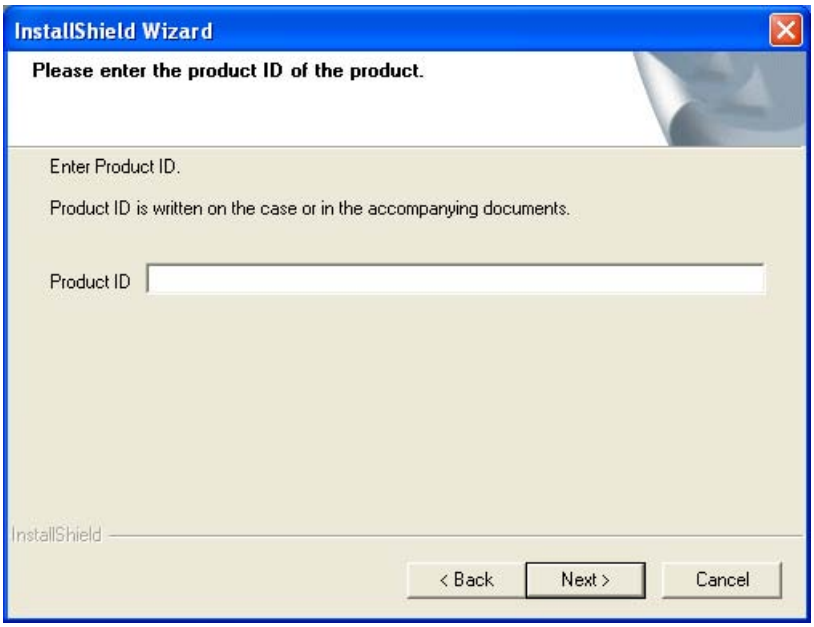


Figure 8-4. Entering Product ID

Wait until the automatic installation of the package finishes. When the installation finishes and the following window is displayed, click the [OK] button.

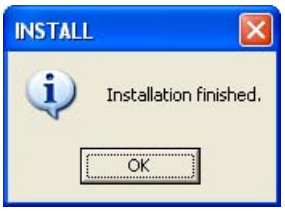


Figure 8-5. Confirmation Dialog Box for Finishing Installation

When the installation finishes and the window shown in Figure 8-1 is redisplayed, click the [Exit] button to close the installer.

This concludes the installation of PM+ and the RA78K0R package.

8.2 Installing CC78K0R Package

Double-click the installer file for the CC78K0R package (cc78k0r_w*XXX*_e.exe, where *XXX* is the version number), which was downloaded from the website for downloading EZ-0010 IO-Link solution. The following window is displayed.

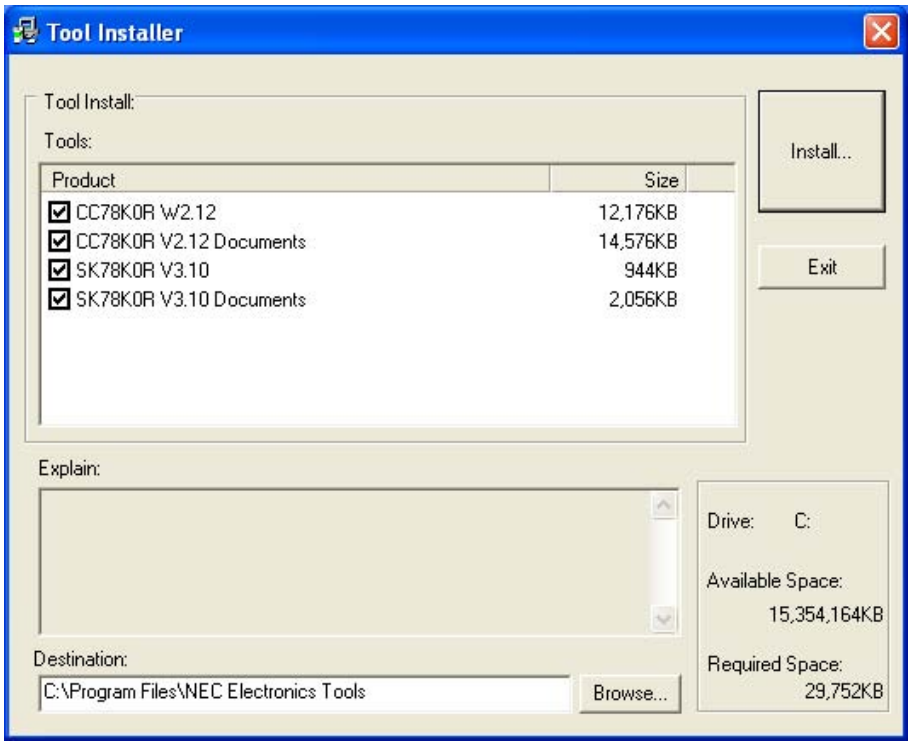


Figure 8-6. CC78K0R Package Installer

Not only CC78K0R but also SK78K0R package is included in this package. This package is a tool used to estimate the stack memory required for the 78K0R. Unless there is a specific reason a tool is unnecessary, select all the check boxes as shown in Figure 8-6.

Next, click the [Install...] button to display the following dialog box, and then click [OK] to start the installation.

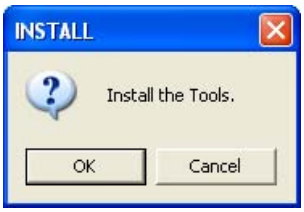


Figure 8-7. Confirmation Dialog Box for Starting Installation

The CC78K0R end user license agreement is displayed in the dialog box below. Read the contents carefully, and then click [Yes] to agree.

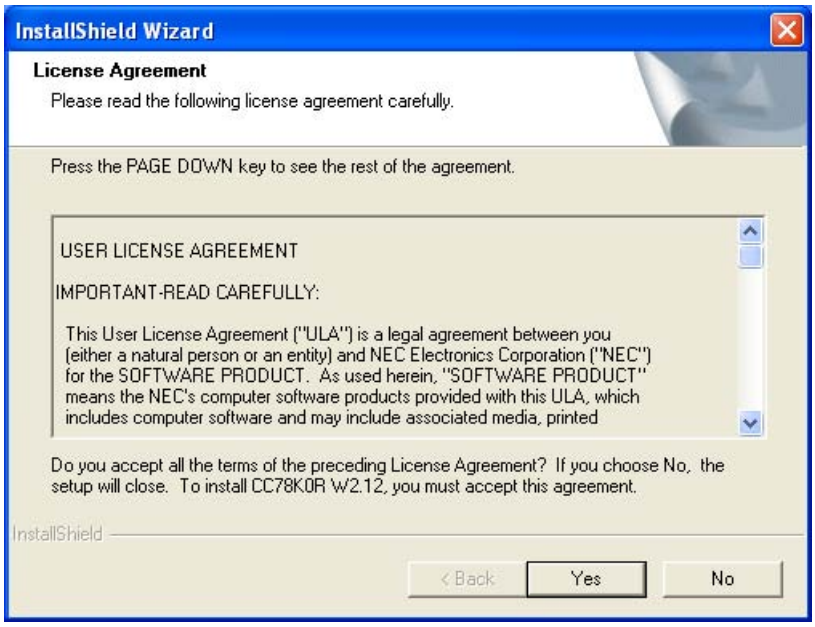


Figure 8-8. CC78K0R Package End User License Agreement

Next, a dialog box for entering the package product ID is displayed. Enter the CC78K0R package product ID provided by NEC Electronics, and then click [Next].

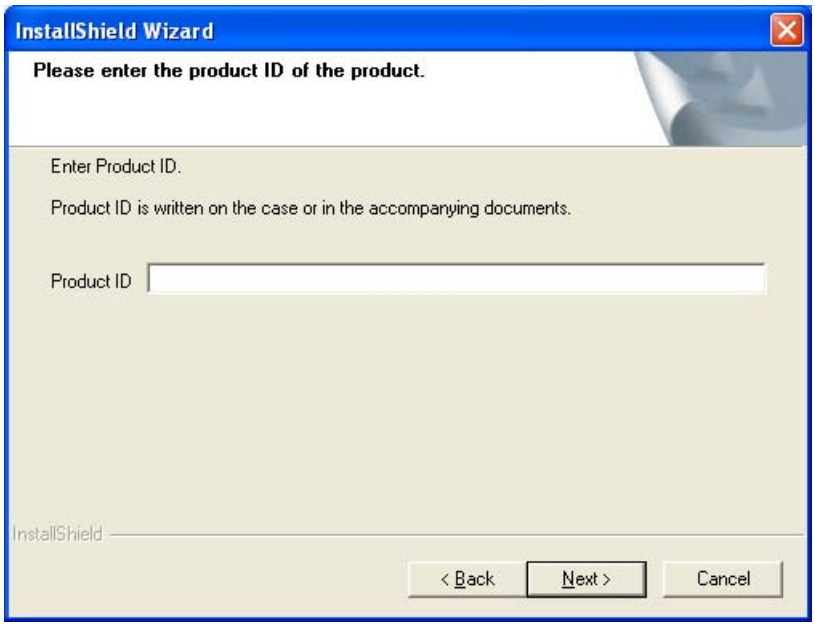


Figure 8-9. Entering Product ID

Wait until the automatic installation of the package finishes. When the installation finishes and the following window is displayed, click the [OK] button.

Figure 8-10. Confirmation Dialog Box for Finishing Installation

When the installation finishs and the window shown in Figure 8-6 is redisplayed, click the [Exit] button to close the installer.

This concludes the installation of CC78K0R package.

8.3 Installing ID78K0R-QB

Double-click the installer file for the ID78K0R-QB (id78k0r-qb_v*XXX*_e.exe, where *XXX* is the version number), which was downloaded from the website for downloading EZ-0010 IO-Link solution. The following window is displayed.

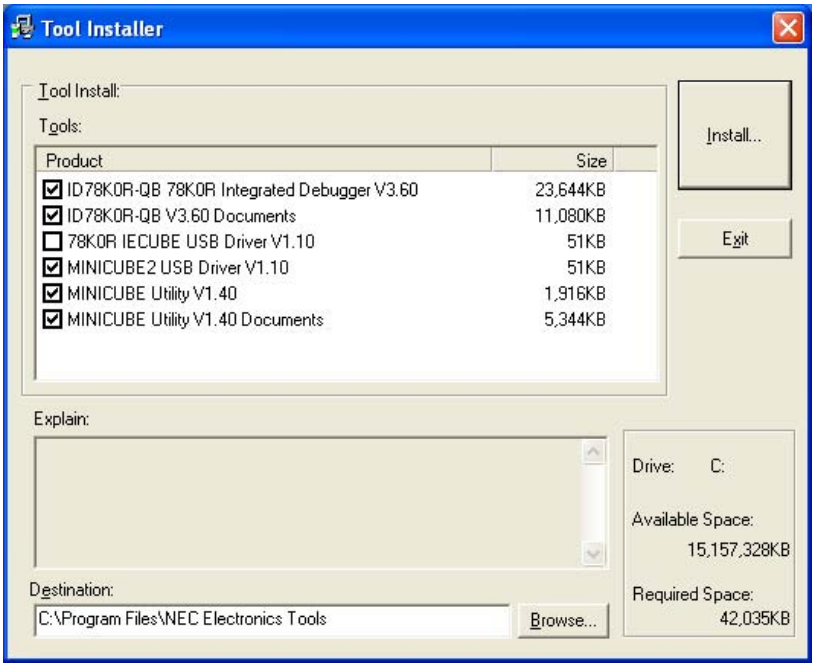


Figure 8-11. ID78K0R-QB Installer

In addition to the ID78K0R-QB debugger software, this package includes a USB driver for MINICUBE2 and utilities. Because these components are necessary when debugging a 78K0R-Link It! board using MINICUBE2, select their check boxes. A USB driver for IECUBE, a fully functional in-circuit emulator for 78K0R, can also be installed, but it is not used in this document, so clear its check box.

Next, click the [Install...] button to display the following dialog box, and then click [OK] to start the installation.

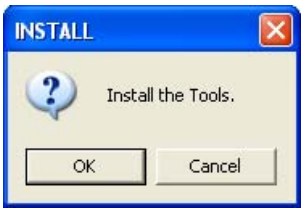


Figure 8-12. Confirmation Dialog Box for Starting Installation

Wait until the automatic installation of the package finishes. When the installation finishes and the following window is displayed, click the [OK] button.

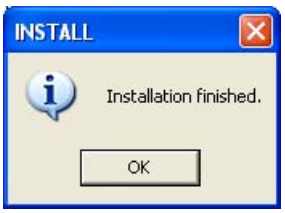


Figure 8-13. Confirmation Dialog Box for Finishing Installation

When the installation finishes and the window shown in Figure 8-11 is redisplayed, click the [Exit] button to close the installer.
This concludes the installation of ID78K0R-QB package.

8.4 Installing Device File

Decompress the file that includes the device file for uPD78F1009 (df781014_v*XXX*.zip, where *XXX* is the version number), which was downloaded from the website for downloading EZ-0010 IO-Link solution.
Next, in the Windows [Start] menu, select [All Programs], [NEC Electronics Tools], and then [Device File Installer] to start the device file installer.

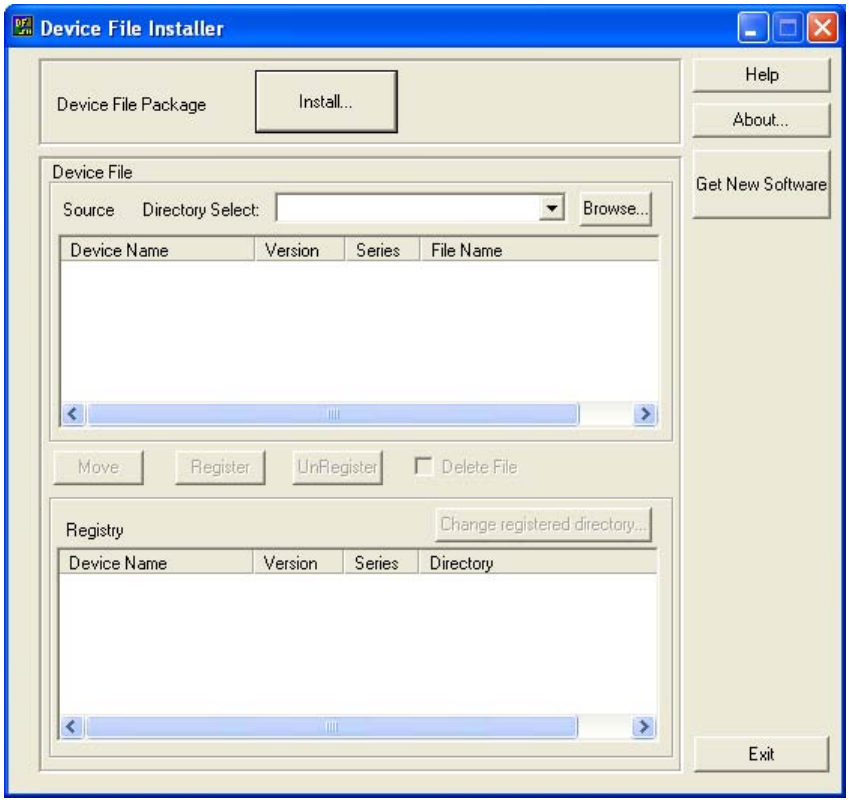


Figure 8-14. Device File Installer

Click the [Install...] button shown in Figure 8-14 to display the dialog box shown in Figure 8-15.
Click the [Browse…] button shown in Figure 8-15, and then open the device file package file “NECSETUP.INI” that was decompressed above.

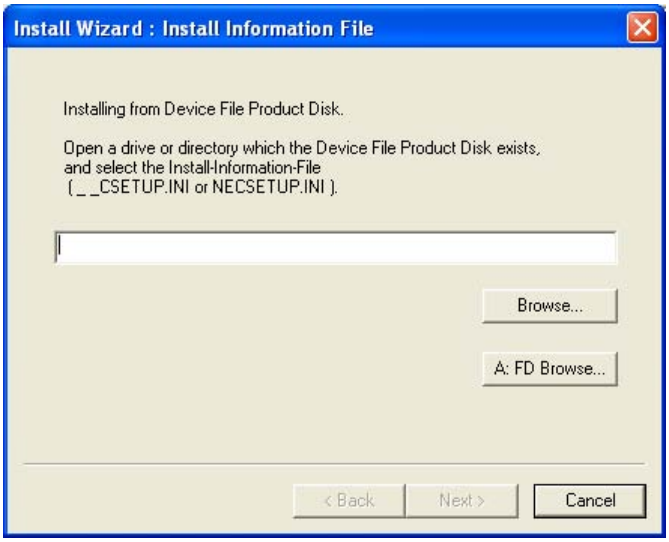


Figure 8-15. Specifying Installation Information File <1>

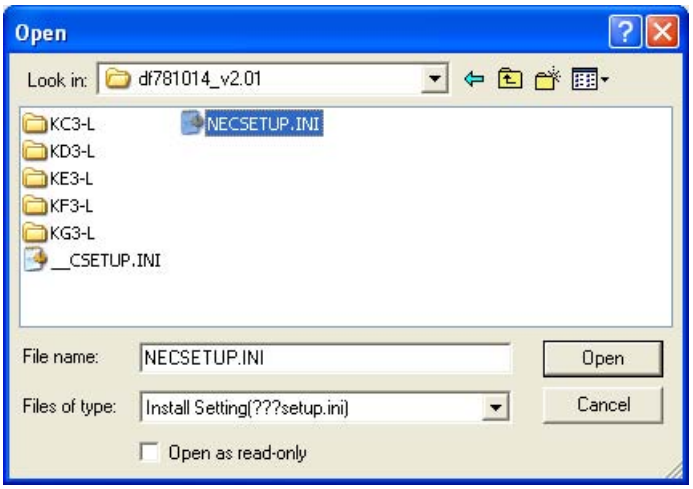


Figure 8-16. Specifying Installation Information File <2>

Select NECSETUP.INI, and then click [Open]. The dialog box shown in Figure 8-17 is displayed.

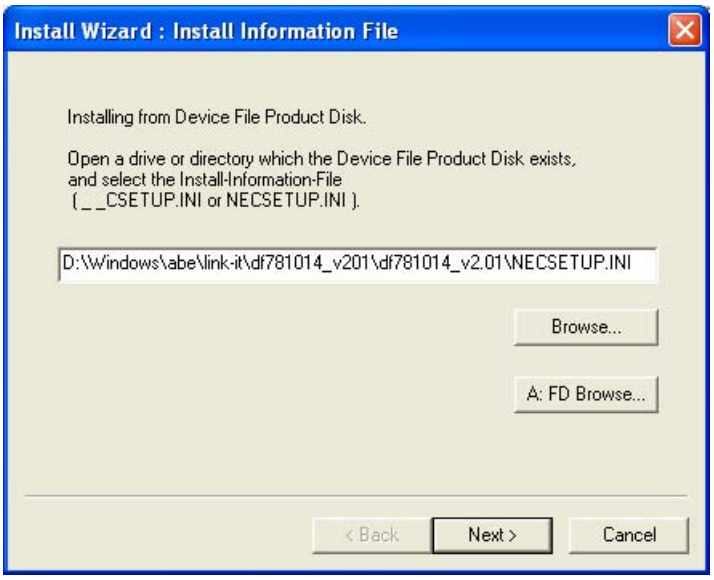


Figure 8-17. Specifying Installation Information File <3>

Click [Next].

Figure 8-18. Software License Agreement

Click [Agree] to agree to the contents displayed in the dialog box and continue.

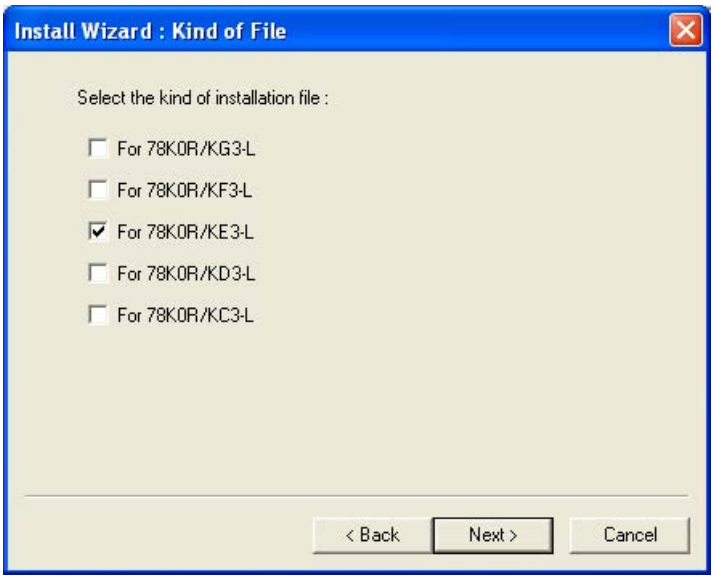


Figure 8-19. Selecting File Type

This package is used for the 78K0R/KE3-L Series uPD78F1009, so select only [For 78K0R/KE3-L], and then click [Next].

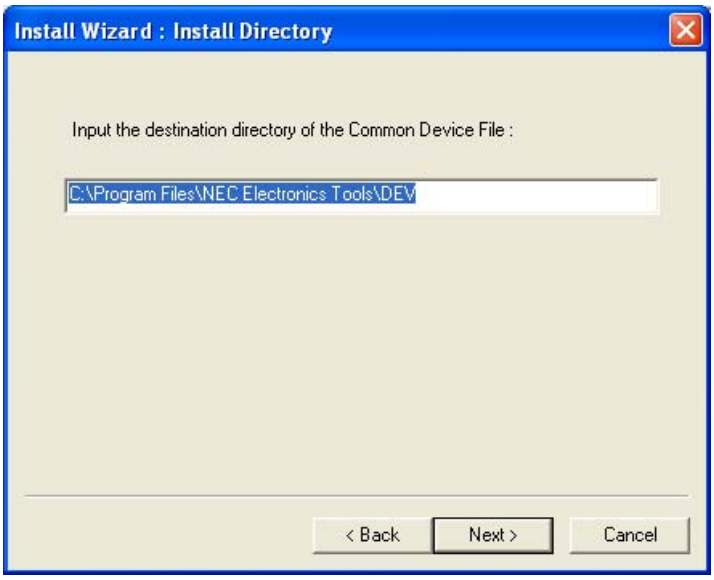


Figure 8-20. Specifying Installation Folder

In the dialog box displayed for selecting the installation folder, click [Next] if the default folder is okay.
(To install the file to a different folder, enter the folder path, and then click [Next].)

Figure 8-21. Starting Installation

In the dialog box displayed for checking the series name and installation folder of the device file to be installed, click [Next] if the information is okay. The device file installation starts.

If the information is incorrect, click [Back] to return to the previous dialog box, and then specify the necessary settings again.

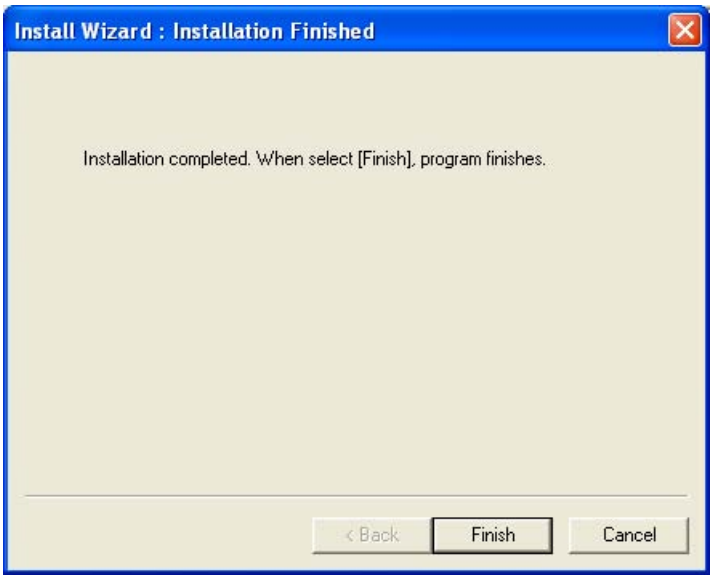


Figure 8-22. Finishing Installation

When the device file installation finishes, click [Finish] in the displayed dialog box for finishing the installation.

When the installation finishes and the window shown in Figure 8-14 is redisplayed, click the [Exit] button to close the installer.

This concludes the installation of the device file.

8.5 Installing WriteEZ5 and Parameter File

8.5.1 Installing WriteEZ5

Decompress the WriteEZ5 package file, which was downloaded from the website for downloading EZ-0010 IO-Link solution, and then save it to an appropriate folder. After decompression, a folder that includes the following contents is generated:

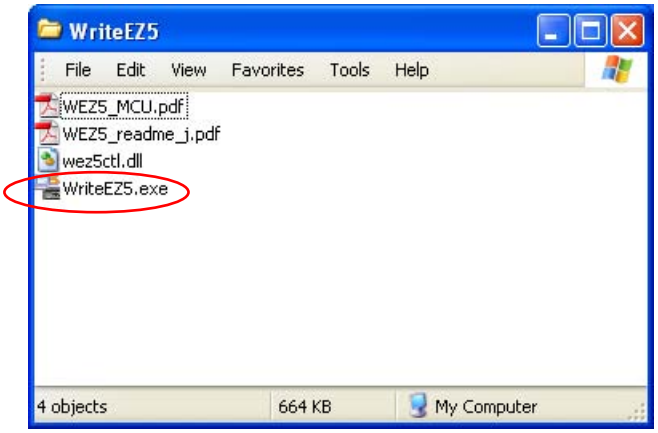


Figure 8-23. WriteEZ5 Package Contents

Double-click the executable file WriteEZ5.exe, which is in the above folder, to start WriteEZ5.

8.5.2 Installing Parameter File for WriteEZ5

WriteEZ5 uses the parameter file for the flash programming performed when using MINICUBE2.

Decompress the parameter file package file, which was downloaded from the website for downloading EZ-0010 IO-Link solution, and then save it to an appropriate folder.

When selecting the parameter file for WriteEZ5, specify this folder.

8.6 Installing EZ-0010 Sample Programs

Decompress the sample program package (SamplePrograms.zip), which was downloaded from the website for downloading EZ-0010 IO-Link solution, and then save it to an appropriate folder.

8.7 Installing XML Device Description File

Decompress the device description file package (DeviceDesc.zip) for the sample programs, which was downloaded from the website for downloading EZ-0010 IO-Link solution.

(1) NEC_Electronics_Europe_IOD_0x018C_0x18C000.jpg
This image file is for the 78K0R-Link It! board. Because the same board is used, this file is used for both the temperature sensor and RPM sensor.

(2) NEC_Electronics_Europe_IOD_0x018C_0x18C000.XML
This is an IO-Link device description file of the temperature sensor application.

(3) NEC_Electronics_Europe_IOD_0x018C_0x18C001.XML
This is an IO-Link device description file of the RPM sensor application.

The above files are used by IO-Link Device Tool, which is manufactured by TMG. Copy these three files to the following folder:

C:¥Program Files¥TMG¥IO-Link Device Tool¥Devices¥IO-Link¥

Note that this folder is not generated unless IO-Link Device Tool, which is manufactured by TMG, is installed. Please contact TMG for details on this software.

8.8 Installing USB Driver

In order to use the 78K0R-Link *It!* board, the USB driver needs to be installed on the host machine. Install the driver according to the following procedure:

8.8.1 Decompressing Driver Package

Decompress the USB driver package package (usbif_v100.zip), which was downloaded from the website for downloading EZ-0010 IO-Link solution, and then save it to an appropriate folder. Continue to 8.8.2 if installing the driver in Windows 2000, or continue to 8.8.3 if installing the driver in Windows XP.

8.8.2 Installing By Windows2000

When the 78K0R Link-it!board is connected to host PC firstly, the dialog box showed during installation is displayed.

Next, the Windows "Found New Hardware Wizard" automatically starts.

Figure 8-24. Found New Hardware Wizard (Windows2000)

Click [Next].

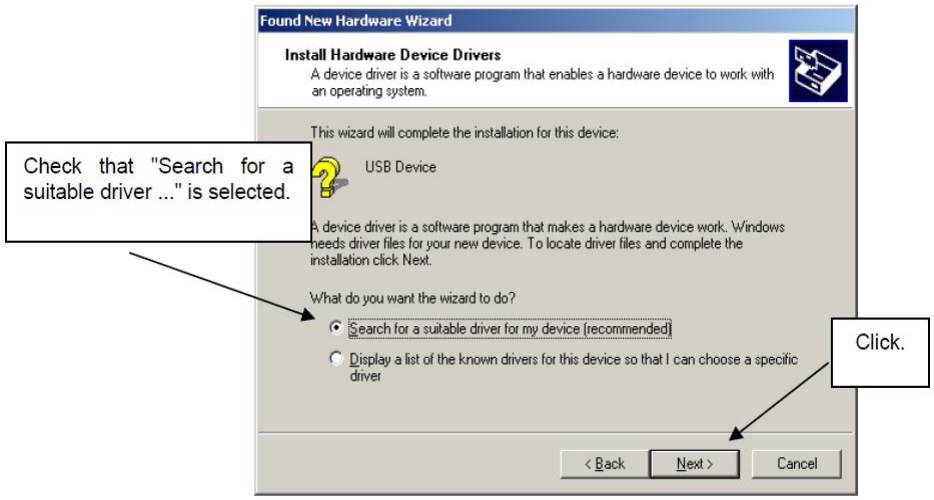


Figure 8-25. Selecting Search Method (Windows2000)

In the displayed dialog box for specifying the method for searching for the driver, which is shown in Figure 8-25, select the [Search for a suitable driver for my device (recommended)] radio button, and then click [Next].

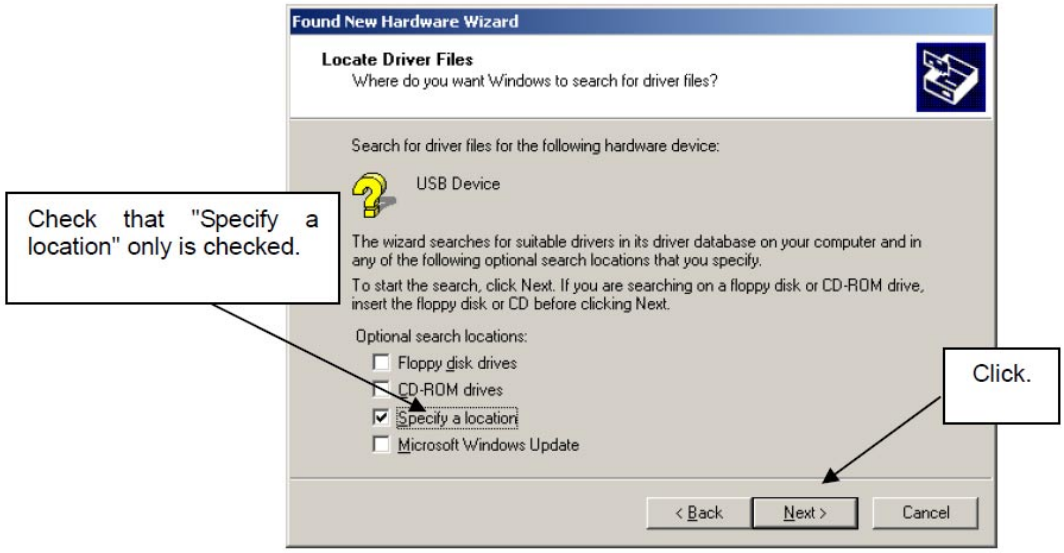


Figure 8-26. Specifying Where to Search for Driver File (Windows2000)

In the displayed dialog box for specifying where to search for the driver file, which is shown in Figure 8-26, select [Specify a location], and then click [Next].

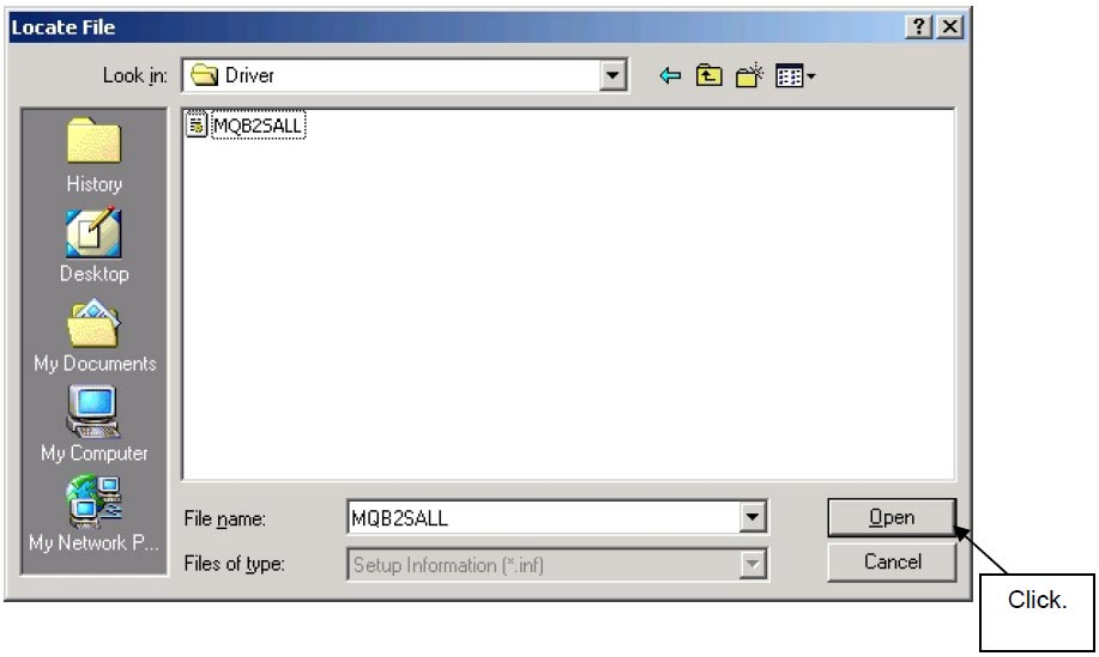


Figure 8-27. Specifying Driver Information File (Windows2000)

In the window that opens for specifying the driver information file, which is shown in Figure 8-27, open the folder in which the driver package was decompressed in 8.1.1, select the "MQB2SALL.inf" file, and then click [Open].

Figure 8-28. Driver File Copy Source (Windows2000)

In the dialog box shown in Figure 8-28, click [OK].

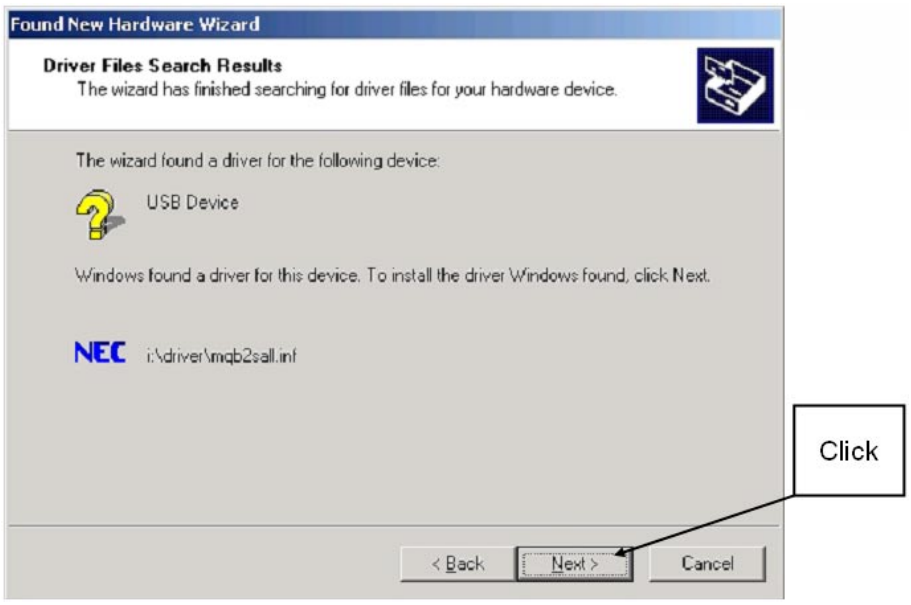


Figure 8-29. Finishing Driver Detection (Windows2000)

In the displayed dialog box for finishing driver detection, which is shown in Figure 8-29, click [Next].

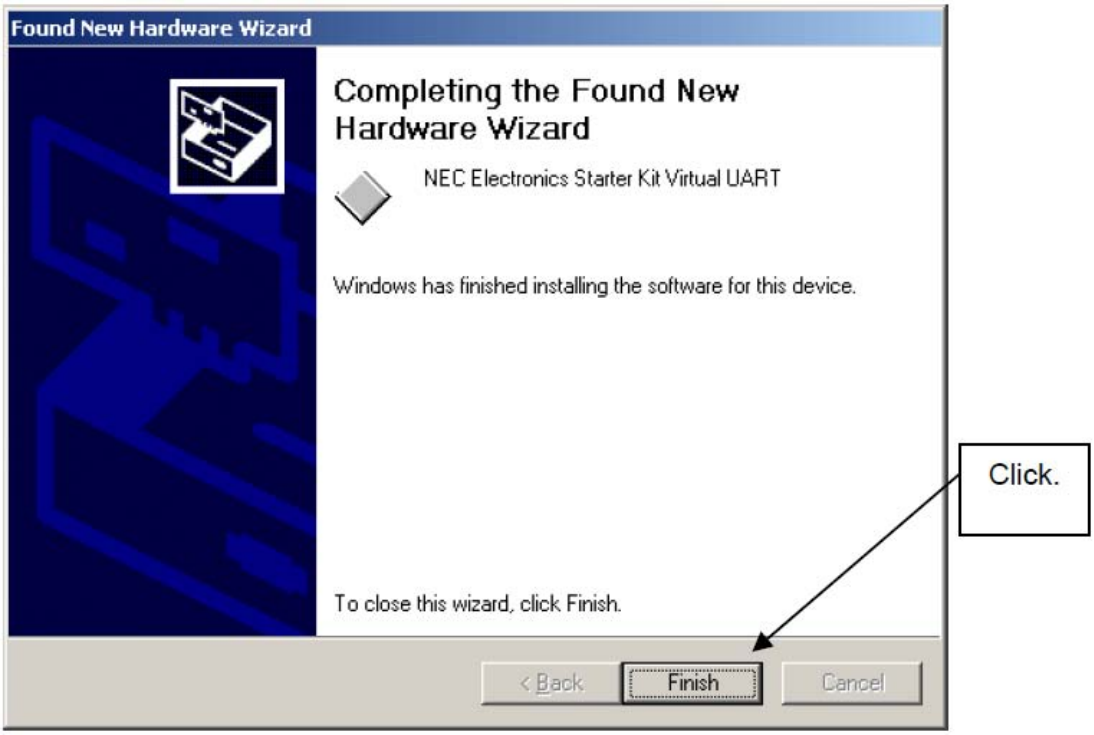


Figure 8-30. Finishing Driver Installation (Windows2000)

Concludes the installation of the driver, the dialog is displayed shown in Figure 8-30. Click [Finish].

This concludes the installation of the driver by Windows2000.

8.8.3 Installation on Windows XP

When the 78K0R Link-it! board is connected to host PC firstly, the following window is displayed.

Figure 8-31. Welcome to the Found New Hardware Wizard (WindowsXP)

Shown in Figure 8-31, select the [Install from a list or specific location (Adbanced)] radio button, and then click [Next].

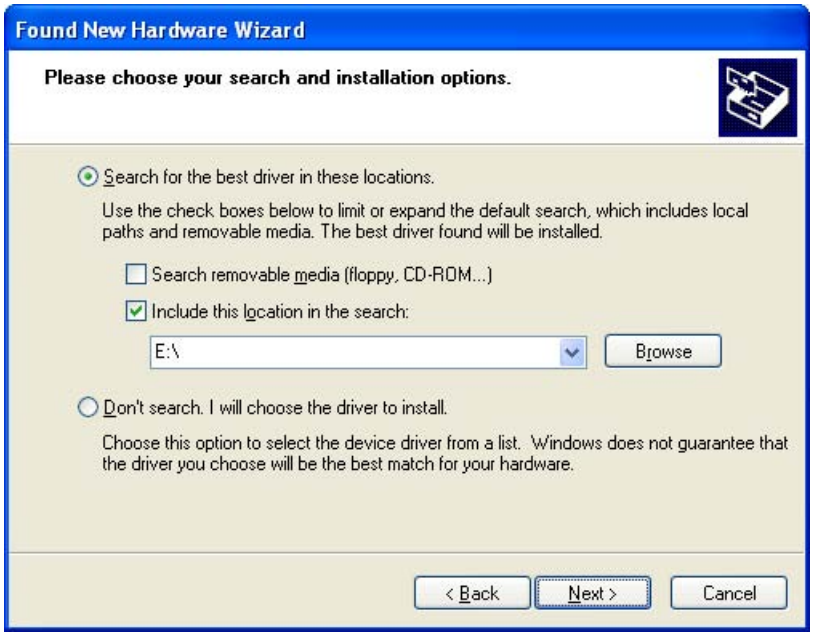


Figure 8-32. Choosing Search Method (WindowsXP)

Select the "Don't search. I will choose the driver to install." radio button shown in Figure 8-32, and then click [Next].

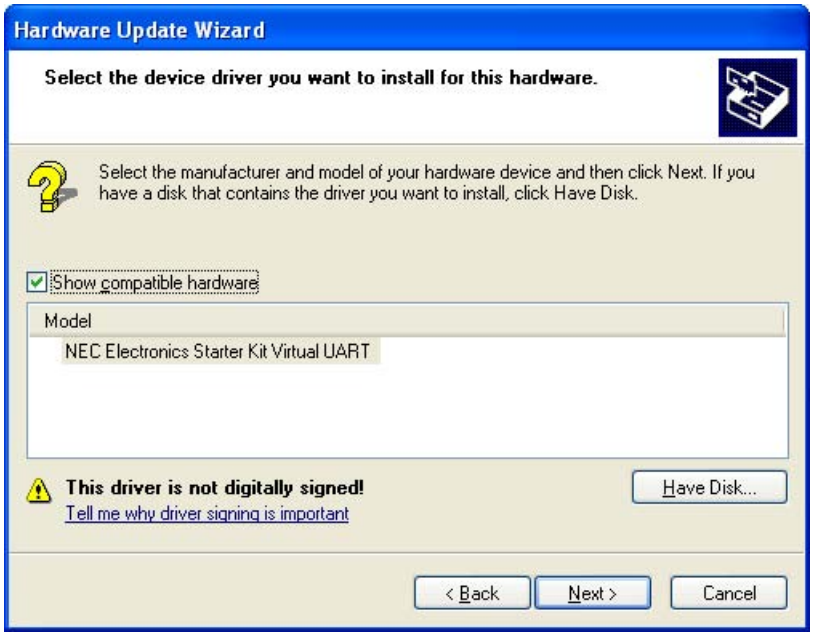


Figure 8-33. Choosing Driver (WindowsXP)

The dialog is displayed shown in Figure 8-33. Click [Have Disk...].

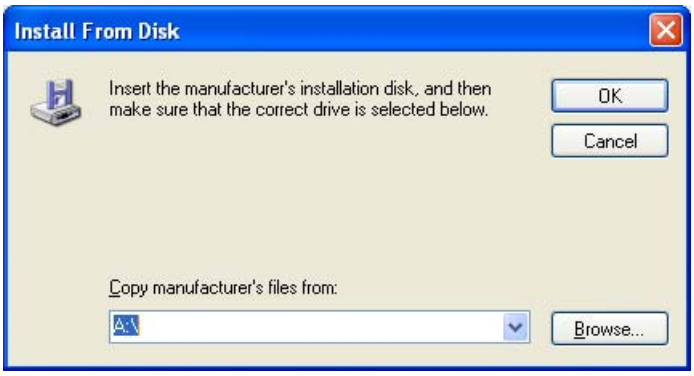


Figure 8-34. Manually Specifying File to Copy from Disk (WindowsXP)

In the window shown in Figure 8-34, click [Browse…], and then browse to the folder decompressed in 8.8.1.

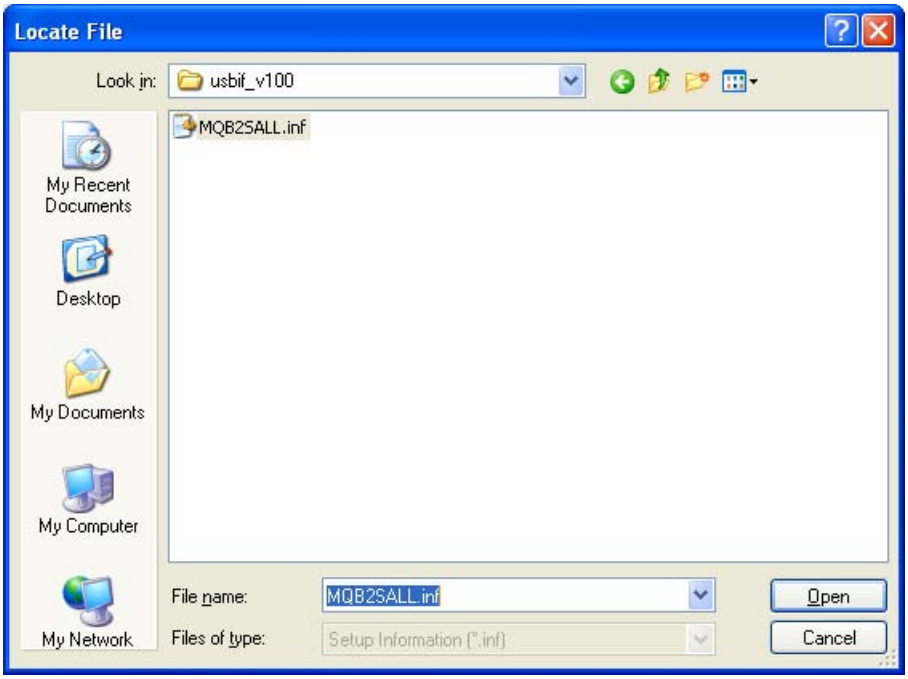


Figure 8-35. Specifying Folder in Which Driver Was Decompressed (WindowsXP)

In the displayed file selection window, browse to the folder in which the driver was decompressed, select MSB2SALL.inf, and then click [Open].

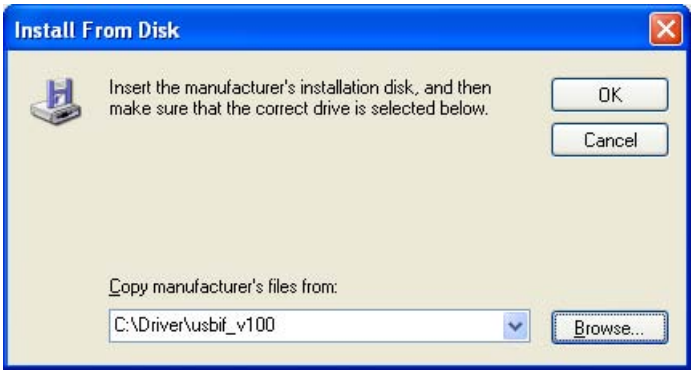


Figure 8-36. Finishing Specifying Folder (WindowsXP)

The dialog is displayed shown in Figure 8-36. Click [OK]. The dialog is displayed shown in Figure 8-37.

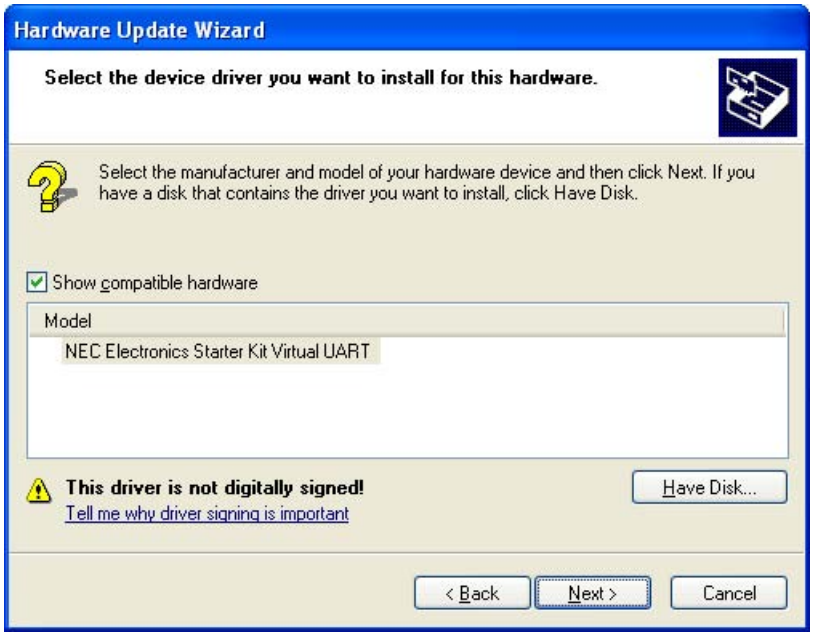


Figure 8-37. Choosing Driver Display (WindowsXP)

In the dialog box shown in Figure 8-37, click [Next].

Figure 8-38. Windows Logo Test Warning (WindowsXP)

The Windows Logo Test warning shown in Figure 8-38 is displayed, but click [Continue Anyway].

Figure 8-39. Dialog Box Displayed During Driver Installation (WindowsXP)

When the driver installation starts, wait a while. When the installation finishes, the following window is displayed,

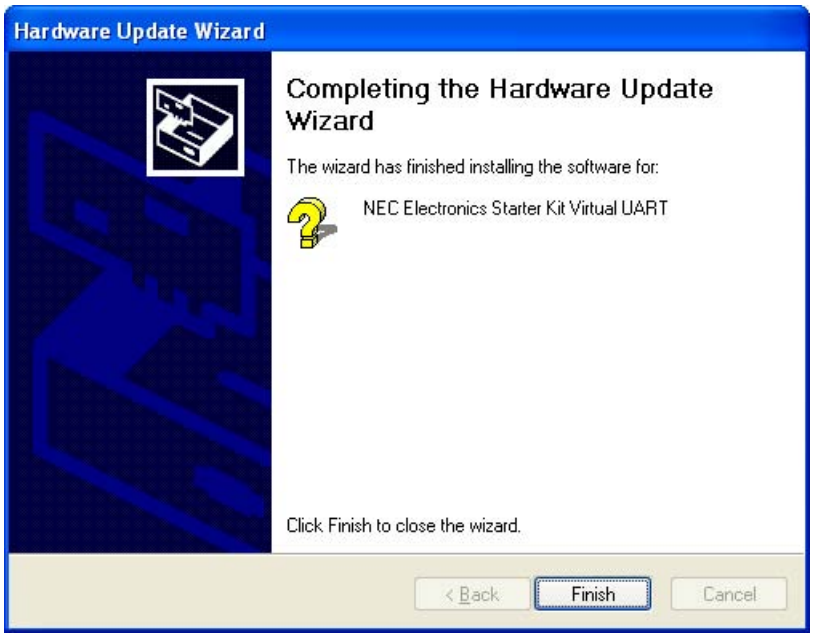


Figure 8-40. Dialog Box Displayed When Driver Installation Finishes (WindowsXP)

Concludes the installation of the driver. Click [Finish] button to close the found new hardware wizard. This concludes the installation of the driver by WindowsXP.

8.9 Confirmation Installing USB Driver

After installing the USB driver, open the Windows 2000 or Windows XP "Device Manager".

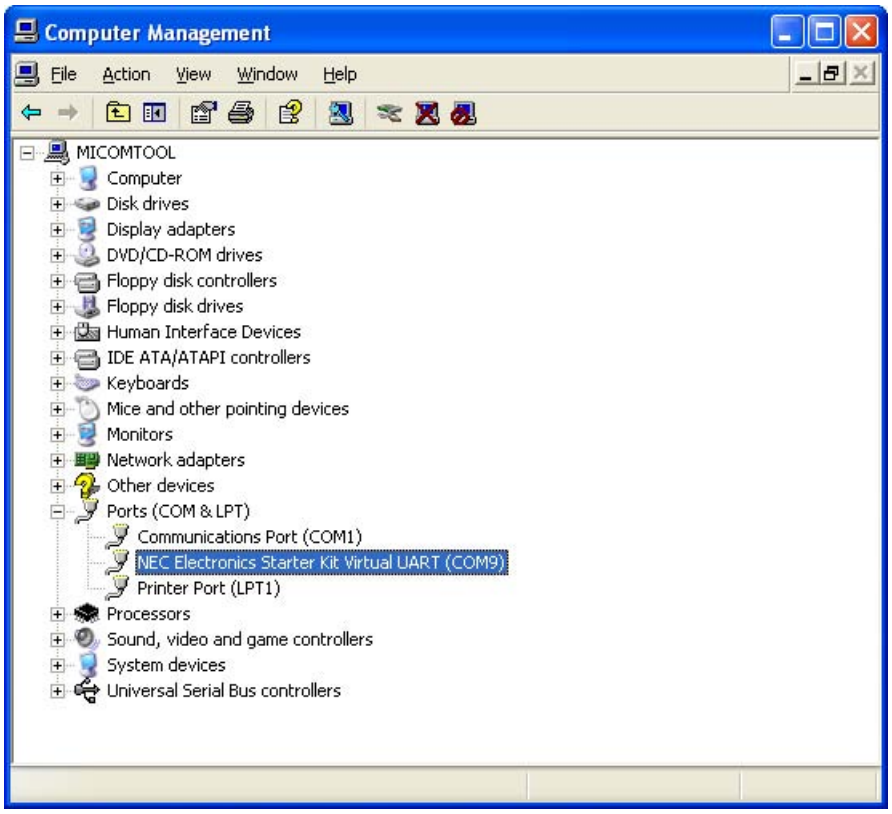


Figure 8-41. Checking Device Manager (WindowsXP)

If "NEC Electronics Starter Kit Virtual UART (COM*x*)" (for which the COM number *x* differs according to the environment) is displayed in the [Ports (COM & LPT)] tree in Device Manager as shown in Figure 8-41, the driver was installed without a problem.

CHAPTER 9 FLASH PROGRAMMING SOFTWARE WRITEEZ5

WriteEZ5 is a flash programming software to flash HEX files to the related device. For installation information refer to the **8.5.1 Installing WriteEZ5**.

9.1 Starting up the GUI Software

Double-click WriteEZ5.exe, which is in the folder generated in **8.5.1 Installing WriteEZ5**, to start the program. When the GUI software is started normally, the following screen appears.

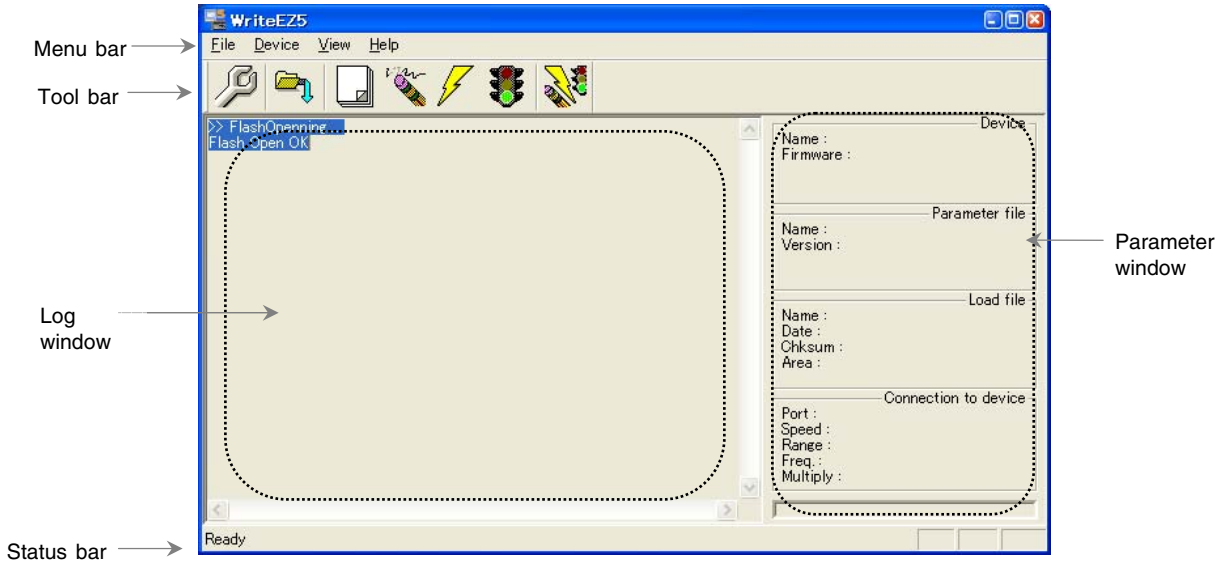


Figure 9-1. WriteEZ5 Startup

This window consists of the following items:

Table 9-1. WriteEZ5 Windows items

Name	Display Information
Menu bar (displayed at the top)	Displays menu items executable by the WriteEZ5
Toolbar. (displayed under the menu bar)	Displays frequently used commands as icons.
Action log window (displayed under the toolbar)	Displays an WriteEZ5 action log
Programmer parameter window (Displayed to the right of the action log window).	Displays programming parameter settings.
Status bar	Displays status..

9.2 Toolbar

The toolbar contains buttons for starting the important procedures of the WriteEZ5.

Table 9-2. Toolbar Buttons

	[Device] → [Setup] button
	[File] → [Load] button
	[Device] → [Blank Check] button
	[Device] → [Erase] button
	[Device] → [Program] button
	[Device] → [Verify] button
	[Device] → [Auto procedure(EPV)] button

9.3 Menu Bar

Depending on the actual device status and device type, some menu items may be enabled or disabled.

9.3.1 [File] menu

Clicking the [File] menu displays the pull-down menu as shown below. This menu mainly contains commands related to file operation.

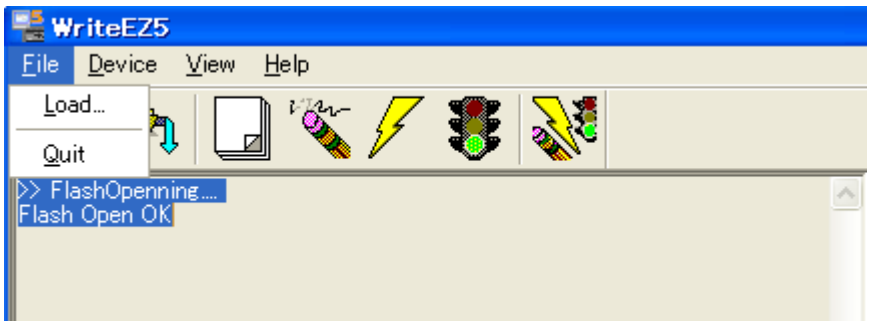


Figure 9-2. [File] menu

(1) [Load] command

The [Load] command allows you to select a program file. The selected program file is programmed into the flash memory of the device by executing the [Program] command or [Autoprocedure (EPV)] command.

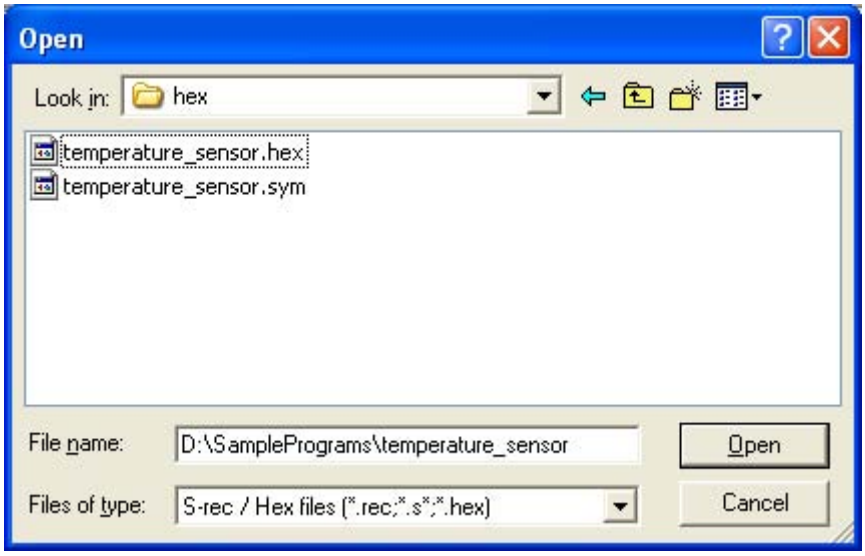


Figure 9-3. HEX File Selection Window

The file selection window for program loading displays the most recently used directory to which a user program has been loaded. After a user program is loaded, a checksum calculation is made and the result is displayed in the programmer parameter window.

[Open button], selects a user program as a program to be written to the target device.

[Cancel button], closes the window without selecting a program.

(2) [Quit] command

The [Quit] menu is the command for terminating the WriteEZ5 GUI software. Clicking the [×] box on the top right side of the task bar also terminates the WriteEZ5 GUI software.

User settings are saved in the WriteEZ5.INI file, so that the GUI software starts up next time with the same settings.

Notes **WriteEZ5.INI is created in the Windows folder when Windows XP is used.**

When Windows 2000 is used, WriteEZ5.INI is created in the Winnt folder.

.

9.3.2 [Device] menu

Clicking the [Device] menu displays the pull-down menu as shown below. This menu mainly contains commands for programming operations such as deletion, programming, and verification on the target device.

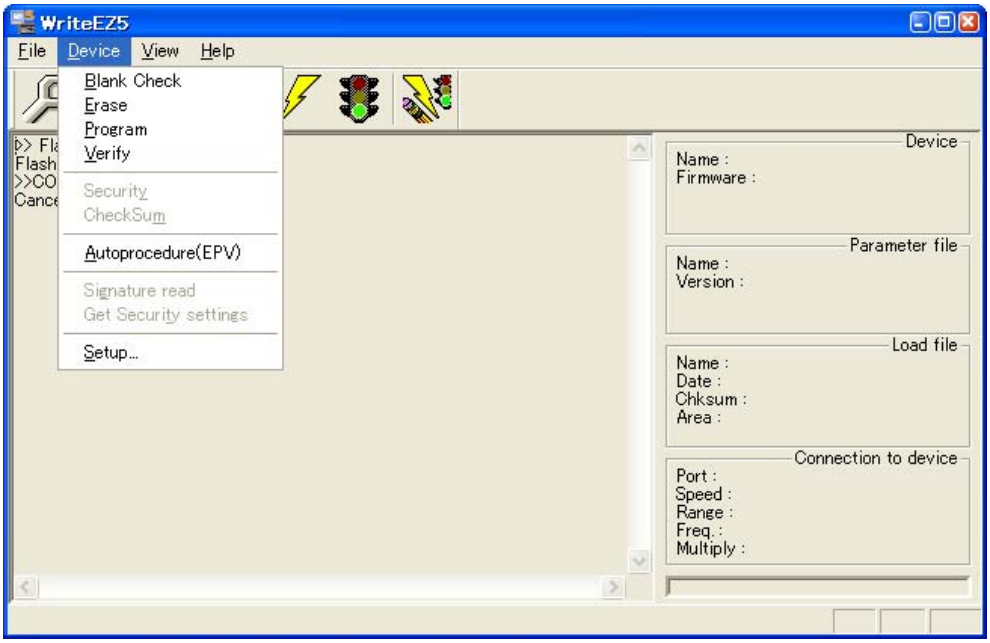


Figure 9-4. [Device] menu

(1) [Blank Check] command

The [Blank Check] command allows you to make a blank check on the target device connected to the WriteEZ5. If the flash memory of the target device is erased, a blank check is terminated normally. If the flash memory is not completely erased, the indication "not blank" is provided. Before starting programming, erase the flash memory of the target device.

(2) [Erase] command

The [Erase] command erases the flash memory of the target device connected to the WriteEZ5. While the flash memory is being erased, the progress status is displayed in the action log window to indicate programmer operation. The execution on the [Blank Check] command before the [Erase] command is executed follows the setting of 'Command options' of the Advance tab displayed by selecting [Device] → [Setup].

Upon completion of [Erase] command execution, the GUI software displays the result of executing the command on the target device.

(3) [Program] command

The [Program] command sends a specified user program to the target device and writes the program to the flash memory. The execution of Verify operation for detecting an error in user program communication from the WriteEZ5 to the target device after the execution of the [Program] command follows the setting of the 'Command options' on the Advance tab displayed by selecting [Device] → [Setup].

During programming, the progress status is displayed in the action log window to indicate programmer operation. This progress status display window displays the progress status on target device programming by percentage.

Upon completion of [Program] command execution, the GUI software displays the result of executing the command on the target device.

(4) [Verify] command

The [Verify] command sends a specified user program to the target device connected with the WriteEZ5, and performs verification against the data written to the flash memory of the target device. During verification, the progress status is displayed in the action log window to indicate programmer operation. This progress status display window displays the progress status of target device verification by percentage.

Upon completion of [Verify] command execution, the GUI software displays the result of executing the command on the target device.

(5) [Security] command

The [Security] command initiates the programming of the security flag of the target device connected to the WriteEZ5. Before executing this command, the security flags must be specified by specifying the various settings for the "Security flag settings" field, which is found by selecting the [Device] menu, [Setup], and then the "Advance" tab.

(6) [Checksum] command

The [Checksum] command reads the checksum value of the target device connected with the WriteEZ5.

This value differs from the value displayed in the parameter window of the main window.

(7) [Autoprocedure (EPV)] command

The [Autoprocedure (EPV)] command executes the [Erase] command, [Program] command and [Verify] command in succession.

To automatically execute read verification after executing the [Autoprocedure (EPV)] command, select "Read verify after Program" in the "Command option" field, which is found by selecting the [Device] menu, [Setup], and then the "Advance" tab.

During EPV execution, the progress status is displayed in the action log window to indicate programmer operation. For a selected command, its execution operation, and messages, refer to Chapter 10.

Upon completion of [Autoprocedure (EPV)] command execution, the GUI software displays the result of executing the command on the target device.

(8) [Signature read] command

The [Signature read] command reads the signature information (device name, flash memory information, and so forth) of the target.

9.3.3 [Setup] command

The [Setup] menu allows you to make settings related to flash memory rewriting according to the user environment and to set command options. Each time the GUI software is started, the most recently used parameter file (.PRM) is read and the settings are displayed.

(1) Standard setup

This menu is used to set the environment for rewriting the flash memory of the target device.

The mode of communication with the target, the operating clock, and so forth differ depending on the device used. For details, refer to the manual of the device used, when making settings.

The window shown below is opened.

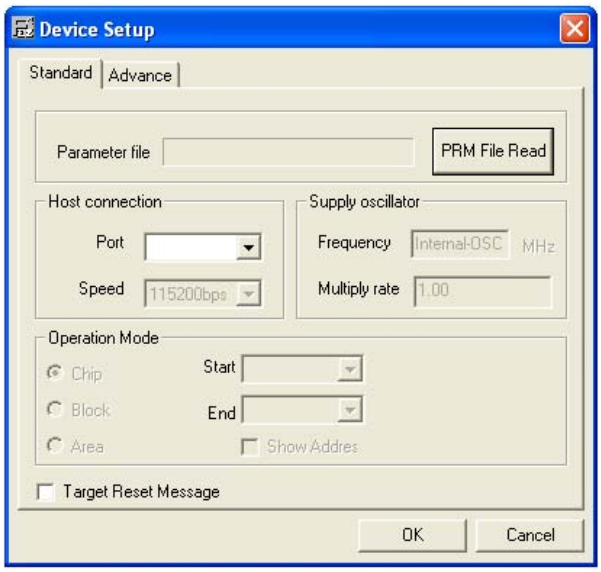


Figure 9-5. Device Setup Window - Standard

This window shows all basic options that can be set in accordance with the user environment and target device.

[OK button]: Clicking the OK button saves the settings on the Standard and Advance menus and closes the window.

[Cancel button]: Clicking the Cancel button closes the window without saving the settings on the Standard and Advance menus.

(1-1) Parameter file

This file holds parameters and timing data required to rewrite the flash memory of the target device. Do not modify the data in the parameter file because the data is related to the guarantee of rewrite data.

The parameter file is protected by the checksum function. If the checksum result indicates an error, the WriteEZ5 does not accept the parameter file.

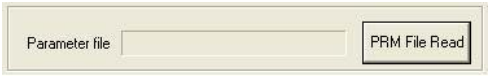


Figure 9-6. Device Setup Window - Parameter File Selection

[PRM File Read button]: A window for specifying a parameter file is displayed.

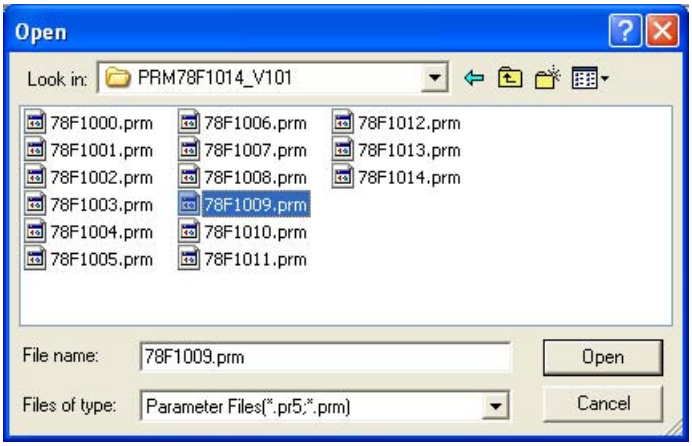


Figure 9-7. Parameter File Selection Window

(1-2) Host connection

"Host connection" is used to select a channel for communication between the *78K0R–Link It!* board and host machine.

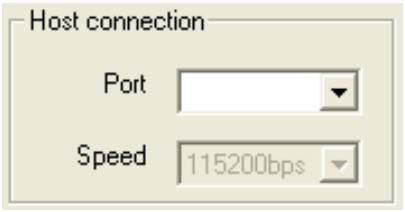


Figure 9-8. Device Setup Window – Host connection

[Port list box]

Select a channel for communication between the *78K0R–Link It!* board and host machine.

- COM1 to COM256

Remark Selectable ports can be checked using Device Manager. For details, refer to **8.9 Confirmation of Installing USB Driver**.

[Speed list box]

Select a communication rate for the selected communication channel from the following:

- 9600 bps
- 19200 bps
- 38400 bps
- 115200 bps

Remark For selectable communication rates, refer to the user's manual of the device used.

(1-3) Supply oscillator

"Supply oscillator" is used to select a clock that determines programming, data transfer, and a transfer rate.

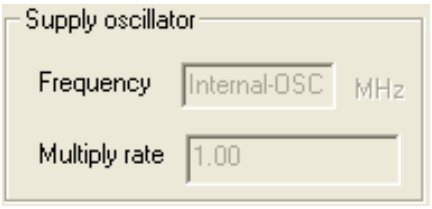


Figure 9-9. Device Setup Window - Supply Oscillator Selection

[Frequency box]

Specify the target system clock frequency. The specifiable frequency range differs according to the device. Therefore, check the target device specifications before specifying the frequency.

[Multiply rate]

Specifies the division rate or multiplication rate of the target device. If the target device has an on-chip PLL circuit, enter a division rate or multiplication rate according to the use environment. The selectable division rate or multiplication rate differs depending on the device.

Check the specifications of the device used before making a setting.

If the target device does not have an on-chip PLL circuit, select "1.0".

On the initial screen, the default setting is displayed according to the parameter file.

(1-4) Operation Mode

The setting of "Operation Mode" may divide the flash memory of some target devices into blocks or areas.

This menu is used to select an operation mode of the flash memory. Some devices do not have the block and area division modes, and some devices have only one of the modes. In these cases, a nonexistent mode is not selectable.

Figure 9-10. Device Setup Window - Operation Mode

[When Chip is selected]

The entire flash memory area of the target device is subject to rewrite processing.

[When Block is selected]

Specify the Block number range subject to rewrite processing by using Start/End.
The Start/End list boxes display the Block numbers where the flash memory of the target device is configured.

[When Area is selected]

Specify the Area number range subject to rewrite processing by using Start/End.
The Start/End list boxes display the Area numbers where the flash memory of the target device is configured.

[Show Address check box]

Specify whether numbers or addresses are displayed in the Start/End list boxes.
If this check box is checked, addresses are displayed.
If this check box is not checked, numbers are displayed.

(1-5) Target Reset Message

By checking the Target Reset Message check box, the window promoting the reset operation manually is displayed even when the reset signal cannot be connected to the target cable.

Figure 9-11. Device Setup Window – Target Reset Message

(2) Advance setup

The Advance setup menu is used to specify the command options and security flag settings.

When "Advance" is clicked, the following window is displayed:

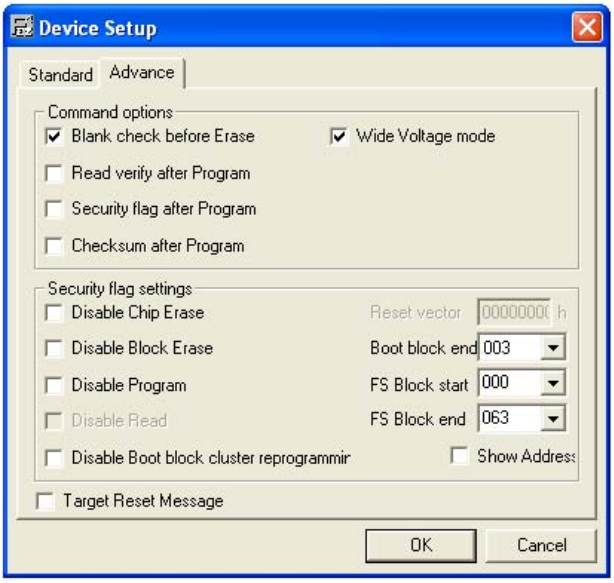


Figure 9-12. Device Setup Window – Advance

(2-1) Command options

This dialog box is used to specify the WriteEZ5 flash processing command options.

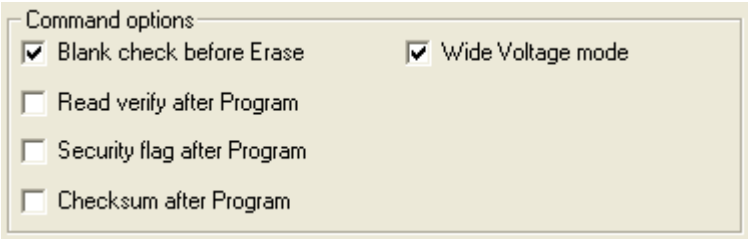


Figure 9-13. Device Setup Window - Command options

[Blank check before Erase check box]

If this check box is checked, blank check is made before the Erase command or EPV command is executed.

If the result of a blank check indicates OK, erase processing is not executed.

[Read verify after Program check box]

If this check box is checked, write data is sent from the programmer after execution of the Program command and EPV command, then the data is verified against the data written to the flash memory.

[Security flag after Program check box]

If this check box is checked, automatic programming of the selected security flag is executed after execution of the [Program] and [Autoprocedure (EPV)] commands.

[Checksum after Program check box]

If this check box is checked, the flash memory checksum value of the target device is read from the target device after execution of the Program command and EPV command.

This value differs from the value displayed in the parameter window of the main window.

(2-2) Security flag settings

The <Security flag settings> dialog box is used to specify which security function is valid.

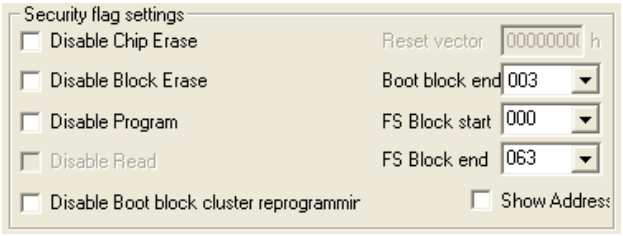


Figure 9-14. Device Setup Window – Security flag settings

Caution The following table is the correspondence between the [Erase] and [Program] Commands when the security functions of a 78K0R/KE3-L microcontroller are valid.

	Chip Erase Command	Block Erase Command	Program Command
Disable Chip Erase	Invalid	Invalid	Valid (Note1)
Disable Block Erase	Valid	Invalid	Valid
Disable Program	Valid	Invalid	Invalid
Disable Boot block cluster reprogramming	Invalid	Valid (Note2)	Valid (Note2)

Notes

1. **Since the [Erase] command is invalid, the data that differs from the data already written in the flash memory cannot be written.**
2. **Valid only for the area other than the area specified as the boot area**

[Disable Chip Erase check box]

If this check box is checked, the [Erase] command becomes invalid in the entire flash memory area of the target device.

At this time, the warning message shown below is displayed.

Figure 9-15. Device Setup Window – Disable Chip Erase

Caution Be aware that if the security flag is set in the target device, neither erasing nor writing to the device can be enabled afterward!!!

[Disable Block Erase check box]

If this check box is checked, the [Erase] command becomes invalid in all the blocks of the flash memory selected under Operation Mode in the Standard Setup menu. This setting is cleared by the [Erase] command when Chip was selected under Operation Mode.

[Disable Program check box]

If this check box is checked, the [Program] command becomes invalid, as does the [Erase] command in all the blocks of the flash memory selected under Operation Mode in the Standard Setup menu.

The [Erase] command for the entire flash memory area is valid. This setting is cleared by the [Erase] command when Chip was selected under the Operation Mode.

[Disable Boot block cluster reprogramming check box]

If this check box is checked, the boot area is set with the boot block set in the Boot block Cluster setting as the last block. At this time, the warning message shown below is displayed.

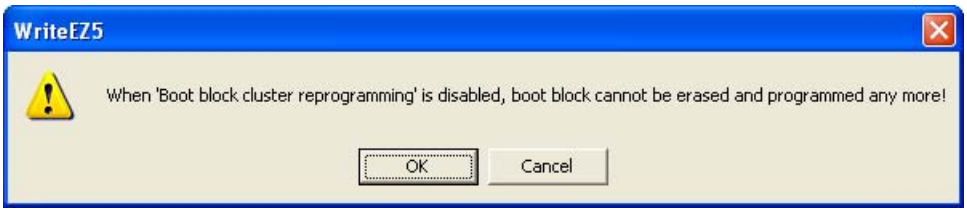


Figure 9-16. Device Setup Window – Disable Boot Cluster Reprogramming warning

Caution Be aware that if the security flag is set in the target device, the boot area cannot be rewritten to the device afterward!!!

9.3.4 [View] menu

Clicking the [View] menu displays the pull-down menu shown below.

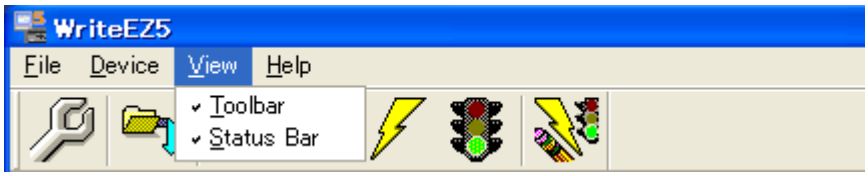


Figure 9-17. [View] Menu

This menu contains commands for setting whether to display the toolbar and status bar.

(1) [Toolbar] command

Checking the [Toolbar] command displays the toolbar. Unchecking the command hides the toolbar.

(2) [Status Bar] command

Checking the [Status Bar] command displays the status bar. Unchecking the command hides the status bar.

9.3.5 [Help] menu

Clicking the [Help] menu displays the following pull-down menu:

Figure 9-18. [Help] Menu

(1) [About WriteEZ5] command

The [About WriteEZ5] command opens the program entry window as shown below and indicates the version.

Clicking OK terminates the display.

Figure 9-19. About WriteEZ5 Window

9.4 Programmer Parameter Window

This window displays the settings of the programming parameters.

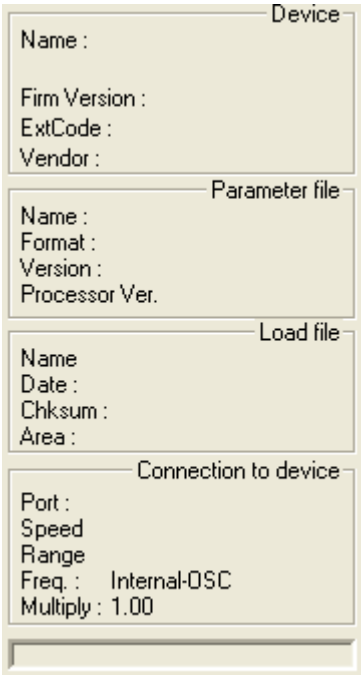


Figure 9-20. Programmer Parameter Window

[Device]

Updated after communication with the target device to display information about the target device.

[Parameter file]

Updated after [Setup] command execution to display information about a read parameter file.

[Load file]

Updated after [Load] command execution to select information about a selected program file.

[Connection to device]

Updated after [Setup] command execution to display information about the connection with the target device.

CHAPTER 10 HOW TO USE WRITEEZ5 FLASH PROGRAMMING SOFTWARE

This chapter explains the basic operation of the WriteEZ5 GUI for programming the *78K0R-Link it!* board. It covers how to start the system, execute the EPV command, and program the target device. Furthermore, the FLASH programming of the 78K0R/KE3-L via WriteEZ5 is shown in the following pages.

The conditions of the series of operations described in this chapter are as follows:

Hardware configuration of EZ-0010:

Base board: *78K0R-Link it!* board

Target device: 78K0R/KE3-L (μPD78F1009)

Clock: Internal Oscillator

Voltage level: 5 V

Software configuration of WriteEZ5:

Parameter file: 78F1009.PRM

Clock setting: Internal Oscillator

Port: COM13 (115200 bps) (The COM number differs according to the environment.)

Operation mode: Chip

Write HEX: temperature_sensor.hex

Option setting: Blank check before Erase

10.1 Installing the WriteEZ5 GUI software

Install the WriteEZ5 GUI software on the host machine you are using, by referring to section **8.5.1 Installing WriteEZ5** (if the software has not been installed yet).

10.2 Installing the driver

Install the USB driver on the host machine you are using, by referring to section **8.8 Installing USB Driver** (if the driver has not been installed yet).

10.3 Installing the parameter file

The parameter file (*.prm) for the μPD78F1009 can be downloaded from the NEC Electronics website.

Copy the parameter file downloaded from the NEC Electronics Web site into sub-directory <WriteEZ5.EXE-install-path>¥PRM created during GUI software setup (If the folder does not exist, create it.) (refer to section **8.5.1 Installing WriteEZ5**).

10.4 Connecting and starting

Set the *78K0R-Link it!* board to the WriteEZ5 FLASH programming mode by switching SW5, SW7 and jumpers JP1 to JP4 to the following setting:

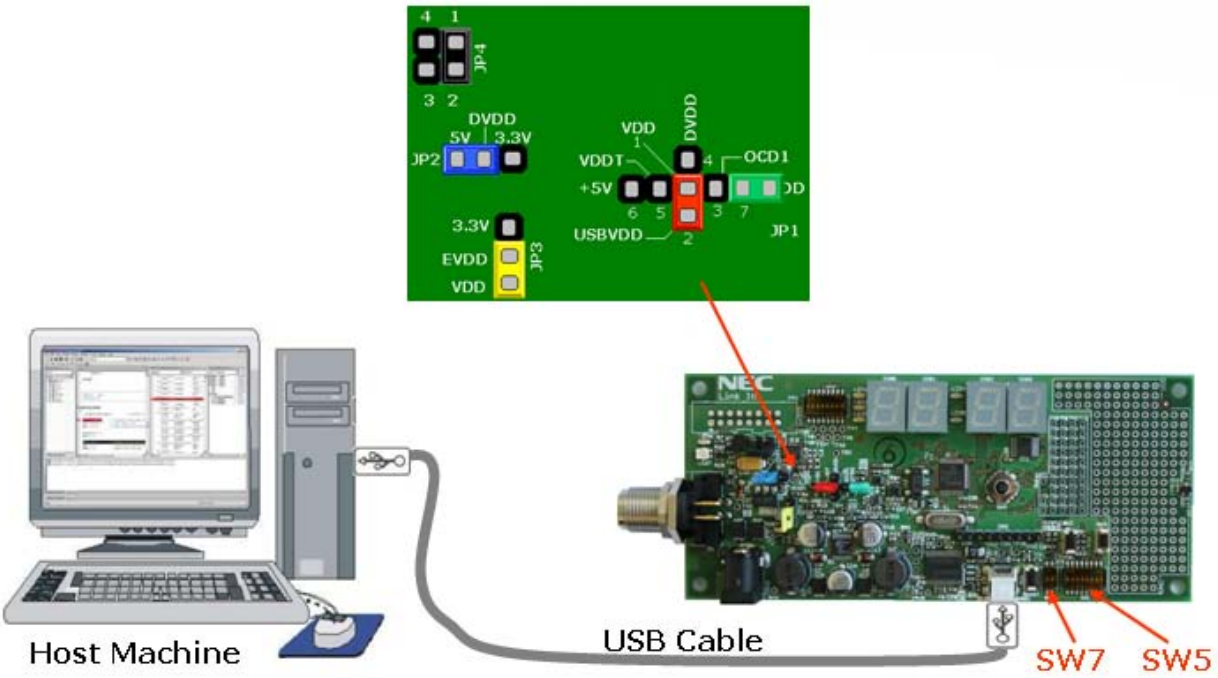


Figure 10-1. WriteEZ5 FLASH programming mode connection

Table 10-1. WriteEZ5 FLASH programming mode

Mode	SW5										SW7	
	1	2	3	4	5	6	7	8	9	10	1	2
WriteEZ5		ON	ON			Don't care			ON	ON	Don't care	ON
	OFF			OFF	OFF							

Use a USB cable to establish a plug and play connection between the 78K0R-Link It! board and host PC.

10.5 Start the WriteEZ5 GUI.

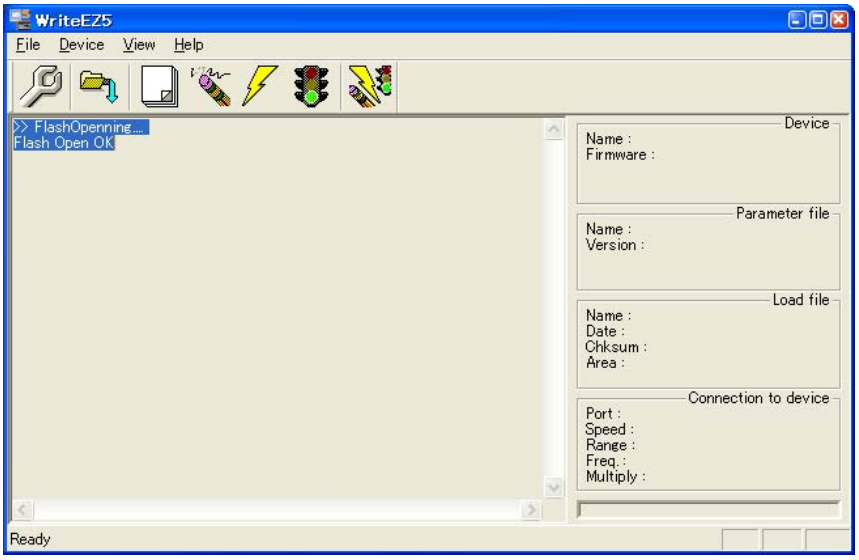


Figure 10-2. GUI Software Startup Screen

10.6 Setting the programming environment

Select [Device] → [Setup] from the menu bar.

The Standard dialog box for device setup is activated.

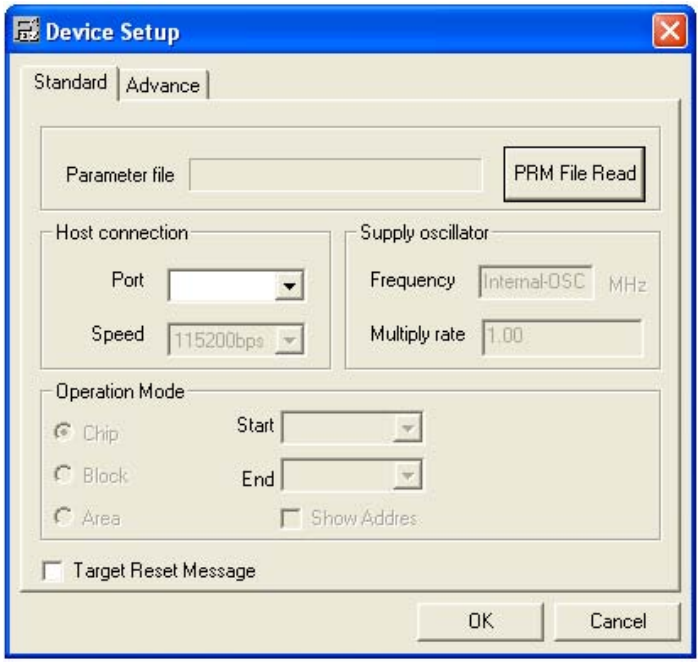


Figure 10-3. <Standard Device Setup> Dialog Box

Click PRM File Read to open the parameter file selection window.

Select the parameter file “78F1009.prm” then click Open.

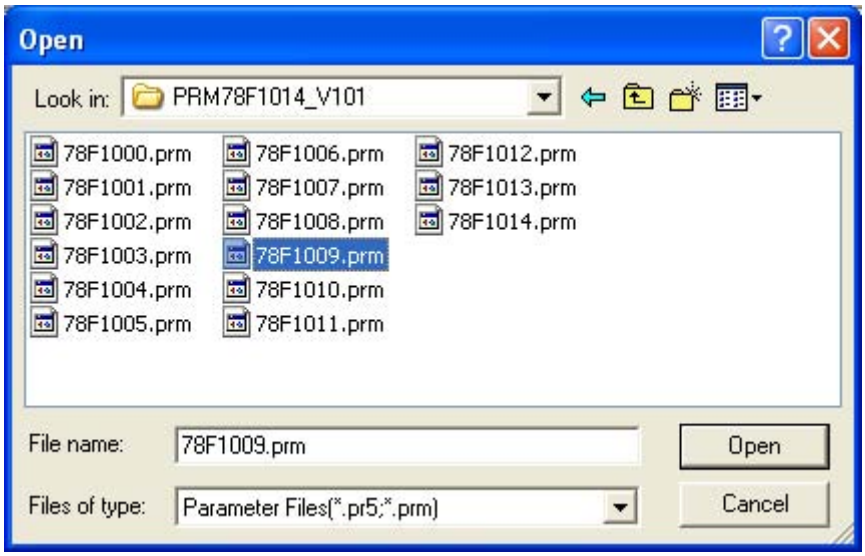


Figure 10-4. Parameter File Selection

From the Port list box, select the communication port that matches the host machine being used. Select the communication speed of the Host connection.

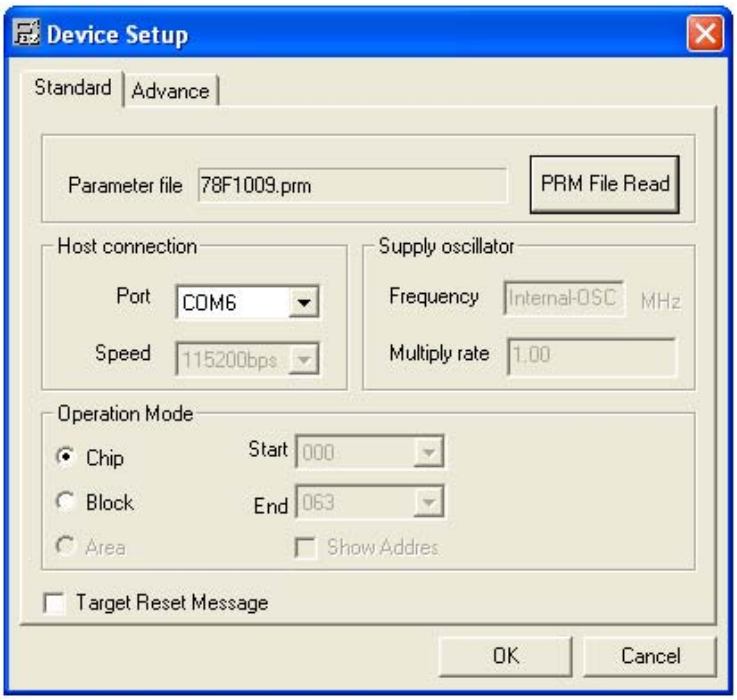


Figure 10-5. Port Selection

Remark Selectable ports can be checked using Device Manager. For details, refer to 8.9 Confirmation Installing USB Driver.

Click the OK button. The GUI software sets the parameters.
When the settings have been completed, the following screen is displayed:

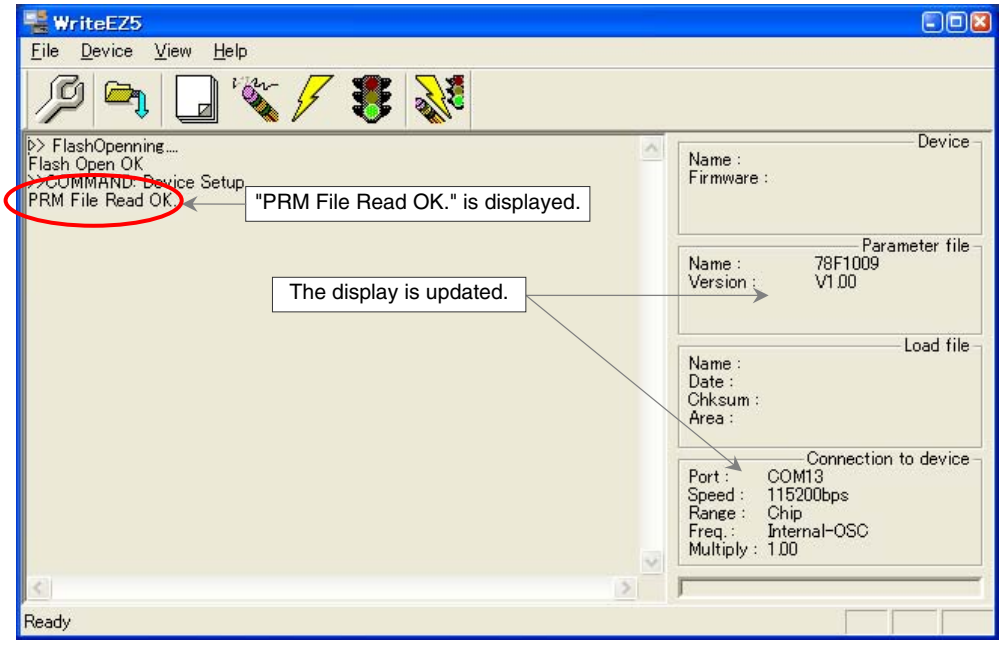


Figure 10-6. Completion of Parameter Setting

10.7 Selecting a user program

Select [File] → [Load]. In the user-created program selection window, select the user-created program file to write to the target device, and then click the [Open] button. The user-created program is downloaded to WriteEZ5.

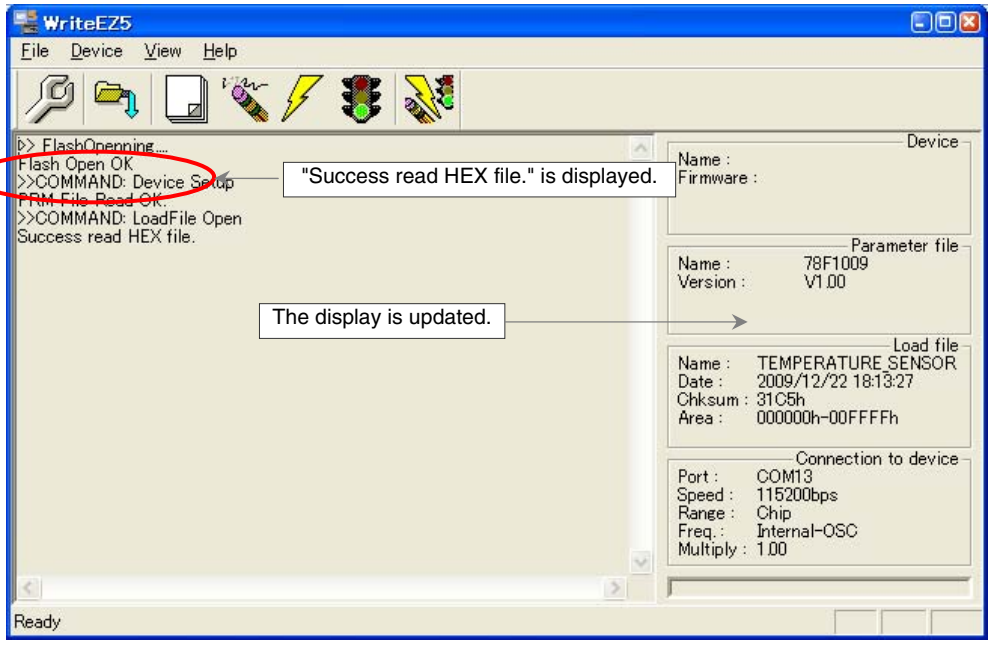


Figure 10-7. After Downloading

10.8 Autoprocedure (EPV)] command execution

Select [Device] → [Autoprocedure (EPV)] from the menu bar.

When the [Autoprocedure (EPV)] command is executed, Blank Check → Erase → Program and FLASH Internal Verify are executed sequentially for the μPD78F1009 device.

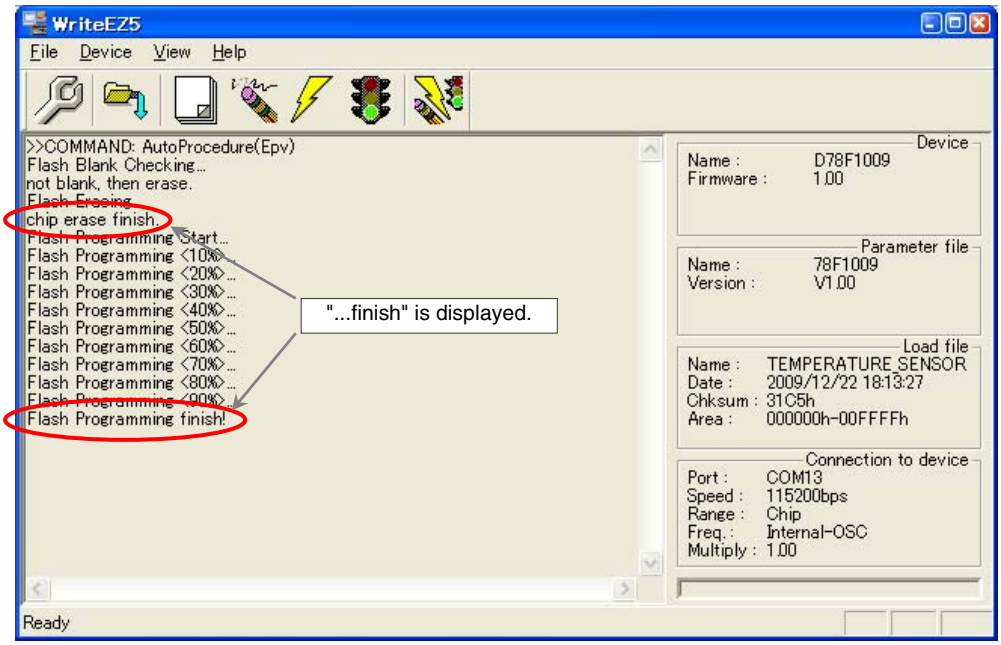


Figure 10-8. After EPV Execution

After flash programming is complete, unplug the USB cable and set SW5, SW7 and the jumpers to their factory settings.

However, if the board must be powered from the USB cable, switch bit 1 of SW5 to 1 to set the board in standalone mode, and press the reset button SW1 to run the application programmed in the microcontroller.

10.9 Terminating the GUI

Select [File] → [Quit] to terminate the GUI software. All settings executed so far are saved in the WriteEZ5.INI file, so that those settings can be reused when the GUI software is restarted.

10.10 Restarting the GUI

When the system is restarted, the same screen as shown in Figure 10-7. After Downloading, appears.

10.11 Erasing the flash memory (using the WriteEZ5)

When the preprogrammed software in the target device is forcibly erased, it is possible to use the WriteEZ5. To erase the flash please perform following steps.

(1) Make sure that the *78K0R–Link It!* board is in the Debugging/Writing mode: (section 10.4.Connecting and starting.)

(2) If not already connected, connect the *78K0R–Link It!* board via USB1 connector to the target device.

(3) Run the WriteEZ5 flash programming software.

(4) Select [Device] → [Erase] from the menu bar, or Click the **erase button**

(5) After finishing the flash erase following message shall apply

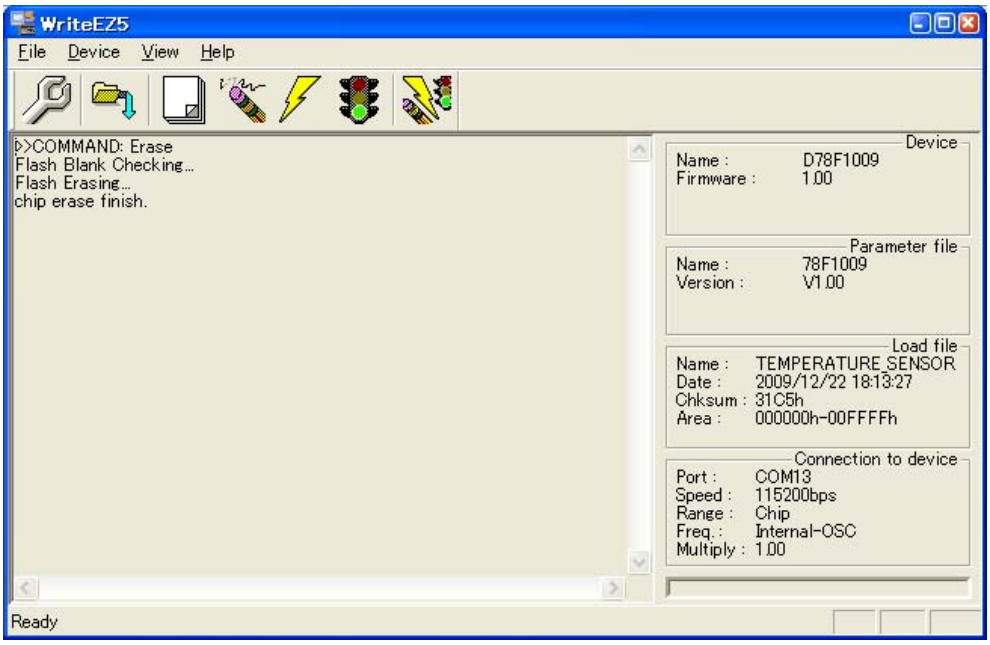


Figure 10-9. WriteEZ5 flash erase successful

CHAPTER 11 EZ-0010 SAMPLE PROGRAMS

The PM+ is a very powerful Integrated Development Environment that allows you to develop and manage a complete embedded application project.

The now described project can be found on the. sample program package of this EZ-0010 package.

To run/debug the project with the ID78K0R, it is necessary to follow the steps described below in this chapter. For more information about PM+ and the included utilities, see the user's manual and other documentation for each tool.

Notes **The EZ-0010 is shipped with the demo software programmed to the 78K0R/KE3-L's internal memory. Make sure that the Flash memory is erased properly before the first project download to the target device. Refer to section.10.11. Erasing the flash memory (using the WriteEZ5)**.

11.1 Hardware setup

To run the program it is necessary to set up the *78K0R–Link It!* board in the following way:

The communication between the starter kit and the ID78K0R debugger running on the host computer is done via the standard UART / USB connection.

To operate the *78K0R-Link it!* board within the TK-78K0R On-Board debug mode, configure switches SW5 to SW7 and Jumpers JP1 to JP4 as instructed in the following figure and Tables:

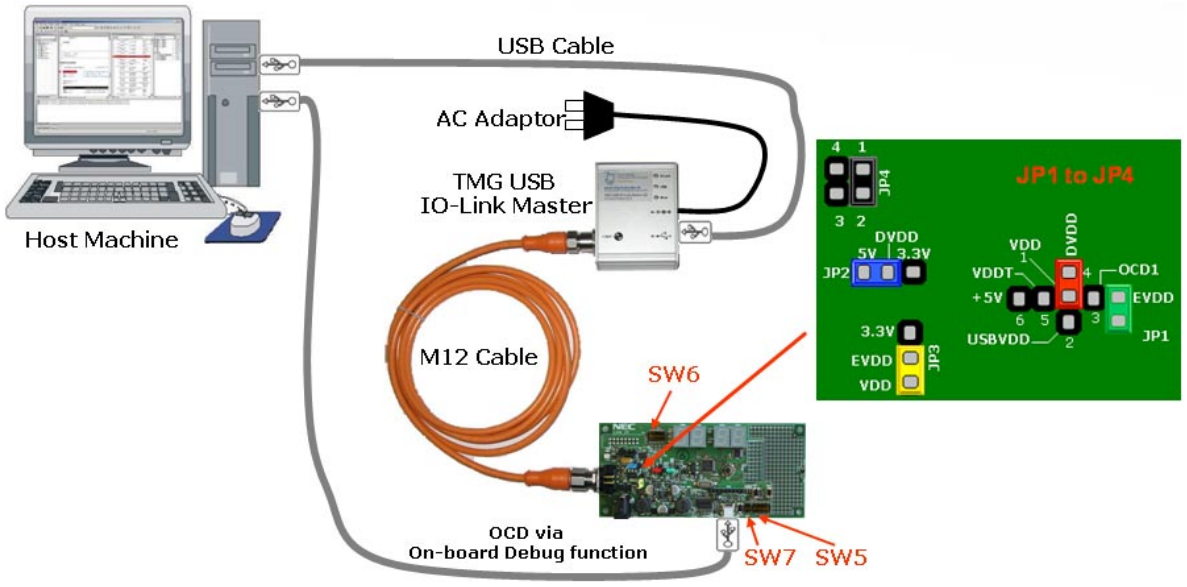


Figure 11-1. IO-Link and On-Chip Debugging via USB

Notes **Figure 11-1 shows the jumper setting when the board is used for an IO-Link application. If you are not using the board for an IO-Link application and USB port power supply is sufficient for your application needs, please use the jumper setting from Figure 10-1 and refer to section 5.2. (OCD via TK-78K0R On-Board debug function) for SW5, SW7 settings**

Table 11-1. SW5 and SW7 setting for On-Chip Debugging via USB

Mode	SW5										SW7	
	1	2	3	4	5	6	7	8	9	10	1	2
TK-78 On-Board debug	ON	ON	ON			ON	ON		ON	ON	ON	ON
				OFF	OFF			OFF				

Table 11-2. SW6 setting for IO-Link Communication

Mode	SW6							
	1	2	3	4	5	6	7	8
Transceiver connected	ON	ON	ON	ON	ON	ON	ON	ON

11.2 Sample Programs

Each of the sample programs is located in a single directory, which will be called main-directory of the sample. This main directory of each sample contains the complete project inclusive all output files of the development tool. The workspace file is located on top of the sample program directories. All sample programs use the same directory structure. Here, the temperature sensor sample program is used as an example. (The RPM sensor sample program has a similar structure.)

Table 11-3. Sample Program Folder Structure (for Temperature Sensor)

SamplePrograms¥	This is the top directory for the sample program project.
temperature_sensor¥	This is the top directory for the temperature sensor sample program. The PM+ configuration file and other files are also saved here.
src¥	Source files and header files used for the sample program are stored in this folder.
obj¥	Object files output by the C compiler are output to this folder.
hex¥	Files output by the linker and object converter are stored in this folder.
stack¥	Files related to the IO-Link device stack are stored in this folder.
temperature_sensor.prw	This is the work space file for the sample program.

The top folder includes only files and folders related to the PM+ project.

The src folder contains source files and header files used for the sample program. The stack folder contains library files and header files for the IO-Link protocol stack.

Object files (*.rel) output by the compiler are output to the obj folder, while load module files (*.lmf) output by the linker and HEX files (*.hex) output by the object converter are output to the hex folder. (HEX files are written to the microcontroller by WriteEZ5.)

11.3 Loading Sample Project

In the Windows start menu, select [All Programs], [NEC Electronics Tools], [PM+], [V6.31], and then [PM+ V6.31] to start PM+. The following screen appears:

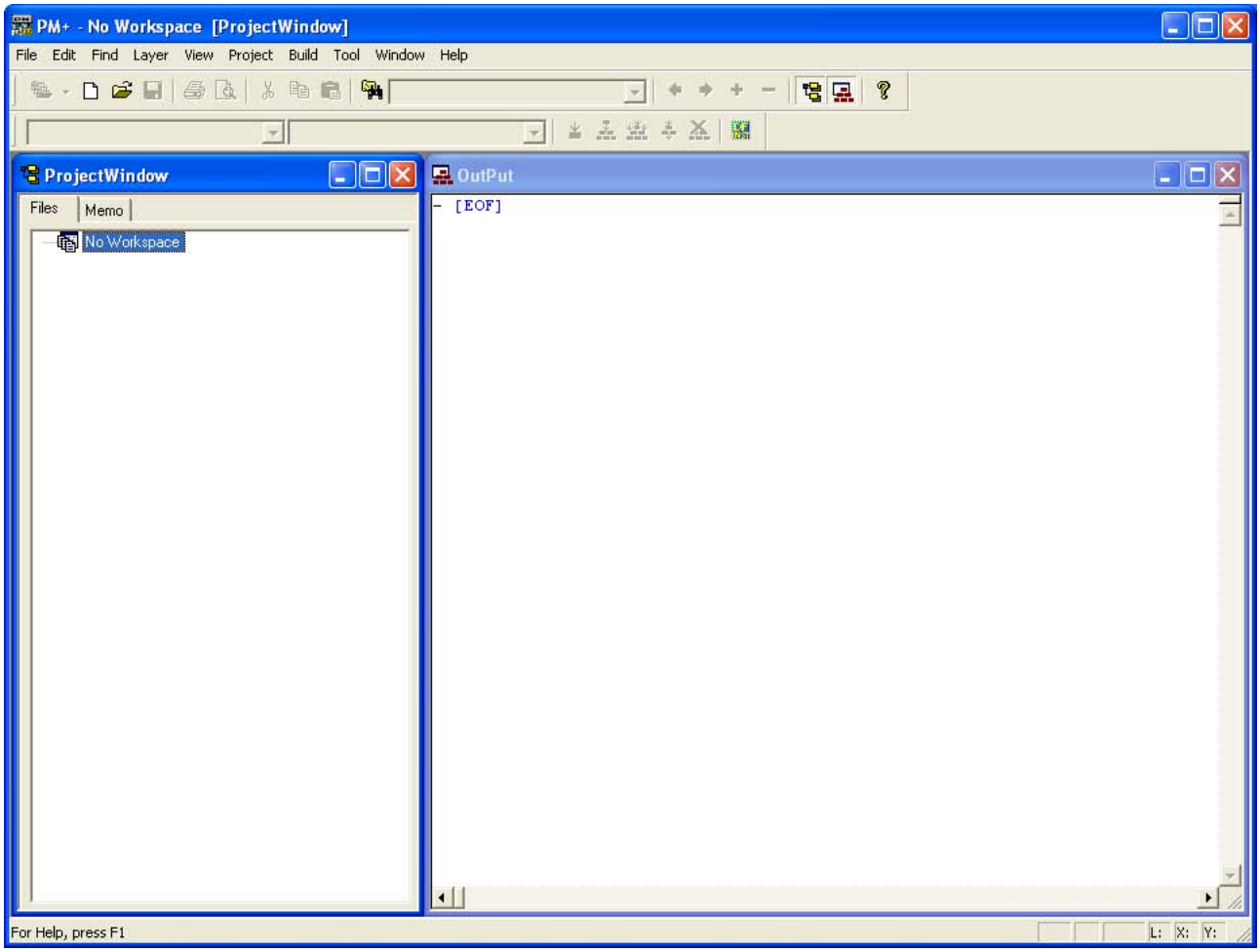


Figure 11-2. PM+ Startup window

Now select the option “Open exiting workspace” from the “File” menu and locate the sample project. Open "temperature_sensor.prw" workspace file. This is the workspace file that contains general information about the demonstration projects and settings.

After the demo workspace has been opened the projects contained in the workspace are displayed. Click on the little “+” sign next to the “Link_It!_Temp_Demo” project to show files part of the project.

The following integrated development environment window is displayed.

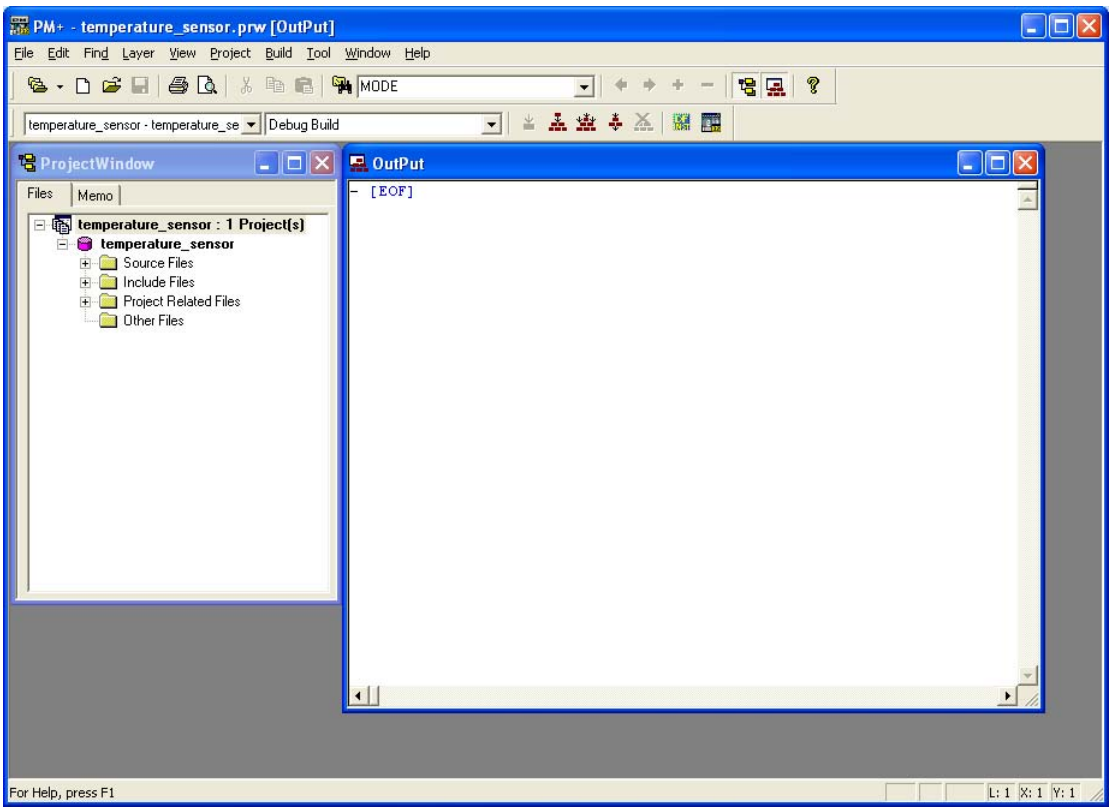


Figure 11-3. PM+ Main window

To confirm that the device to be debugged is correctly selected, select [Project Settings] from the [Project] menu, and then, in the displayed window, confirm that [Microcontroller name] is "78K0R" and that [Device name] is "uPD781009_64"..

Caution An on-board debugging interface is used, but use the drivers for MINICUBE2. These drivers are in the folder in which "usbif_v100.zip" was decompressed.

The corresponding COM port where the *78K0R-Link it!* board is connected to the host PC will be detected automatically by the ID78K0R-QB debugger.

Now after everything has been setup correctly it's time to compile and link the demonstration project. Close the Options menu and select "Rebuild All" from the "Project" menu. If the project is compiled and linked without errors or warnings it can now be downloaded to the *78K0R-Link It!* board and debugged.

To start the ID78K0R-QB debugger select the option "Debug" from the "Project" menu or press the "Debugger" button.

When first starting the debugger connection, after building the project, it is necessary to set up the right connection settings. Please take care that the following settings are chosen:

- ID code = FFFFFFFFFFFFFFFFFFFF (10 FFs)

  * In the window below, the ID code is displayed using "*".

- Main clock = External 18.432 MHz
- Sub clock = External 32.768 KHz
- Target connect = TOOL0+TOOL1

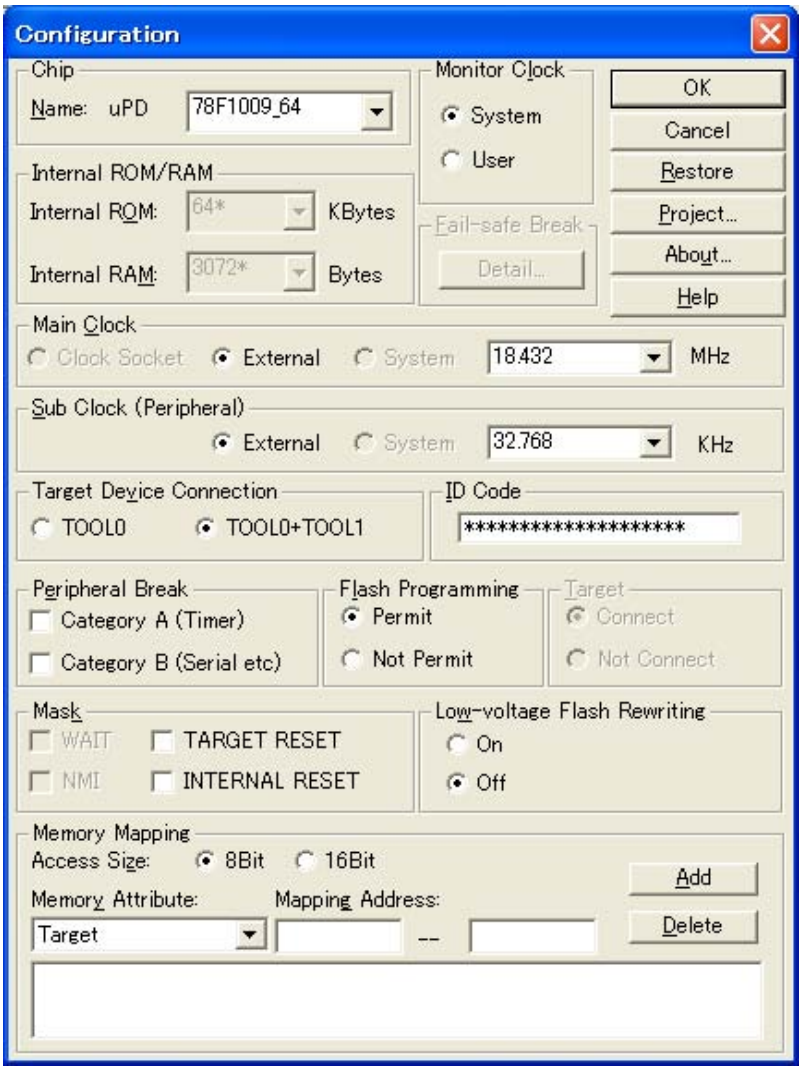


Figure 11-4. Debugger settings

Now the debugger is started and the demo project is downloaded to the *78K0R-Link It! board*. The progress of downloading is indicated by blue dots in the MINICUBE Emulator window. Please note that downloading of larger executables may take some time.

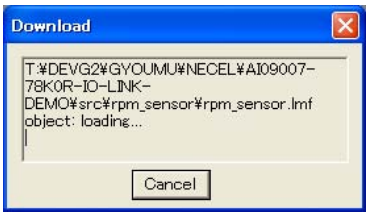


Figure 11-5. Progress of Downloading

After the download was completed all debug features of ID78K0R-QB debugger are available, i.e. Single Stepping, Step Over/-In/-Out, Go-Execution, Breakpoints, Register / Memory view etc.

To get more details on the debugger configuration and capabilities please refer to the User's Manual of the ID78K0R-QB.

CHAPTER 12 IO-LINK TEMPERATURE SENSOR SAMPLE

The demo application shows the IO-link communication protocol as well as different features present on the board such as the display, the Sensor (Temperature), the TMG Protocol Stack, the Elmos IO-Link transceiver, analog digital conversion and more.

The *78K0R–Link It!* board is shipped with the demo preprogrammed to the microcontroller. When the board is connected to an IO-Link Master and communication is established, the temperature is measured and transmitted through IO-Link.

For the purpose of this demo the TMG USB IO-Link Master has been used. The screenshots in the following section are those of the TMG device tool software, operating with the TMG master device.

If you have the TMG Master device, a XML device description file is needed. For more details on installing the XML Device Description file, please refer to section **8.7 Installing XML Device Description file**.

Once the Device tools software has been updated, the *78K0R-Link it!* board can be controlled from the software GUI. For further details please refer to **12.2 Using *78K0R-Link It!* Board demo with TMG IO-Link Device Tool**.

Connect the IO-Link Master to the host computer. The *78K0R-Link-It!* board should be now connected to the Master via the M12 cable see Figure 2-1. IO-Link system configuration

Caution jumpers and switches on the *78K0R-Link-It!* board must be set to their factory settings as described in section 7.1. Factory Settings of jumpers and switches.

12.1 Demo Features

Once IO-Link communication is established with the Master, the board displays the current room temperature in 0,5 °C variation steps as per Figure 12-1.

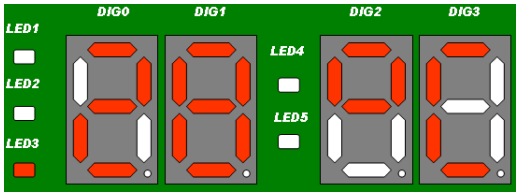


Figure 12-1. 78K0R–Link It! Temperature Sensor Demo display (28 °C)

The current measured temperature value is displayed with 2 digits (DIG0 and DIG1). The decimal point behind the last digit is the 0.5°C indicator (for 28.5 °C it would be lit).

The user can set two (Upper/Lower) thresholds, set by default to 32°C, 29°C respectively, when the board is connected. LED1 to LED3 are alarm LEDs representing thresholds set by the user.

- LED1 lights up when the measured temperature is above the Upper Threshold
- LED2 lights up when the measured temperature is between the two Thresholds
- LED3 lights up when the measured temperature is below the Lower Threshold

12.1.1 Min/Max Temperature display

The minimum/maximum temperatures measured since the sensor has been connected to the IO-Link master can be readout on the display at any time when pressing SW2 for the Min Temperature, and SW3 for the Max Temperature. Refer to Figures 12-2 and 12-3.

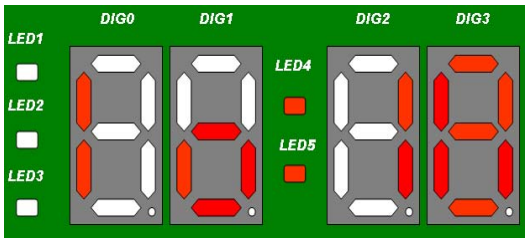


Figure 12-2. Minimum Temperature Readout

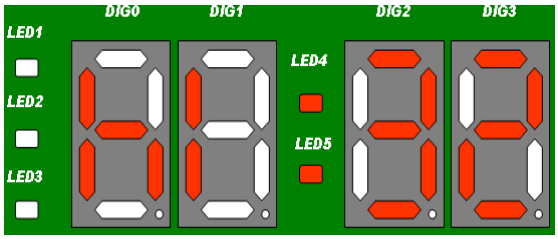


Figure 12-3. Maximum Temperature Readout

12.1.2 Temperature thresholds setting

The user has the ability to set two thresholds (a Lower and an Upper threshold).

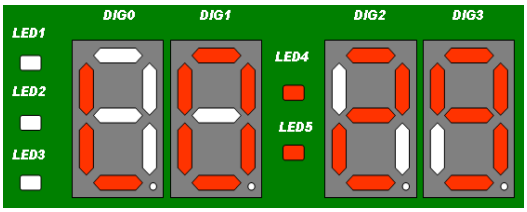


Figure 12-4. Default Lower Threshold

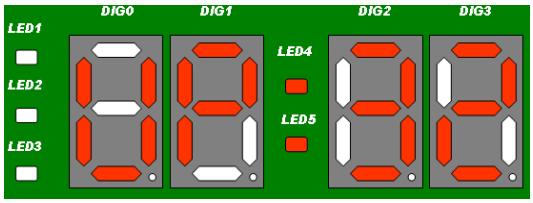


Figure 12-5. Default Upper Threshold

(1) To enter the settings menu, press the Center button on SW4. The display thereafter shows the current value of the lower threshold. Refer to Figures 12-4 and 12-5.

- Pushing the SW4 button to the right increments the lower threshold in 0,5 °C steps
- Pushing the SW4 button to left decrements the lower threshold in 0,5 °C steps
- Pushing the SW4 button up, increments the upper threshold it in 0,5 °C steps
- Pushing the SW4 button down decrements the upper threshold in 0,5 °C steps

The upper threshold can never be lower or equal to the lower threshold and vice-versa.
The actual set threshold temperature as well as the min/max value, are displayed with 2 digits only. The point behind the last digit is the 0.5 indicator (therefore for 32.5 °C it would be lit).

(2) To confirm the settings, press the center button on SW4 again. The display thereafter, reverts to the current measured temperature display.

All of the temperature values, thresholds values and settings can also be changed or read using the TMG USB IO-LINK Master device tool.

12.2 Using *78K0R-Link It!* Board demo with TMG IO-Link Device Tool.

The IO-Link device tool can be used for the configuration of IO-Link masters, setting and steering parameters as well as the diagnosis of IO-Link devices.

IO-Link master initiates the communication, and channels information from the board to the host machine. The operator is able to see the information on the host machine via the installed IO-Link Device Tool.

12.2.1 Getting started with the TMG master:

The user has the ability to set two thresholds (a Lower and an Upper threshold).

(1) Launch the IO-Link device Tool.

(2) Once the tool is up and running, two panes can be seen in the GUI (Topology on the left, and Catalog on the right),see Figure 12-6

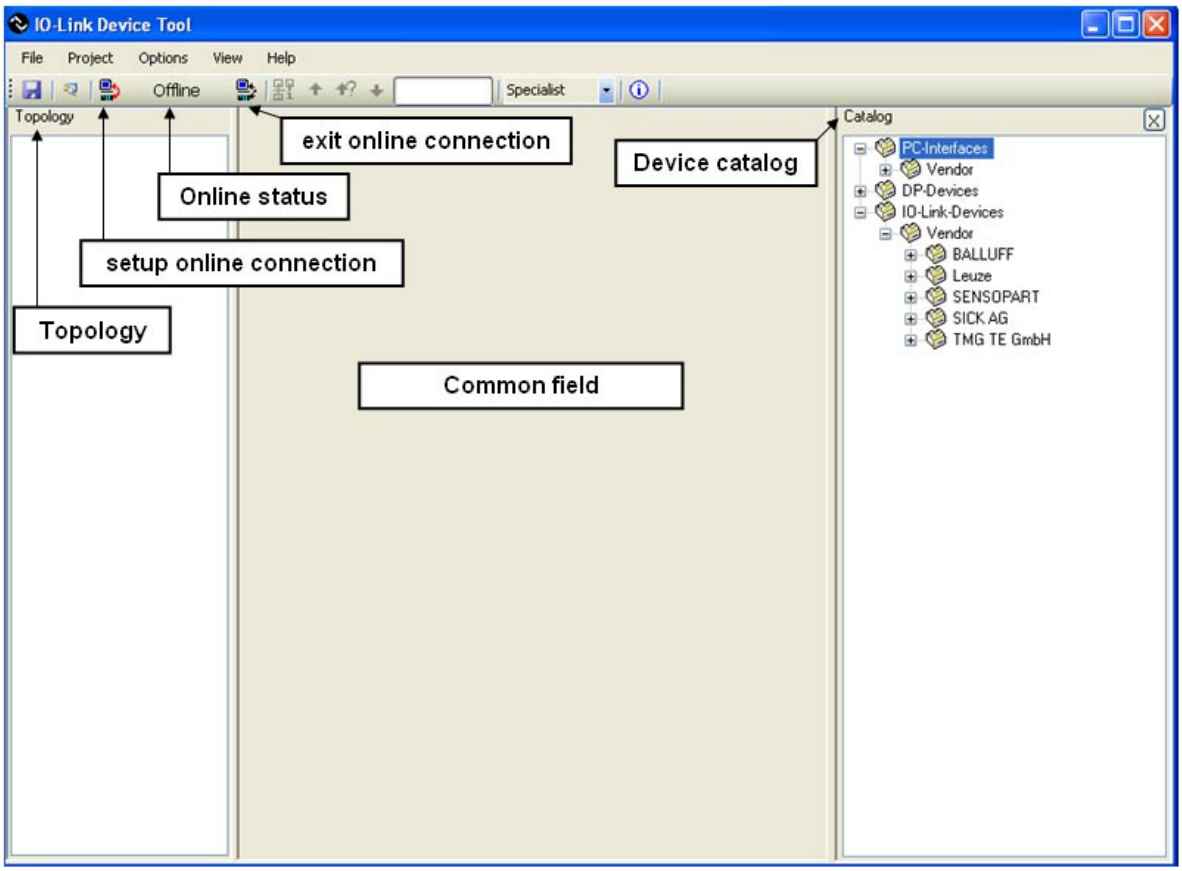


Figure 12-6. IO-Link Device Tool GUI

The topology pane shows the topology from the PC interfaces to the IO-Link Devices The Catalog pane shows all the Devices installed with the tool.

On the symbol bar, two icons (setup/exit online connection) allow to set the IO-Link line status.

When the line is Online, the “Online status” symbol will blink green with the symbol “online”

The common field is currently blank but it will display the description of the devices present in the Topology view

12.2.2 Updating the IO-Link Device catalog

Before the 78K0R-Link It! Temperature sensor can be displayed in the device tool, the IO-Link devices catalog must be updated

(1) In the Catalog pane, right click on “IO-Link Devices” and select [Actuate catalog]

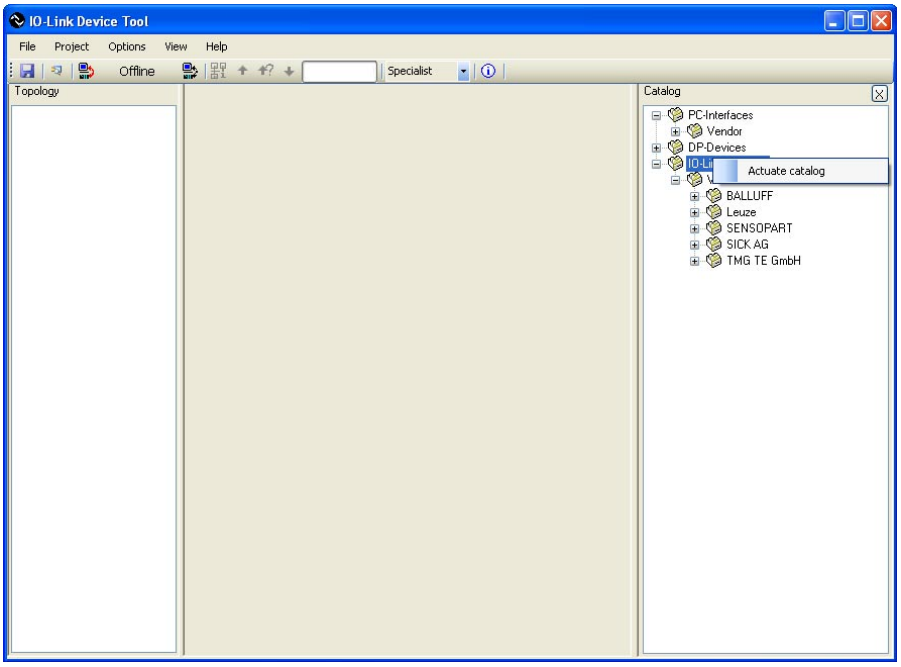


Figure 12-7. Catalog update <1>

(2) Press [Yes] in the small window with the message [Do you want to insert the 78K0R–Link It! Temperature sensor of NEC Electronics (Europe) into the catalog]

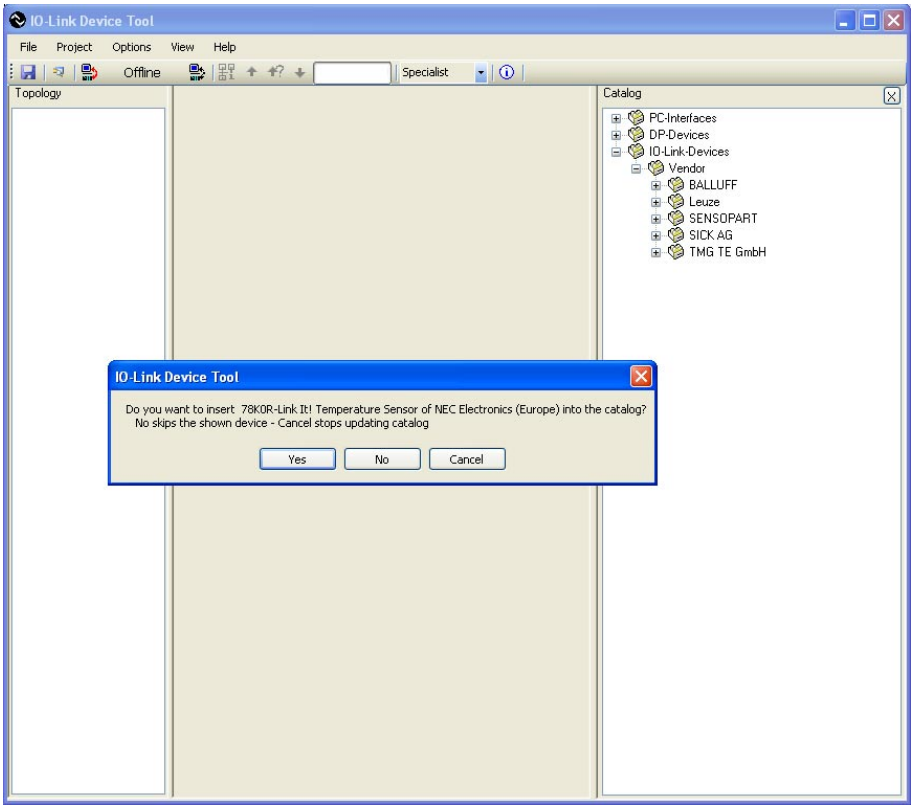


Figure 12-8. Catalog update <2>

12.2.3 Catalog update confirmation

A successful update will show the NEC Electronics (Europe) vendor and the *78K0R–Link It!* Temperature sensor, in the IO-Link Devices section of the catalog. The TMG USB IO-Link master can also be seen under PC Interfaces section of the catalog.

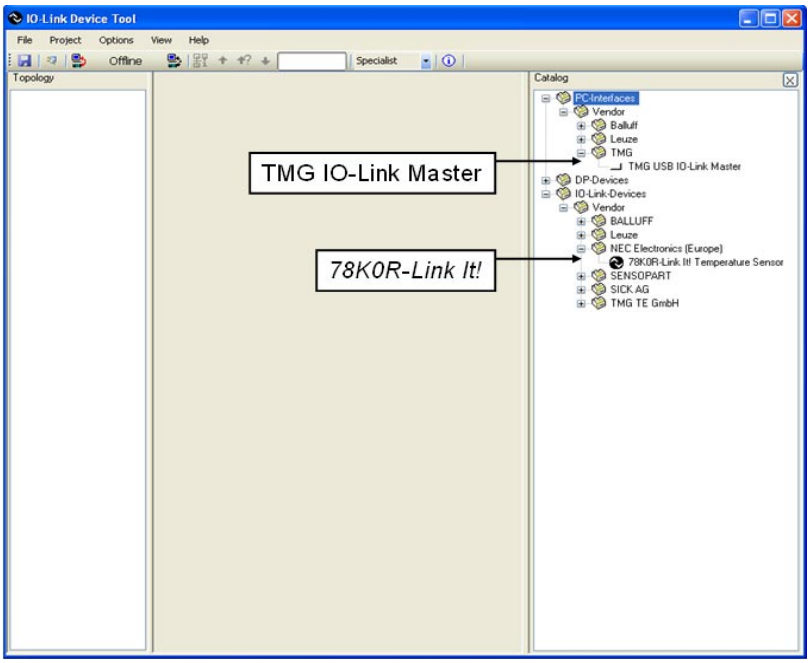


Figure 12-9. Catalog update confirmation

12.2.4 IO-Link communication set-up

(1) From the Catalog pane, drag and drop the TMG USB IO-Link Master into the Topology pane.

(2) Left click on TMG USB IO-Link Master DE in the Topology pane, and the IO-Link Master's details can be seen in the Common pane (centre section of the window).

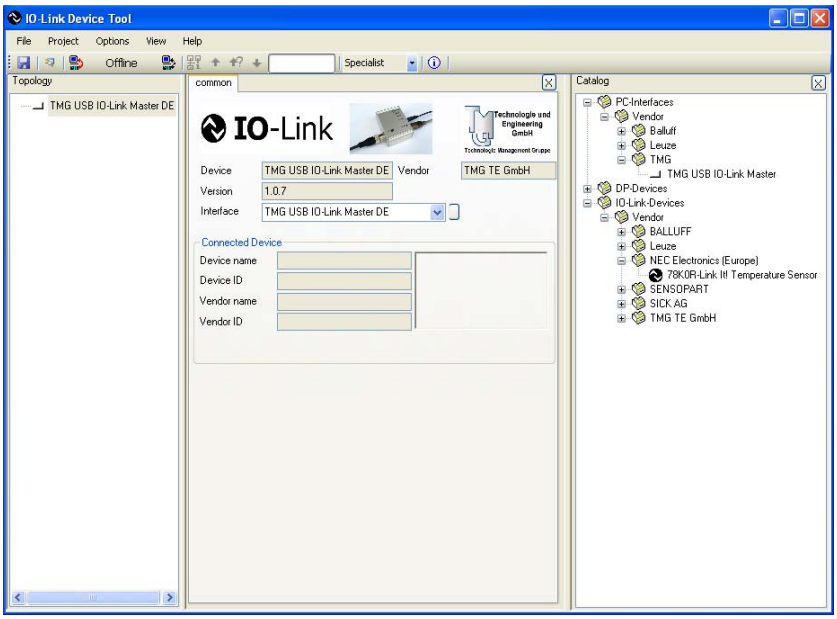


Figure 12-10. TMG USB IO-Link Master

(3) Left click on the "Setup online connection" icon in the Device Tool GUI, will show the "Check Config" button under "Connected Device" in the Common section.

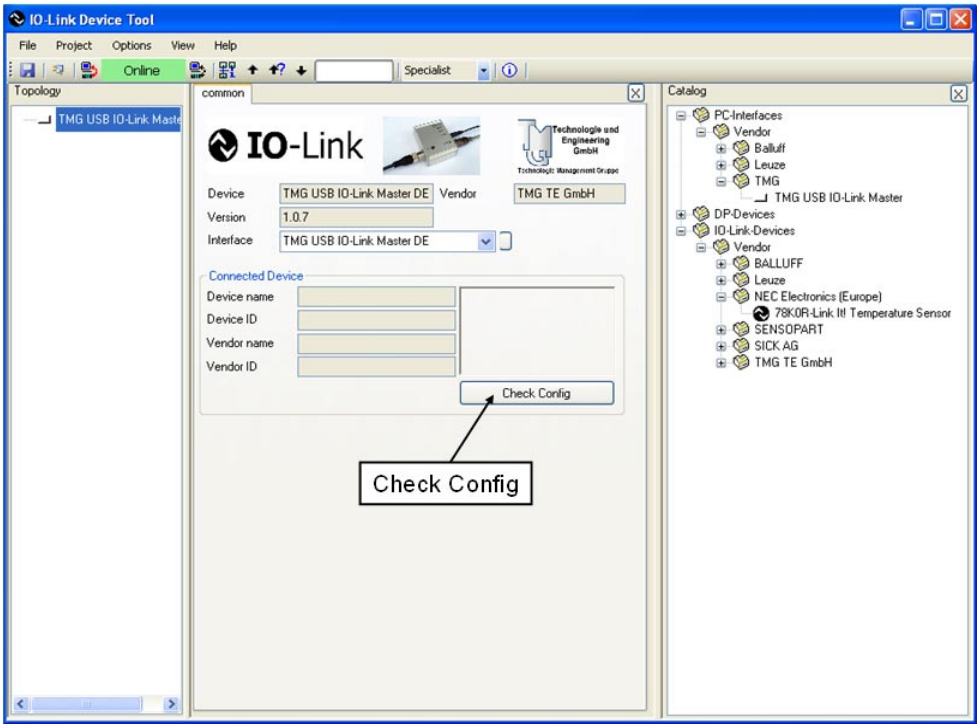


Figure 12-11. Online status, Check Config

(4) Click on the "Check Config" button.

The LEDs on the Master will blink as the master tries to connect to the *78K0R–Link It!* board If the *78K0R–Link It!* board is working and the master can connect to it, a small window will show with information on the board and a “Take over type of device into engineering” button. See **Figure 12-12. Check Config successful.** If the master cannot connect to the *78K0R–Link It!* board a window with an error message “can’t read configuration” will appear. See **Figure 12-13. Check Config failure.**

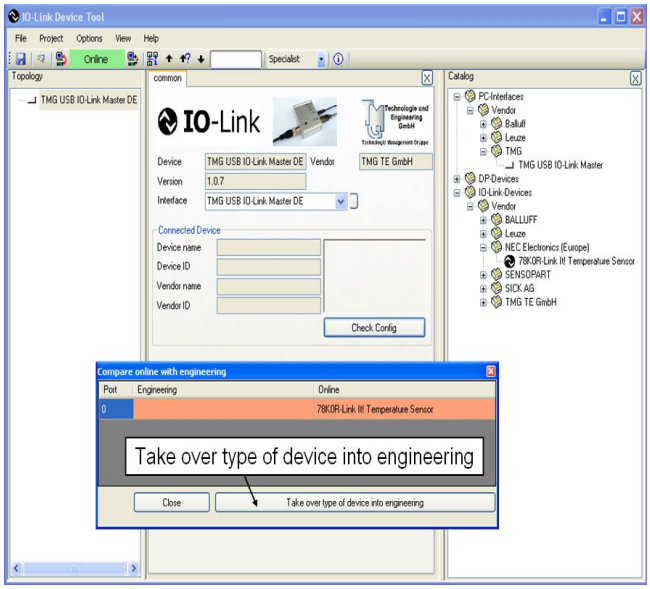


Figure 12-12. Check Config successful

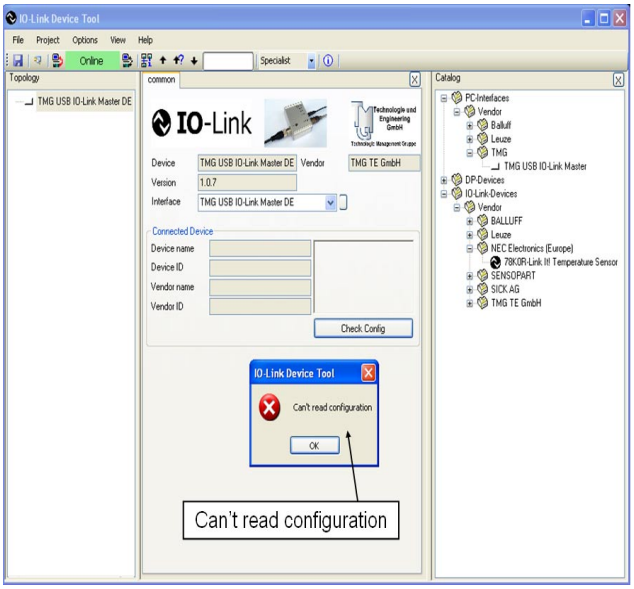


Figure 12-13. Check Config failure

Caution **It can happen sometime that the “can’t read configuration” message appears in the Device Tool when the Master is not properly initialized. Unplugging the master from the USB port of the host machine and re-starting the Device tool solves this problem. Also please check the board settings are properly set for IO-link communication and that the board is properly connected to the IO-Link master.**

(5) Click on the “Take over type of device into engineering” button

The *78K0R–Link It!* Temperature Sensor will now appear under the TMG USB IO-Link Master DE in the topology pane, and details on the board can be read in the Common pane

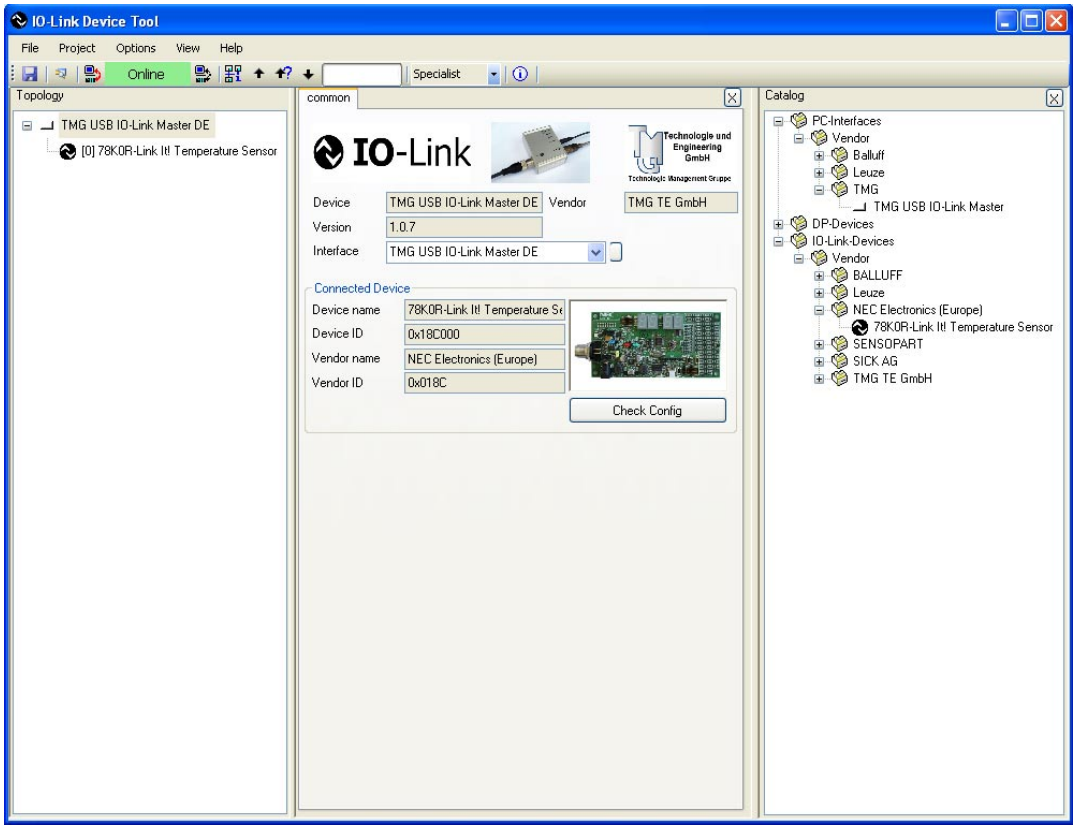


Figure 12-14. IO-Link communication active

The IO-Link communication is now active and we can have a closer look at the device and the sensor application running on the board

The Common pane provides information on the connected device such as:

- Device name: 78K0R-Link It! Temperature Sensor
- Device ID: 0x18C000
- Vendor name: NEC Electronics (Europe)
- Vendor ID: 0x018C
- Picture of the Io-Link device connected to the master

12.2.5 *78K0R-Link It!* Board sensor demo in IO-Link device tool

In this section we see how the *78K0R–Link It!* Board works with the IO-Link Master and how the demo's features explained earlier can be used from the IO-Link Device tool GUI.

Left click on the *78K0R–Link It!* Temperature Sensor in the topology pane to display generic information on the board in the Common pane. Next to the Common tab, two other tabs can be seen (Process data and Parameter).

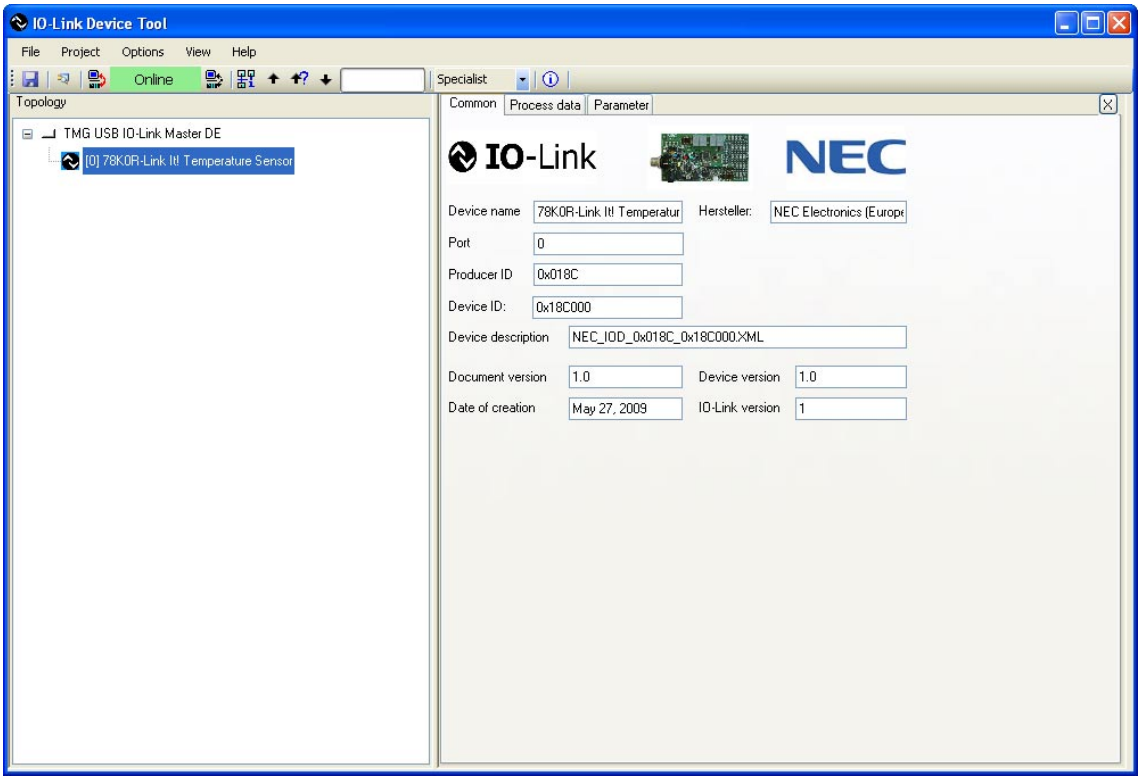


Figure 12-15. *78K0R–Link It!* Temperature Sensor *in device tool*

(1) Common tab

The Common pane provides information on the device such as:

- Picture of the device
- Device name, Product ID, Device ID, vendor name
- Port used on the master. Here the kit is connected to port 0.
- Device description file, its version and date of creation.
- Device version
- IO-Link version.

(2) Process data tab

The Process data tab displays the sensor measurements (process data inputs from the sensor).

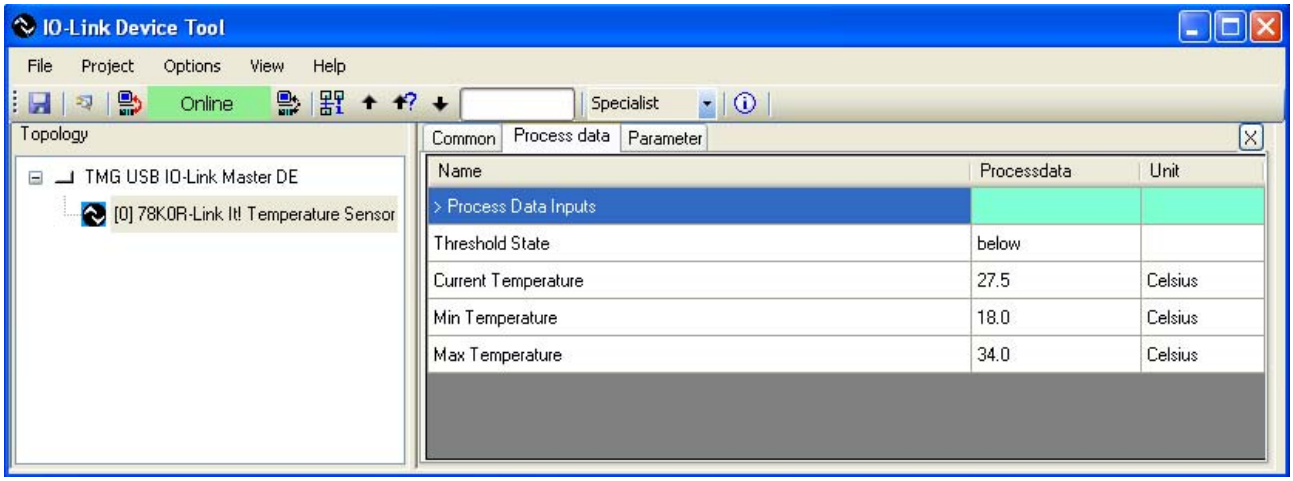


Figure 12-16. *78K0R–Link It!* Temperature Sensor Process data

The user can find information such as the Min/Max Temperature values, the Current Temperature, and the Threshold State (above, between, and below, representing LED1, LED2, and LED3 respectively). (The actual threshold values are displayed on the Parameter tab.) The values measured by the sensor are displayed on the Process data tab. (This is process data input from the sensor.)

(3) Parameter tab

The Parameter tab displays the default settings of the sensor, but also enables the user to teach the sensor by writing new thresholds values to the board.

From the tabs on the parameter page we can find general read only vendor specific parameters such as:

- Vendor and product names
- Hardware, firmware revisions

From the tabs on the parameter page we can find read/write device specific parameters such as:

- Upper/Lower Thresholds

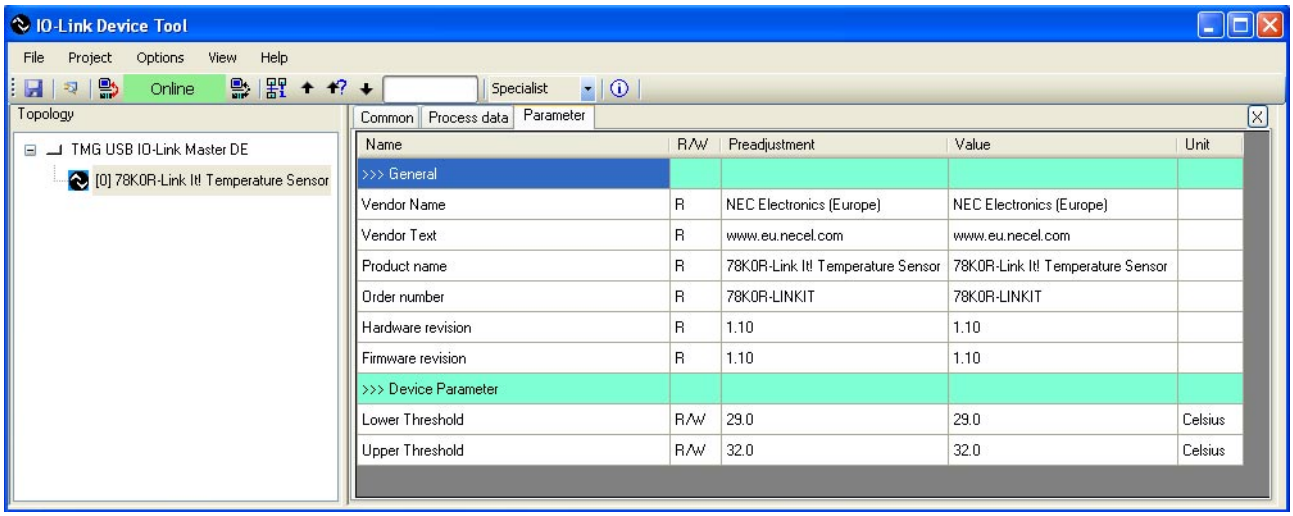


Figure 12-17. *78K0R–Link It!* Temperature Sensor Parameter page

12.2.6 *78K0R-Link It!* Board sensor Teach-in

When the user opens the parameter page for the first time, the device specific parameters are set to their default values. The “Pre-adjustment” and “Value” columns match each other in the parameter page, see **Figure 12-17**.

The “Pre-adjustment” column displays the default sensor settings. The “Value” column displays the current sensor settings

(1) Read parameter from the *78K0R-Link It!* Board sensor

To read or refresh the display of the current parameter, click Left in the cell corresponding to the parameter you want to read from under the “Value” column. The cell is highlighted in blue and the current parameter value is displayed.

Figure 12-18 shows the reading of the lower temperature threshold of 29°C set on the board.

The same procedure is used to read the Upper threshold set on the board.

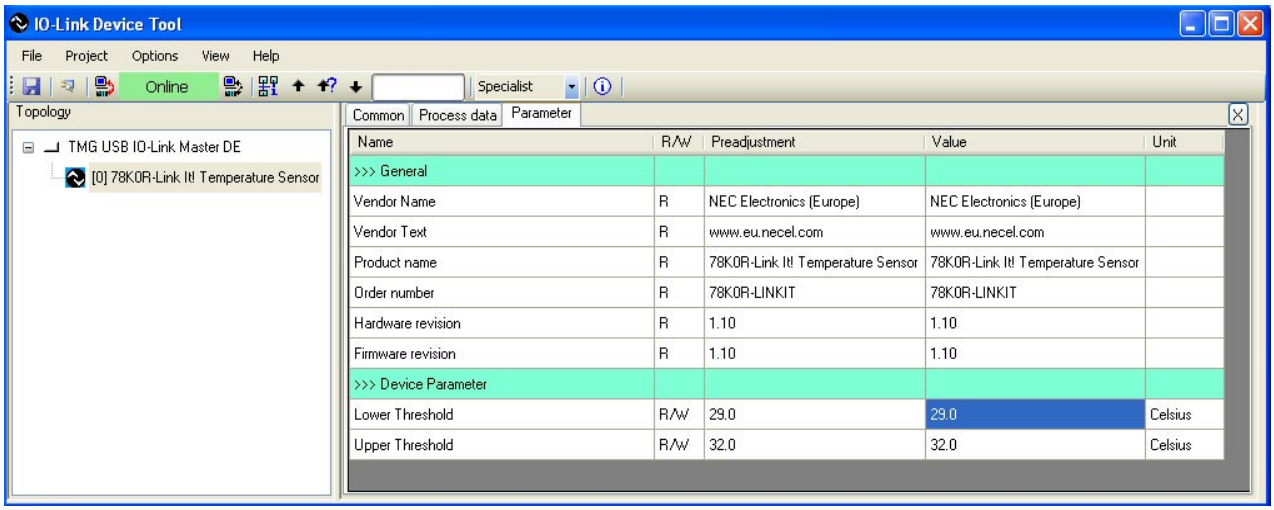


Figure 12-18. Lower Threshold value reading

(2) Write parameter to the *78K0R-Link It!* Board sensor

The teach-in functionality allows changing the valid temperature range on the fly by modifying the Upper/Lower thresholds.

To write or teach a parameter to the board, click Right in the cell corresponding to the parameter you want to change from under the “Value” column. The cell is highlighted in blue and a dialog box will open to enter the new value.

Enter a new value in the box and press the [Enter] key on your keyboard. The new parameter is now passed to the board. You can check with the SW4 joystick that the parameter has been passed correctly to the board.

The Thresholds LEDs will automatically display what the current measured temperature is with regards to the new range.

Figure 12-19 shows the dialog box and the setting of the Lower threshold to 25°C.

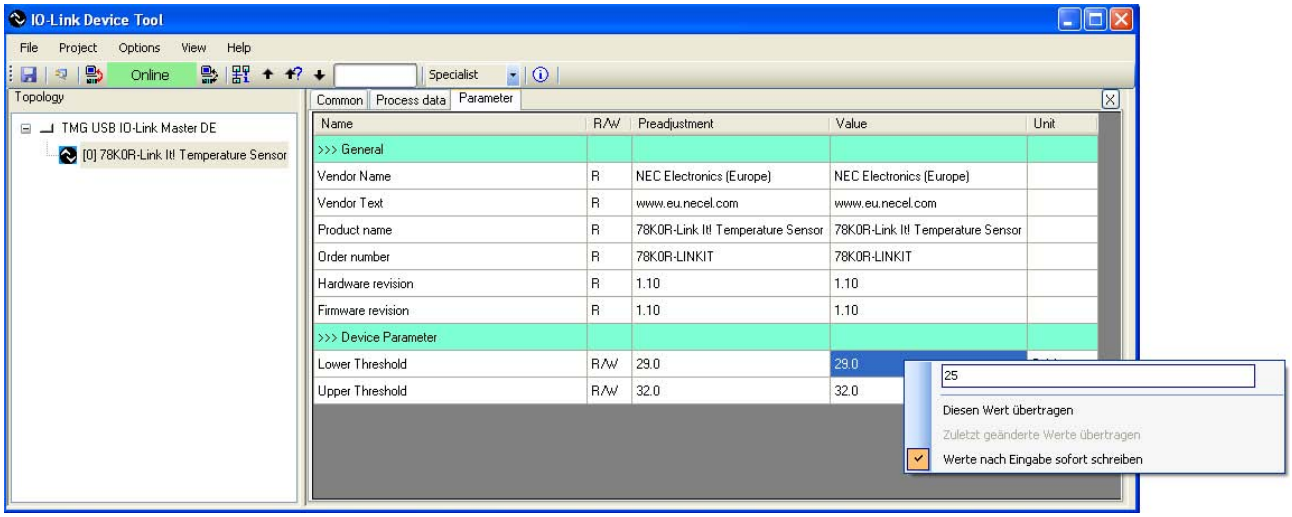


Figure 12-19. *78K0R–Link It!* Temperature Sensor Thresholds setting

12.3 IO-Link Temperature Sensor Sample Program Flowchart

This section provides the flowcharts for this sample program. This program is made up of the functions below. (Below, items that are divided according to these functions are called *modules*.)

(1) Main reset

(2) Temperature sensor control

(3) Display

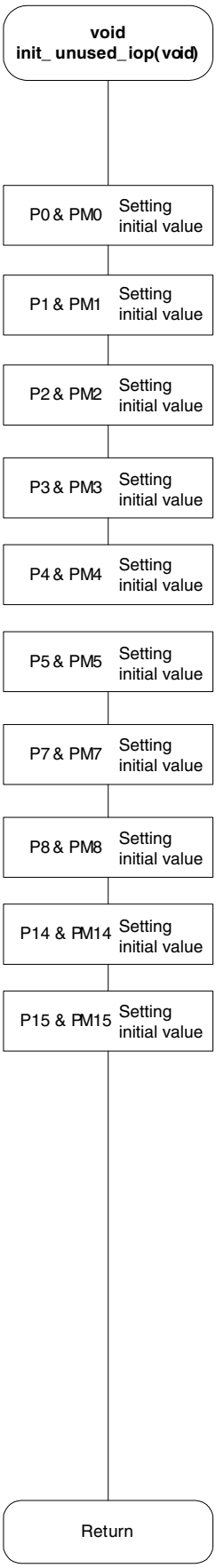
(4) I^2C driver

(5) Key input

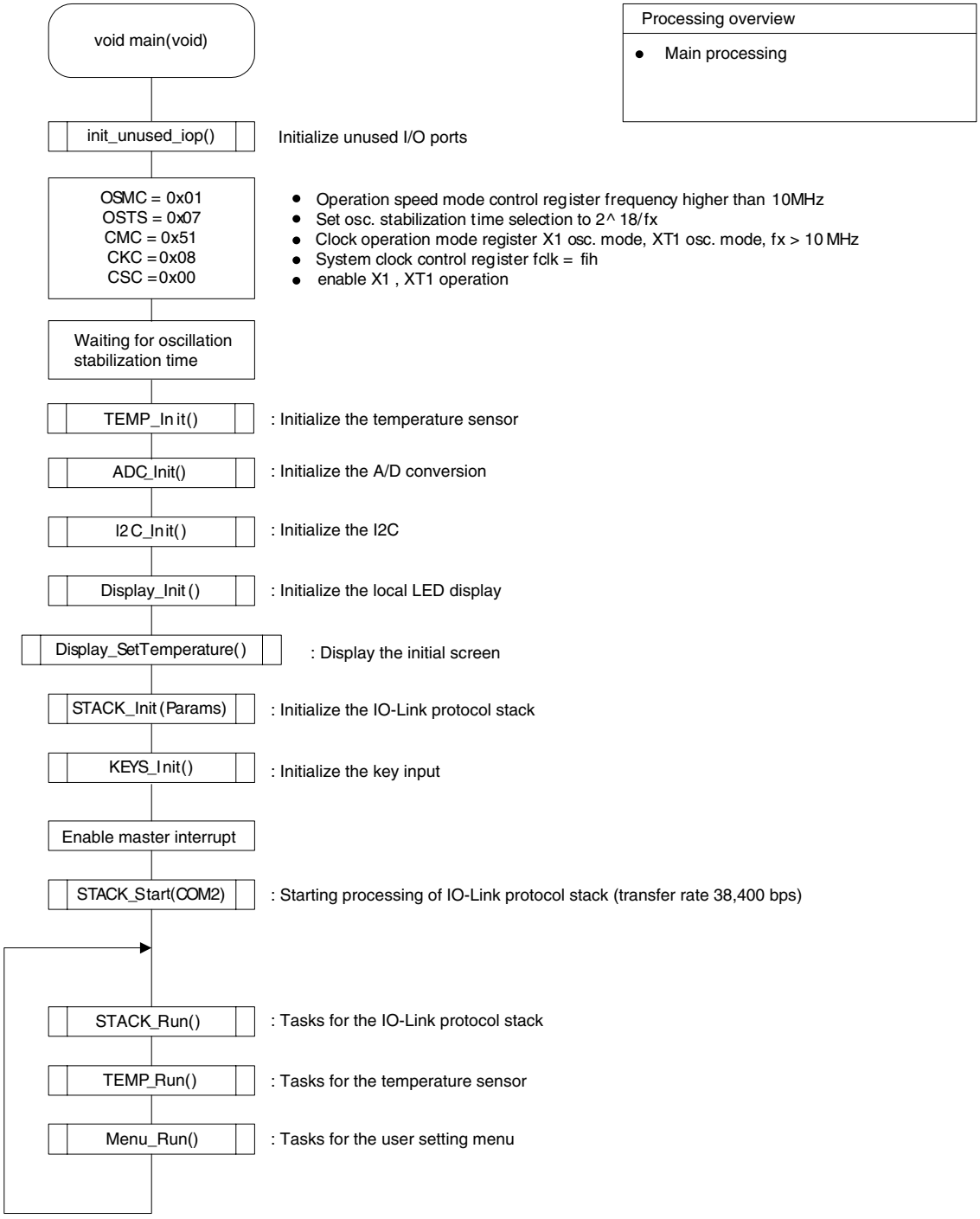
(6) A/D conversion

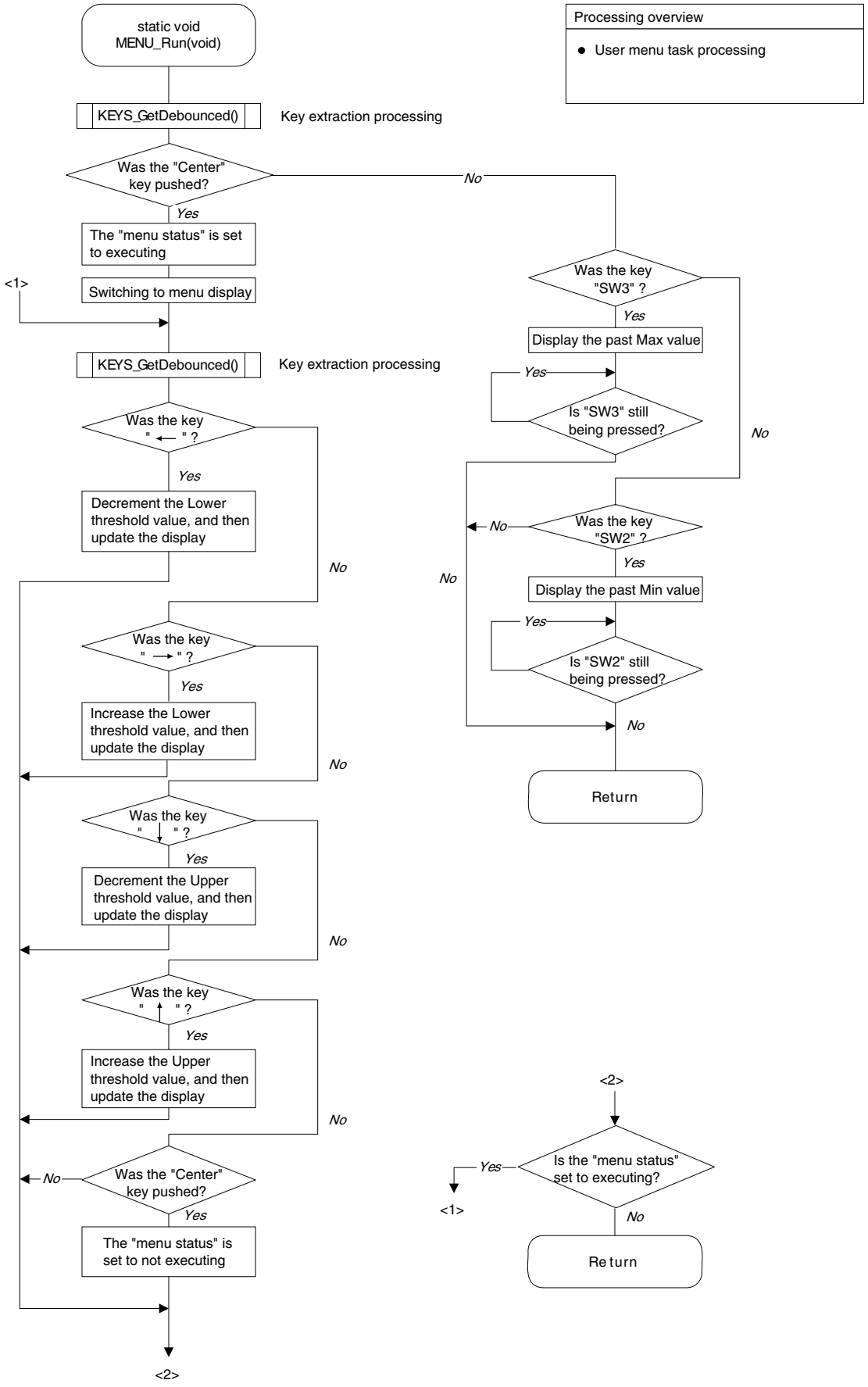
Flowcharts of the functions included in the above modules are provided below.

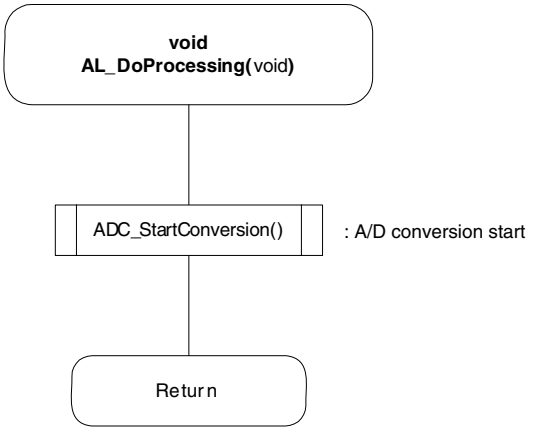
12.3.1 Main reset module flowchart



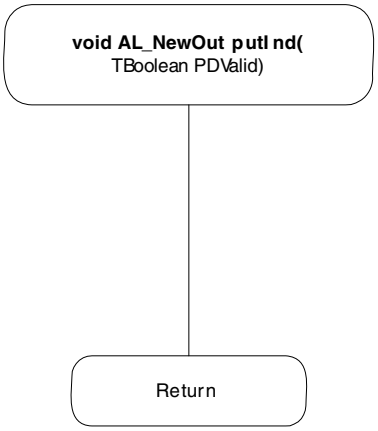
Processing overview
• Initialize unused I/O ports. • Set unused I/O ports to low output.


void main(void)
Processing overview
Main processing
init_unused_iop()
Initialize unused I/O ports
OSMC = 0x01
OSTS = 0x07
CMC = 0x51
CKC = 0x08
CSC = 0x00
Operation speed mode control register frequency higher than 10MHz
Set osc. stabilization time selection to 2^ 18/fx
Clock operation mode register X1 osc. mode, XT1 osc. mode, fx > 10 MHz
System clock control register fclk = fih
enable X1 , XT1 operation
Waiting for oscillation stabilization time
TEMP_Init()
: Initialize the temperature sensor
ADC_Init()
: Initialize the A/D conversion
I2C_Init()
: Initialize the I2C
Display_Init()
: Initialize the local LED display
Display_SetTemperature()
: Display the initial screen
STACK_Init(Params)
: Initialize the IO-Link protocol stack
KEYS_Init()
: Initialize the key input
Enable master interrupt
STACK_Start(COM2)
: Starting processing of IO-Link protocol stack (transfer rate 38,400 bps)
STACK_Run()
: Tasks for the IO-Link protocol stack
TEMP_Run()
: Tasks for the temperature sensor
Menu_Run()
: Tasks for the user setting menu

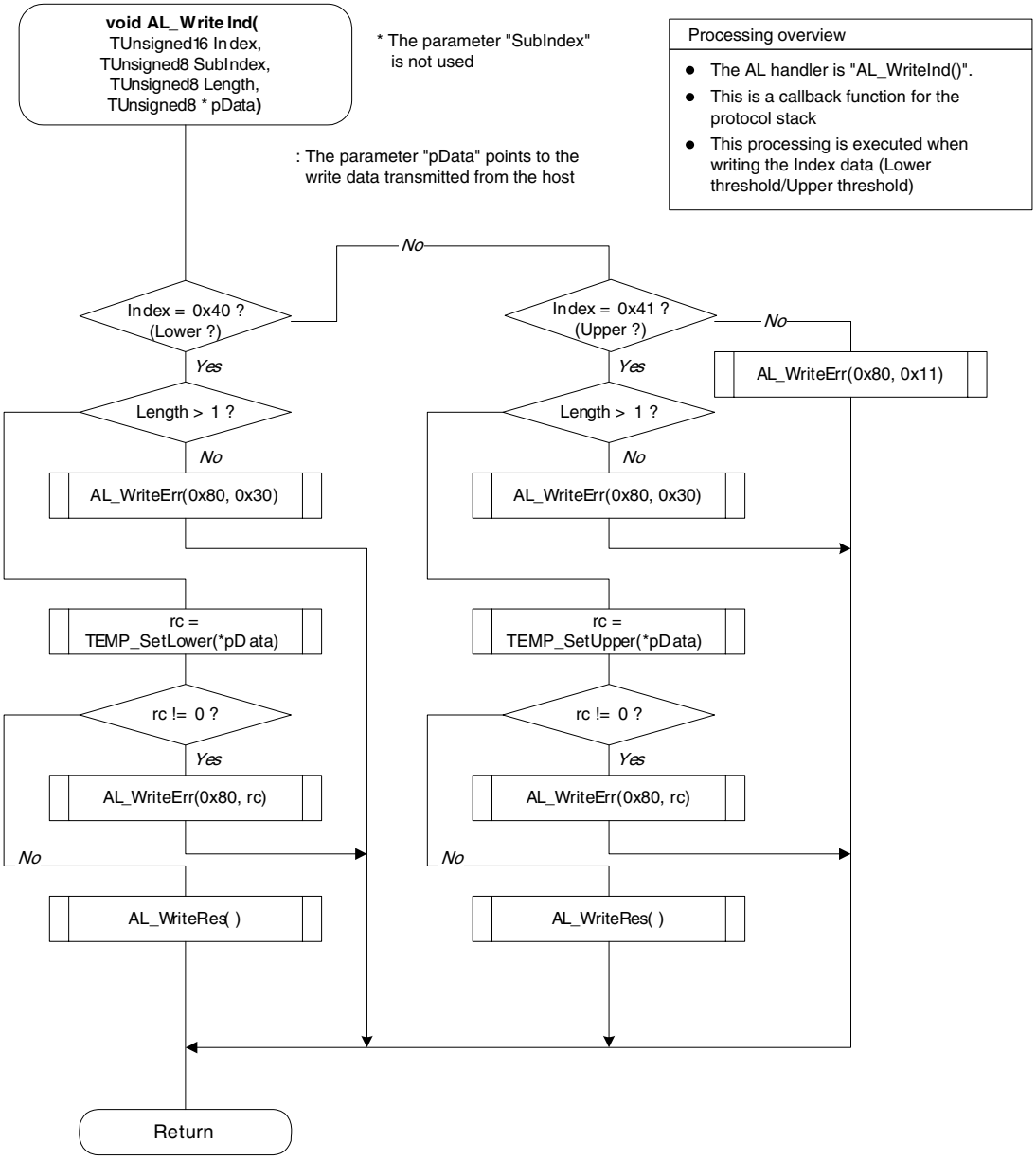

static void
MENU_Run(void)
Processing overview
● User menu task processing
KEYS_GetDebounced()
Key extraction processing
Was the "Center" key pushed?
No
Yes
The "menu status" is set to executing
Switching to menu display
<1>
KEYS_GetDebounced()
Key extraction processing
Was the key " ← " ?
Yes
Decrement the Lower threshold value, and then update the display
No
Was the key " → " ?
Yes
Increase the Lower threshold value, and then update the display
No
Was the key " ↓ " ?
Yes
Decrement the Upper threshold value, and then update the display
No
Was the key " ↑ " ?
Yes
Increase the Upper threshold value, and then update the display
No
Was the "Center" key pushed?
No
Yes
The "menu status" is set to not executing
<2>
Was the key "SW3" ?
Yes
Display the past Max value
Yes
Is "SW3" still being pressed?
No
No
Was the key "SW2" ?
Yes
Display the past Min value
Yes
Is "SW2" still being pressed?
No
No
Return
<2>
Is the "menu status" set to executing?
Yes
<1>
No
Return

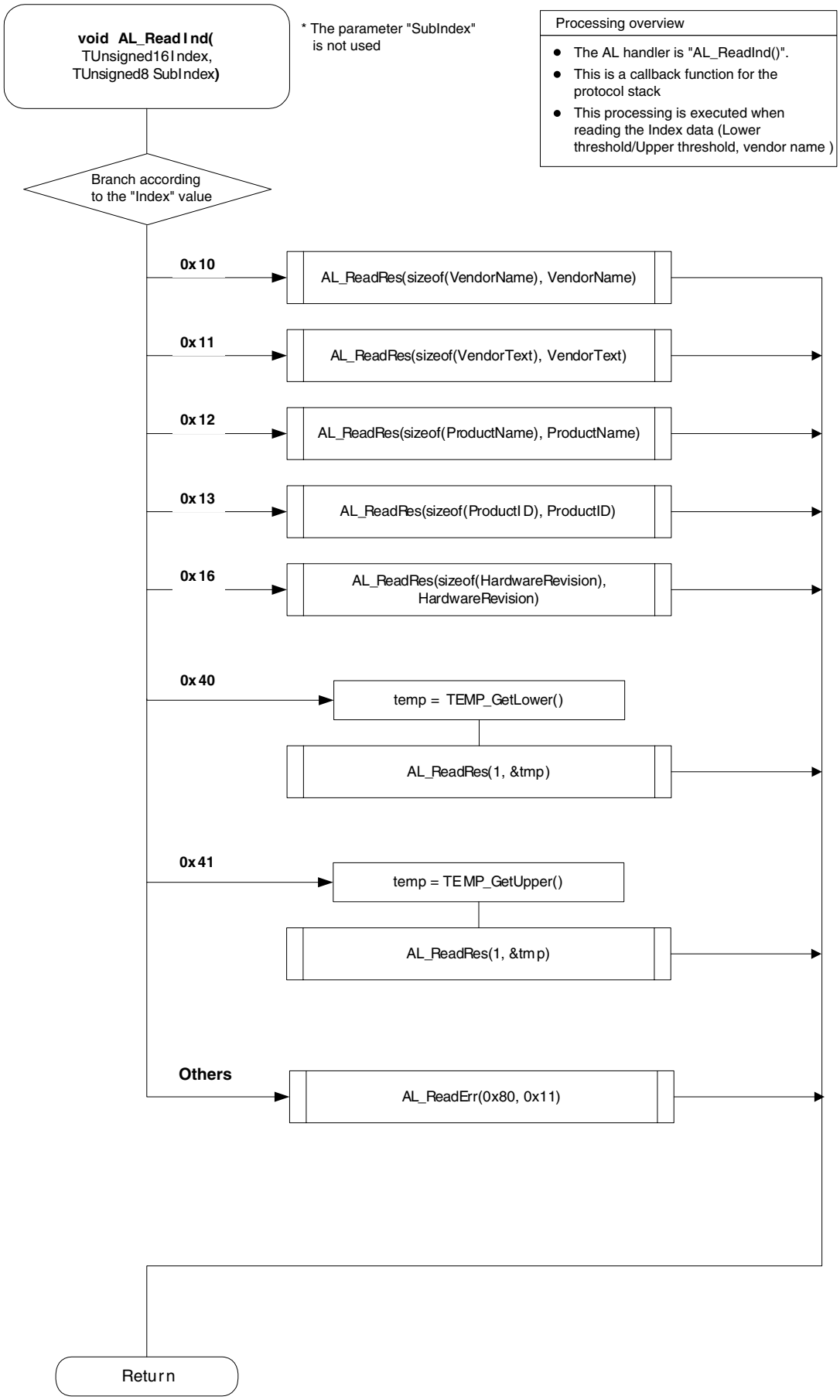


Processing overview
• The AL handler is "AL_DoProcessing()". • This is a callback routine for the protocol stack.

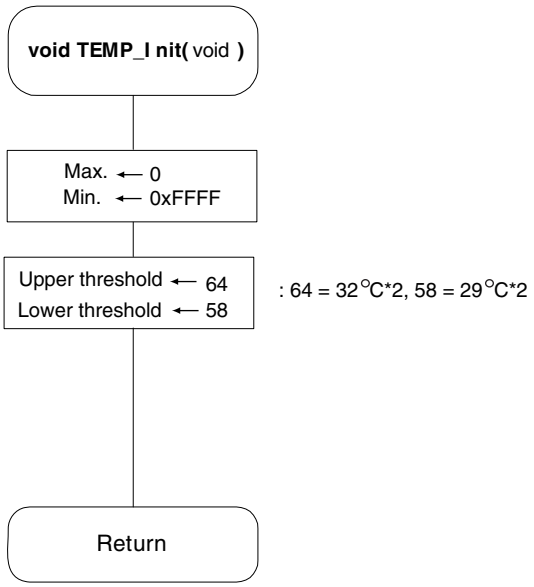


Processing overview
• The AL handler is "AL_NewOutputInd()". • This is a callback routine for the protocol stack.

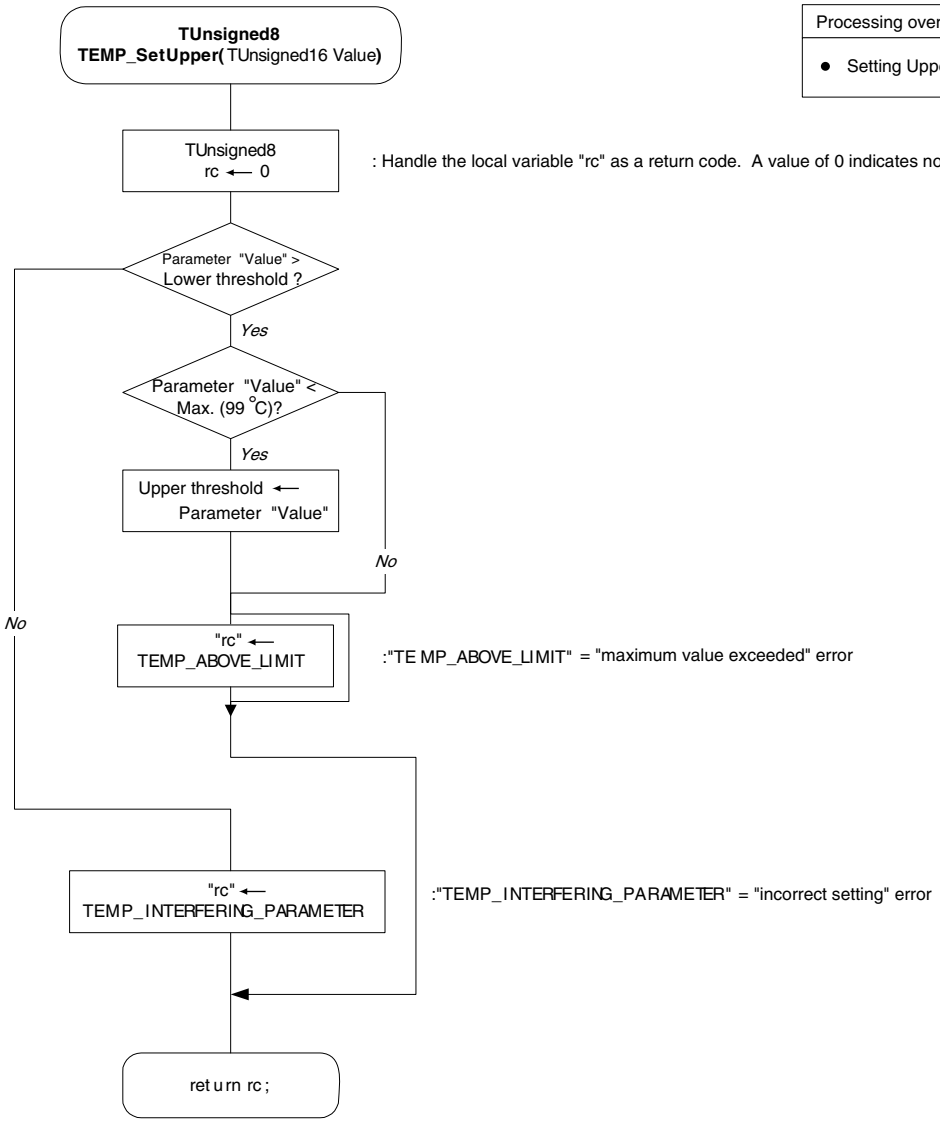

void AL_WriteInd(
TUnsigned16 Index,
TUnsigned8 SubIndex,
TUnsigned8 Length,
TUnsigned8 * pData)
* The parameter "SubIndex" is not used
: The parameter "pData" points to the write data transmitted from the host
Processing overview
The AL handler is "AL_WriteInd()".
This is a callback function for the protocol stack
This processing is executed when writing the Index data (Lower threshold/Upper threshold)
Index = 0x40 ? (Lower ?)
No
Yes
Length > 1 ?
No
AL_WriteErr(0x80, 0x30)
rc = TEMP_SetLower(*pData)
rc != 0 ?
Yes
AL_WriteErr(0x80, rc)
No
AL_WriteRes( )
Index = 0x41 ? (Upper ?)
No
AL_WriteErr(0x80, 0x11)
Yes
Length > 1 ?
No
AL_WriteErr(0x80, 0x30)
rc = TEMP_SetUpper(*pData)
rc != 0 ?
Yes
AL_WriteErr(0x80, rc)
No
AL_WriteRes( )
Return

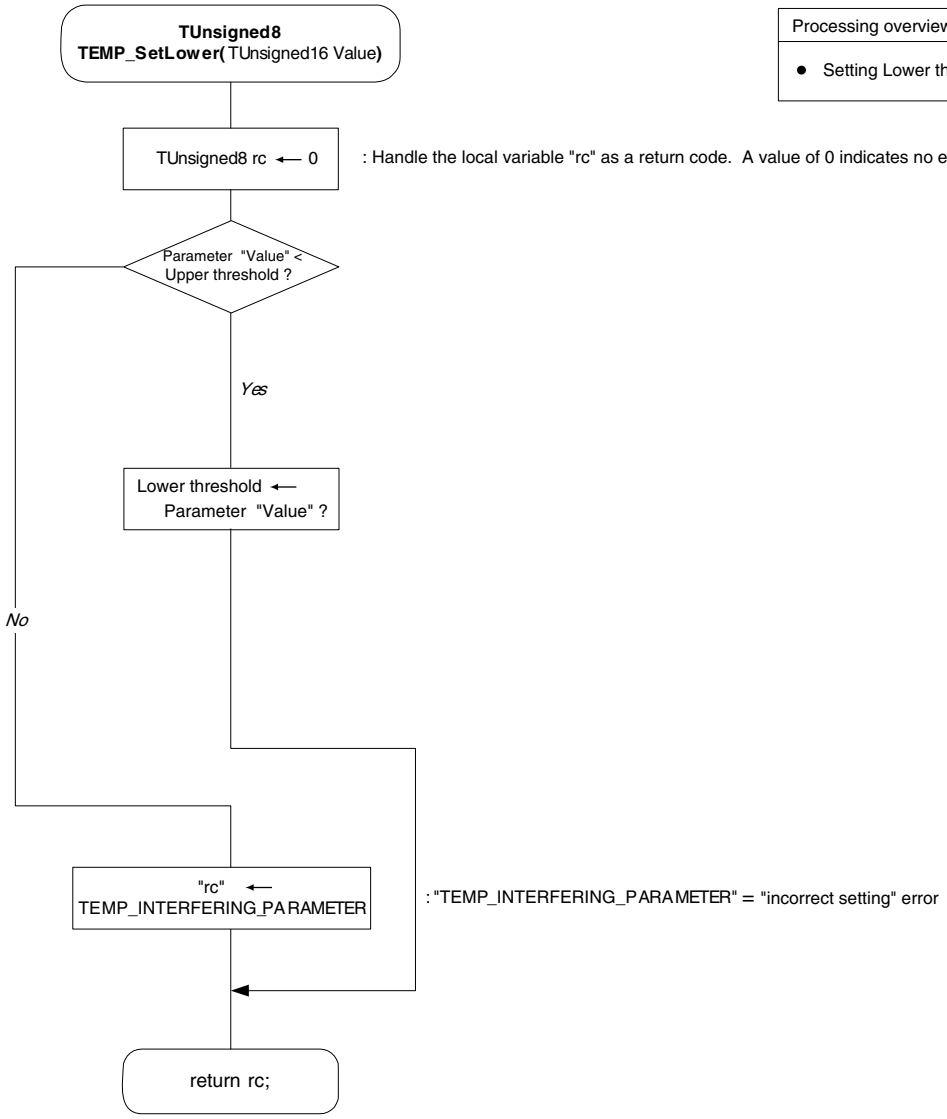

void AL_ReadInd(
TUnsigned16 Index,
TUnsigned8 SubIndex)
* The parameter "SubIndex" is not used
Processing overview
The AL handler is "AL_ReadInd()".
This is a callback function for the protocol stack
This processing is executed when reading the Index data (Lower threshold/Upper threshold, vendor name)
Branch according to the "Index" value
0x10
AL_ReadRes(sizeof(VendorName), VendorName)
0x11
AL_ReadRes(sizeof(VendorText), VendorText)
0x12
AL_ReadRes(sizeof(ProductName), ProductName)
0x13
AL_ReadRes(sizeof(ProductID), ProductID)
0x16
AL_ReadRes(sizeof(HardwareRevision), HardwareRevision)
0x40
temp = TEMP_GetLower()
AL_ReadRes(1, &tmp)
0x41
temp = TEMP_GetUpper()
AL_ReadRes(1, &tmp)
Others
AL_ReadErr(0x80, 0x11)
Return

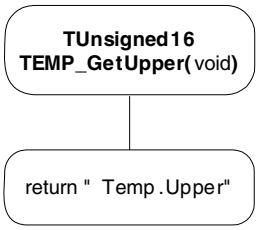
12.3.2 Sensor control module flowchart



Processing overview
• Initialize the temperature sensor • Initialize relevant variables


TUnsigned8
TEMP_SetUpper(TUnsigned16 Value)
Processing overview
Setting Upper threshold
TUnsigned8
rc ← 0
: Handle the local variable "rc" as a return code. A value of 0 indicates no error.
Parameter "Value" >
Lower threshold ?
Yes
No
Parameter "Value" <
Max. (99 °C)?
Yes
No
Upper threshold ←
Parameter "Value"
"rc" ←
TEMP_ABOVE_LIMIT
:"TEMP_ABOVE_LIMIT" = "maximum value exceeded" error
"rc" ←
TEMP_INTERFERING_PARAMETER
:"TEMP_INTERFERING_PARAMETER" = "incorrect setting" error
return rc;

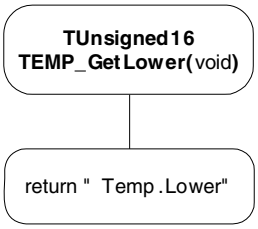

TUnsigned8
TEMP_SetLower(TUnsigned16 Value)
Processing overview
Setting Lower threshold
TUnsigned8 rc ← 0
: Handle the local variable "rc" as a return code. A value of 0 indicates no error.
Parameter "Value" <
Upper threshold ?
Yes
Lower threshold ←
Parameter "Value" ?
No
"rc" ←
TEMP_INTERFERING_PARAMETER
: "TEMP_INTERFERING_PARAMETER" = "incorrect setting" error
return rc;



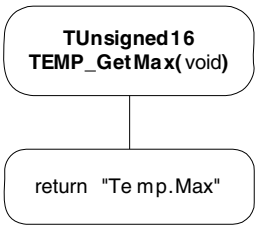
: Return the value of the Upper threshold variable.

: The Temp structure includes values such as the maximum, minimum, and threshold values.

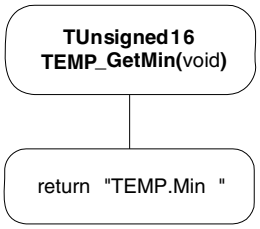
Processing overview
• Acquire the Upper threshold value • Acquire the Lower threshold value • Acquire the maximum-rotation value • Acquire the minimum-rotation value



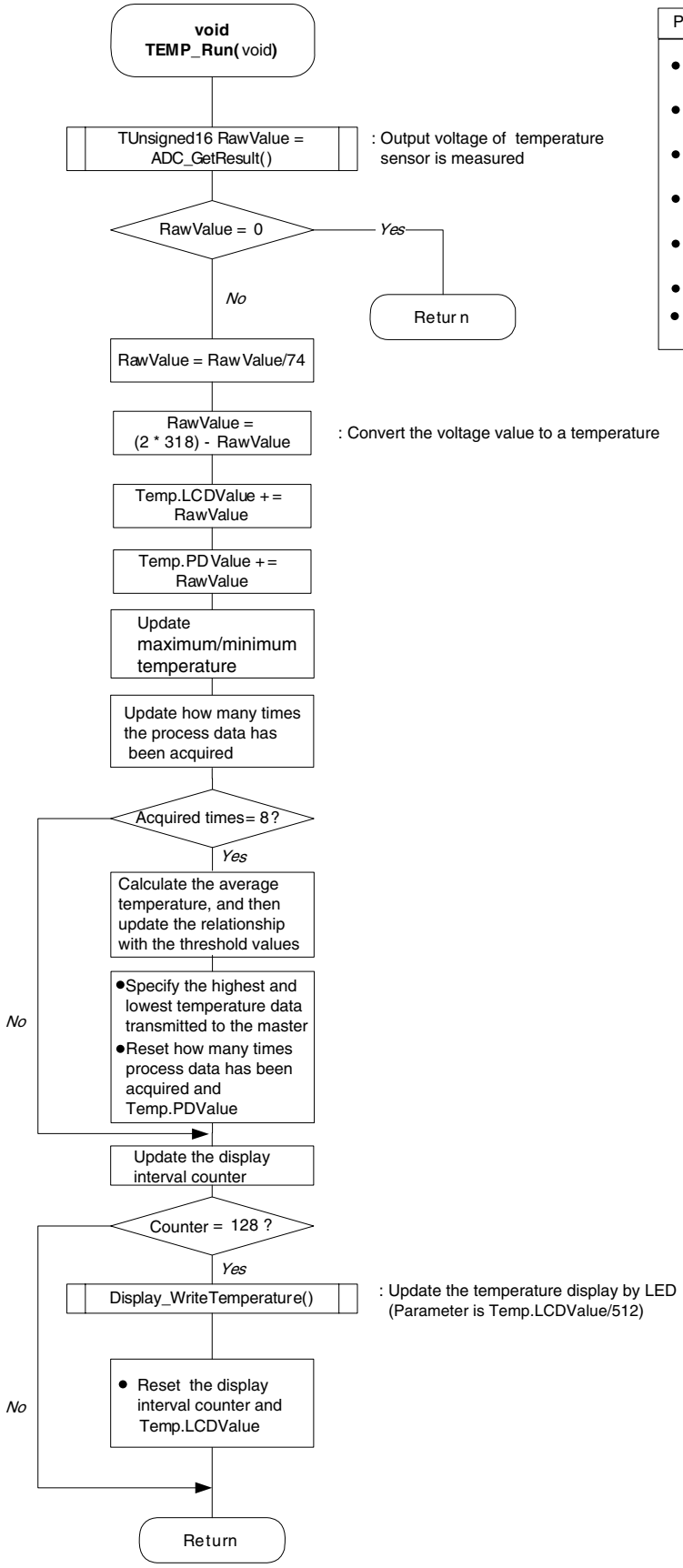
: Return the value of the Lower threshold variable.



:Return the value of the maximum temperature variable.



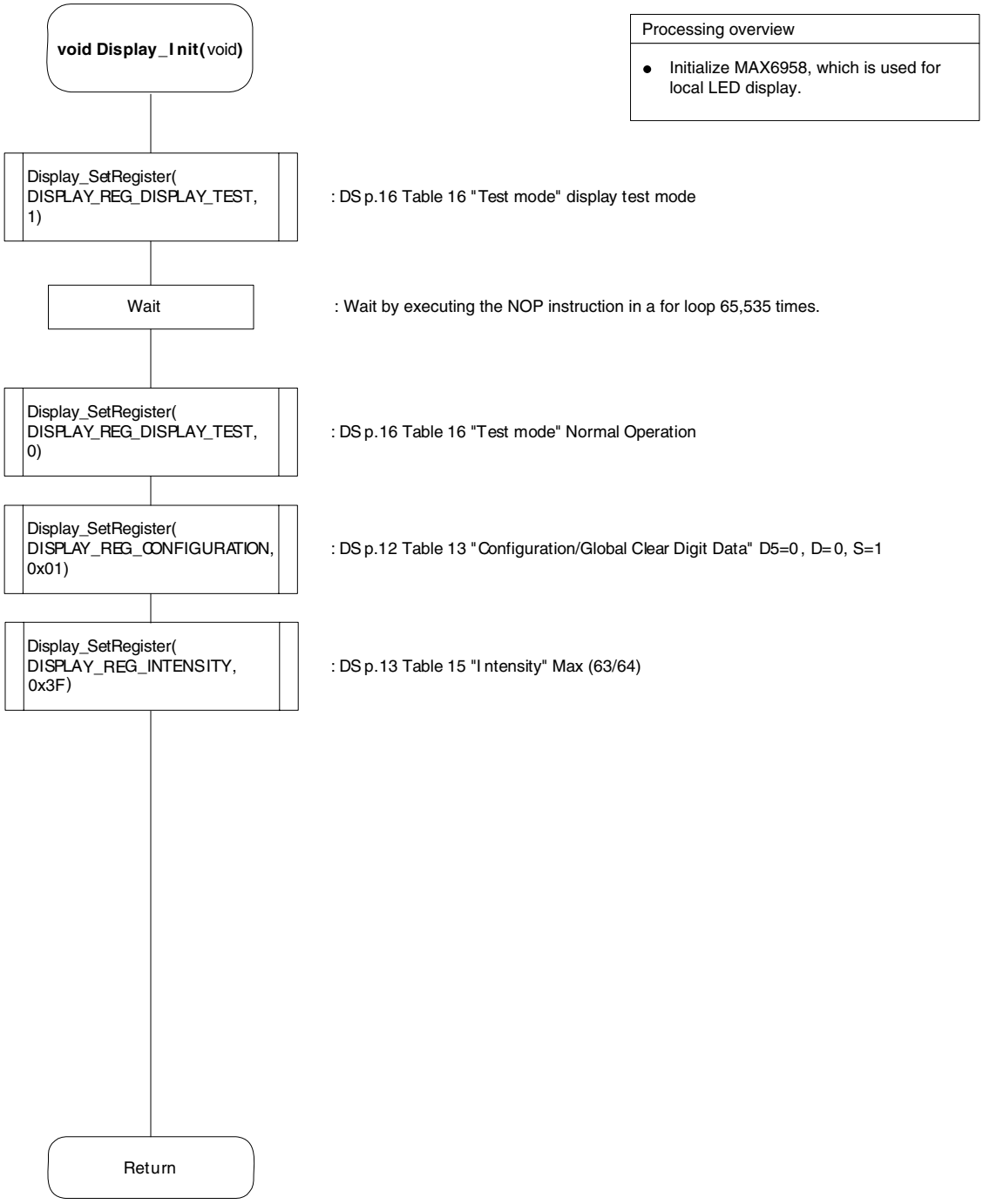
:Return the value of the minimum temperature variable.

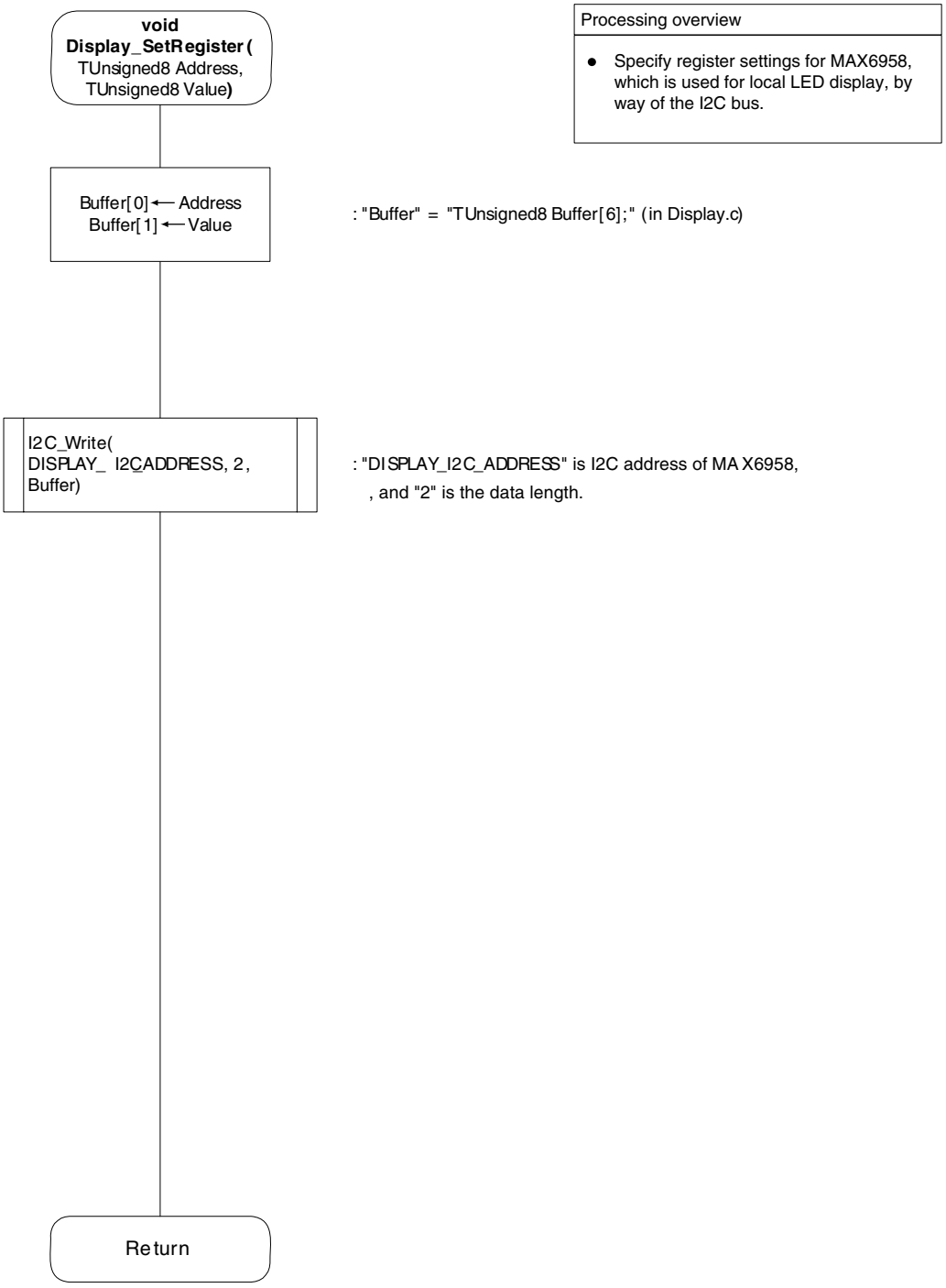


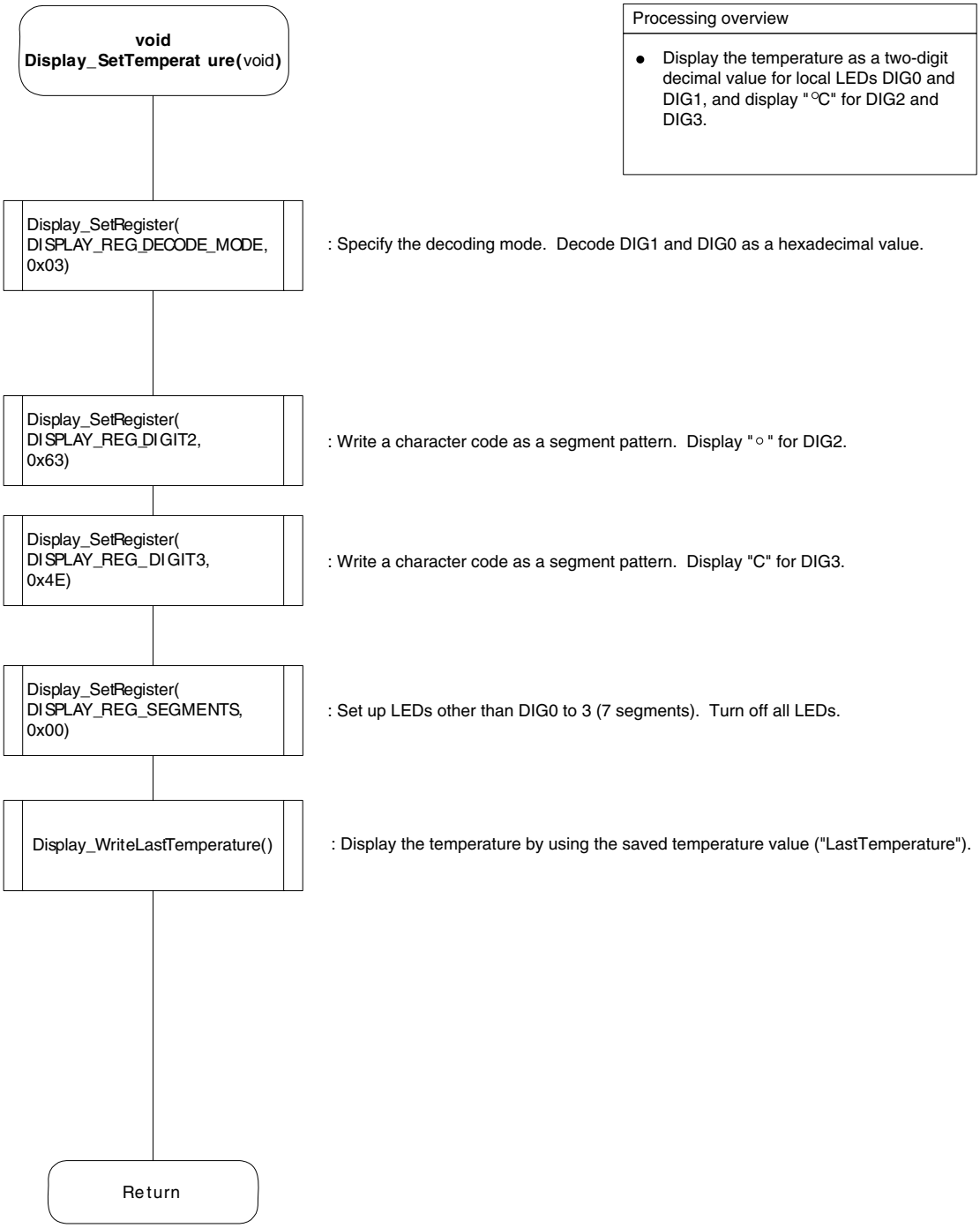
Processing overview

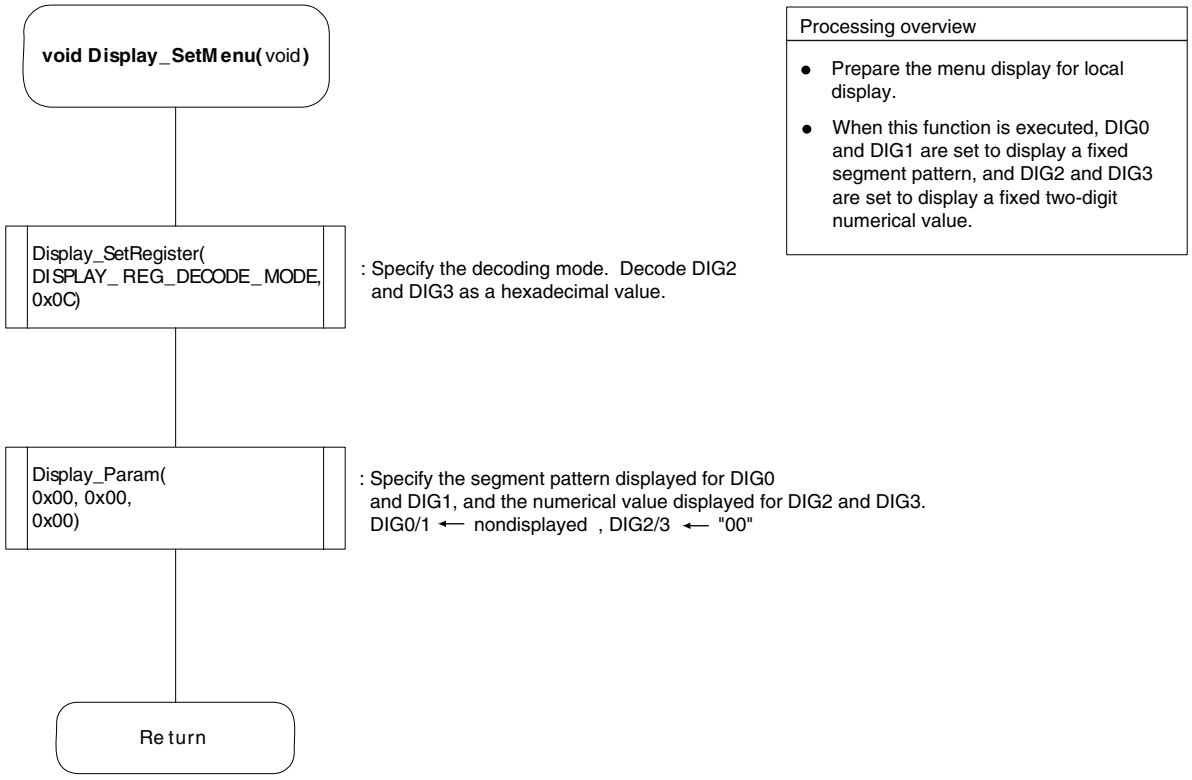
- This is the main task processing for the temperature sensor.
- Constantly execute this processing from the main loop.
- Perform the following processing by using a counter to determine the timing:
- → Convert the voltage value in the sensor output to a temperature value.
- → Average results eight times to determine the temperature value.
- → Update process data.
- → Update the local display.

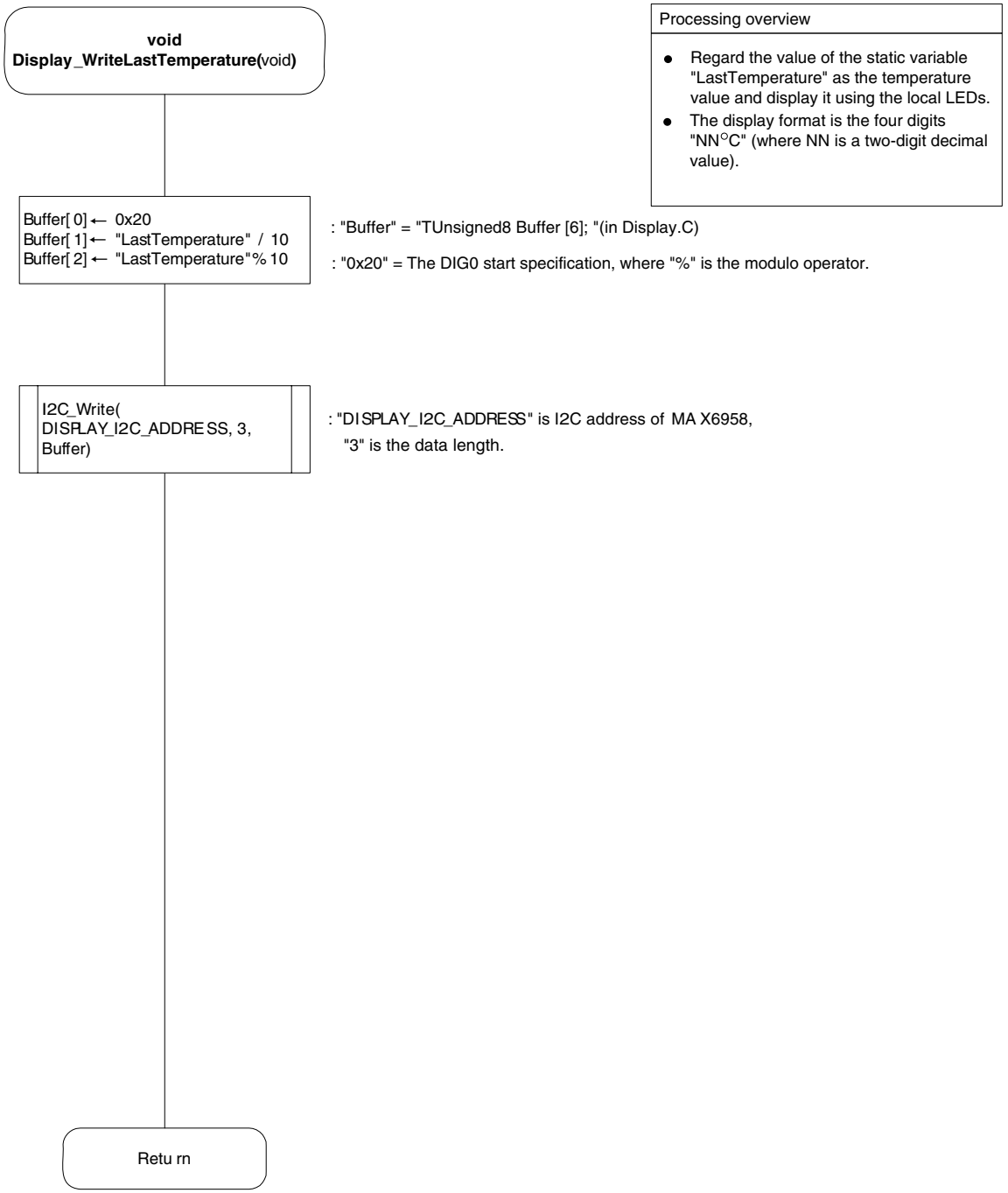
12.3.3 Display control module flowchart

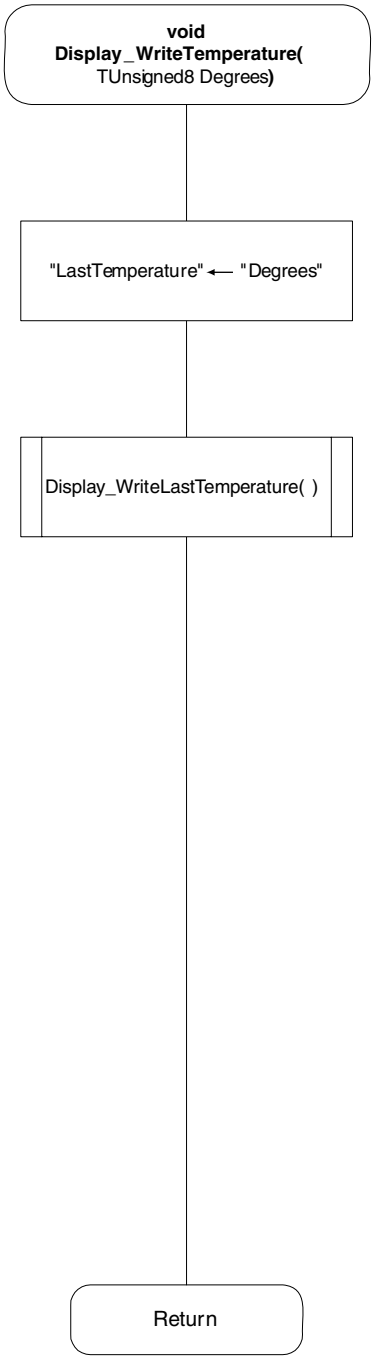



void
Display_SetRegister(
TUnsigned8 Address,
TUnsigned8 Value)
Processing overview
Specify register settings for MAX6958, which is used for local LED display, by way of the I2C bus.
Buffer[0] ← Address
Buffer[1] ← Value
: "Buffer" = "TUnsigned8 Buffer[6];" (in Display.c)
I2C_Write(
DISPLAY_ I2CADDRESS, 2,
Buffer)
: "DISPLAY_I2C_ADDRESS" is I2C address of MAX6958,
, and "2" is the data length.
Return


void
Display_SetTemperature(void)
Processing overview
● Display the temperature as a two-digit decimal value for local LEDs DIG0 and DIG1, and display "°C" for DIG2 and DIG3.
Display_SetRegister(DISPLAY_REG_DECODE_MODE, 0x03)
: Specify the decoding mode. Decode DIG1 and DIG0 as a hexadecimal value.
Display_SetRegister(DISPLAY_REG_DIGIT2, 0x63)
: Write a character code as a segment pattern. Display "○" for DIG2.
Display_SetRegister(DISPLAY_REG_DIGIT3, 0x4E)
: Write a character code as a segment pattern. Display "C" for DIG3.
Display_SetRegister(DISPLAY_REG_SEGMENTS, 0x00)
: Set up LEDs other than DIG0 to 3 (7 segments). Turn off all LEDs.
Display_WriteLastTemperature()
: Display the temperature by using the saved temperature value ("LastTemperature").
Return

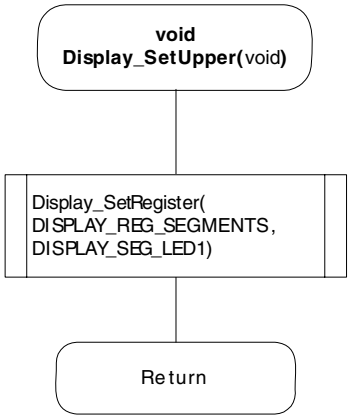

void Display_SetMenu(void)
Display_SetRegister(
DISPLAY_REG_DECODE_MODE,
0x0C)
: Specify the decoding mode. Decode DIG2
and DIG3 as a hexadecimal value.
Display_Param(
0x00, 0x00,
0x00)
: Specify the segment pattern displayed for DIG0
and DIG1, and the numerical value displayed for DIG2 and DIG3.
DIG0/1 ← nondisplayed , DIG2/3 ← "00"
Return
Processing overview
Prepare the menu display for local display.
When this function is executed, DIG0 and DIG1 are set to display a fixed segment pattern, and DIG2 and DIG3 are set to display a fixed two-digit numerical value.


void
Display_WriteLastTemperature(void)
Processing overview
Regard the value of the static variable "LastTemperature" as the temperature value and display it using the local LEDs.
The display format is the four digits "NN°C" (where NN is a two-digit decimal value).
Buffer[0] ← 0x20
Buffer[1] ← "LastTemperature" / 10
Buffer[2] ← "LastTemperature" % 10
: "Buffer" = "TUnsigned8 Buffer [6]; "(in Display.C)
: "0x20" = The DIG0 start specification, where "%" is the modulo operator.
I2C_Write(
DISPLAY_I2C_ADDRESS, 3,
Buffer)
: "DISPLAY_I2C_ADDRESS" is I2C address of MAX6958,
"3" is the data length.
Return



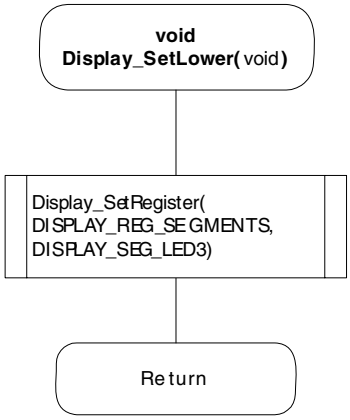
Processing overview

- Regard the value of the parameter "Degrees" as the temperature value and display it using the local LEDs.
- Update the static variable "LastTemperature" with the value of "Degrees".
- The display format is the four digits "NN°C" (where NN is a two-digit decimal value).

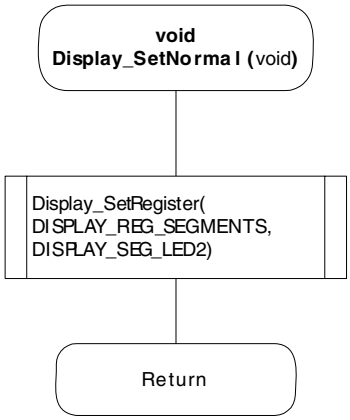


: Specify segment control.
Turns on LED1 (Above) only.

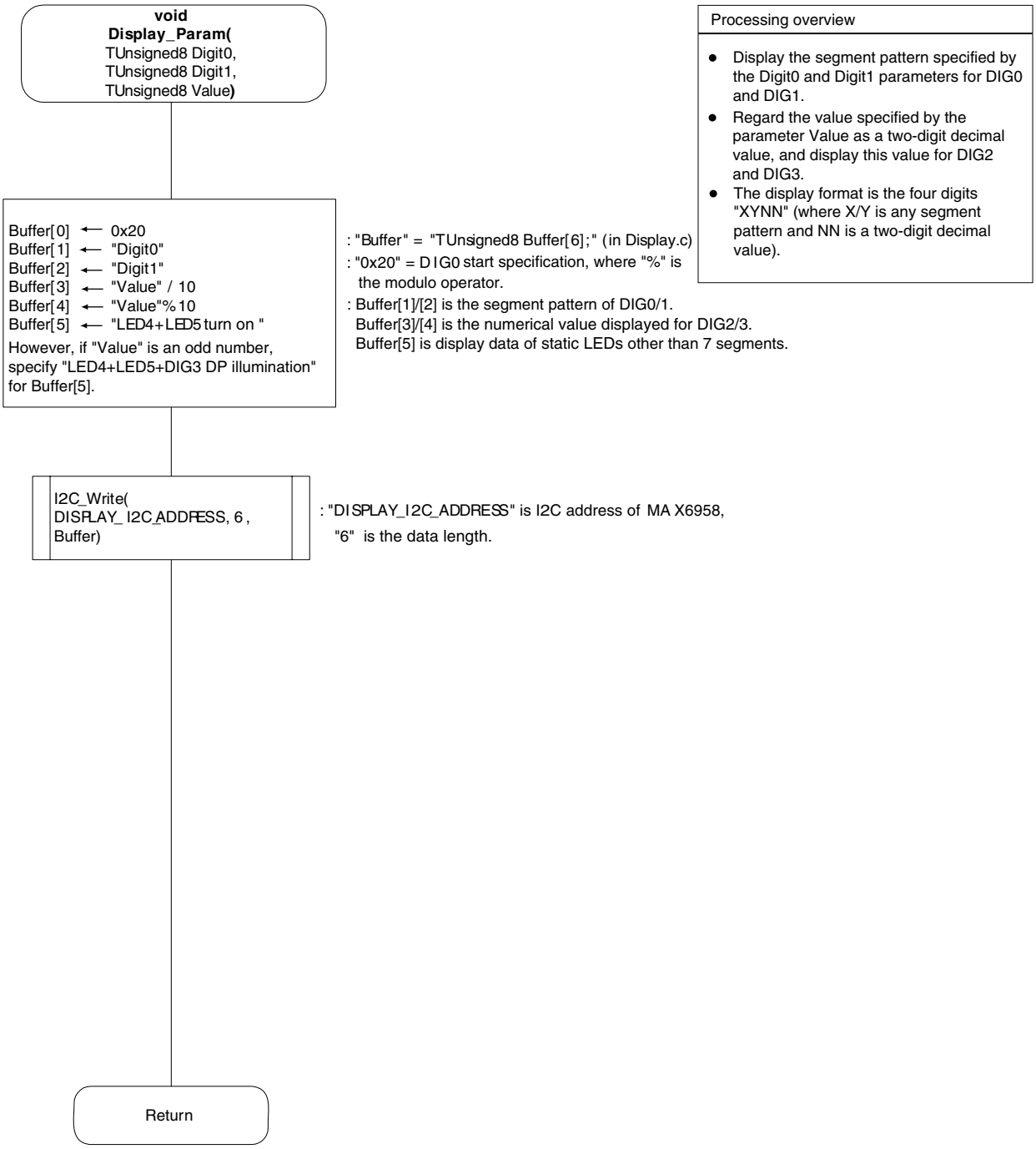
Processing overview
• Control "Above (LED1)", "Between (LED2)", and "Below (LED3)", the static LEDs used for local display. • Display_SetUpper() only turns on "Above". • Display_SetLower() only turns on "Below". • Display_SetNormal() only turns on "Between".



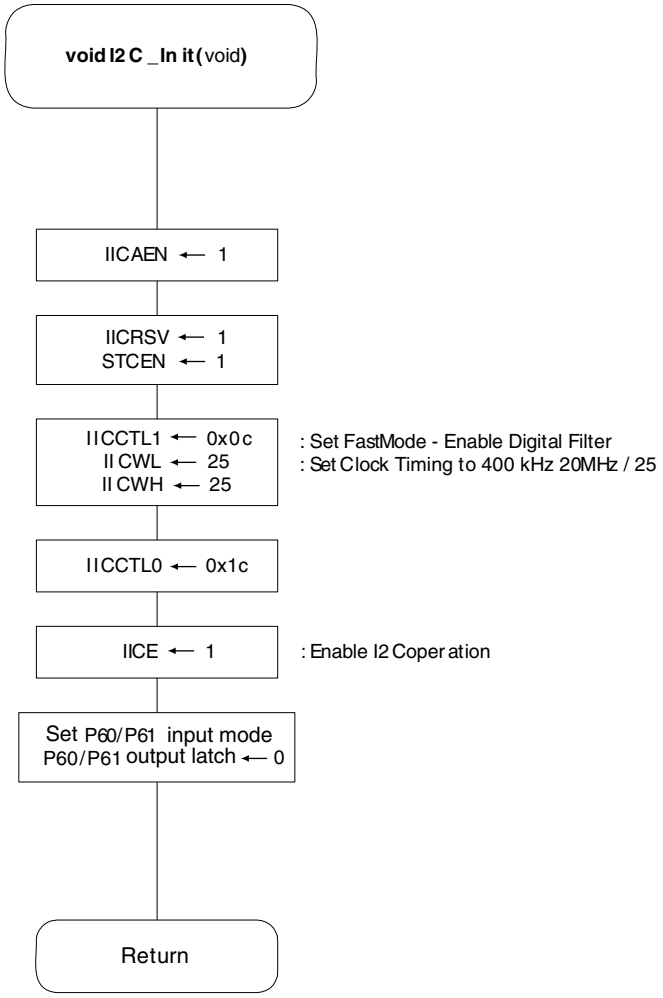
: Specify segment control.
Turns on LED3 (Below) only.



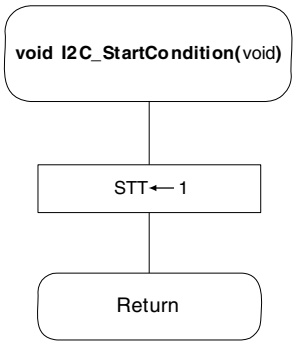
: Specify segment control.
Turns on LED2 (Between) only.


void
Display_Param(
TUnsigned8 Digit0,
TUnsigned8 Digit1,
TUnsigned8 Value)
Buffer[0] ← 0x20
Buffer[1] ← "Digit0"
Buffer[2] ← "Digit1"
Buffer[3] ← "Value" / 10
Buffer[4] ← "Value"%10
Buffer[5] ← "LED4+LED5 turn on "
However, if "Value" is an odd number, specify "LED4+LED5+DIG3 DP illumination" for Buffer[5].
: "Buffer" = "TUnsigned8 Buffer[6];" (in Display.c)
: "0x20" = DIG0 start specification, where "%" is the modulo operator.
: Buffer[1]/[2] is the segment pattern of DIG0/1.
Buffer[3]/[4] is the numerical value displayed for DIG2/3.
Buffer[5] is display data of static LEDs other than 7 segments.
I2C_Write(
DISPLAY_I2C_ADDRESS, 6 ,
Buffer)
: "DISPLAY_I2C_ADDRESS" is I2C address of MAX6958,
"6" is the data length.
Return
Processing overview
Display the segment pattern specified by the Digit0 and Digit1 parameters for DIG0 and DIG1.
Regard the value specified by the parameter Value as a two-digit decimal value, and display this value for DIG2 and DIG3.
The display format is the four digits "XYNN" (where X/Y is any segment pattern and NN is a two-digit decimal value).

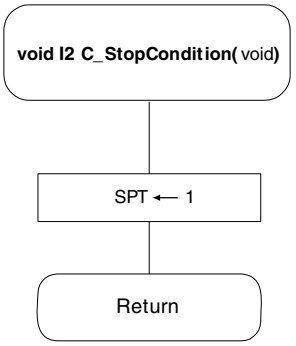
12.3.4 I2C driver module flowchart



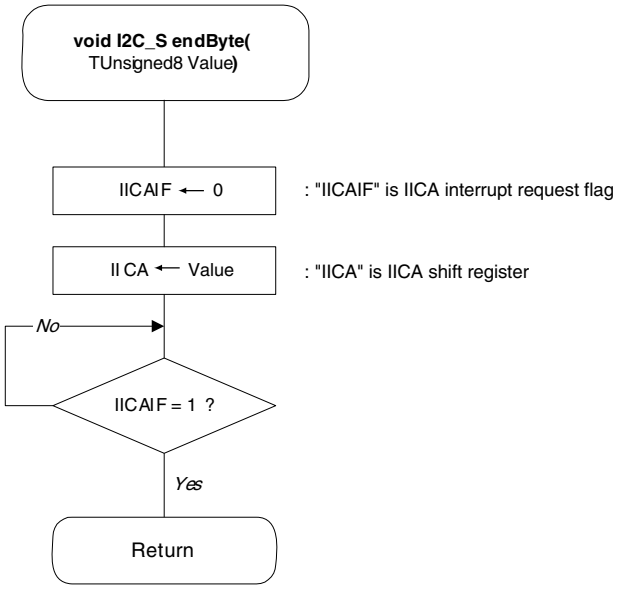
Processing overview
• Initialize the hardware (the "IICA" macro) used for I2C communication.



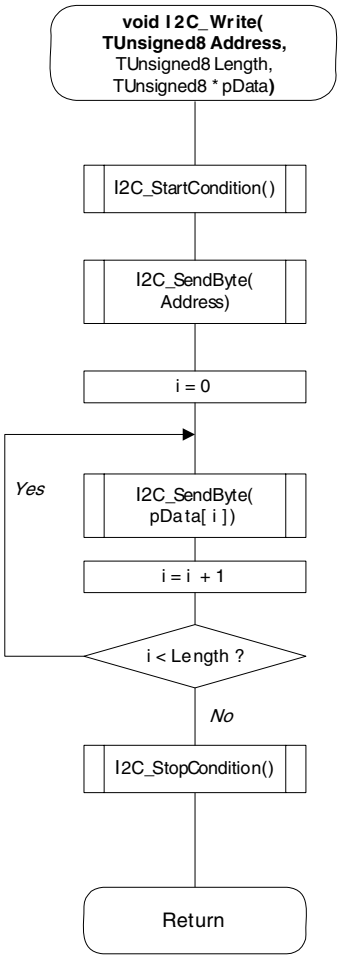
Processing overview
• Issue the I2C start condition.



Processing overview
• Issue the I2C stop condition.



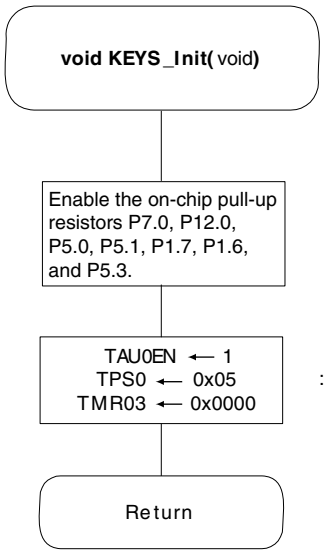
Processing overview
• Transmit the eight bits of data specified by the parameter "Value" to the I2C bus.



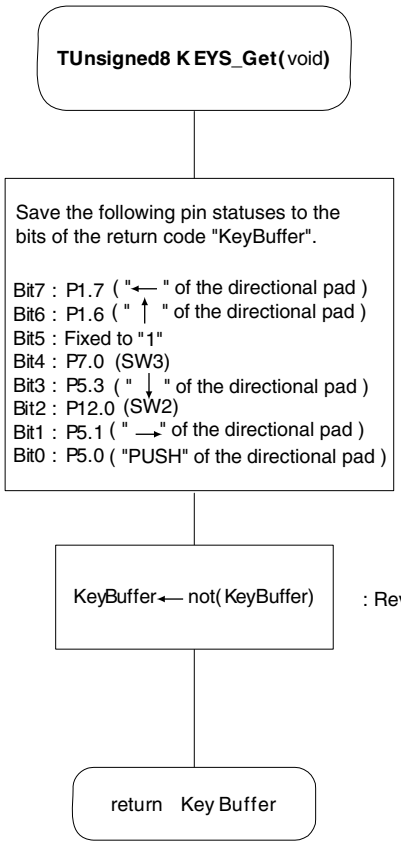
: Use the local variable "i" as a counter.

Processing overview
• Transmit one frame of data to the I2C bus. • The parameter "Address" is the I2C address, Length is the frame size, and pData is a pointer to the transmitted data.

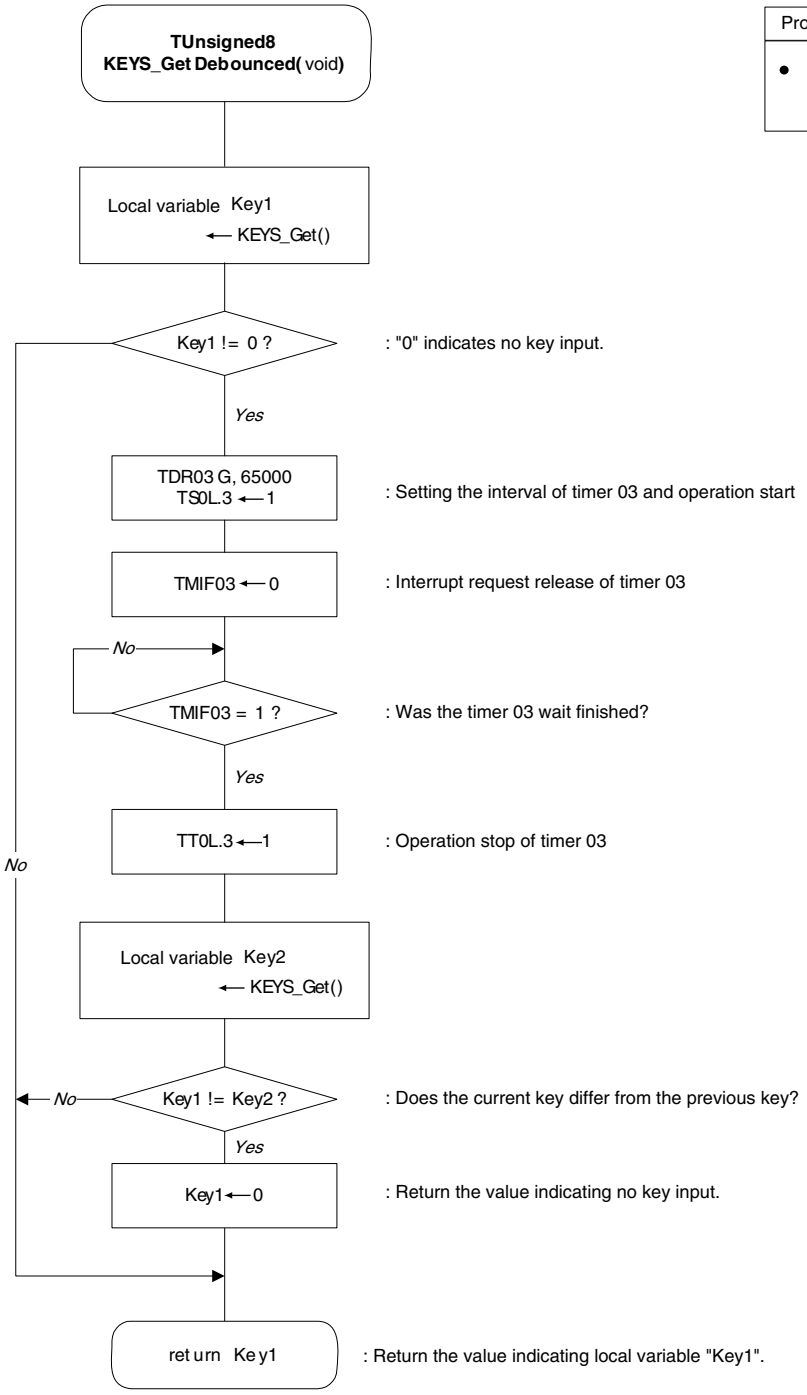
12.3.5 Key input module flowchart



Processing overview
• Initialize key-input hardware.

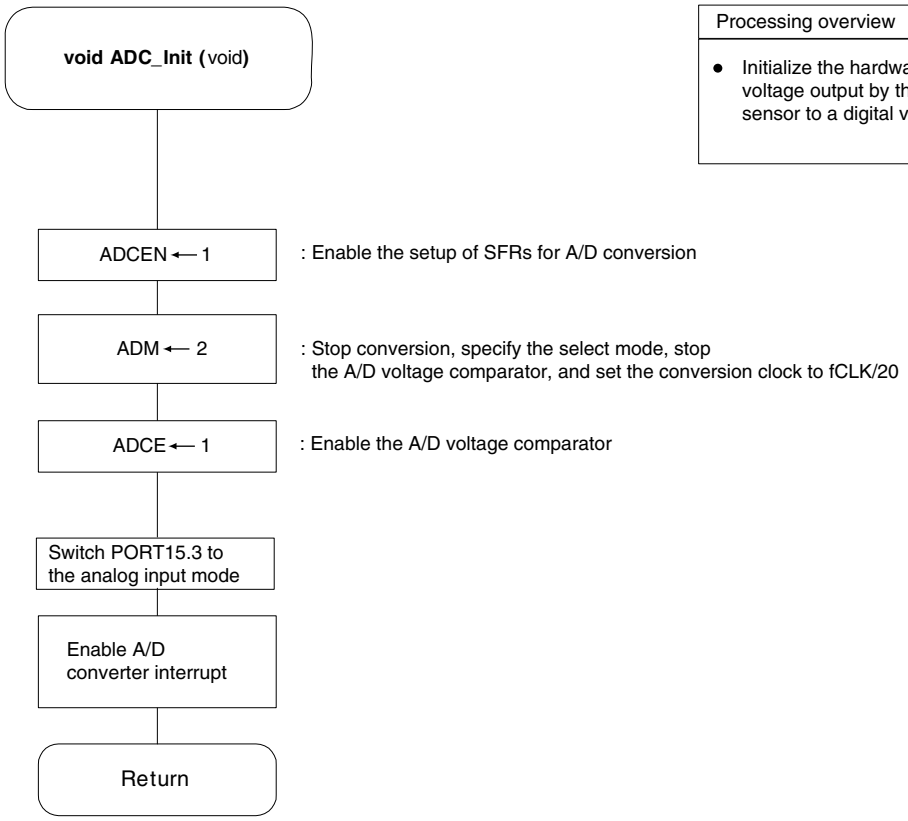


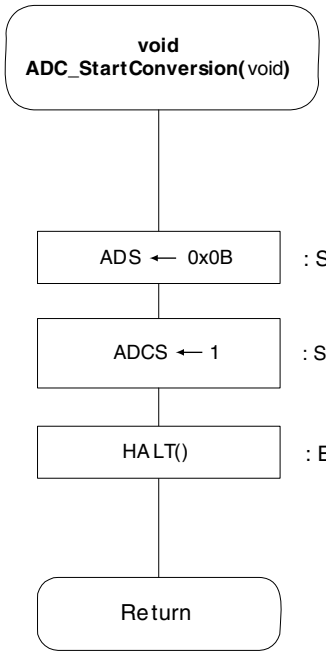
Processing overview
• Acquire the current key input status as a bitmap. • In the return value, set bits corresponding to keys being pressed to "1".



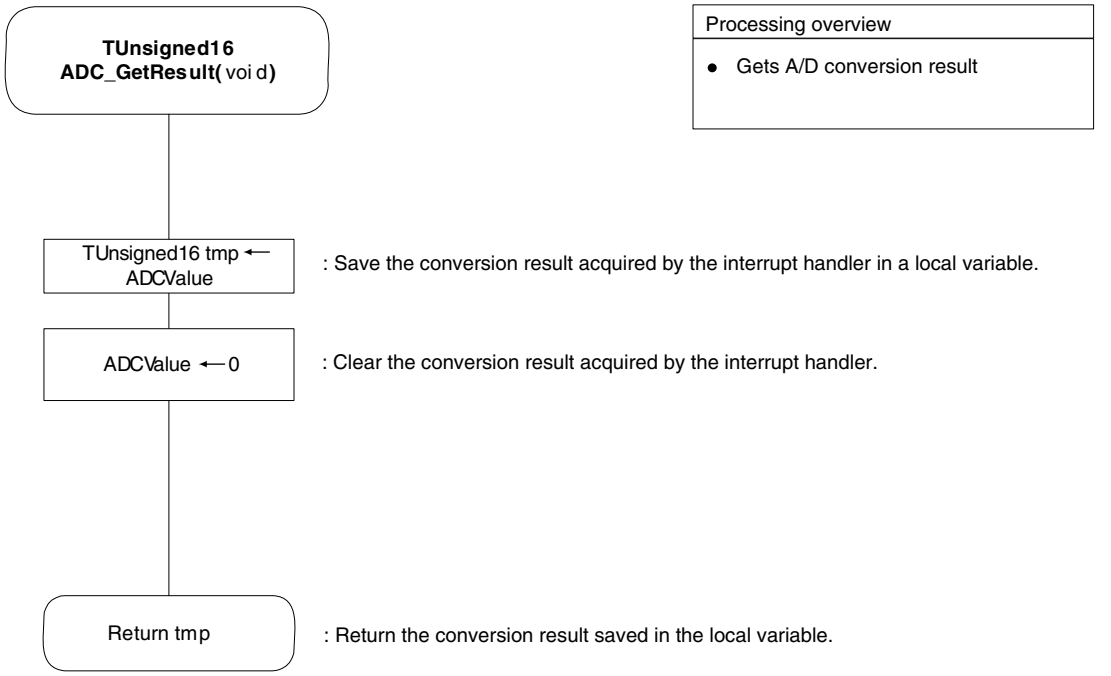
Processing overview
• Acquire the status of a stable key, and return it.

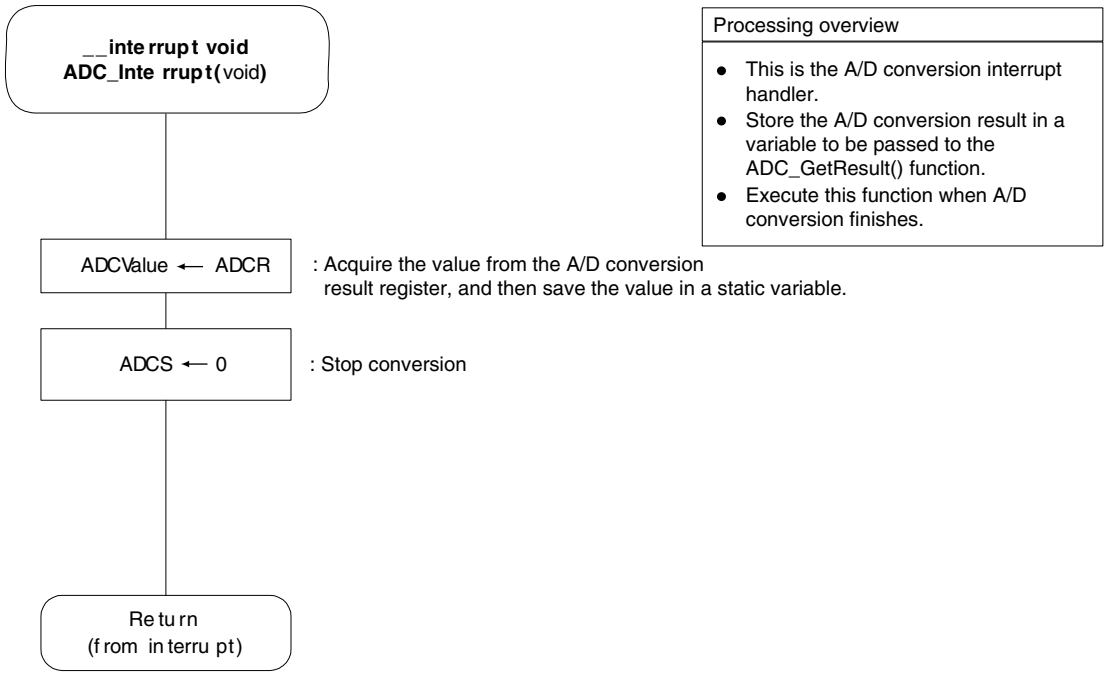
12.3.6 A/D conversion module flowchart





Processing overview
• A/D conversion starts and HALT mode is set.


TUnsigned16
ADC_GetResult(void)
Processing overview
Gets A/D conversion result
TUnsigned16 tmp ← ADCValue
: Save the conversion result acquired by the interrupt handler in a local variable.
ADCValue ← 0
: Clear the conversion result acquired by the interrupt handler.
Return tmp
: Return the conversion result saved in the local variable.


__interrupt void
ADC_Interrupt(void)
Processing overview
This is the A/D conversion interrupt handler.
Store the A/D conversion result in a variable to be passed to the ADC_GetResult() function.
Execute this function when A/D conversion finishes.
ADCValue ← ADCR
: Acquire the value from the A/D conversion result register, and then save the value in a static variable.
ADCS ← 0
: Stop conversion
Return
(from interrupt)

CHAPTER 13 IO-LINK RPM SENSOR SAMPLE

For the 78K0R-Link It! board, the demo software introduced in chapter 12 is already written to the microcontroller at shipment.
This chapter introduces another sample program, that for the RPM sensor.

To use the sample program for the RPM sensor, the program must be written to the 78K0R-Link It! board. The file to write is SamplePrograms¥rpm_sensor¥hex¥rpm_sensor.hex. For further details please refer to CHAPTER 4 and CHAPTER 10.

The basic usage method, notes, and other information are the same as for the temperature sensor.

This sample program uses the magnetic sensor on the 78K0R-Link It! board, which is introduced in 3.19. This sensor generates pulses according to the magnetic field strength and switches of the north and south poles. When a magnet is attached to an object that can be rotated and the object is rotated near the sensor on the board, the number of pulses generated per fixed period of time increases and decreases according to the rotation speed. This can be used to measure RPM. In addition, by changing a constant defined in the source code, it is possible to adjust the RPM calculation method according to how many magnets are attached to the object to be rotated.

Caution When operating the 78K0R-Link It! board in the standalone mode without using a debugger, the jumpers and switches on the board must have the factory settings described in 7.1 Factory Settings for Jumpers and Switches.

13.1 Demo Features

After writing the sample program for the RPM sensor to the board, if IO-Link communication with the master device is established, the current board RPM value is displayed. All RPM values are displayed in rpm.

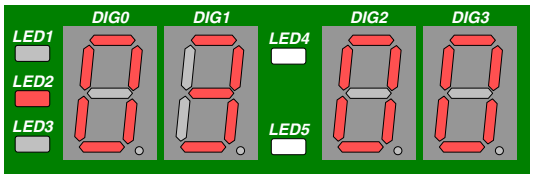


Figure 13-1. EZ-0010 RPM Sensor Demo Display

The current RPM is displayed with 4 digits (DIG0 to DIG4). The user can set two (maximum-rotation threshold value/minimum-rotation threshold value) thresholds, set by default to 500 rpm, 200 rpm respectively, when the board is connected. LED1 to LED3 are alarm LEDs representing thresholds set by the user.

- LED1 lights up when the measured RPM is above the maximum-rotation threshold value
- LED2 lights up when the measured RPM is between the two thresholds
- LED3 lights up when the measured RPM is below the minimum-rotation threshold value

13.1.1 Min/Max Rotation Display

The minimum/maximum rotation measured since the sensor has been connected to the IO-Link master can be readout on the display at any time when pressing SW2 for the minimum-rotation, and SW3 for the maximum-rotation. Refer to Figures 13-2 and 13-3.

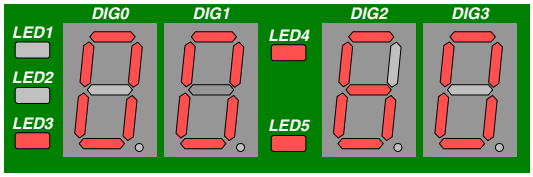


Figure 13-2. Minimum-Rotation Readout (60 rpm)

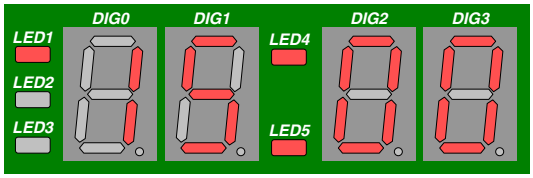


Figure 13-3. Maximum-Rotation Readout (1500 rpm)

13.1.2 RPM thresholds setting

The user has the ability to set two thresholds (a Lower and an Upper threshold).

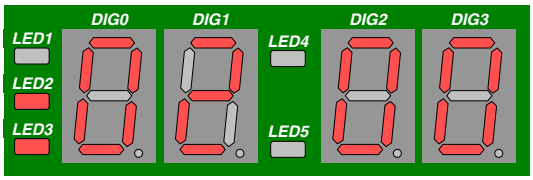


Figure13-4. Default Lower Threshold (200 rpm)

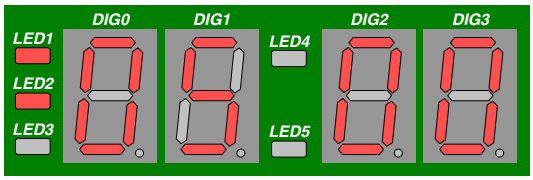


Figure 13-5. Default Lower Threshold (500 rpm)

(1) To enter the settings menu, press the Center button on SW4. The display thereafter shows the current value of the lower threshold. Refer to Figures 13-4 and 13-5.

- Pushing the SW4 button to the right increments the lower threshold in 100rpm steps
- Pushing the SW4 button to left decrements the lower threshold in 100rpm steps
- Pushing the SW4 button up, increments the upper threshold it in 100rpm steps
- Pushing the SW4 button down decrements the upper threshold in 100rpm steps

When displaying or setting the maximum-rotation threshold, LED1 and LED2 turn on. Similarly, when displaying or setting the minimum-rotation threshold, LED2 and LED3 turn on. The upper threshold can never be lower or equal to the lower threshold.

(2) To confirm the settings, press the center button on SW4 again. The display thereafter, reverts to the current measured RPM display.

All of the RPM values, thresholds values and settings can also be changed or read using the TMG USB IO-LINK Master device tool.

13.2 Using *78K0R-Link It!* Board demo with TMG IO-Link Device Tool.

The IO-Link device tool can be used for the configuration of IO-Link masters, setting and steering parameters as well as the diagnosis of IO-Link devices.

IO-Link master initiates the communication, and channels information from the board to the host machine. The operator is able to see the information on the host machine via the installed IO-Link Device Tool.

13.2.1 Getting started with the TMG master

The user has the ability to set two thresholds (a Lower and an Upper threshold).

(1) Launch the IO-Link device Tool.

(2) Once the tool is up and running, two panes can be seen in the GUI (Topology on the left, and Catalog on the right),see Figure 13-6.

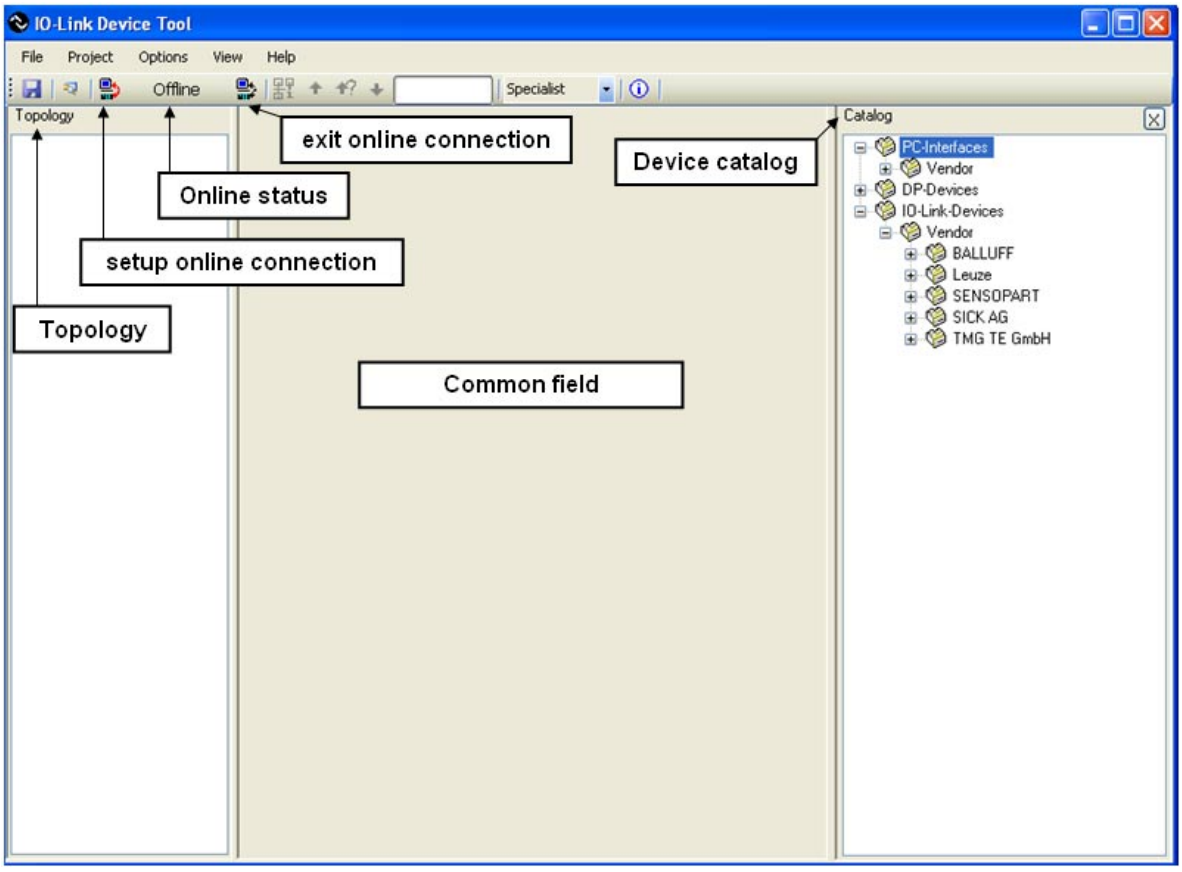


Figure 13-6. IO-Link Device Tool GUI

The topology pane shows the topology from the PC interfaces to the IO-Link Devices. The Catalog pane shows all the Devices installed with the tool.

On the symbol bar, two icons (setup/exit online connection) allow to set the IO-Link line status.

When the line is Online, the “Online status” symbol will blink green with the symbol “online”

The common field is currently blank but it will display the description of the devices present in the Topology view.

13.2.2 Updating the IO-Link Device catalog

Before the 78K0R-Link It! R.P.M. sensor can be displayed in the device tool, the IO-Link devices catalog must be updated

(1) In the Catalog pane, right click on “IO-Link Devices” and select [Actuate catalog]

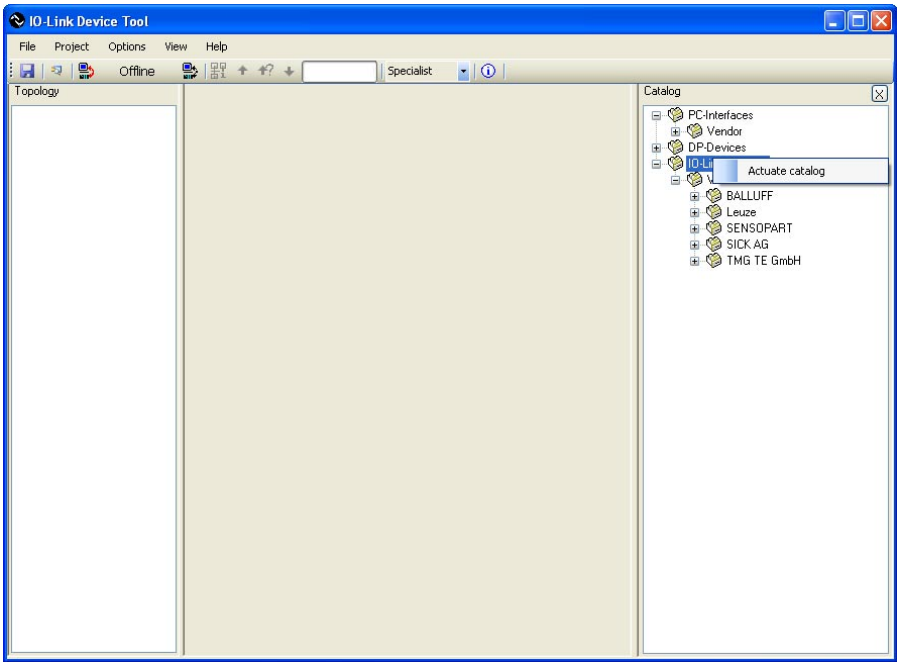


Figure 13-7. Catalog update <1>

(2) Press [Yes] in the small window with the message [Do you want to insert 78K0R–Link It! R.P.M. Sensor of NEC Electronics (Europe) into the catalog? No skips the shown device - Cancel stops updating catalog]

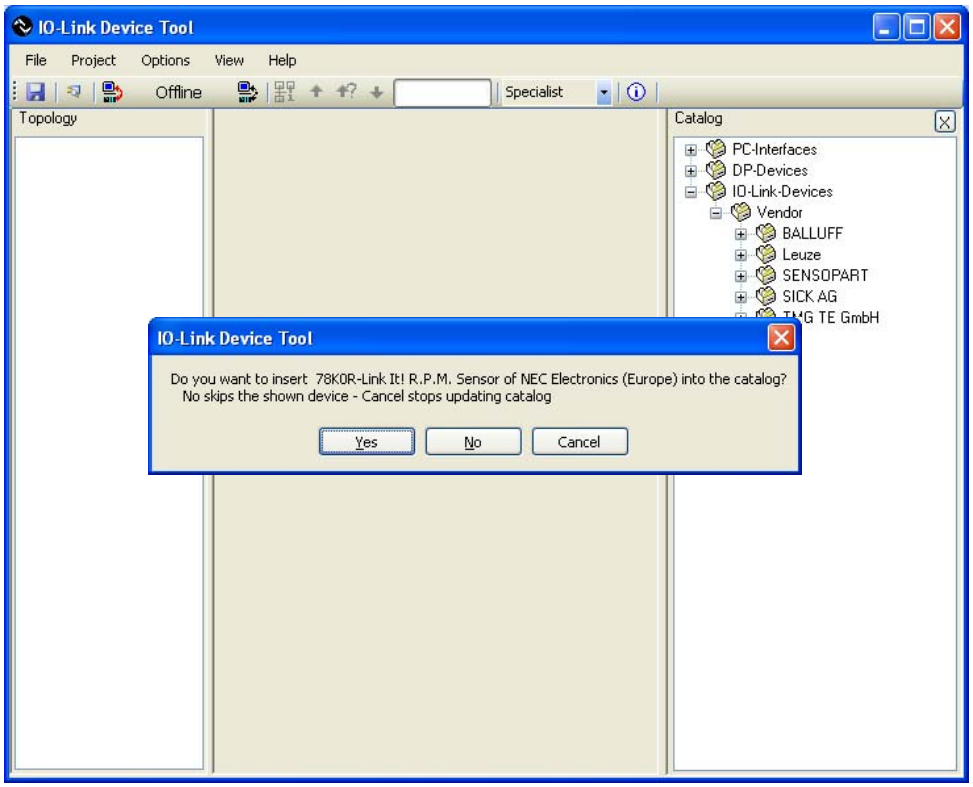


Figure 13-8. Catalog update <2>

13.2.3 Catalog update confirmation

A successful update will show the NEC Electronics (Europe) vendor and the *78K0R–Link It!* R.P.M. sensor, in the IO-Link Devices section of the catalog. The TMG USB IO-Link master can also be seen under PC Interfaces section of the catalog.

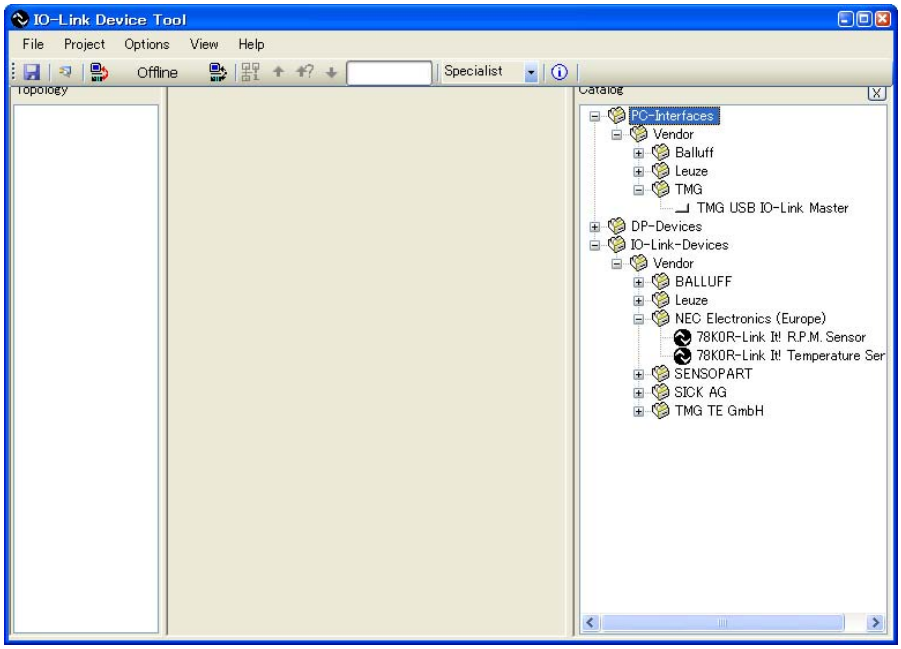


Figure 13-9. Catalog update confirmation

13.2.4 IO-Link communication set-up

(1) From the Catalog pane, drag and drop the TMG USB IO-Link Master into the Topology pane.

(2) Left click on TMG USB IO-Link Master DE in the Topology pane, and the IO-Link Master's details can be seen in the Common pane (centre section of the window).

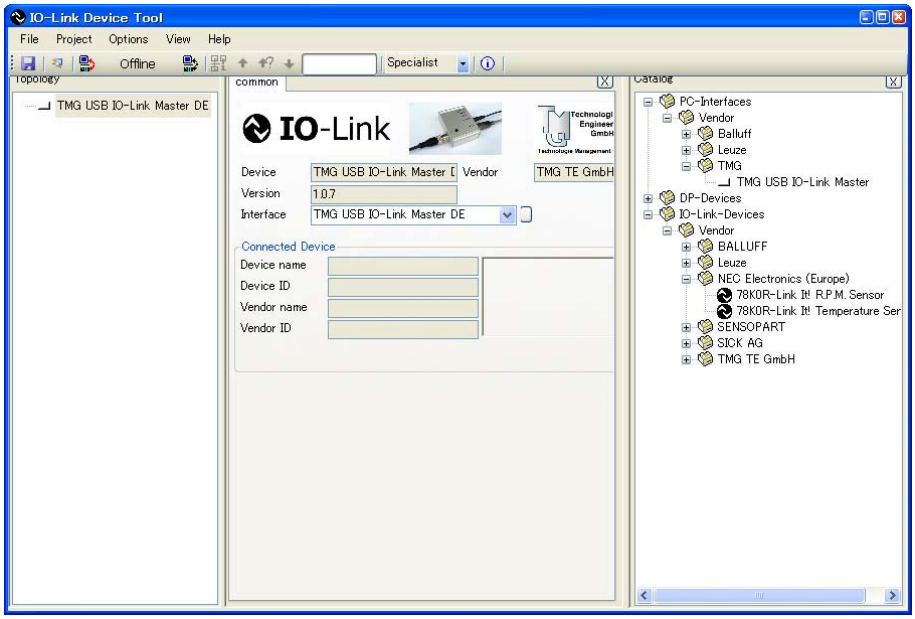


Figure 13-10. TMG USB IO-Link Master

(3) Left click on the "Setup online connection" icon in the Device Tool GUI, will show the "Check Config" button under "Connected Device" in the Common section.

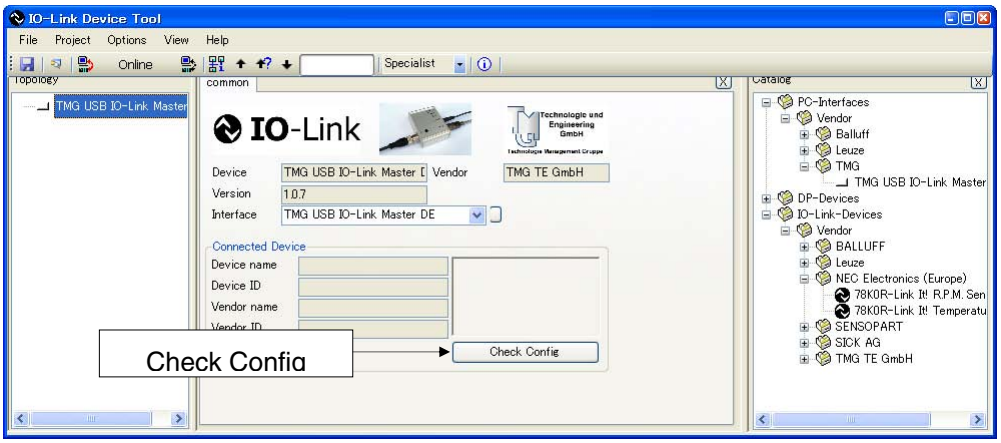


Figure 13-11. Online status, Check Config

(4) Click on the "Check Config" button.

The LEDs on the Master will blink as the master tries to connect to the *78K0R–Link It!* board If the *78K0R–Link It!* board is working and the master can connect to it, a small window will show with information on the board and a “Take over type of device into engineering” button. See **Figure 13-12. Check Config successful.** If the master cannot connect to the *78K0R–Link It!* board a window with an error message “can’t read configuration” will appear. See **Figure 13-13. Check Config failure.**

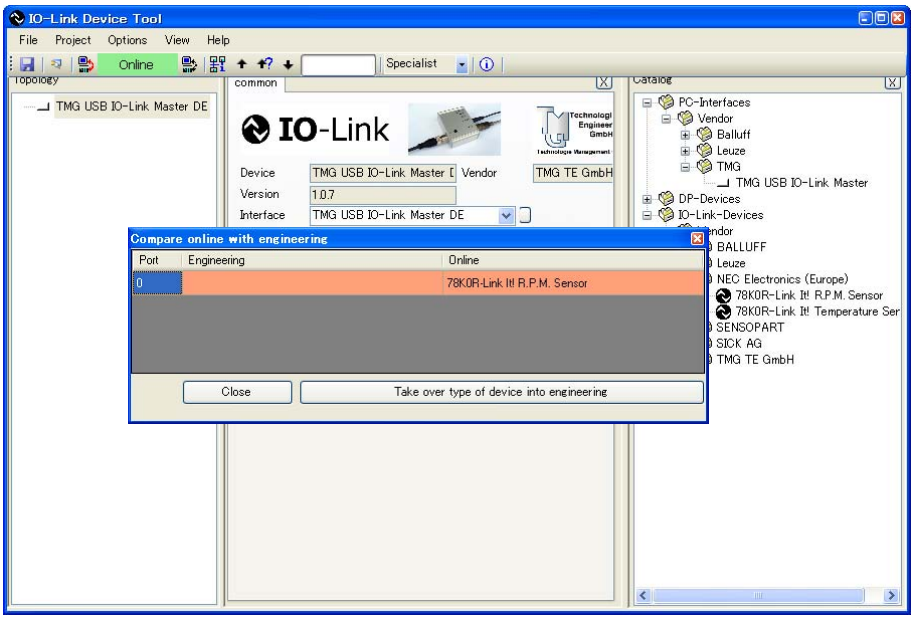


Figure 13-12. Check Config successful

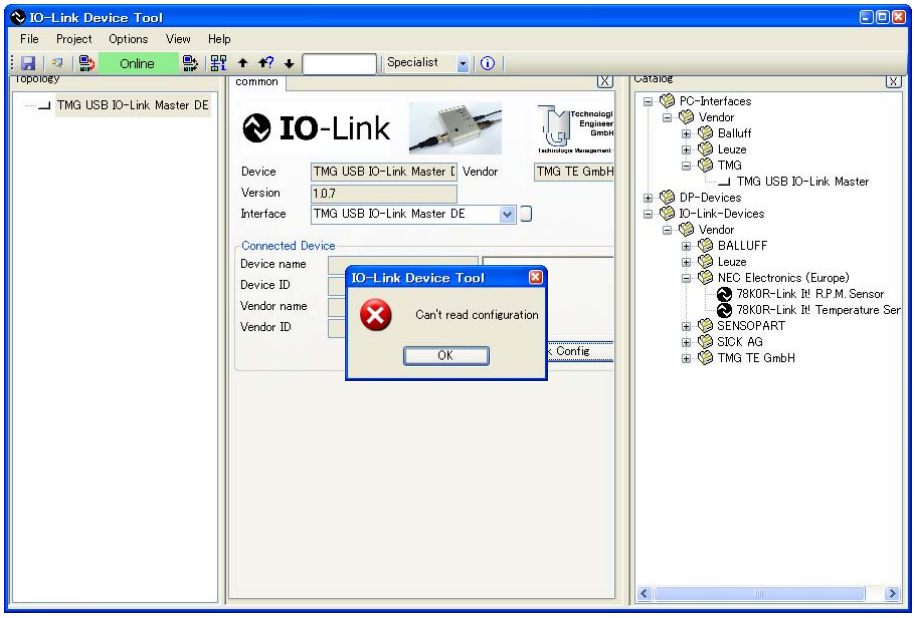


Figure 13-13. Check Config failure

Caution **It can happen sometime that the “can’t read configuration” message appears in the Device Tool when the Master is not properly initialized. Unplugging the master from the USB port of the host machine and re-starting the Device tool solves this problem. Also please check the board settings are properly set for IO-link communication and that the board is properly connected to the IO-Link master.**

(5) Click on the “Take over type of device into engineering” button

The *78K0R–Link It!* R.P.M. Sensor will now appear under the TMG USB IO-Link Master DE in the topology pane, and details on the board can be read in the Common pane

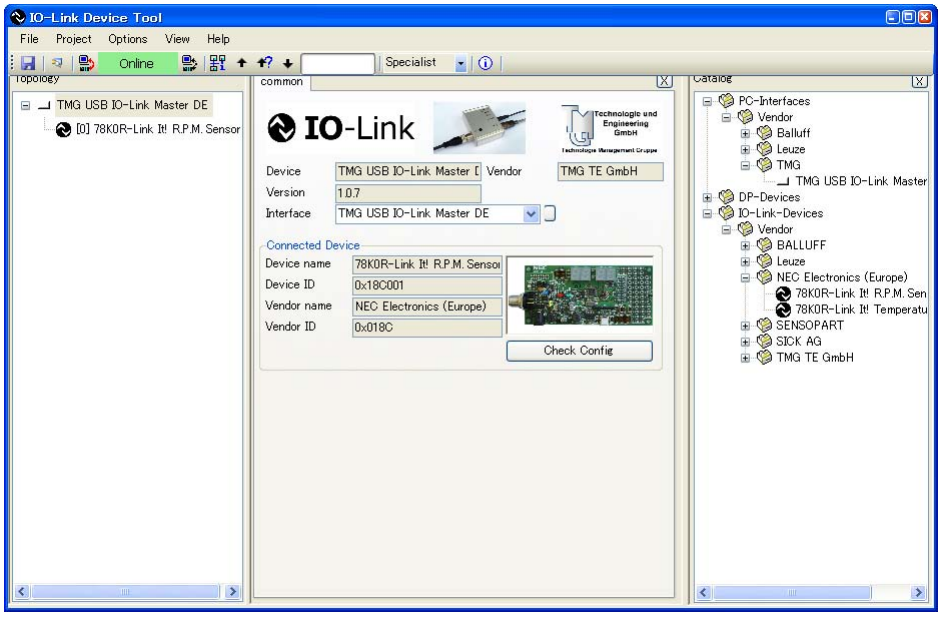


Figure 13-14. IO-Link communication active

The IO-Link communication is now active and we can have a closer look at the device and the sensor application running on the board

The Common pane provides information on the connected device such as:

- Device name: 78K0R-Link It! R.P.M. Sensor
- Device ID: 0x18C000
- Vendor name: NEC Electronics (Europe)
- Vendor ID: 0x018C
- Picture of the Io-Link device connected to the master

13.2.5 *78K0R-Link It!* Board sensor demo in IO-Link device tool

In this section we see how the *78K0R–Link It!* Board works with the IO-Link Master and how the demo's features explained earlier can be used from the IO-Link Device tool GUI.

Left click on the *78K0R–Link It!* R.P.M. Sensor in the topology pane to display generic information on the board in the Common pane. Next to the Common tab, two other tabs can be seen (Process data and Parameter).

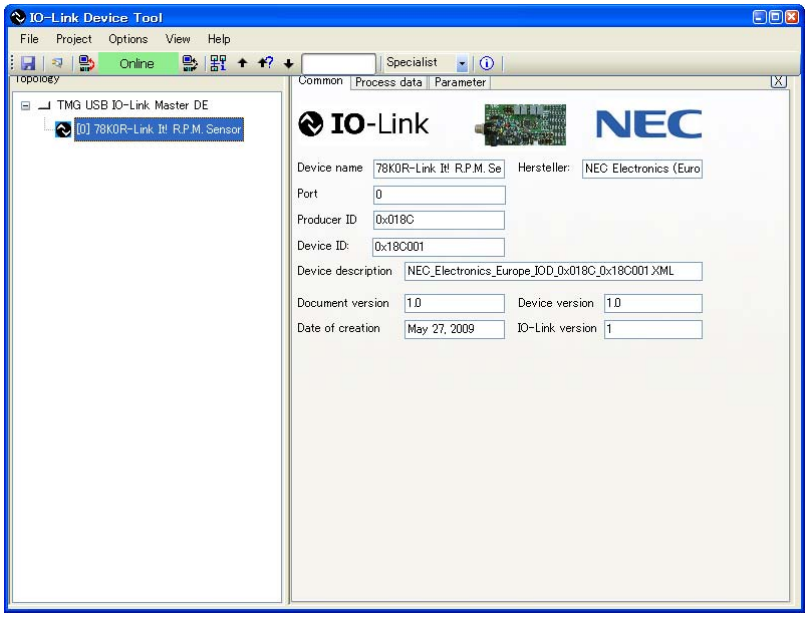


Figure 13-15. *78K0R–Link It!* R.P.M. Sensor *in device tool*

(1) Common tab

The Common pane provides information on the device such as:

- Picture of the device
- Device name, Product ID, Device ID, vendor name
- Port used on the master. Here the kit is connected to port 0.
- Device description file, its version and date of creation.
- Device version
- IO-Link version.

(2) Process data tab

The Process data tab displays the sensor measurements (process data inputs from the sensor).

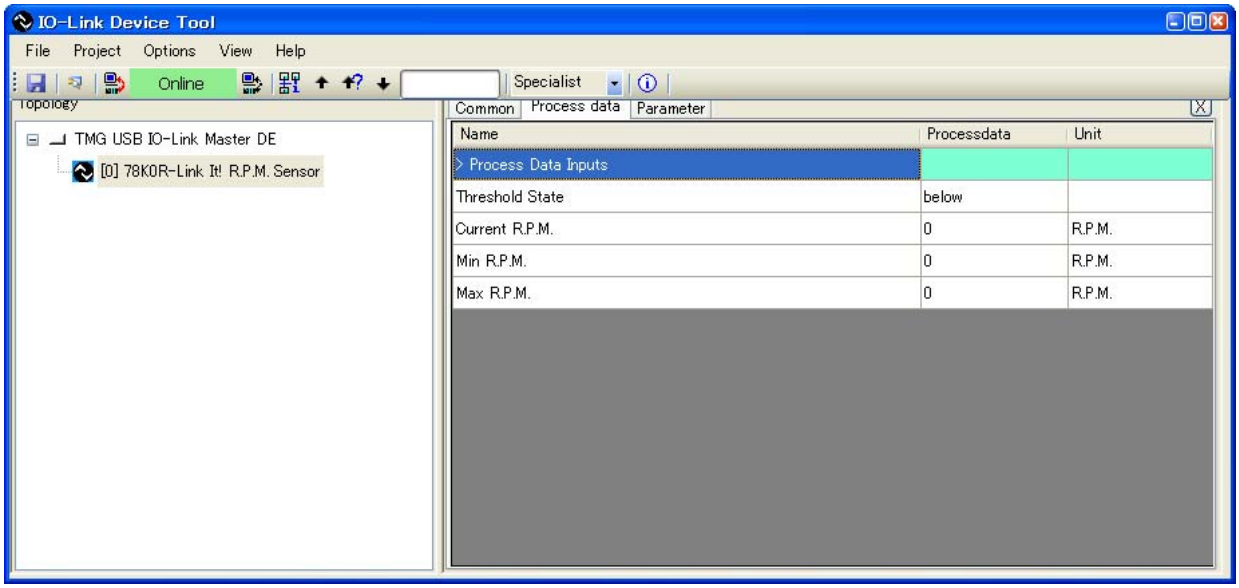


Figure 13-16. R.P.M. Sensor Process data

The user can find information such as the Min/Max RPM, the Current RPM, and the Threshold State (above, between, and below, representing LED1, LED2, and LED3 respectively). (The actual threshold values are displayed on the Parameter tab.) The values measured by the sensor are displayed on the Process data tab. (This is process data input from the sensor.)

(3) Parameter tab

The Parameter tab displays the default settings of the sensor, but also enables the user to teach the sensor by writing new thresholds values to the board.

From the tabs on the parameter page we can find general read only vendor specific parameters such as:

- Vendor and product names
- Hardware, firmware revisions

From the tabs on the parameter page we can find read/write device specific parameters such as:

- Upper/Lower Thresholds

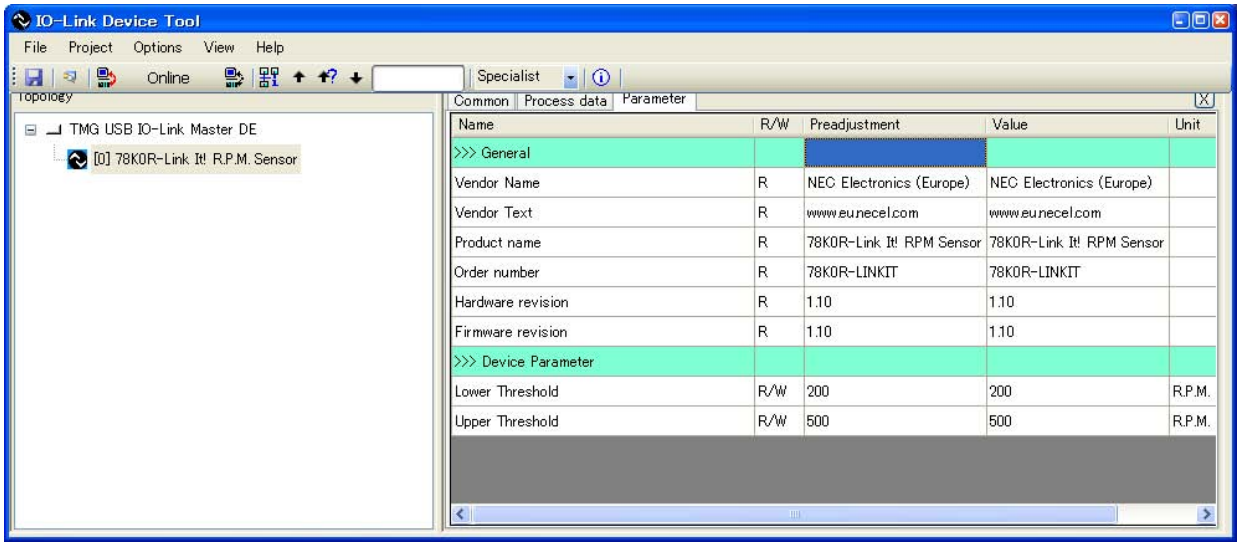


Figure 13-17. R.P.M. Sensor Parameter page

13.2.6 *78K0R-Link It!* RPM Board sensor Teach-in

When the user opens the parameter page for the first time, the device specific parameters are set to their default values. The “Pre-adjustment” and “Value” columns match each other in the parameter page, see **Figure 13-17**.

The “Pre-adjustment” column displays the default sensor settings. The “Value” column displays the current sensor settings

(1) Read parameter from the *78K0R-Link It!* Board sensor

To read or refresh the display of the current parameter, click Left in the cell corresponding to the parameter you want to read from under the “Value” column. The cell is highlighted in blue and the current parameter value is displayed.

Figure 13-18 shows the reading of the lower RPM threshold of 200rpm set on the board.

The same procedure is used to read the Upper threshold set on the board.

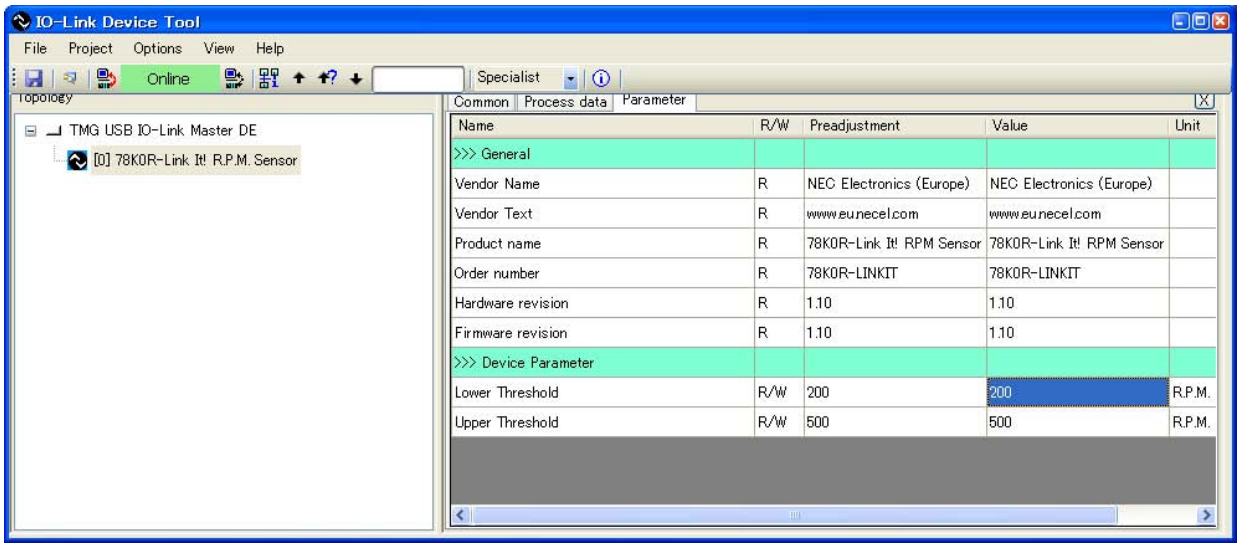


Figure 13-18. Lower Threshold value reading

(2) Write parameter to the *78K0R-Link It!* Board sensor

The teach-in functionality allows changing the valid RPM range on the fly by modifying the Upper/Lower thresholds.

To write or teach a parameter to the board, click Right in the cell corresponding to the parameter you want to change from under the “Value” column. The cell is highlighted in blue and a dialog box will open to enter the new value.

Next, enter the new value in the dialog box as a multiple of 100 times, and then press the Enter key on the keyboard. The new parameter is now passed to the board. You can check with the SW4 joystick that the parameter has been passed correctly to the board.

The Thresholds LEDs will automatically display what the current measured RPM is with regards to the new range.

Caution For this sample software, because the maximum and minimum-speed threshold values are managed as multiples of 100 times, be sure to enter 0 for the tens and ones places. If a number other than 0 is specified, it is discarded. Any multiple of 100 from 0 to 9,900 can be specified.

Figure 13-19 shows the dialog box and the setting of the Lower threshold to 300rpm.

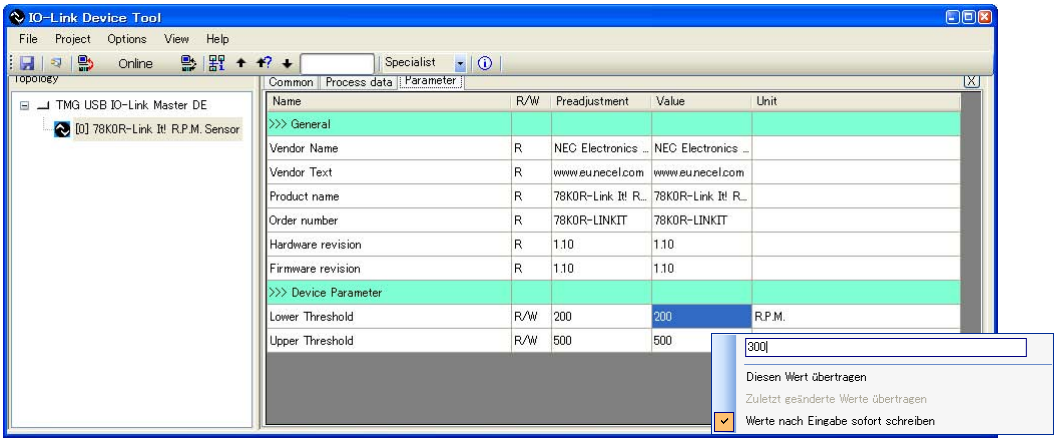


Figure 13-19 *78K0R–Link It!* R.P.M. Sensor Thresholds setting

The maximum-rotation threshold value can be specified in the same way. In addition, specify values such that:

Minimum-rotation threshold value < maximum-rotation threshold value

13.3 Adjusting Number of Pulses Per Rotation

For this sample application, it is assumed that three pulses are input per rotation by default.

The number of pulses is generally determined by how many magnets are attached to the object to be rotated, but the number of pulses (the number of magnets) can be changed by changing the value of the PULSE_PER_ROUND constant defined in source code "sensor.c".

The default is as follows:
```
#define    PULSE_PER_ROUND (3)      /* pulse output per 1 round */
```

As an example, to change the number of pulses per rotation to eight (when there are eight magnets), change the above definition as follows:

```
#define    PULSE_PER_ROUND (8)      /* pulse output per 1 round */
```

Next, rebuild the project, write the modified HEX file to the board, and then begin operation.

13.4 Flowcharts for IO-Link RPM Detection Sensor Sample Program

This section provides the flowcharts for this sample program. This program is made up of the functions below. (Below, items that are divided according to these functions are called *modules*.)

(1) Main reset

(2) Sensor control

(3) Display

(4) Display (RPM sensor)

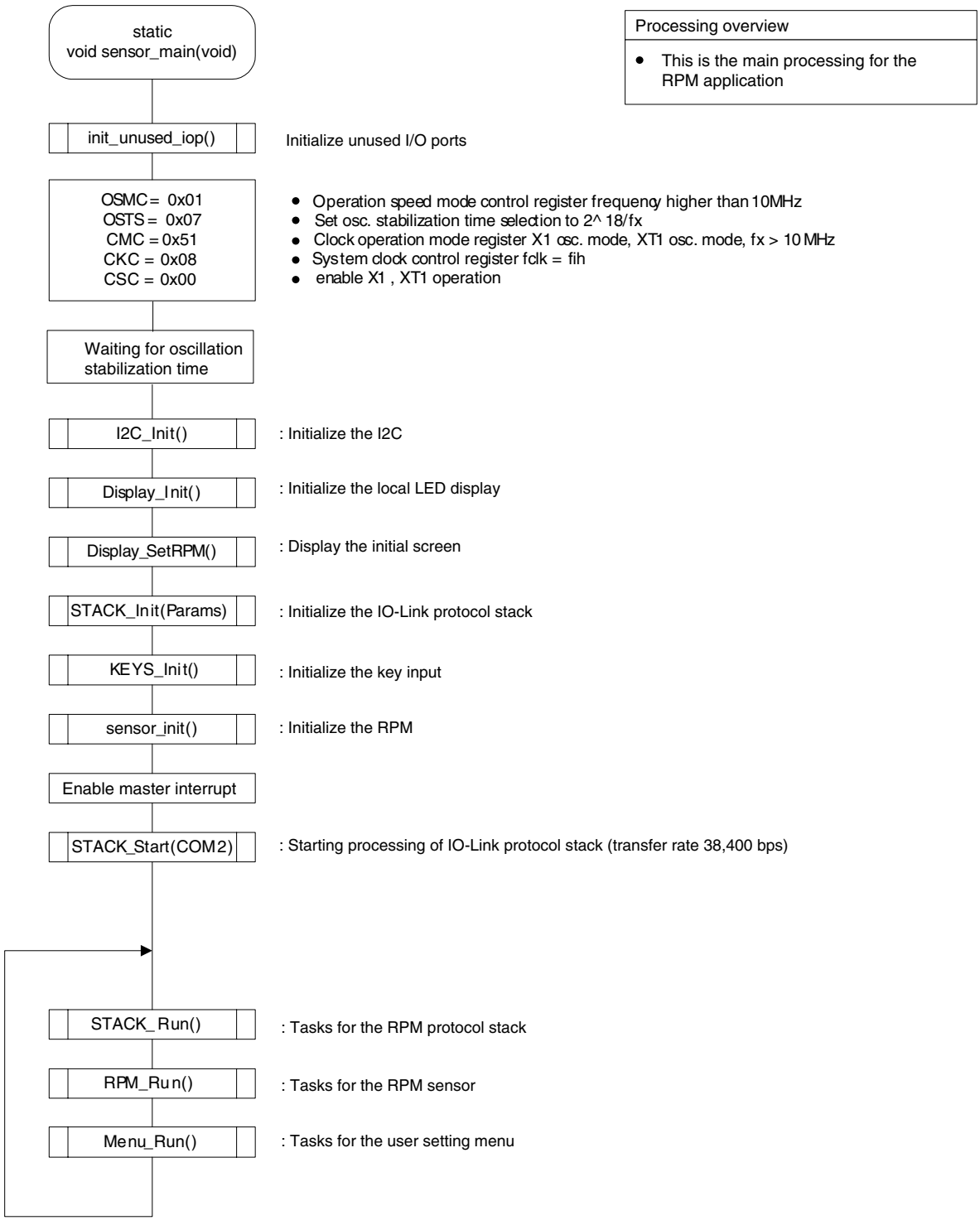
(5) Application layer for the IO-Link protocol stack

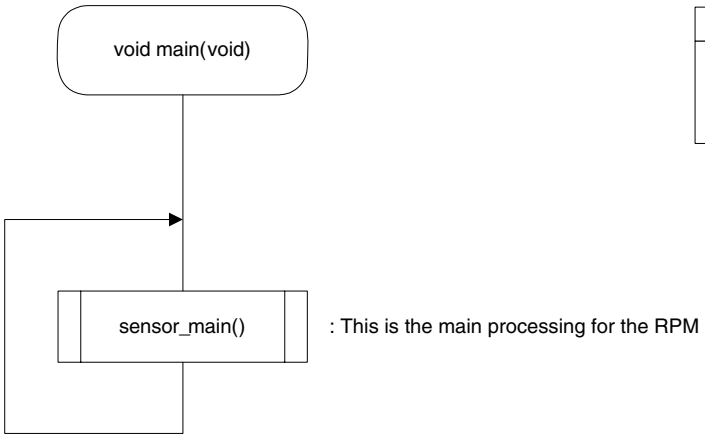
(6) I^2C driver

(7) Key input

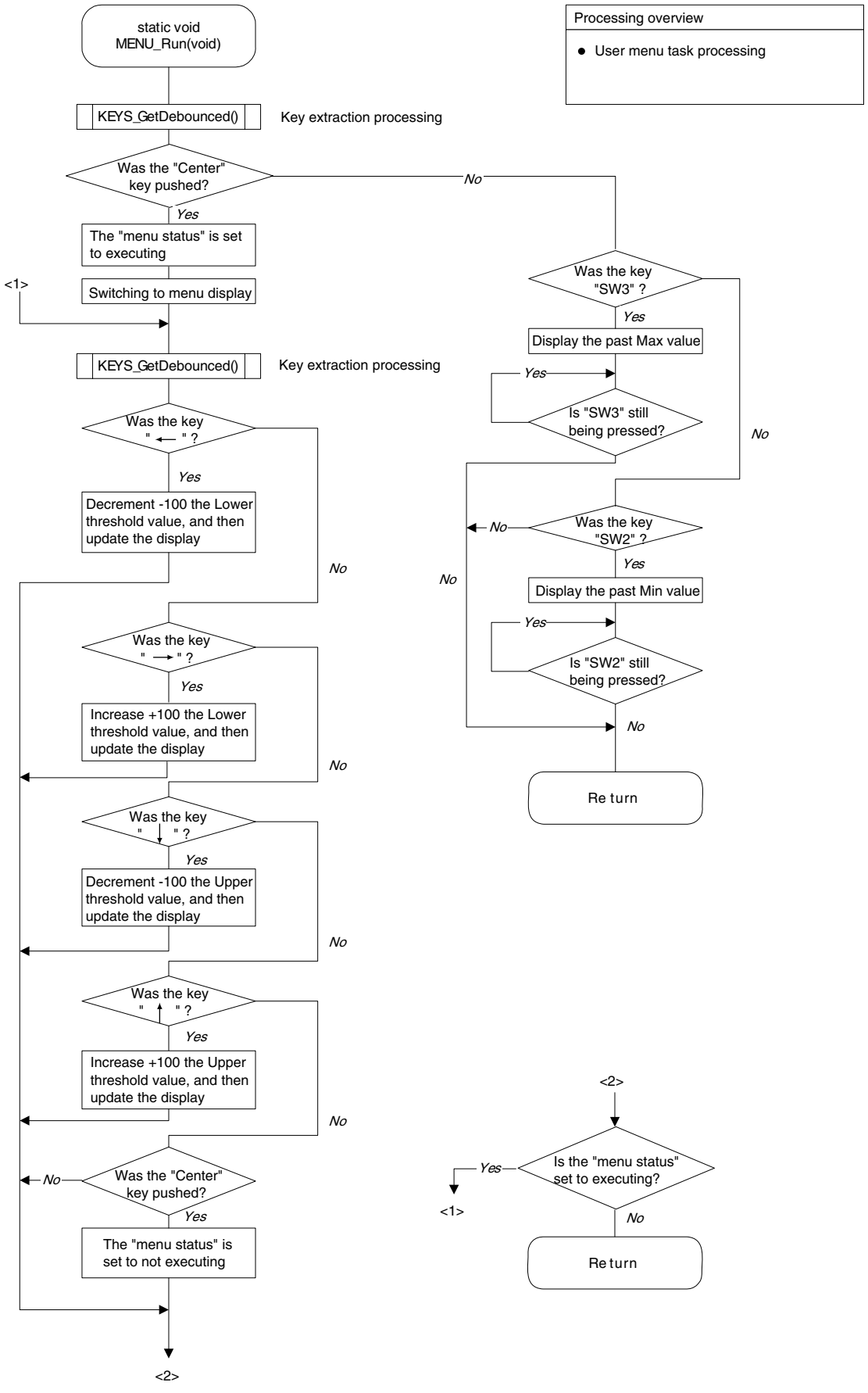
Flowcharts of the functions included in the above modules are provided below.

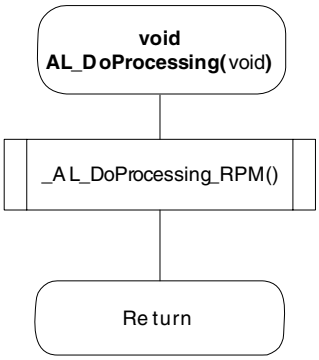
13.4.1 Main reset module flowchart



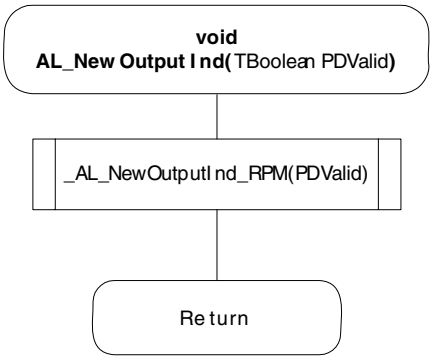


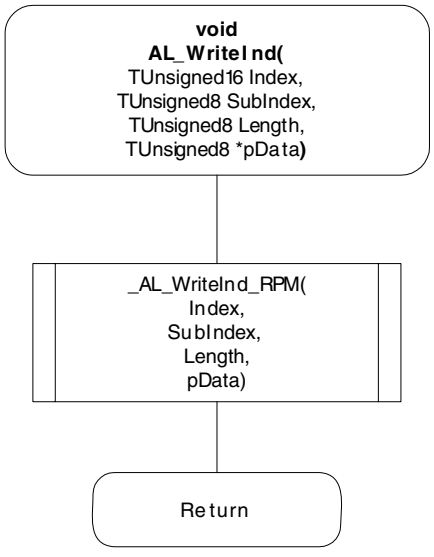
Processing overview
• Main processing

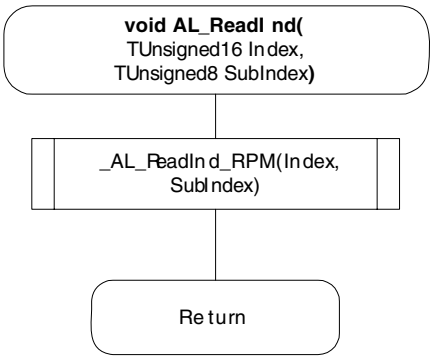

static void
MENU_Run(void)
Processing overview
● User menu task processing
KEYS_GetDebounced()
Key extraction processing
Was the "Center" key pushed?
No
Yes
The "menu status" is set to executing
<1>
Switching to menu display
KEYS_GetDebounced()
Key extraction processing
Was the key " ← " ?
Yes
Decrement -100 the Lower threshold value, and then update the display
No
Was the key " → " ?
Yes
Increase +100 the Lower threshold value, and then update the display
No
Was the key " ↓ " ?
Yes
Decrement -100 the Upper threshold value, and then update the display
No
Was the key " ↑ " ?
Yes
Increase +100 the Upper threshold value, and then update the display
No
No
Was the "Center" key pushed?
Yes
The "menu status" is set to not executing
<2>
Was the key "SW3" ?
Yes
Display the past Max value
Yes
Is "SW3" still being pressed?
No
No
Was the key "SW2" ?
Yes
No
Display the past Min value
Yes
Is "SW2" still being pressed?
No
Re turn
<2>
Yes
Is the "menu status" set to executing?
<1>
No
Re turn



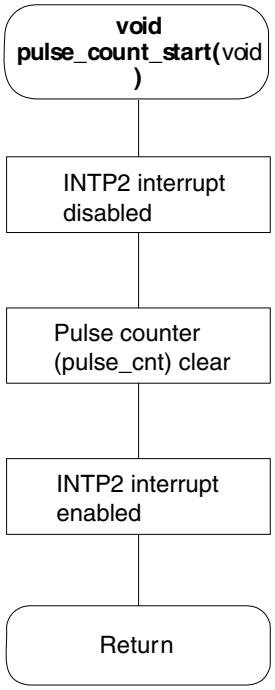
Processing overview
• Application layer function • This is a callback routine for the protocol stack. * Each function that starts with "_AL_" is in the "al_rpm" module.





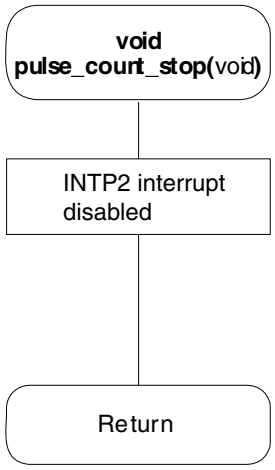


13.4.2 Sensor control module flowchart

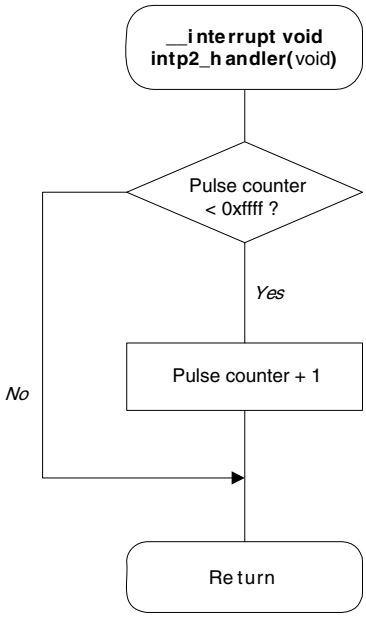


: Pulse count start processing

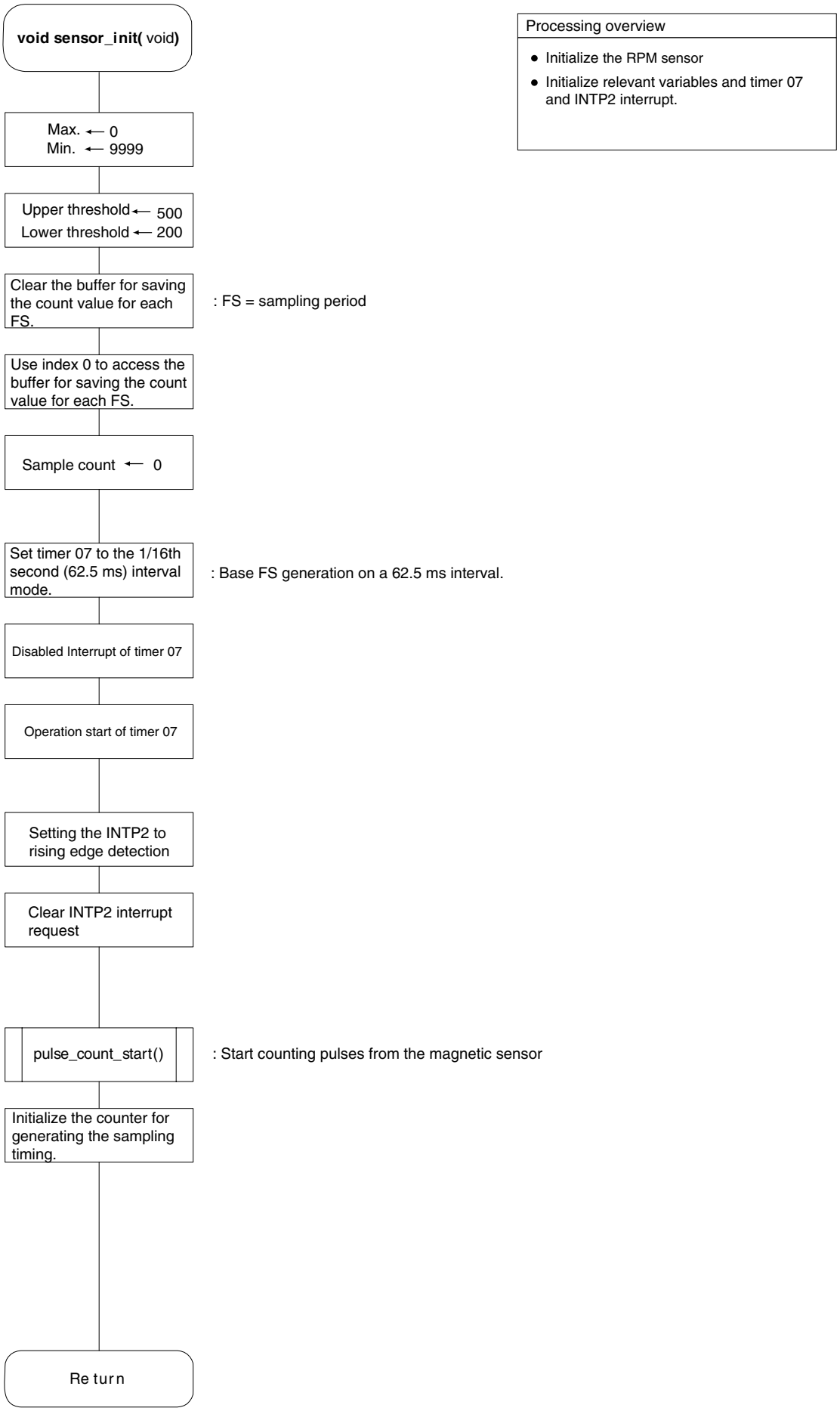
Processing overview
• Start or stop counting pulses from the magnetic sensor.

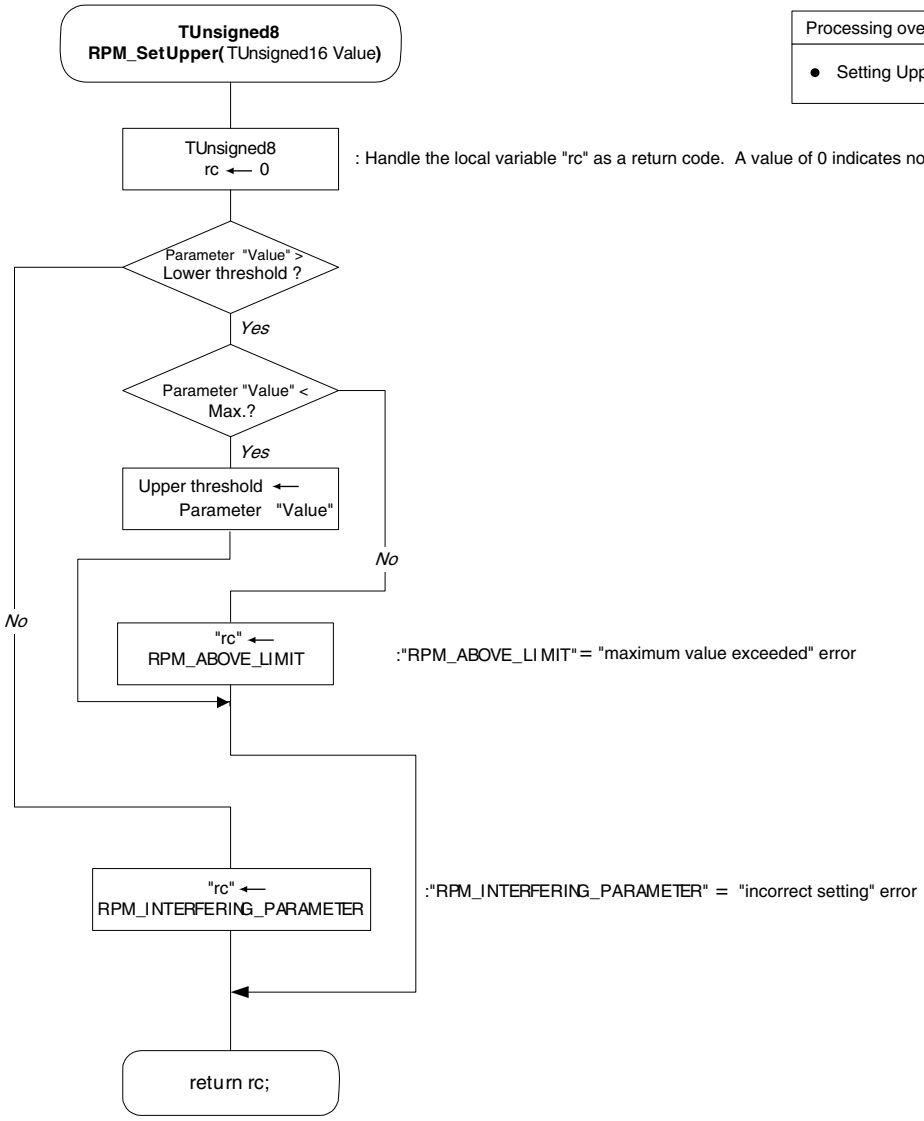


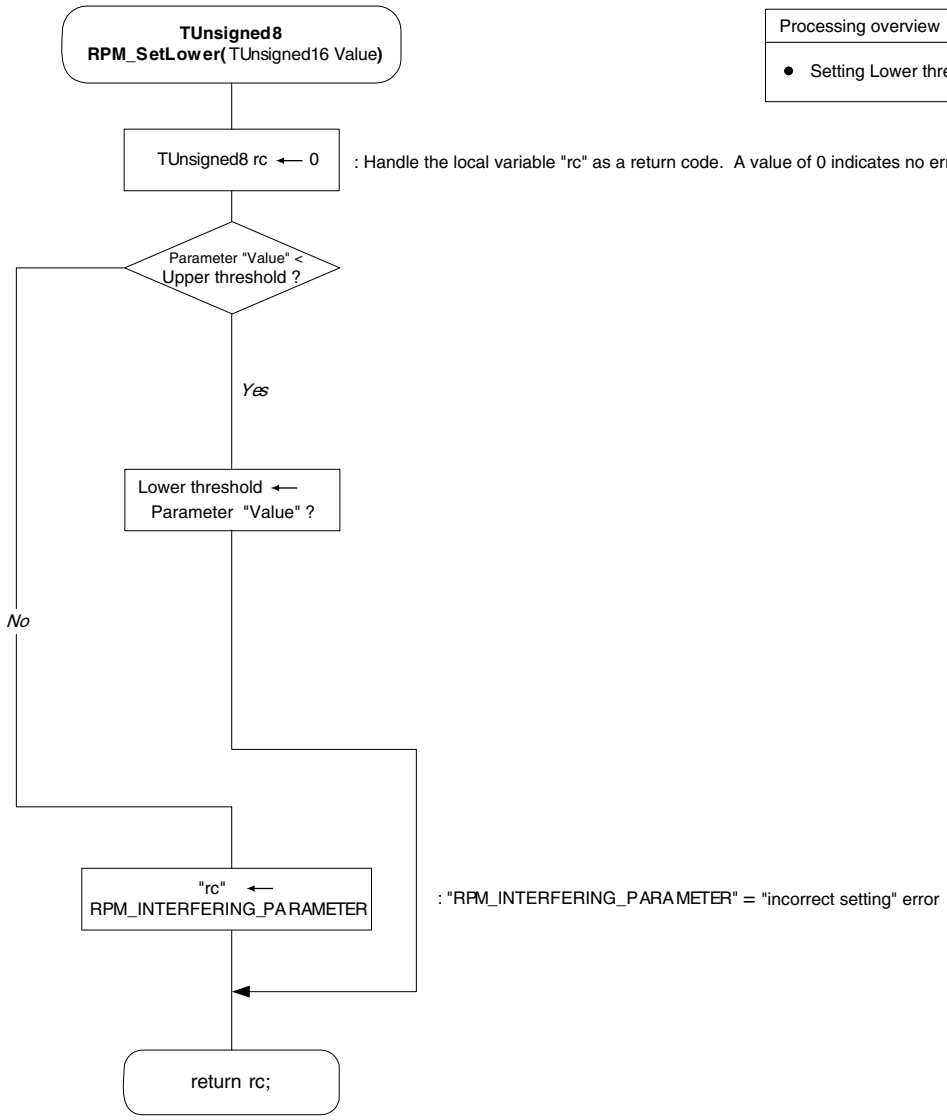
: Pulse count stop processing

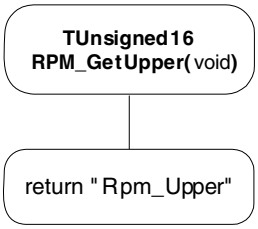


Processing overview
• This is the INTP2 interrupt handler. • Specify the use of RB3. • Only perform processing at the rising edge of INTP2.


void sensor_init(void)
Processing overview
Initialize the RPM sensor
Initialize relevant variables and timer 07 and INTP2 interrupt.
Max. ← 0
Min. ← 9999
Upper threshold ← 500
Lower threshold ← 200
Clear the buffer for saving the count value for each FS.
: FS = sampling period
Use index 0 to access the buffer for saving the count value for each FS.
Sample count ← 0
Set timer 07 to the 1/16th second (62.5 ms) interval mode.
: Base FS generation on a 62.5 ms interval.
Disabled Interrupt of timer 07
Operation start of timer 07
Setting the INTP2 to rising edge detection
Clear INTP2 interrupt request
pulse_count_start()
: Start counting pulses from the magnetic sensor
Initialize the counter for generating the sampling timing.
Return

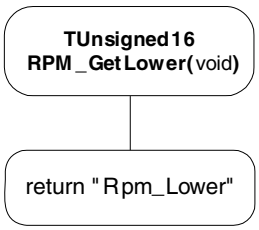

TUnsigned8
RPM_SetUpper(TUnsigned16 Value)
Processing overview
Setting Upper threshold
TUnsigned8
rc ← 0
: Handle the local variable "rc" as a return code. A value of 0 indicates no error.
Parameter "Value" >
Lower threshold ?
Yes
Parameter "Value" <
Max.?
Yes
Upper threshold ←
Parameter "Value"
No
No
"rc" ←
RPM_ABOVE_LIMIT
:"RPM_ABOVE_LIMIT" = "maximum value exceeded" error
"rc" ←
RPM_INTERFERING_PARAMETER
:"RPM_INTERFERING_PARAMETER" = "incorrect setting" error
return rc;


TUnsigned8
RPM_SetLower(TUnsigned16 Value)
Processing overview
Setting Lower threshold
TUnsigned8 rc ← 0
: Handle the local variable "rc" as a return code. A value of 0 indicates no error.
Parameter "Value" <
Upper threshold ?
Yes
No
Lower threshold ←
Parameter "Value" ?
"rc" ←
RPM_INTERFERING_PARAMETER
: "RPM_INTERFERING_PARAMETER" = "incorrect setting" error
return rc;

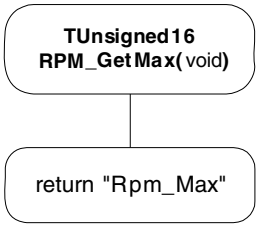


: Return the value of the Upper threshold variable.

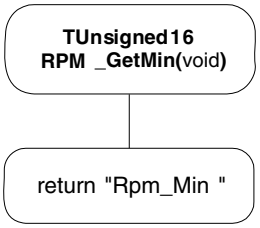
Processing overview
• Acquire the Upper threshold value • Acquire the Lower threshold value • Acquire the maximum-rotation value • Acquire the minimum-rotation value



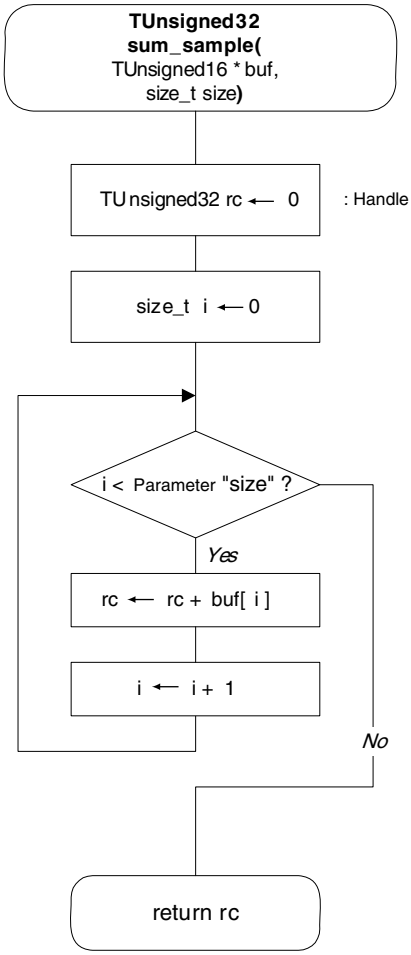
: Return the value of the Lower threshold variable.



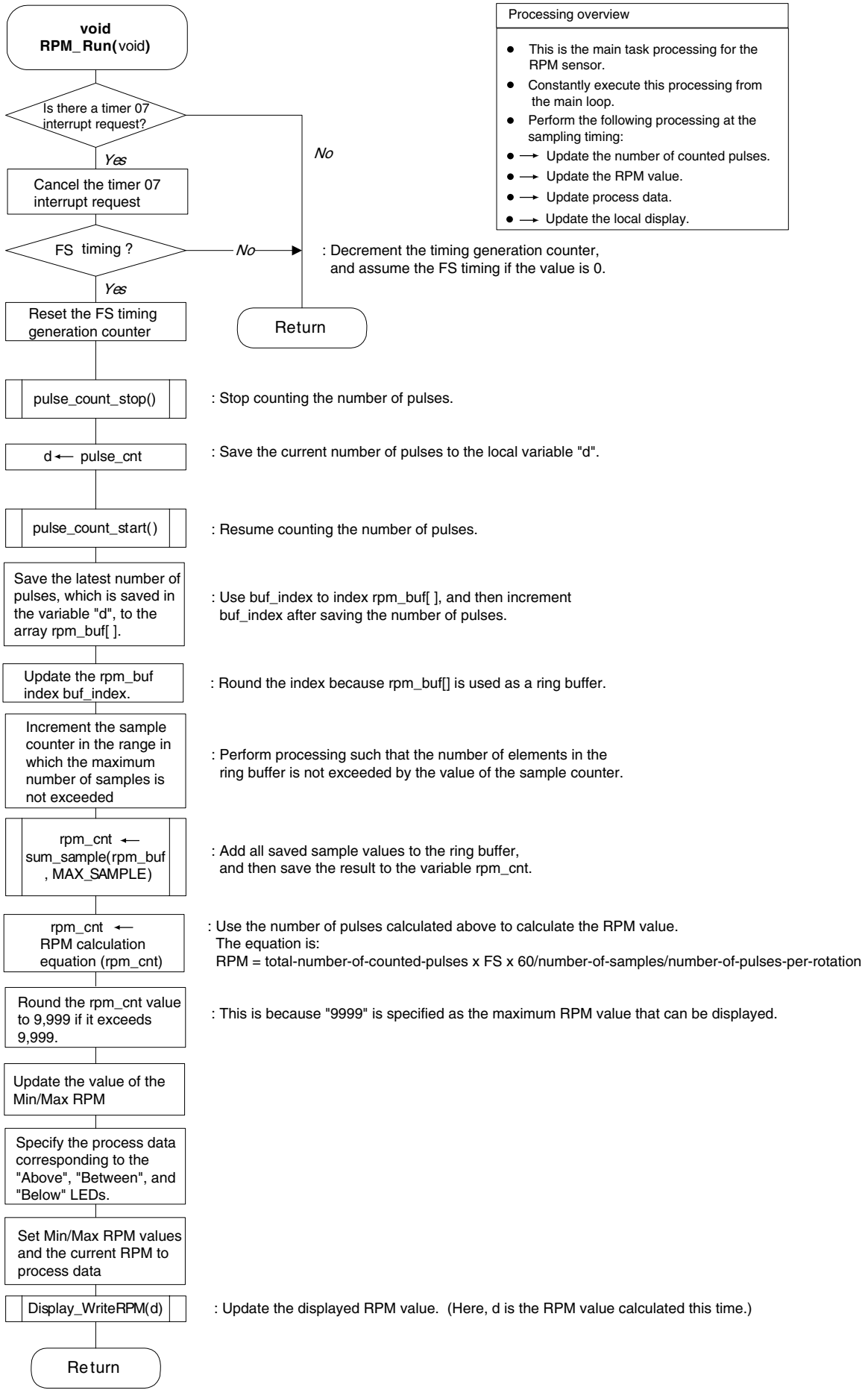
:Return the value of the maximum RPM variable.



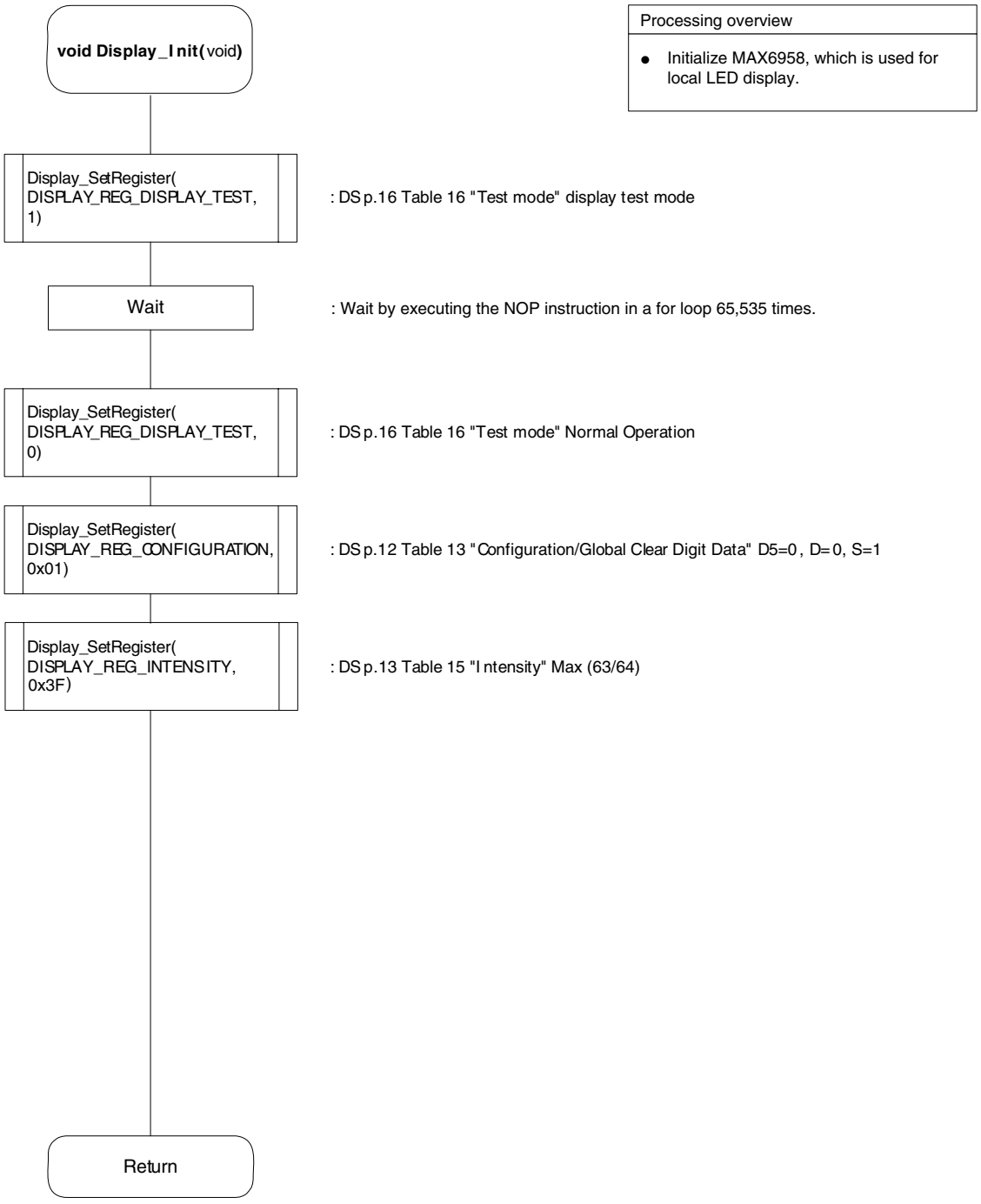
:Return the value of the minimum RPM variable.

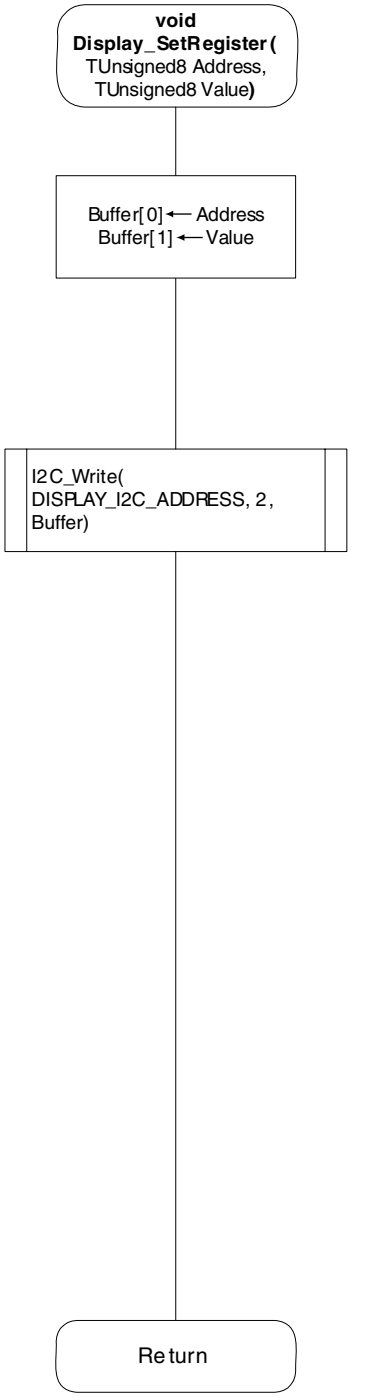


Processing overview
● Add the number of values indicated by the "size" parameter to the array elements indicated by the "buf" parameter, and then return the result.


void
RPM_Run(void)
Is there a timer 07 interrupt request?
No
Yes
Cancel the timer 07 interrupt request
FS timing ?
No
: Decrement the timing generation counter, and assume the FS timing if the value is 0.
Yes
Reset the FS timing generation counter
Return
pulse_count_stop()
: Stop counting the number of pulses.
d ← pulse_cnt
: Save the current number of pulses to the local variable "d".
pulse_count_start()
: Resume counting the number of pulses.
Save the latest number of pulses, which is saved in the variable "d", to the array rpm_buf[ ].
: Use buf_index to index rpm_buf[ ], and then increment buf_index after saving the number of pulses.
Update the rpm_buf index buf_index.
: Round the index because rpm_buf[] is used as a ring buffer.
Increment the sample counter in the range in which the maximum number of samples is not exceeded
: Perform processing such that the number of elements in the ring buffer is not exceeded by the value of the sample counter.
rpm_cnt ← sum_sample(rpm_buf , MAX_SAMPLE)
: Add all saved sample values to the ring buffer, and then save the result to the variable rpm_cnt.
rpm_cnt ← RPM calculation equation (rpm_cnt)
: Use the number of pulses calculated above to calculate the RPM value.
The equation is:
RPM = total-number-of-counted-pulses x FS x 60/number-of-samples/number-of-pulses-per-rotation
Round the rpm_cnt value to 9,999 if it exceeds 9,999.
: This is because "9999" is specified as the maximum RPM value that can be displayed.
Update the value of the Min/Max RPM
Specify the process data corresponding to the "Above", "Between", and "Below" LEDs.
Set Min/Max RPM values and the current RPM to process data
Display_WriteRPM(d)
: Update the displayed RPM value. (Here, d is the RPM value calculated this time.)
Return
Processing overview
This is the main task processing for the RPM sensor.
Constantly execute this processing from the main loop.
Perform the following processing at the sampling timing:
→ Update the number of counted pulses.
→ Update the RPM value.
→ Update process data.
→ Update the local display.

13.4.3 Display control module flowchart



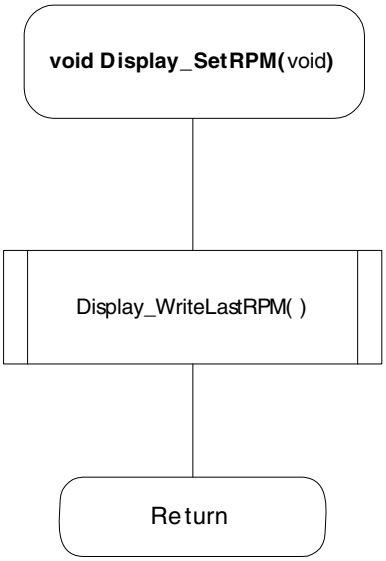


Processing overview
• Specify register settings for MAX6958, which is used for local LED display, by way of the I2C bus.

: "Buffer" = "TUnsigned8 Buffer[6];" (in Display.c)

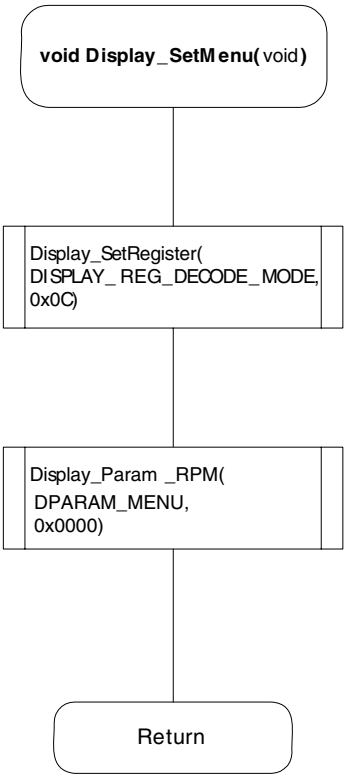
: "DISPLAY_I2C_ADDRESS" is I2C address of MAX6958,
"2" is the data length.

13.4.4 Display (RPM sensor) module flowchart



Processing overview
• Display the RPM as a four-digit decimal value for local LEDs DIG0 to DIG3.

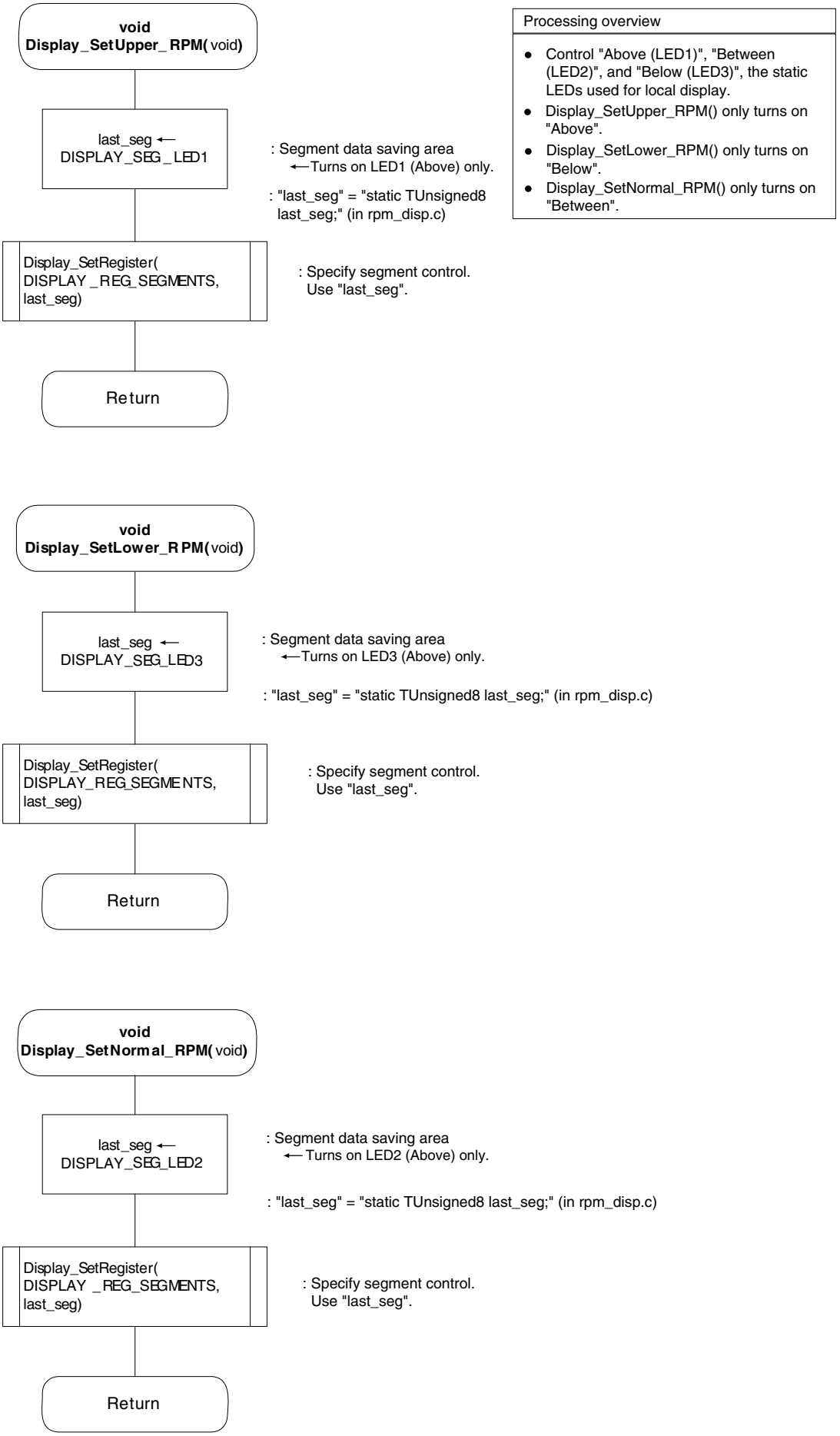
: Display the RPM by using the saved RPM value ("LastValue").

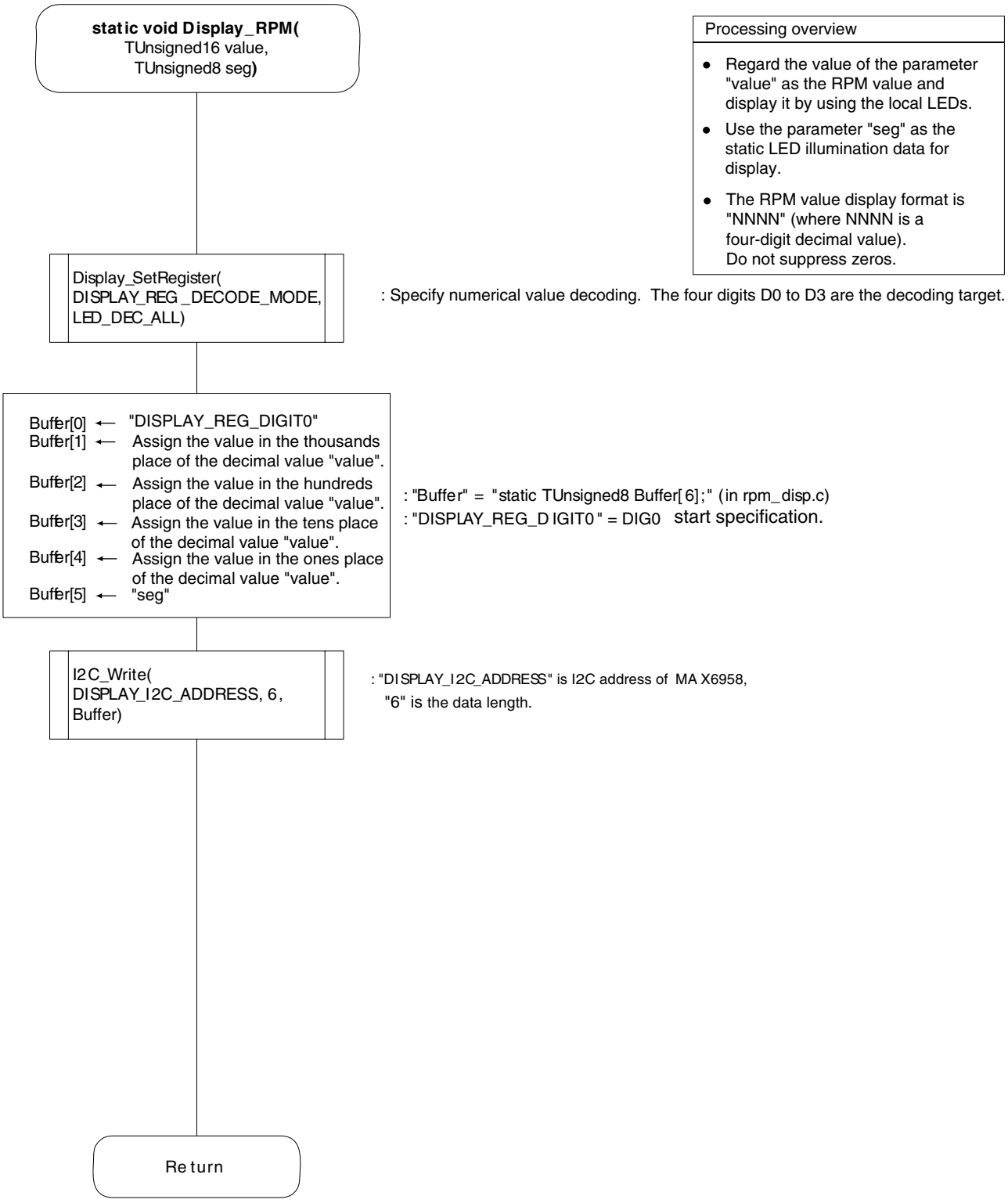


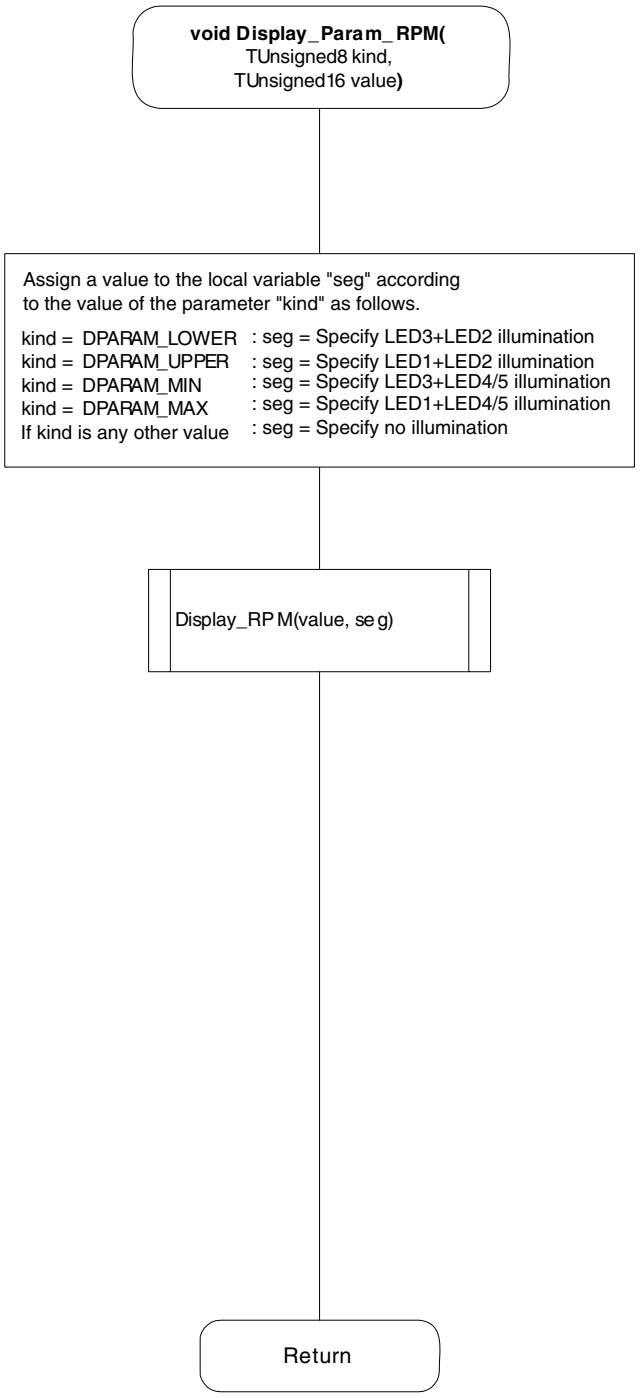
Processing overview
• Prepare the menu display for local display. • When this function is executed, DIG0 and DIG1 are set to display a fixed segment pattern, and DIG2 and DIG3 are set to display a fixed two-digit numerical value.

: Specify the decoding mode. Decode DIG2 and DIG3 as a hexadecimal value.

: Specify no segment display for "DPARAM_MENU". "0x0000" is a temporary RPM value.

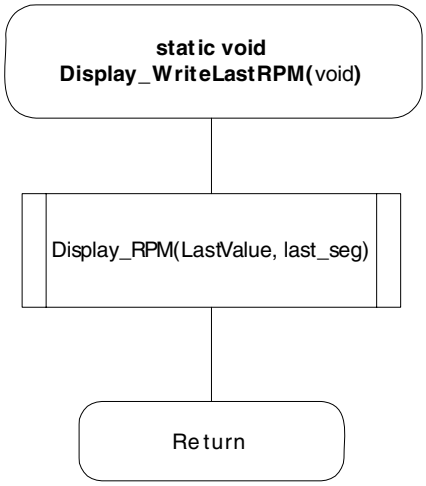

void
Display_SetUpper_RPM(void)
last_seg ← DISPLAY_SEG_LED1
: Segment data saving area
←Turns on LED1 (Above) only.
: "last_seg" = "static TUnsigned8 last_seg;" (in rpm_disp.c)
Display_SetRegister(DISPLAY_REG_SEGMENTS, last_seg)
: Specify segment control. Use "last_seg".
Return
Processing overview
Control "Above (LED1)", "Between (LED2)", and "Below (LED3)", the static LEDs used for local display.
Display_SetUpper_RPM() only turns on "Above".
Display_SetLower_RPM() only turns on "Below".
Display_SetNormal_RPM() only turns on "Between".
void
Display_SetLower_RPM(void)
last_seg ← DISPLAY_SEG_LED3
: Segment data saving area
←Turns on LED3 (Above) only.
: "last_seg" = "static TUnsigned8 last_seg;" (in rpm_disp.c)
Display_SetRegister(DISPLAY_REG_SEGMENTS, last_seg)
: Specify segment control. Use "last_seg".
Return
void
Display_SetNormal_RPM(void)
last_seg ← DISPLAY_SEG_LED2
: Segment data saving area
←Turns on LED2 (Above) only.
: "last_seg" = "static TUnsigned8 last_seg;" (in rpm_disp.c)
Display_SetRegister(DISPLAY_REG_SEGMENTS, last_seg)
: Specify segment control. Use "last_seg".
Return


static void Display_RPM(
TUnsigned16 value,
TUnsigned8 seg)
Processing overview
• Regard the value of the parameter "value" as the RPM value and display it by using the local LEDs.
• Use the parameter "seg" as the static LED illumination data for display.
• The RPM value display format is "NNNN" (where NNNN is a four-digit decimal value). Do not suppress zeros.
Display_SetRegister(
DISPLAY_REG_DECODE_MODE,
LED_DEC_ALL)
: Specify numerical value decoding. The four digits D0 to D3 are the decoding target.
Buffer[0] ← "DISPLAY_REG_DIGIT0"
Buffer[1] ← Assign the value in the thousands place of the decimal value "value".
Buffer[2] ← Assign the value in the hundreds place of the decimal value "value".
Buffer[3] ← Assign the value in the tens place of the decimal value "value".
Buffer[4] ← Assign the value in the ones place of the decimal value "value".
Buffer[5] ← "seg"
: "Buffer" = "static TUnsigned8 Buffer[6];" (in rpm_disp.c)
: "DISPLAY_REG_DIGIT0" = DIG0 start specification.
I2C_Write(
DISPLAY_I2C_ADDRESS, 6,
Buffer)
: "DISPLAY_I2C_ADDRESS" is I2C address of MAX6958,
"6" is the data length.
Return

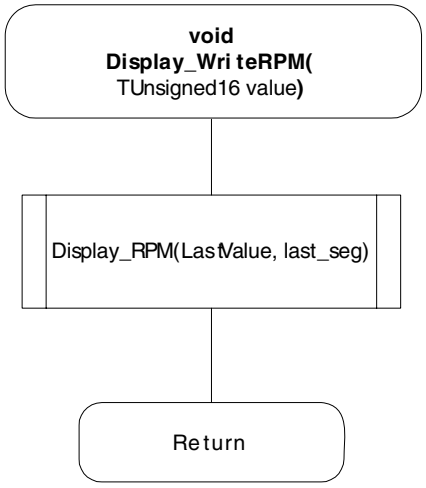


Processing overview

- Regard the value of the parameter "value" as the RPM value and display it by using the local LEDs.
- Use the parameter "kind" as the static LED illumination data for display.
- The RPM value display format is "NNNN" (where NNNN is a four-digit decimal value).
  Do not suppress zeros.

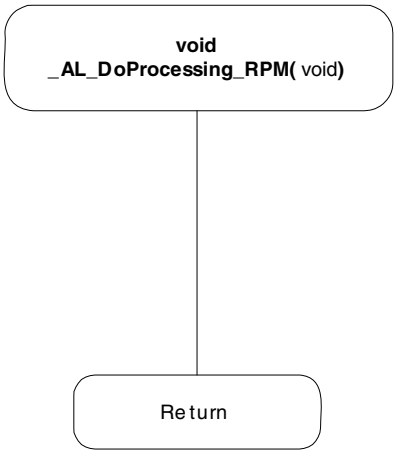


Processing overview
• Use the values saved in the static variables "LastValue" and "lase_seg" to perform display processing for the RPM value and static LEDs.

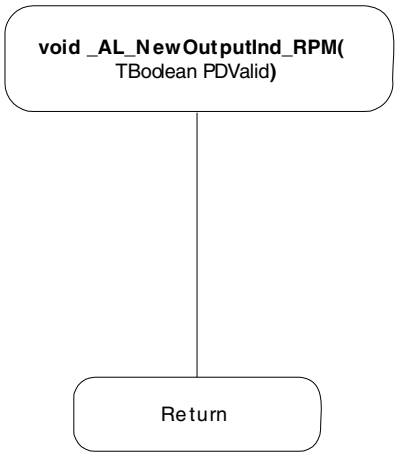


Processing overview
• Use the values saved in the parameter "Value" and static variables "lase_seg"to perform display processing for the RPM value and static LEDs.

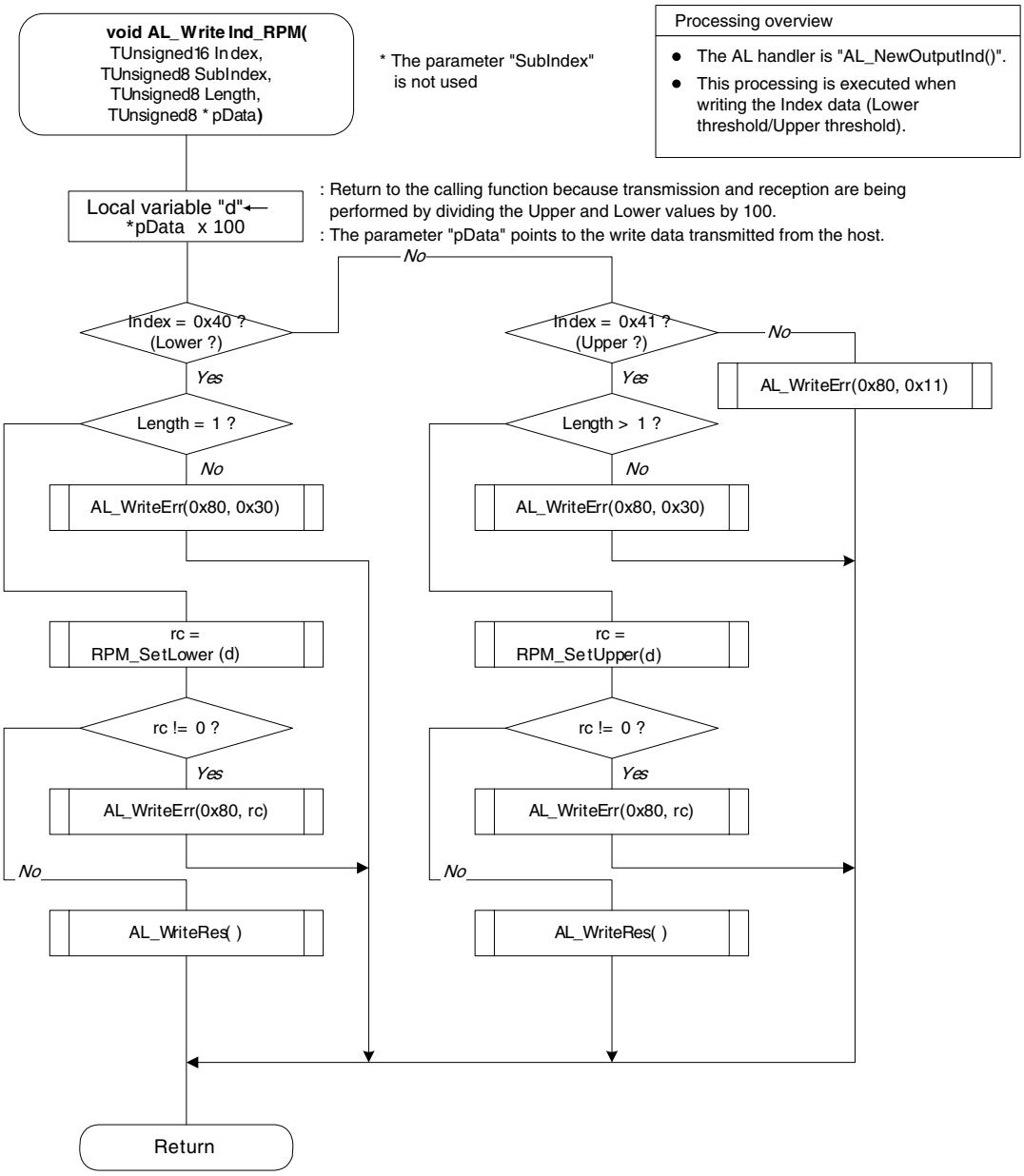
13.4.5 IO-Link protocol stack application layer module flowchart

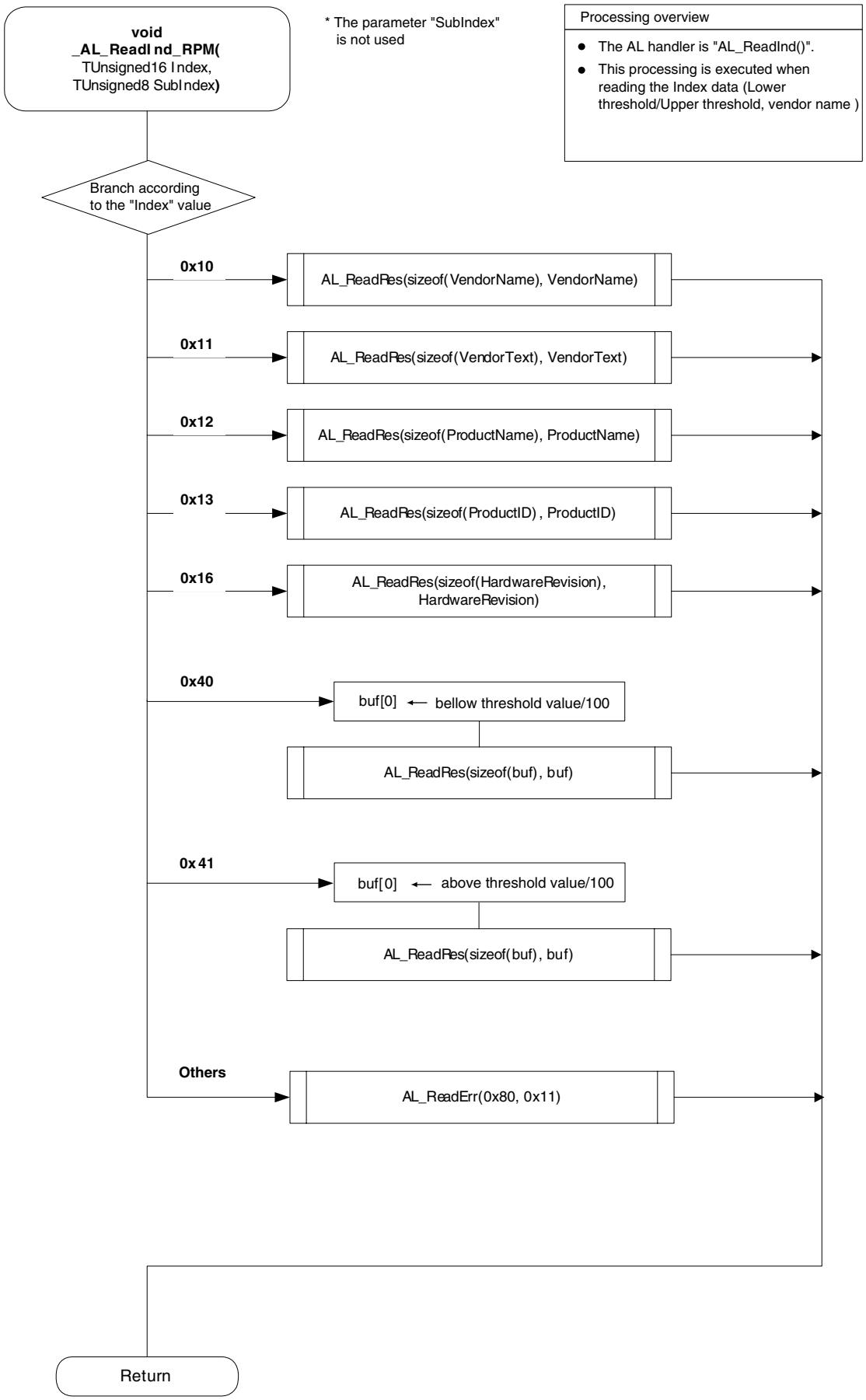


Processing overview
• The AL handler is "AL_DoProcessing()".

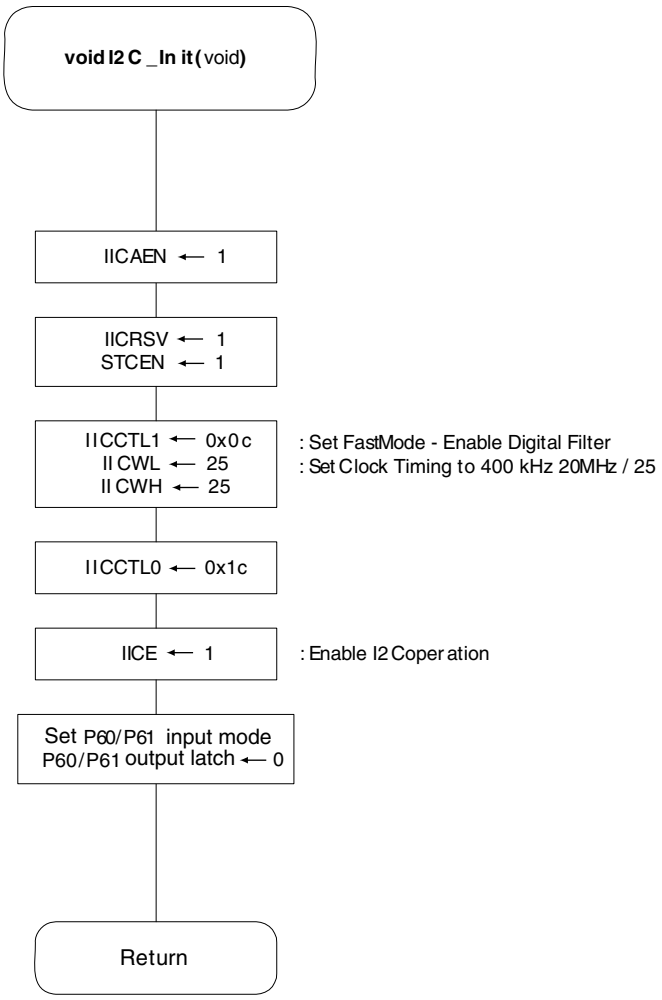


Processing overview
• The AL handler is "AL_NewOutputInd()".

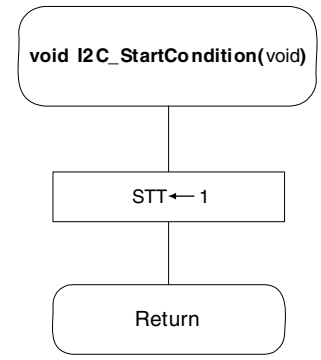

void AL_WriteInd_RPM(
TUnsigned16 Index,
TUnsigned8 SubIndex,
TUnsigned8 Length,
TUnsigned8 * pData)
* The parameter "SubIndex" is not used
Processing overview
The AL handler is "AL_NewOutputInd()".
This processing is executed when writing the Index data (Lower threshold/Upper threshold).
Local variable "d" ← *pData x 100
: Return to the calling function because transmission and reception are being performed by dividing the Upper and Lower values by 100.
: The parameter "pData" points to the write data transmitted from the host.
No
Index = 0x40 ? (Lower ?)
Index = 0x41 ? (Upper ?)
No
Yes
AL_WriteErr(0x80, 0x11)
Yes
Length = 1 ?
Length > 1 ?
No
No
AL_WriteErr(0x80, 0x30)
AL_WriteErr(0x80, 0x30)
rc = RPM_SetLower (d)
rc = RPM_SetUpper(d)
rc != 0 ?
rc != 0 ?
Yes
Yes
AL_WriteErr(0x80, rc)
AL_WriteErr(0x80, rc)
No
No
AL_WriteRes( )
AL_WriteRes( )
Return


void
_AL_ReadInd_RPM(
TUnsigned16 Index,
TUnsigned8 SubIndex)
* The parameter "SubIndex" is not used
Processing overview
The AL handler is "AL_ReadInd()".
This processing is executed when reading the Index data (Lower threshold/Upper threshold, vendor name)
Branch according to the "Index" value
0x10
AL_ReadRes(sizeof(VendorName), VendorName)
0x11
AL_ReadRes(sizeof(VendorText), VendorText)
0x12
AL_ReadRes(sizeof(ProductName), ProductName)
0x13
AL_ReadRes(sizeof(ProductID), ProductID)
0x16
AL_ReadRes(sizeof(HardwareRevision), HardwareRevision)
0x40
buf[0] ← bellow threshold value/100
AL_ReadRes(sizeof(buf), buf)
0x41
buf[0] ← above threshold value/100
AL_ReadRes(sizeof(buf), buf)
Others
AL_ReadErr(0x80, 0x11)
Return

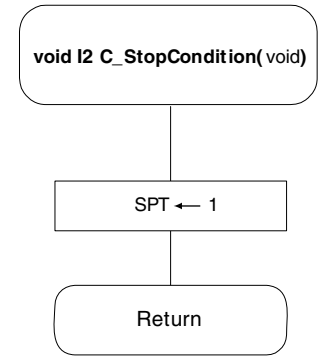
13.4.6 I²C driver module flowchart



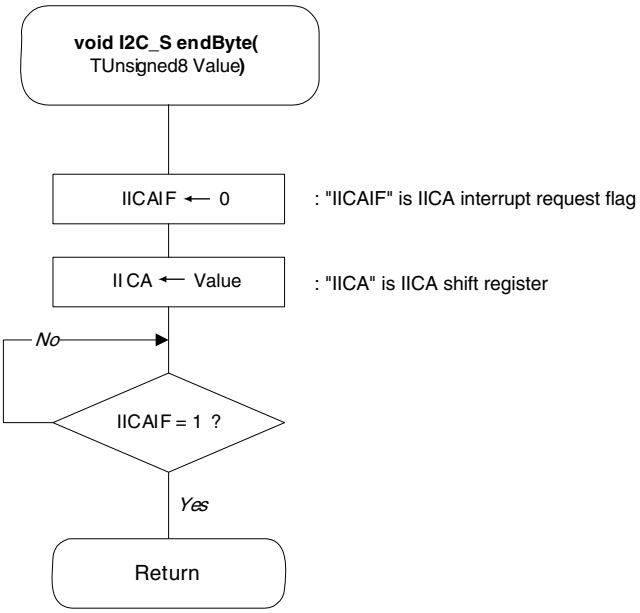
Processing overview
• Initialize the hardware (the "IICA" macro) used for I2C communication.



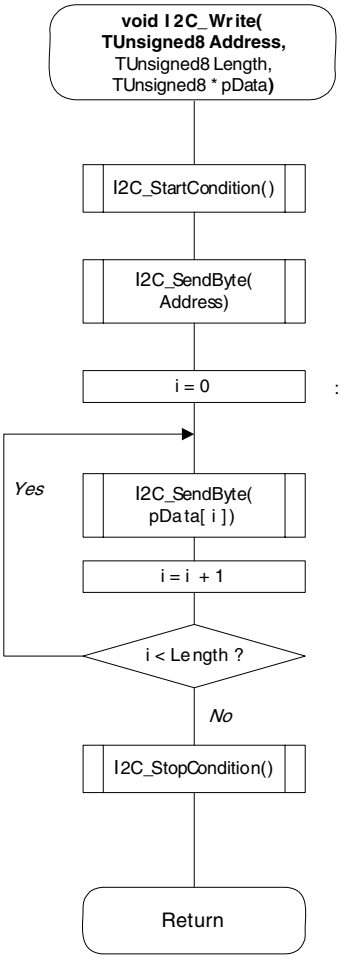
Processing overview
• Issue the I2C start condition.



Processing overview
• Issue the I2C stop condition.



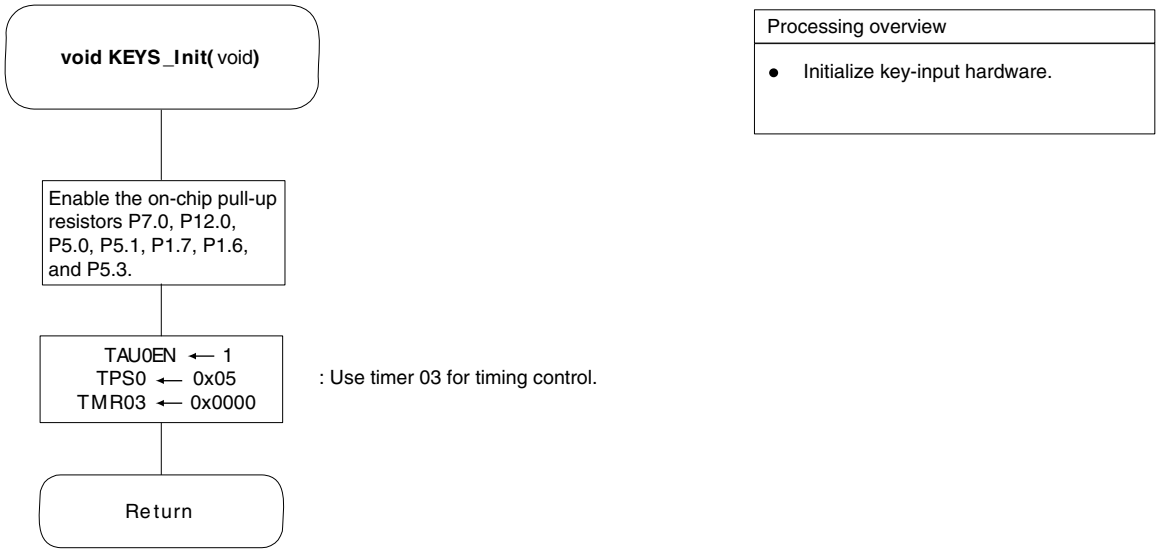
Processing overview
• Transmit the eight bits of data specified by the parameter "Value" to the I2C bus.

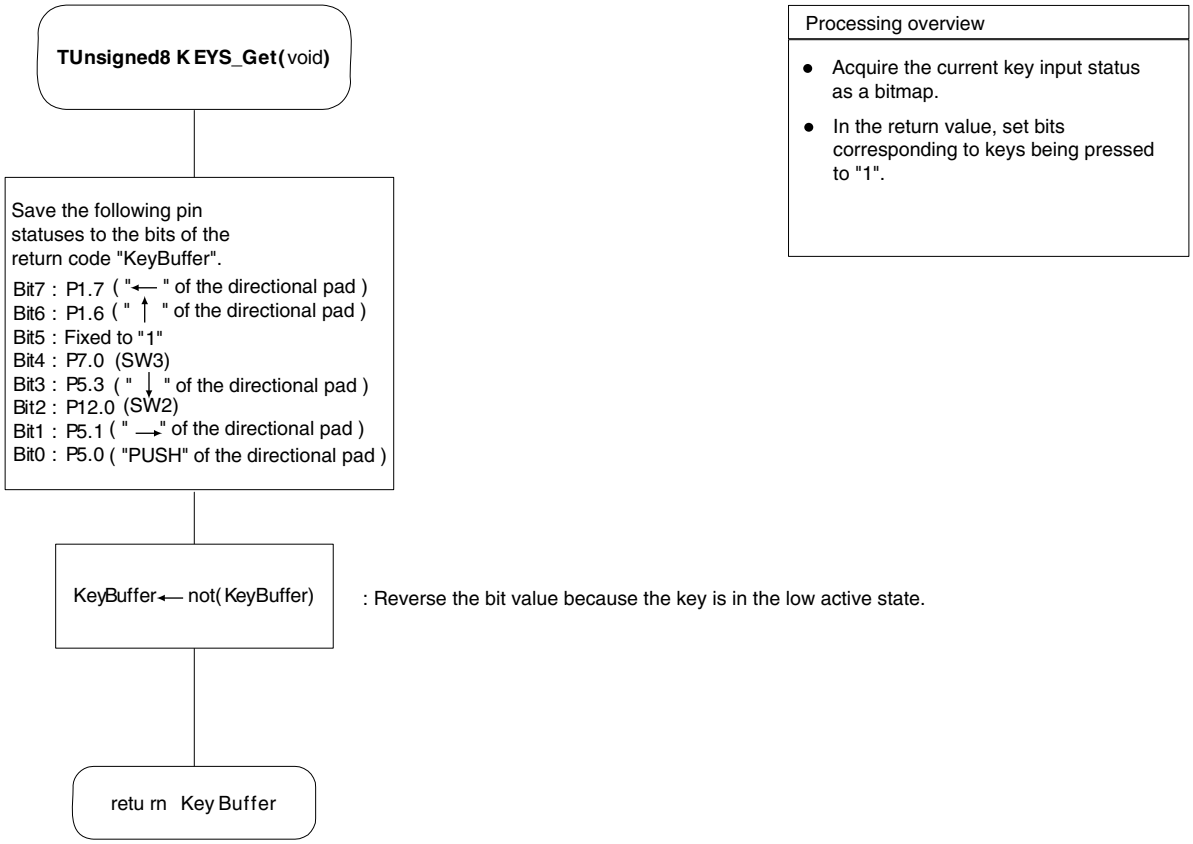


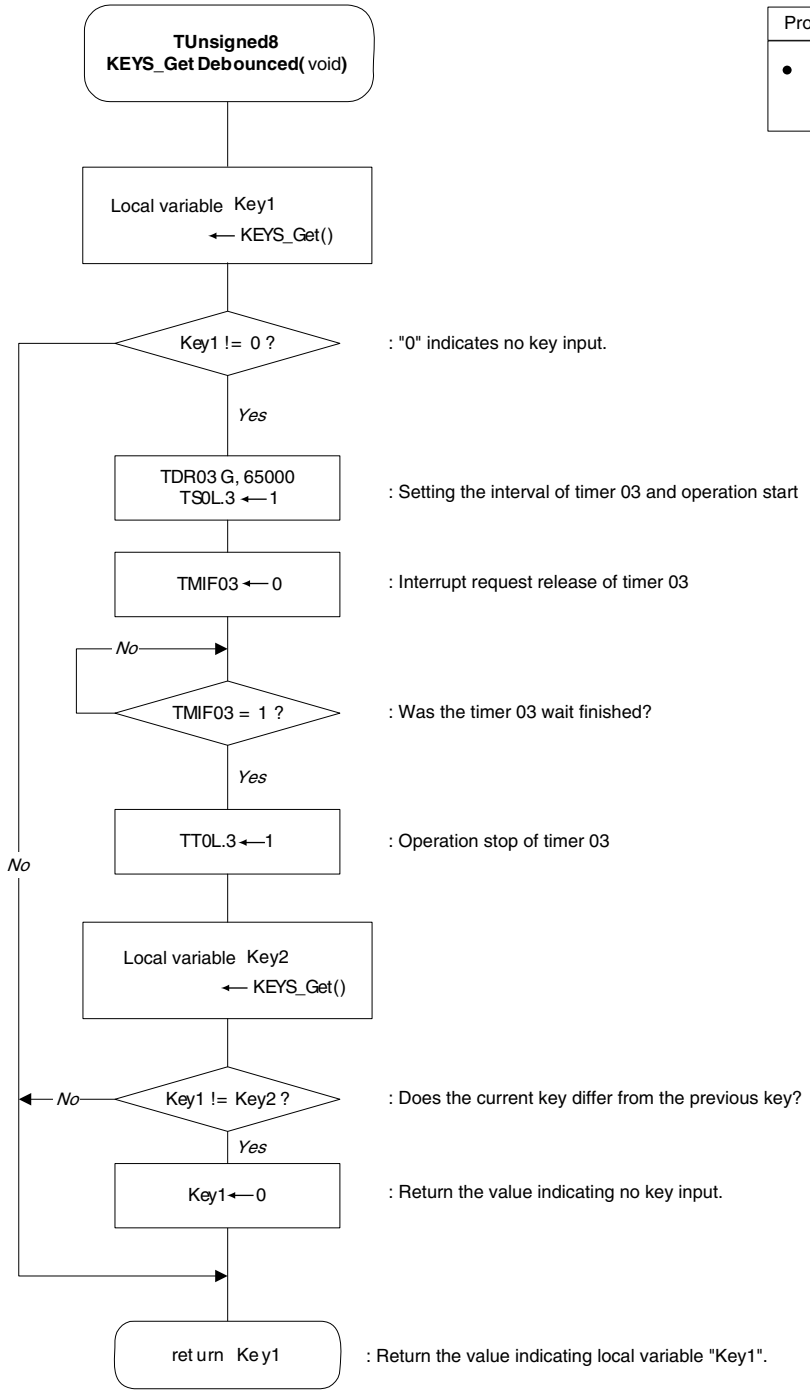
: Use the local variable "i" as a counter.

Processing overview
• Transmit one frame of data to the I2C bus. • The parameter "Address" is the I2C address, Length is the frame size, and pData is a pointer to the transmitted data.

13.4.7 Key input module flowchart







Processing overview

- Acquire the status of a stable key, and return it.

CHAPTER 14 CABLES

14.1 USB interface cable (Mini-B type)

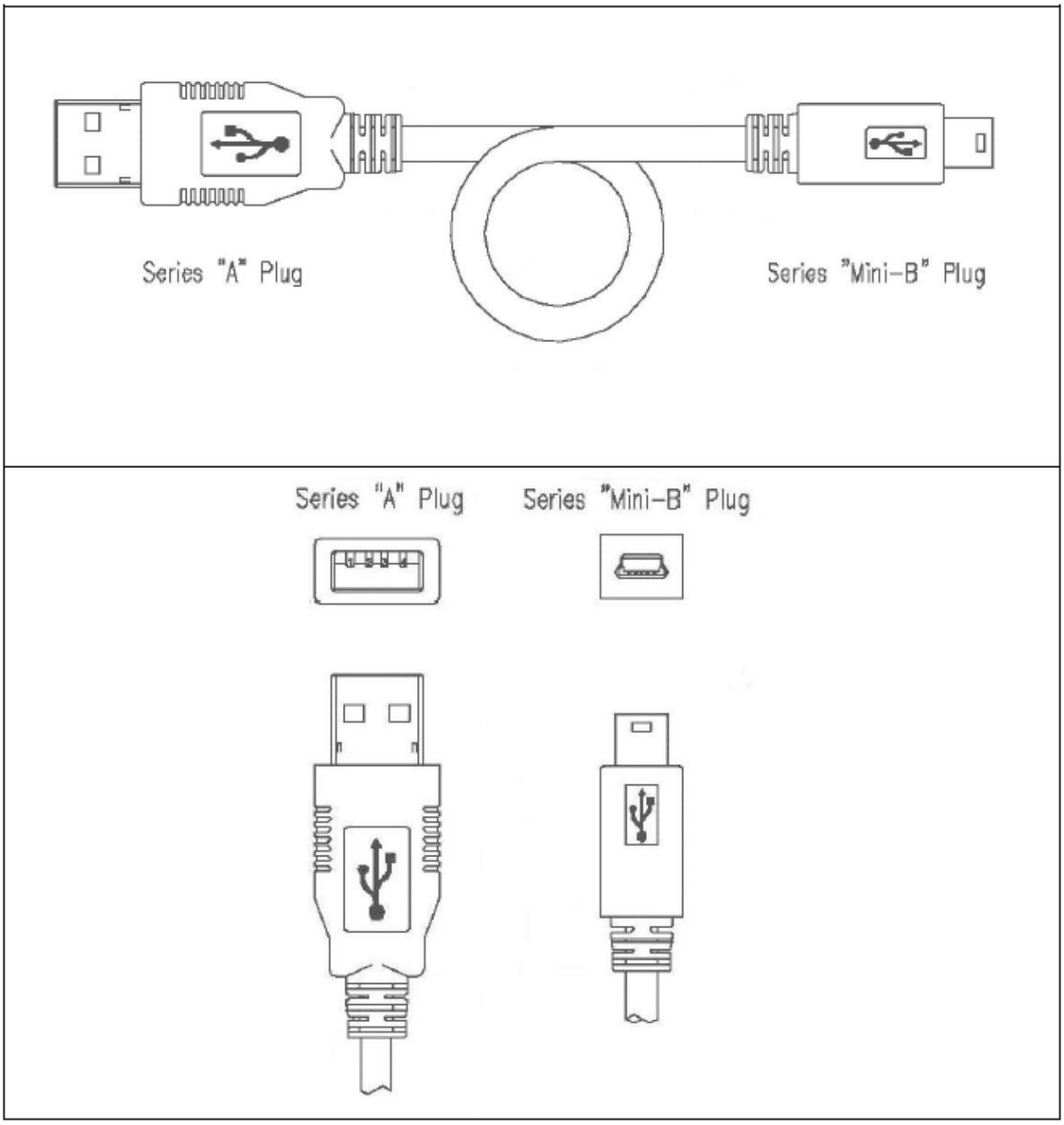


Figure 14-1. USB interface cable (Mini-B type)

Published by: NEC Electronics Corporation (http://www.necel.com/)
Contact: http://www.necel.com/support/