NEC

User's Manual

TK-78F0730

USB-Demonstration Kit for the 78K0 8-bit microcontroller family

Document No. U19557EE1V1UM00
Date Published May 2009

CAUTION

This is a Test- and Measurement equipment with possibility to be significantly altered by user through hardware enhancements/modifications and/or test or application software. Thus, with respect to Council Directive 89/336/EEC (Directive on compliance with the EMC protection requirements), this equipment has no autonomous function. Consequently this equipment is not marked by the CE-symbol.

EEDT-ST-005-10

CAUTION

This equipment should be handled like a CMOS semiconductor device. The user must take all precautions to avoid build-up of static electricity while working with this equipment. All test and measurement tool including the workbench must be grounded. The user/operator must be grounded using the wrist strap. The connectors and/or device pins should not be touched with bare hands.

EEDT-ST-004-10

For customers in the European Union only

Redemption of Waste Electrical and Electronic Equipment (WEEE) in accordance with legal regulations applicable in the European Union only: This equipment (including all accessories) is not intended for household use. After use the equipment cannot be disposed of as household waste. NEC Electronics (Europe) GmbH offers to take back the equipment. All you need to do is register at http://www.eu.necel.com/weee

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

NEC Electronics Inc. (U.S.)
Santa Clara, California
Tel: 408-588-6000
800-366-9782
Fax: 408-588-6130
800-729-9288

NEC Electronics (Europe) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 0
Fax: 0211-65 03 1327

Sucursal en España
Madrid, Spain
Tel: 091- 504 27 87
Fax: 091- 504 28 60

Succursale Française
Vélizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

Filiale Italiana
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

Branch The Netherlands
Eindhoven, The Netherlands
Tel: 040-244 58 45
Fax: 040-244 45 80

Branch Sweden
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

United Kingdom Branch
Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Singapore Pte. Ltd.
Singapore
Tel: 65-6253-8311
Fax: 65-6250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-2719-2377
Fax: 02-2719-5951

NEC do Brasil S.A.
Electron Devices Division
Guarulhos, Brasil
Tel: 55-11-6465-6810
Fax: 55-11-6465-6829

Revision History

Date	Revision	Chapter	Description
12-01-2009	V1.0	---	First release
08-05-2009	V1.1	1	Incorrectly information of analog I/O function removed

Table of Contents

List of Figures

List of Tables

1. Introduction

TK-78F0730 is a USB-demonstration kit for the NEC 78K0 8-bit microcontroller family. It allows the development of an USB system based on the 78K0 78F0730 device. It supports onboard debugging and real time execution of application programs. The board is prepared to be connected to user hardware parts such as digital I/O signals.

1.1 Main features of *TK-78F0730*

- Easy to use device demonstration capabilities
  TK-78F0730 contains elements to easily demonstrate simple I/O-functions, i.e. I/O lines, UART serial interface, USB interface etc.

- On-Board debug function (TK-78K0 debugging)
  The *TK-78F0730* supports an On-Board debug function by using the IAR C-SPY debugger without a need of additional debug hardware. It allows FLASH downloading and standard debug functions like code execution, single stepping, breakpoints, memory manipulation etc.

- Power supply by USB interface or via external power supply

- Various input / output signals available, such as

  - I/O ports prepared to be connected to user hardware
  - Timer input / output signals
  - Two serial interfaces
  - Virtual UART interface, via the µPD78F0730 78K0 8-bit microcontroller with on-board USB interface

- The IAR Embedded Workbench for 78K and the IAR C-SPY debugger / simulator are included. These packages are restricted in such that maximum program code size is limited to 16 KB.

- Full documentation is included for the NEC 78K0 78F0730 microcontroller, IAR Systems Embedded Workbench and IAR Systems C-SPY debugger / simulator.

***TK-78F0730* is not intended for code development. NEC does not allow and does not support in any way any attempt to use *TK-78F0730* in a commercial or technical product.**

1.2 System requirements

HOST PC	A PC supporting Windows 2000, Windows XP or Windows Vista is required for the IAR Systems Embedded Workbench demo-version. 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)

1.3 Package contents

Please verify that you have received all parts listed in the package contents list attached to the *TK-78F0730* package. If any part is missing or seems to be damaged, please contact the dealer from whom you received your *TK-78F0730*.

Note: Updates of the IAR Embedded Workbench for 78K, documentation and/or utilities for *TK-78F0730*, if available, may be downloaded from the NEC WEB page(s) at **http://www.eu.necel.com/TK-78F0730**

1.4 Trademarks

IAR Embedded Workbench, visualSTATE, IAR MakeApp and C-SPY are registered trademarks of IAR Systems AB. Microsoft and Windows are registered trademarks of Microsoft Corporation. Adobe and Acrobat Reader are registered trademarks of Adobe Systems Incorporated.

All other product names are trademarks or registered trademarks of their respective owners.

2. *TK-78F0730* system configuration

The *TK-78F0730* system configuration is given in the diagram below:

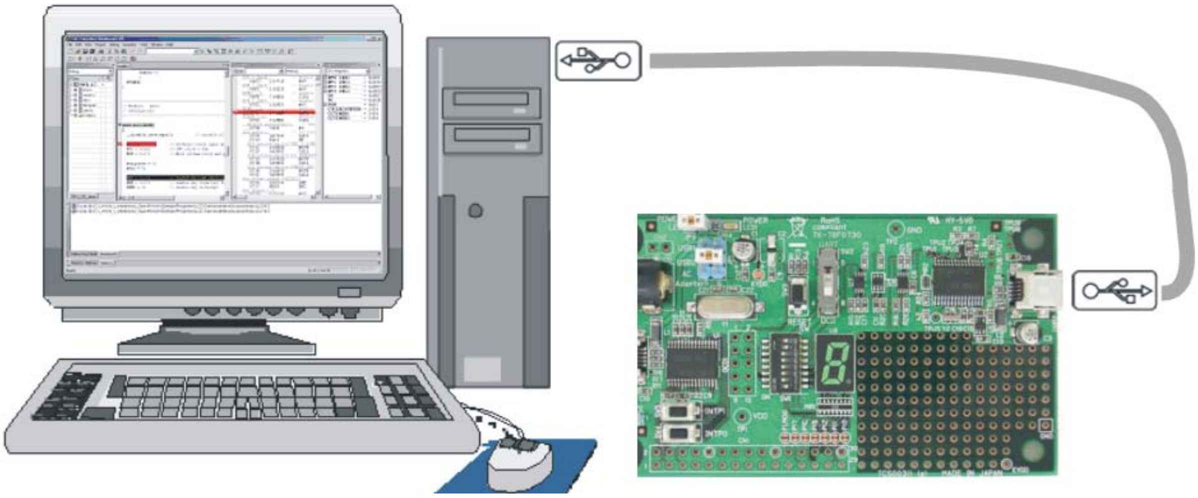


Figure 1: *TK-78F0730* system configuration

2.1 TK-78F0730

TK-78F0730 is a USB-demonstration kit for the 78F0730 8-bit microcontroller of the 78K0 family. The demonstration board is connected to the host system via USB interface cable. The host system may be used for On-Chip debugging by using the IAR C-SPY debugger and to allow execution of application programs on *TK-78F0730* starterkit.

TK-78F0730 runs the microcontroller at 16 MHz operating speed.

2.2 Host computer

The USB host interface enables communication to the *TK-78F0730* 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 *TK-78F0730* board is powered by the USB interface. Optional the power supply can be applied via the connectors JACK1.

3. *TK-78F0730* components

The *TK-78F0730* board is equipped with USB-connector and with several connectors in order to be connected to host computers, FLASH programmer or any external target hardware.

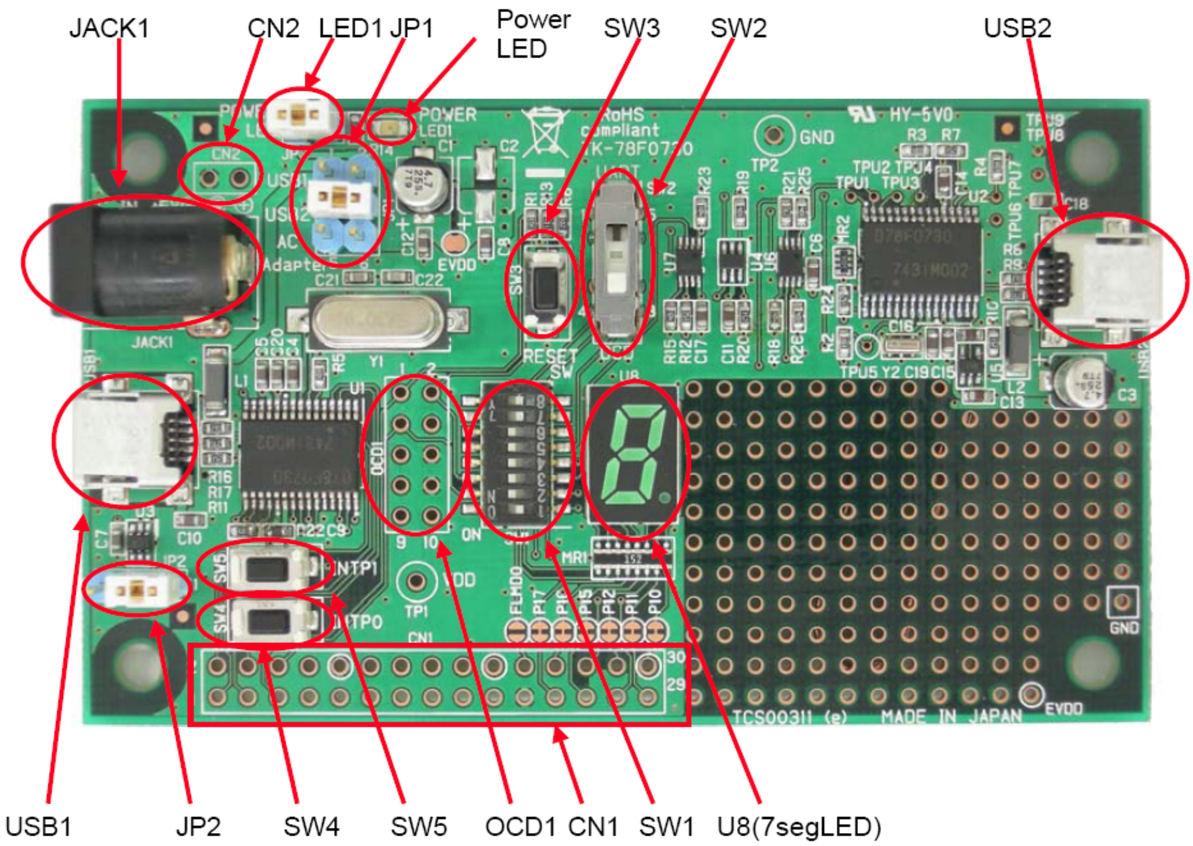


Figure 2: TK-78F0730 Components

Some of the *TK-78F0730* components are free for user application hardware and software. Please read the user's manual of the 78F0730 device carefully to get information about the electrical specification of the available I/O ports before you connect any external signals to the *TK-78F0730* board.

3.1 SW1, Configuration Switch (DIP-Switch)

The different operation modes of the *TK-78F0730* board can be set by switch SW1. The bits 1-5 of DIP switch SW1 are for the mode setting of the board, bits 5-6 are DIP switches for the seven segment LEC and bits 7-8 are connected to the pins “P00/TI000” and P01/TI010/TO00” of the 78K0 microcontroller and can be used for user application purpose.

The TK-78F0730 starterkit can be used in the following operation modes:

- Stand alone mode
  - Run a program stored in built-in flash memory of the 78F0730 device
- On-Board debug mode
  - Start a debug session using the TK-78K0 Interface
- MINICUBE2 debug mode
  - Start a debug session using MINICUBE2 emulator
- Flash-Programming Mode
  - Program an application to the build-in flash memory of 78F0730 by WriteEZ3-programmer

SW1 / bit	On-Board Debug Mode	Stand alone Mode	Flash Programming Mode	MINICUBE2 Debug Mode
1	OFF	OFF	OFF	OFF
2	ON	OFF	ON	OFF
3	ON	OFF	ON	OFF
4	ON	OFF	ON	OFF
5	ON / OFF [1]	ON / OFF [1]	OFF	ON / OFF[1]
6	ON / OFF [1]	ON / OFF [1]	OFF	ON / OFF[1]

Table 1: Mode setting, switch SW1

Note: After changing the configuration of SW1 bits1-5 it is necessary to power-up the *TK-78F0730 board* to make changing active. This can be done by simply dis- and re-connecting the USB interface cable.

[1] If P13/P14 is not used for seven-segment-LED then turn off SW1 bit 5-6 and separate the seven-segment-LED.

3.2 SW2, Configuration switch

The different operation modes of the *TK-78F0730* board can be set by switch SW2.

	On-Board Debug Mode	Stand alone Mode	Flash Programming Mode	MINICUBE2 Debug Mode
SW2	OCD side	UART side or centre position	UART side	OCD side or centre position

Table 2: Mode setting, switch SW2

Note: After changing the position of SW2 it is necessary to power-up the *TK-78F0730 board* to make changing active. This can be done by simply dis- and re-connecting the USB interface cable.

3.3 SW3, RESET button

SW3 is the reset button. It activates the power on reset. Switch SW3 controls the reset input signal of the 78F0730 microcontroller.

3.4 SW4, Switch (INTP0)

SW4 is a push button connecting VSS to external interrupt input INTP0 of the microcontroller. This is equal to port "P120/INTP0" of the 78F0730 device. The port may be programmed to generate the external interrupt INTP0. The necessary initialization for this purpose is described in the user's manual of the 78F0730 device. Please note, when using SW2 turn ON the built-in pull-up resistor of the 78F0730 device, register PU12.

3.5 SW5, Switch (INTP1)

SW5 is a push button connecting VSS to external interrupt input INTP1 of the microcontroller. This is equal to port "P30/INTP1" of the 78F0730 device. The port may be programmed to generate the external interrupt INTP1. The necessary initialization for this purpose is described in the user's manual of the 78F0730 device. Please note, when using SW3 turn ON the built-in pull-up resistor of the 78F0730 device, register PU3.

3.6 JP1, Power Supply selector

Jumper JP1 is the power supply selector of the *TK-78F0730 board*.

JP1	Configuration	Power Supply Source
1-2	Closed	USB connector USB1
3-4	Closed (default)	USB connector USB2
5-6	Closed	AC adapter via connector JACK1

Table 3: Power supply selector, JP1

Note: If multiple connections made to USB1, USB2 and JACK1, set JP1 to the preferred power supply route.

3.7 JP2, Pole selector

Jumper JP2 defines the pole level of USBPUC

JP2	Configuration	D+ pull –up
1-2	Closed	Valid if USBPUC is low level
2-3	Closed	Valid if USBPUC is high level

Table 4: Pole select of USB D+ pull-up, JP2

3.8 JP3, Power LED

Jumper JP3 enables or disables power LED1

JP2	Configuration	Power LED1
1-2	Closed	Enabled
1-2	Opened	Disabled

Table 5: Power LED1 setting

3.9 LED1, power LED

LED1 is the power LED of the *TK-78F0730 board*. It indicates if power is applied to the *TK-78F0730 board*.

3.10 JACK1, AC power supply connector

JACK1 is the AC power supply connector of the *TK-78F0730 board*. Caution, please connect only a power supply of maximum +5V DC to the board. There is no voltage regulator assembled on the *TK-78F0730 board*. Higher supply voltage can damage the board.

JACK1	Input
Centre	VDD (+5V)
Ring	GND

Table 6: JACK1 connector

3.11 OCD1, MINICUBE2 connector

Connector OCD1 (not assembled) allows connecting the QB-MINI2 On-Chip debug emulator (MINICUBE2) to *TK-78F0730 board*. Please note, the QB-MINI2 On-Chip debug emulator is a separate product from NEC and it is not included in this starterkit package.
Please take care of pin 1 position.

3.12 U8, Seven-segment-LED

The seven-segment-LED U8 is controlled by port P10 to P17. Please set port mode to output and output a low signal to light the corresponding segment.

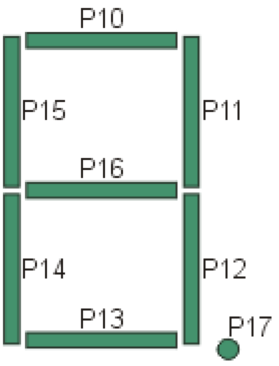


Figure 3: Seven-segment-LED U8

To display the characters ‘0’ to ‘9’ please write the following values to register P1:

Character	P1 value
0	0xC0
1	0xF9
2	0xA4
3	0xB0
4	0x99
5	0x92
6	0x83
7	0xf8
8	0x80
9	0x98

Table 7: Display Examples

3.13 CN2, power connector

JCN is the external power supply connector of the *TK-78F0730 board*. Caution, please connect only a power supply of maximum +5V DC to the board. There is no voltage regulator assembled on the *TK-78F0730 board*. Higher supply voltage can damage the board.

3.14 Universal Wrap field

For the integration of additional application hardware and user circuits the *TK-78F0730 board* offers a wrap field. Please read the user’s manual of the 78F0730 device carefully to get information about the electrical specification of the available I/O ports before you connect any external signals to the *TK-78F0730 board*.

3.15 USB1, serial interface connector

This interface allows connecting the IAR C-SPY debugger to the *TK-78F0730 board* in order to use the On-Board debug function (TK-78K debugging). The TK-78K 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 UART3 of the 78F0730 device can be redirected to the USB1 connector via the µPD78F0730 USB microcontroller.

The power supply of the *TK-78F0730 board* is also provided by the USB1 connector.

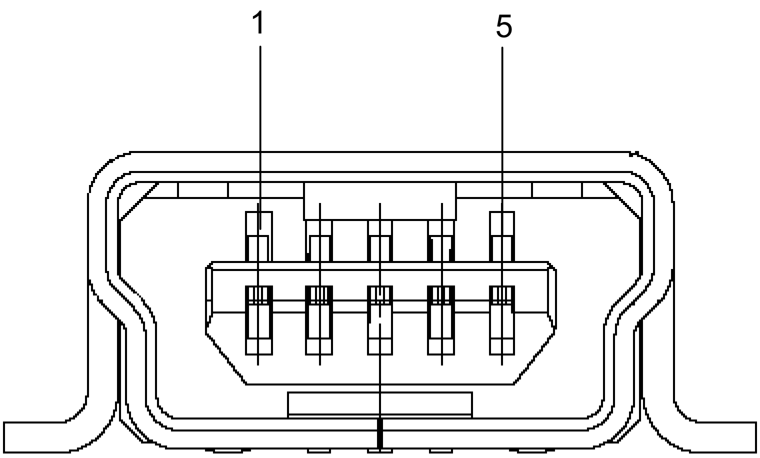


Figure 4: USB1, USB Mini-B Type Host Connector Pin Configuration

Connector USB1	Signal Name
1	VBUS
2	D-
3	D+
4	ID_NC
5	GND

Table 8: Pin Configuration of Connector USB1

For connection with the host machine, use a USB cable (Mini-B type). For confirmation, NEC Electronics used only the USB cable delivered with the *TK-78F0730 board*.

3.16 Layout of solder-short pads

Several pins of the 78F0730 microcontroller are connected to solder short-pads. The pads can be opened by the user to add user specific functions. The signal connected to each solder-short pad is printed on the TK-78F07030 board.

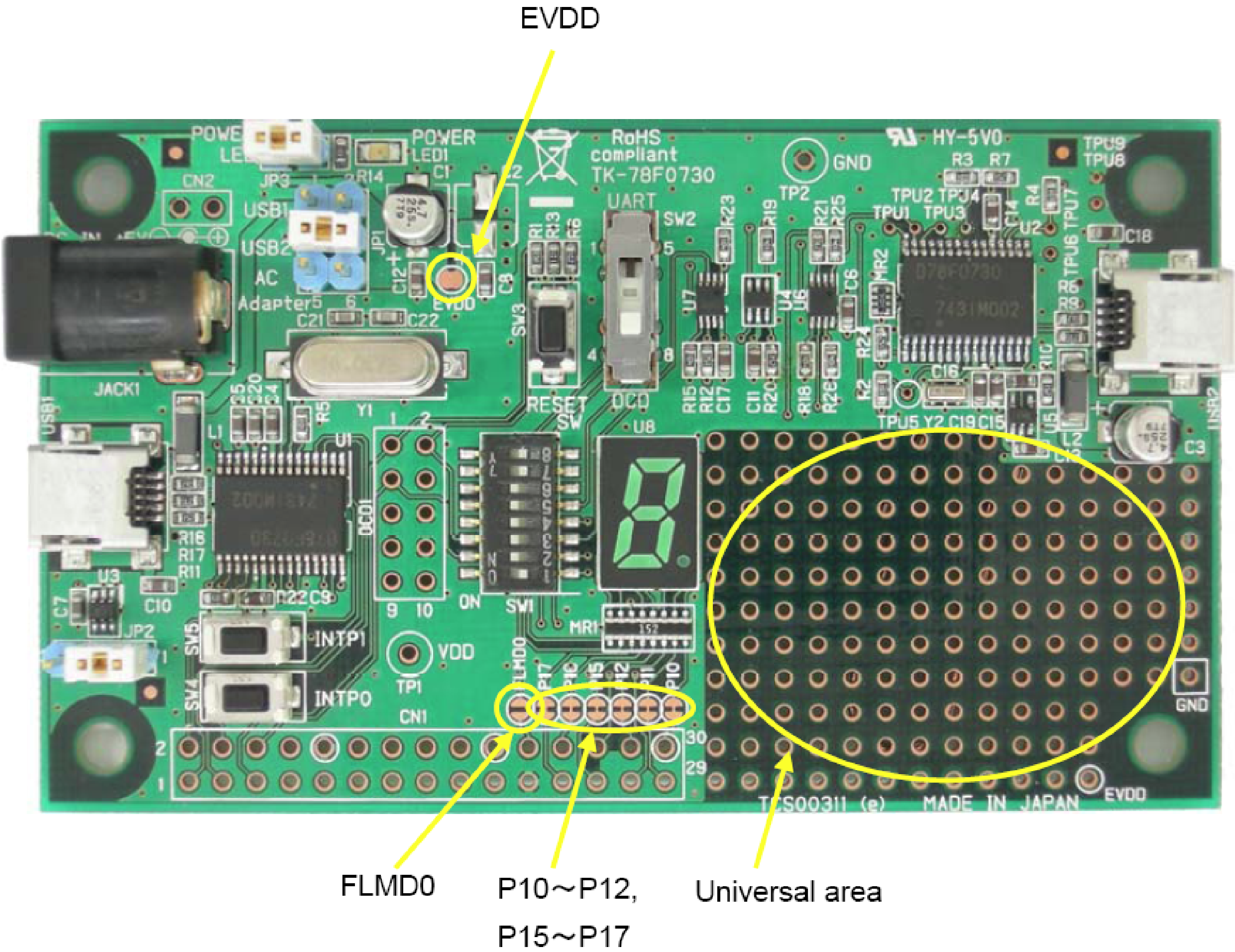


Figure 5: Solder-short pad Layout

To open a circuit, cut the narrow part of the pad with a knife. To short a circuit again, join the separated pad with a soldering iron.

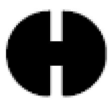


Figure 6: Solder-short pad opened shape

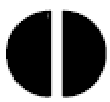


Figure 7: Solder-short pad shortened shape

Solder-short pad name	Shipping state	Connection
P10-P12,P15-P17	shortened	connection to seven-segment LED Open pads if port pins shall have other usage.
FLMD0	opened	FLMDO to CPU port P33 Shorten pad if flash self programming is used
EVDD	shortened	EVDD to VDD Open if EVDD is driven by other voltage

Table 9: Solder-short pad connection

4. On-Chip debugging

The *TK-78F0730 board* offers two possibilities to use On-Chip debugging (OCD). The TK-78K0R On-Board debug function of *TK-78F0730* 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.

Additionally *TK-78F0730* supports the QB-MINI2 On-Chip debug emulator in order to use On-Chip debug function of the 78F0730 device. The system configuration for On-Chip debugging is shown in figure below.

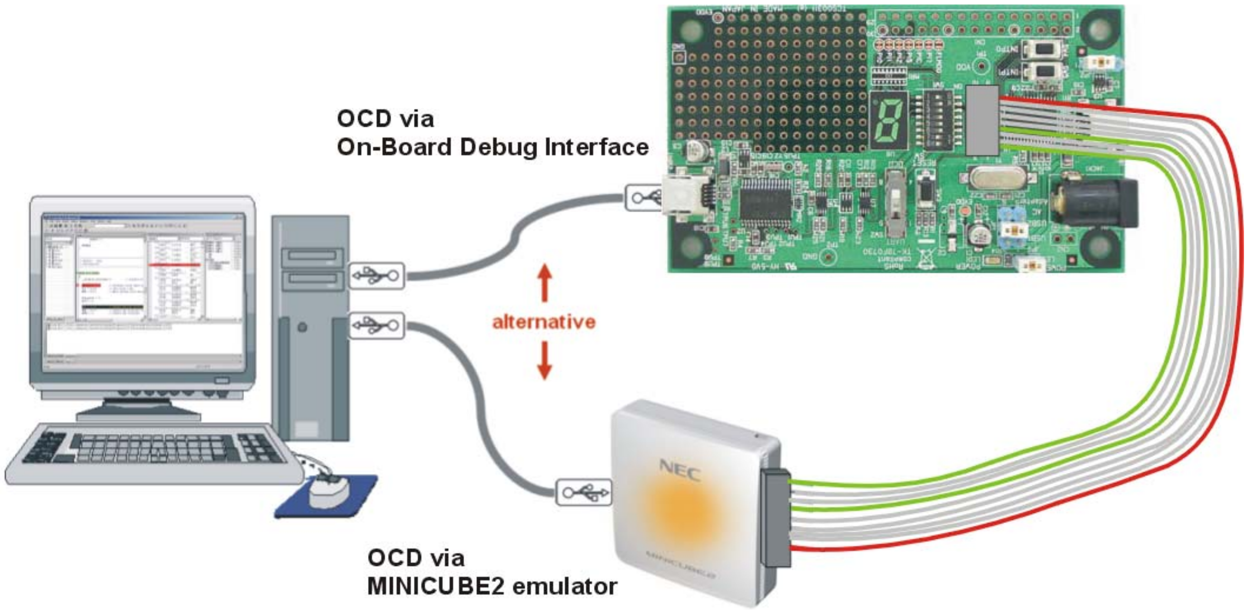


Figure 8: On-Chip debugging

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

To operate the *TK-78F0730 board* within the On-Board debug mode, configure switch SW5 bits1-5 as following:

SW5/bit	Configuration
1	ON / OFF (*)
2	ON
3	ON
4	OFF
5	OFF

Table 10: OCD via TK-78K0R On-Board debug function

(*) = individual selectable by user.

4.2 OCD via QB-MINI2 emulator

To operate the *TK-78F0730 board* together with the QB-MINI2 On-Chip debug emulator, configure switch SW5 bits1-5 as following:

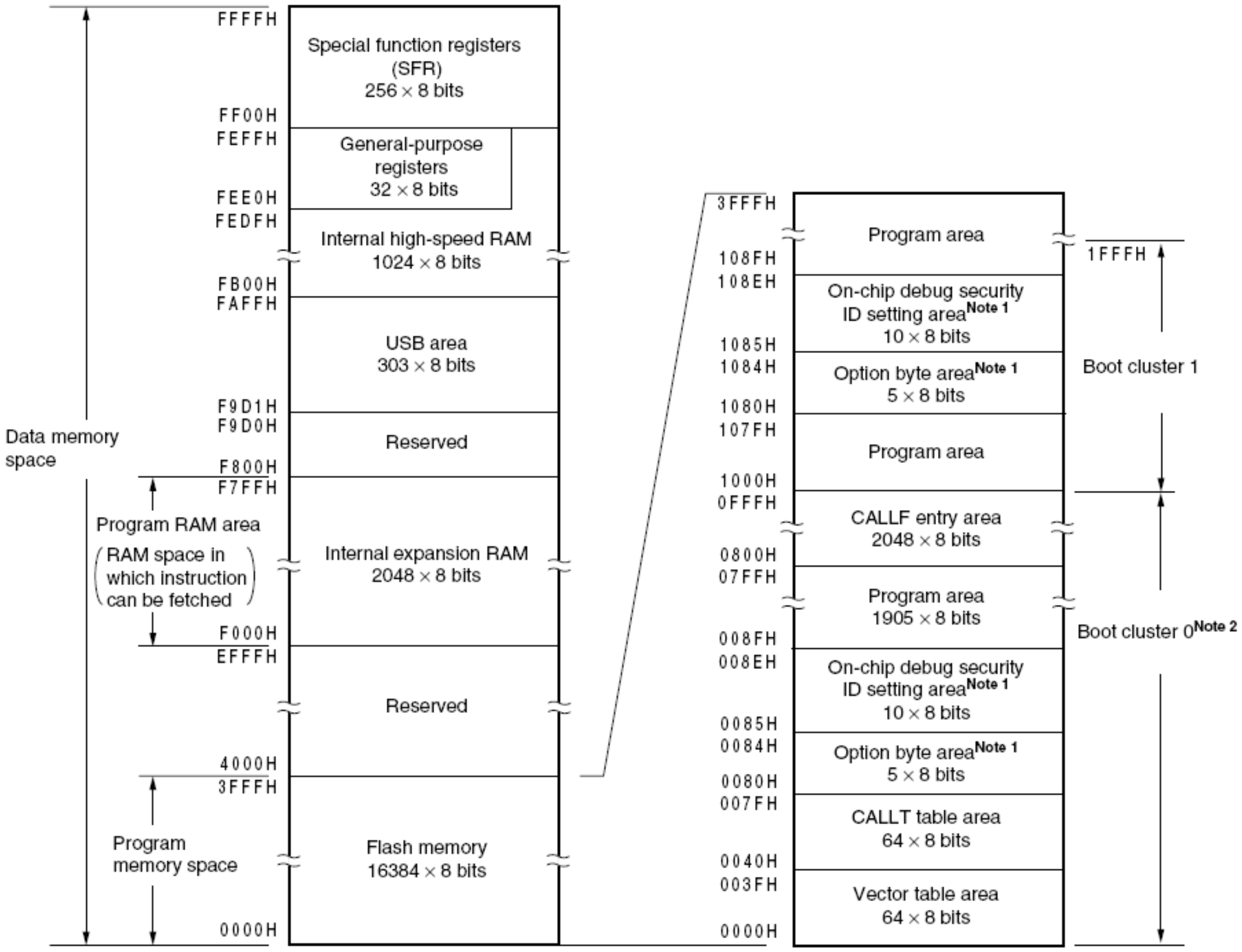
SW5/bit	Configuration
1	OFF
2	OFF
3	OFF
4	OFF / ON (*)
5	OFF / ON (*)

Table 11: OCD via QB-MINI2 emulator

(*) = individual selectable by user.

5. 78F0730 Memory Layout

The memory layout of 78F0730 device is shown in the figure below.



Notes 1. When boot swap is not used: Set the option bytes to 0080H to 0084H, and the on-chip debug security IDs to 0085H to 008EH.

When boot swap is used: Set the option bytes to 0080H to 0084H and 1080H to 1084H, and the on-chip debug security IDs to 0085H to 008EH and 1085H to 108EH.

2. Writing boot cluster 0 can be prohibited depending on the setting of security

Figure 9: 78F0730 Memory Map

The TK-78F0730 uses the standard resources necessary of the 78K0 OCD interface, consequently the resources have to be reserved by the application software.

5.1 Resources used by 78K OCD Interface

Debugging via 78K OCD Interface uses the user memory spaces (shaded portions in Figure 10) to implement communication with the target device, or each debug functions. The areas marked with a dot (•) are always used for debugging, and other areas are used for certain debug functions. Refer to the following descriptions and secure these spaces in the user program.

- addresses 0x0002, 0x0003: Interrupt vector of debug monitor
  This area is automatically reserved.
  It is not allowed to use this area for any application segment.

- addresses 0x007E, 0x007F: CALLT table entry used for software breakpoints
  This area must be reserved in the linker control file (*.xcl)
  It is not allowed to use this area for any application segment.

- address 0x0084: Option byte to configure the OCD Interface
  The option byte must be defined to configure the OCD Interface.
  Details are described in the µPD78F0730 user's manual.

- addresses 0x0085-0x008F Security ID
  The security ID must be defined to configure the OCD Interface
  Details are described in the µPD78F0730 user's manual.

- addresses 0x008F – 0x018F: Debug Monitor area
  This area must be reserved in the linker control file (*.xcl).
  It is not allowed to use this area for any application segment.

- addresses 0x0190 – 0x028F: Pseudo Realtime RAM Mirror area (RRM area)
  This area must be reserved in the linker control file (*.xcl), if the RRM Feature is used (e.g. Live Watch Window)
  It is not allowed to use this area for any application segment.

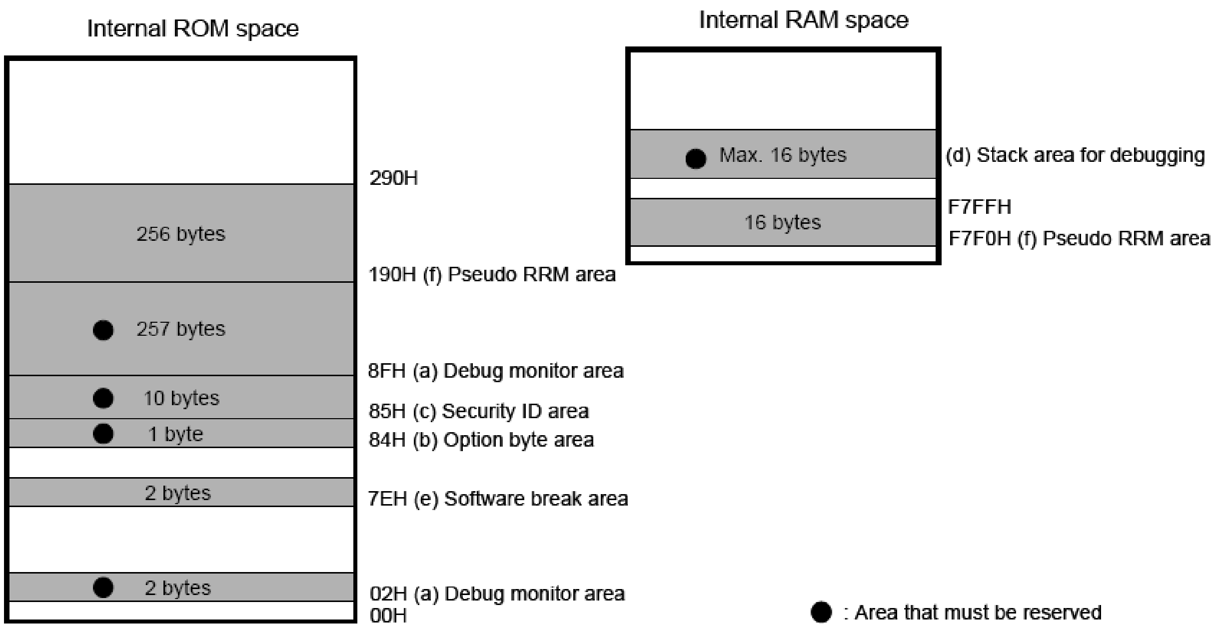


Figure 10: 78K OCD Interface Resources

6. *TK-78F0730* installation and operation

6.1 Getting started

The IAR Embedded Workbench including the C-SPY debugger allows building and downloading application programs to the *TK-78F0730* starterkit. As communication interface between the PC host system and the *TK-78F0730 board* a standard USB interface line is needed. Before you can download and run a program, software and hardware have to be installed properly.

6.1.1 CD-ROM contents

The CD-ROM shows following directory structure:

NEC TK-78F0730	CD-ROM ROOT
Acrobat	- Acrobat Reader for 32Bit Windows OS
Doc	- Documentation
IAR	- IAR Embedded Workbench for 78K
SamplePrograms	- Sample programs for TK-78F0730 *including:* o *TK-78F0730 COM sample* o *TK-78F0730 HID sample* o *TK-78F0730 Demonstration sample*
WriteEZ3	o Flash Programmer WriteEZ3 incl. PRM file for µPD78F0730

Table 12: *TK-78F0730* CD-ROM directory structure

6.1.2 Using the Pre-Programmed Demo-Application

The demo application described in chapter 12.2 had been programmed to TK-78F0730 during production. To start the demo-application please check that the configuration switches SW1 (table 1) and SW2 (table 2 are configured for free-running mode and connect a power supply according to the setting of JP1 (table 3). After pressing switch SW4 the random number generator is started visualized by a flashing LED segment. Pressing switch SW5 stops the generator and the number is displayed.

7. Hardware installation

After unpacking *TK-78F0730,* connect the board via connector USB2 to your host computer using the provided USB interface cable. When *TK-78F0730* is connected, the USB driver needs to be installed on the host machine. Please refer to the following **CHAPTER 8 SOFTWARE INSTALLATION**.

8. Software installation

The *TK-78F0730* package comes with the following software demo packages:

- IAR Systems Embedded Workbench for 78K, including C compiler, assembler, linker, librarian and IAR C-SPY debugger / simulator

- Sample programs

The IAR Systems Embedded Workbench must be installed on your PC. For detailed installation hints, refer to the following chapters and to the corresponding documentation of the IAR Embedded Workbench.

8.1 IAR Systems Embedded Workbench for 78K installation

To install the IAR Systems Embedded Workbench for 78K including C-SPY debugger / simulator, select the `AUTORUN` program in the directory `\IAR\` of the CDROM. The setup dialogues will guide you through the installation process.

8.2 Sample program installation

To install the sample/demonstration programs for the *TK-78F0730 board* select the `SETUP` program in the directory `\SamplePrograms\` of the CDROM. The setup dialogues will guide you through the installation process.

8.3 USB Driver Installation

In order to use the *TK-78F0730 board* for On-Chip debugging the USB driver needs to be installed on the host machine. Install the driver according to the following procedure:

Note: The USB driver is part of the IAR Embedded Workbench software package. Therefore please install the IAR Embedded Workbench first.

8.3.1 Installation on Windows 2000

1. When the *TK-78F0730 board* is connected with the host machine, the board is recognized by <Plug and Play>, and the wizard for finding new hardware is started. Click Next>.

Figure 11: Found New Hardware Wizard (Windows 2000)

2. Following the window below is displayed. So, check that "Search for a suitable driver ..." is selected, then click Next>.

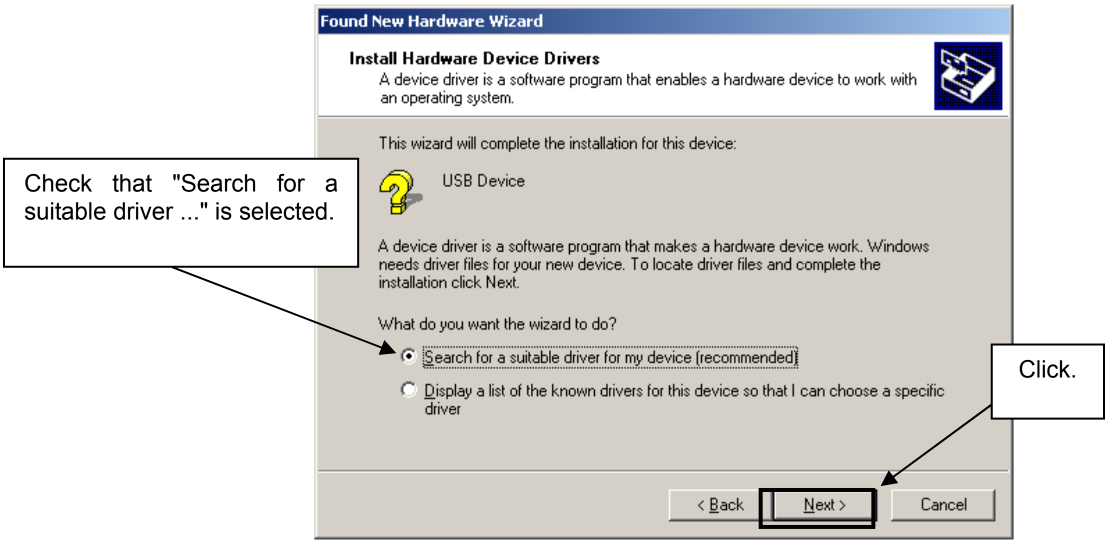


Figure 12: Search Method (Windows 2000)

3. Check the "Specify a location" check box only, then click Next>.

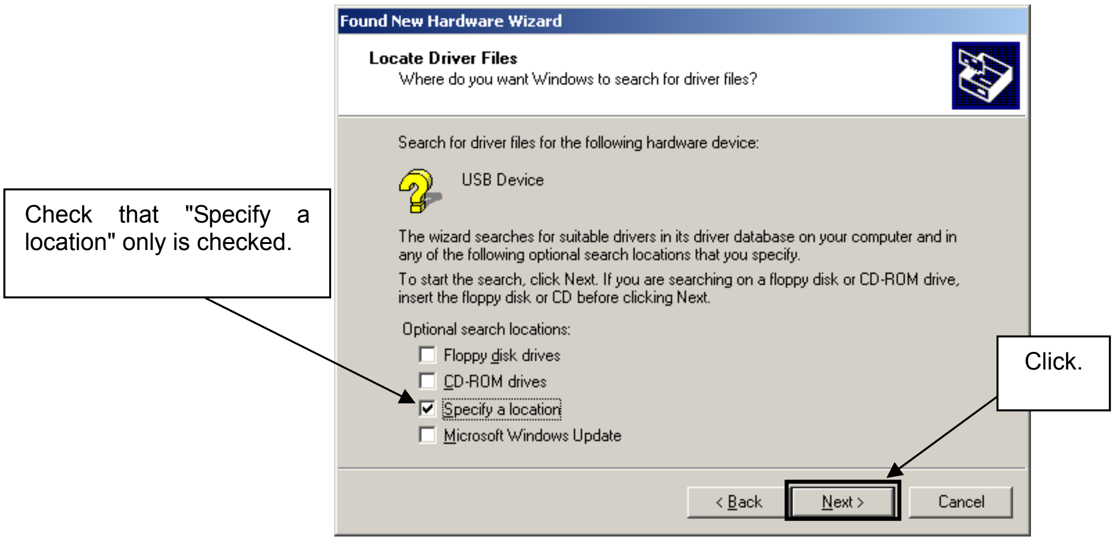
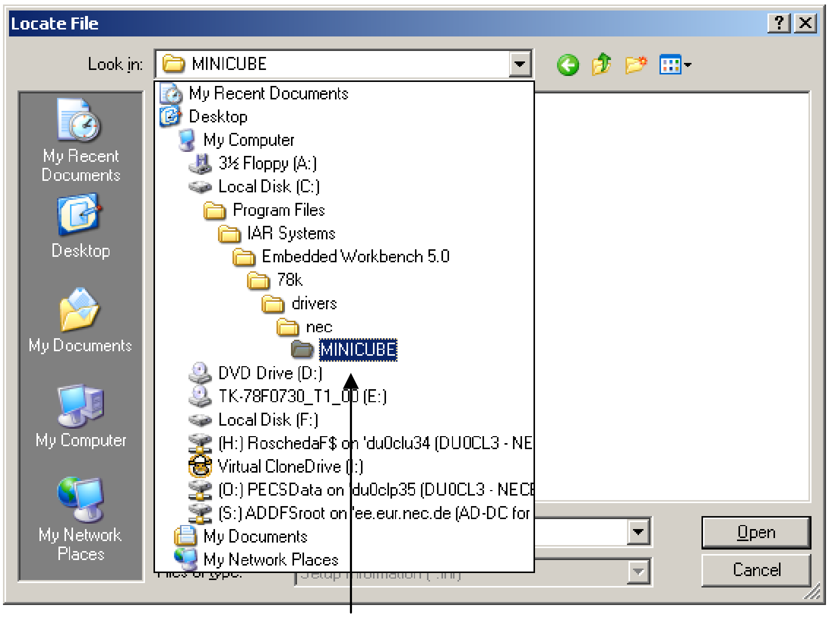


Figure 13: Driver File Location (Windows 2000)

4. Locate to the folder "C:\Program Files\IAR Systems\Embedded Workbench 5.0\78K\config\nec\drivers\MINICUBE”.



Locate to “C:\Program Files\IAR Systems\Embedded Workbench 5.0\78K\config\nec\drivers\NEC\MINICUBE”

Figure 14: Address Specification 1 (Windows 2000)

Remark If the installation destination folder is changed at the time of IAR Embedded Workbench installation, enter "new-folder\78K\config\nec\ie_pc_driver\MINICUBE".

5. The setup information file “MQB2ALL.inf” is automatic selected, then click Open to proceed within driver installation.

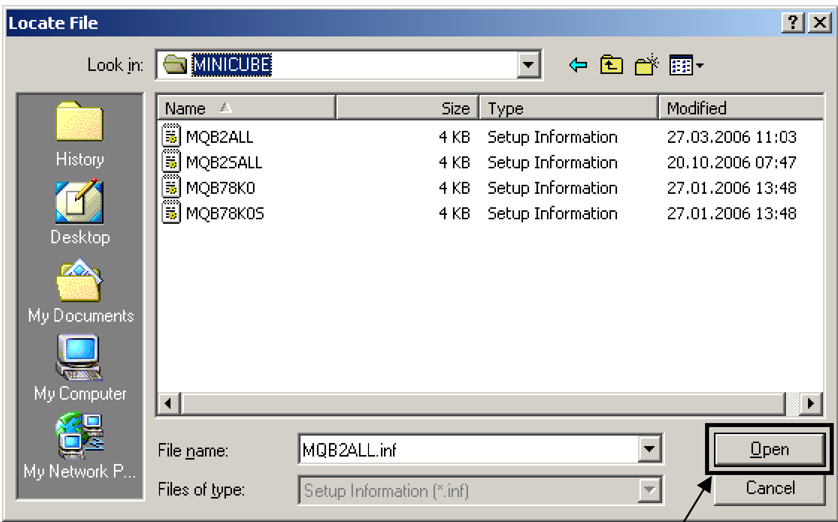


Figure 15: Address Specification 2 (Windows 2000)

Click.

6. After the location of the USB driver has been specified click OK to proceed.

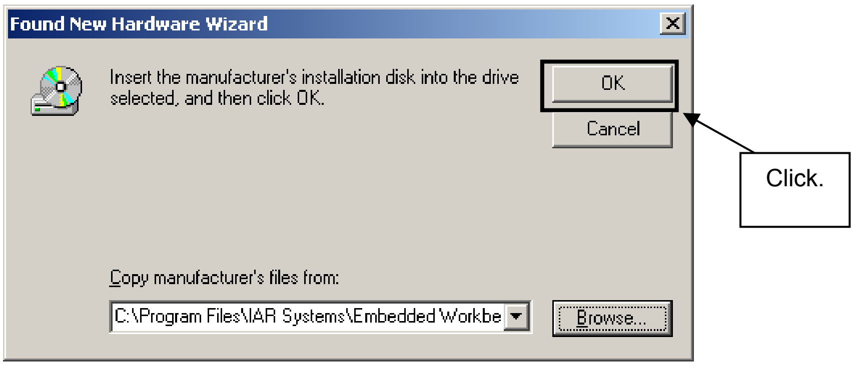


Figure 16: Address Specification 3 (Windows 2000)

5. Click Next>.

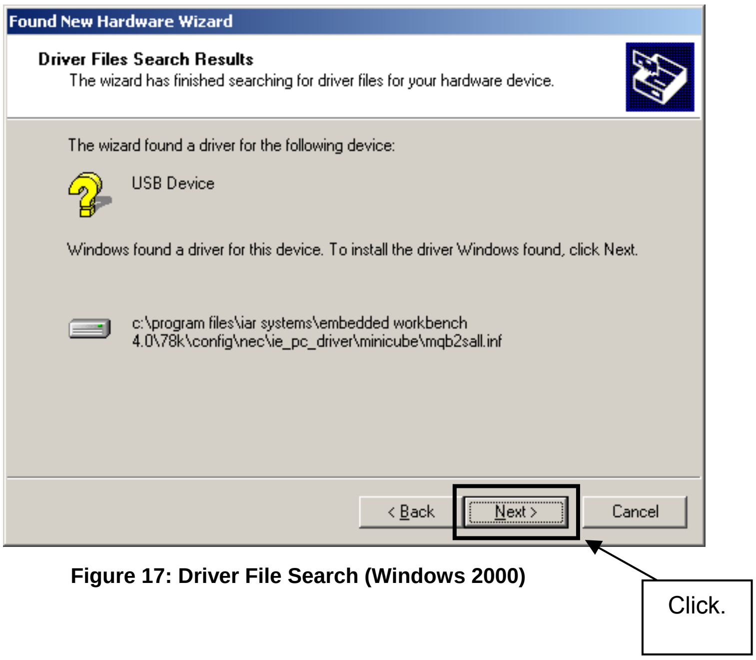


Figure 17: Driver File Search (Windows 2000)

6. Click Finish to complete the installation of the USB driver.

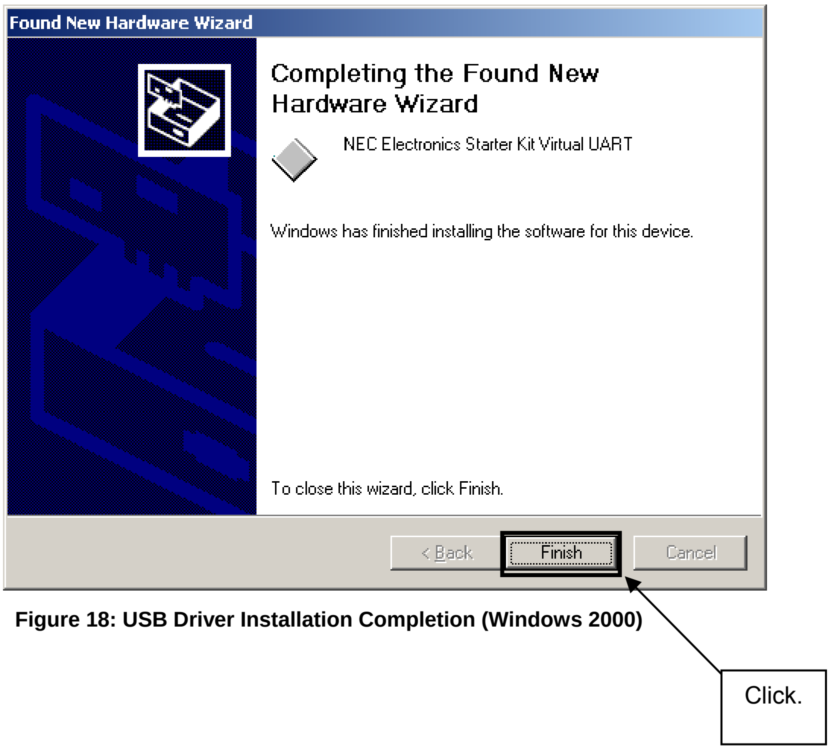


Figure 18: USB Driver Installation Completion (Windows 2000)

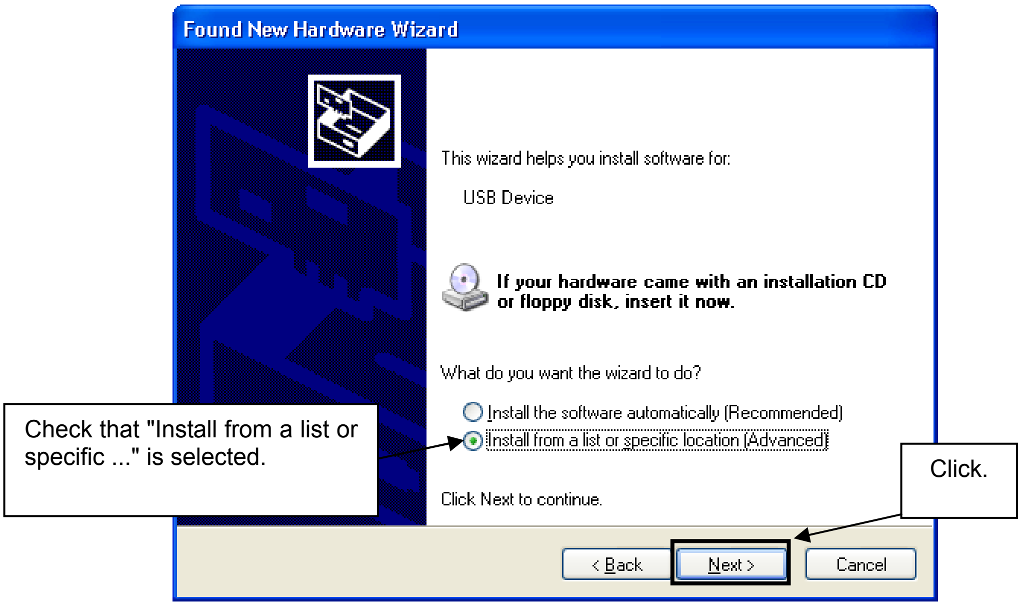
8.3.2 Installation on Windows XP

1. When the *TK-78F0730 board* is connected with the host machine, the board is recognized by Plug and Play, and the wizard for finding new hardware is started. At first the hardware wizard will ask if windows should search on the windows update web, check "No, not this time" and then click Next>.

Figure 19: Found New Hardware Wizard 1 (Windows XP)

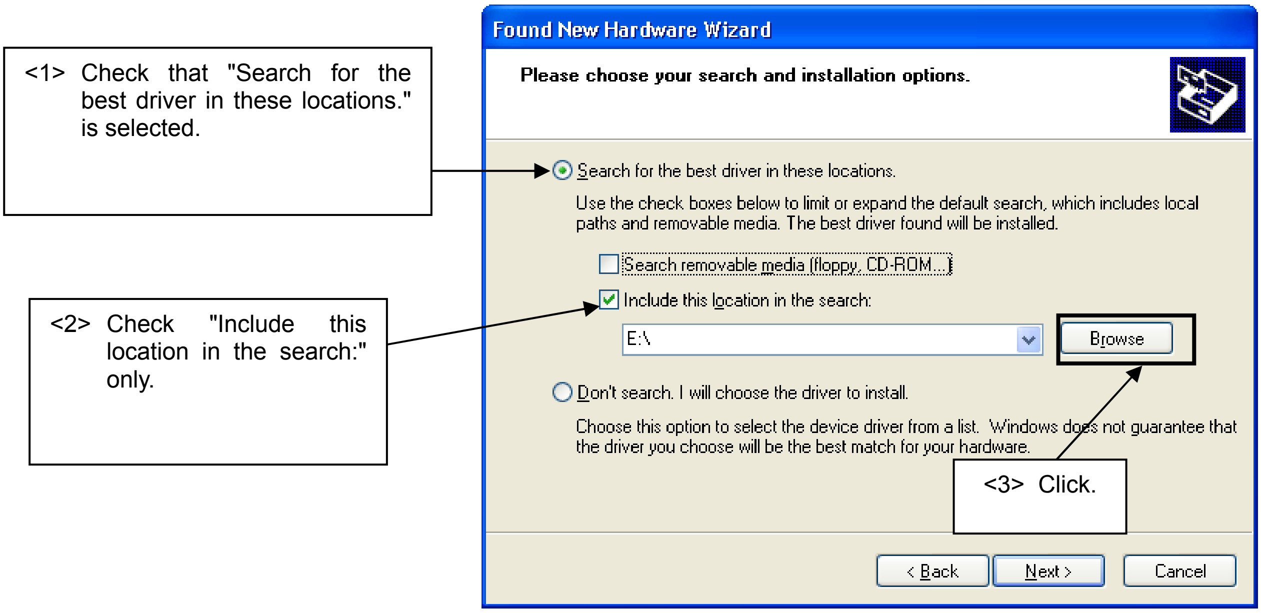
2. Check that "Install from a list or specific location (Advanced)" is selected, then click Next>.

Figure 20: Found New Hardware Wizard 2 (Windows XP)



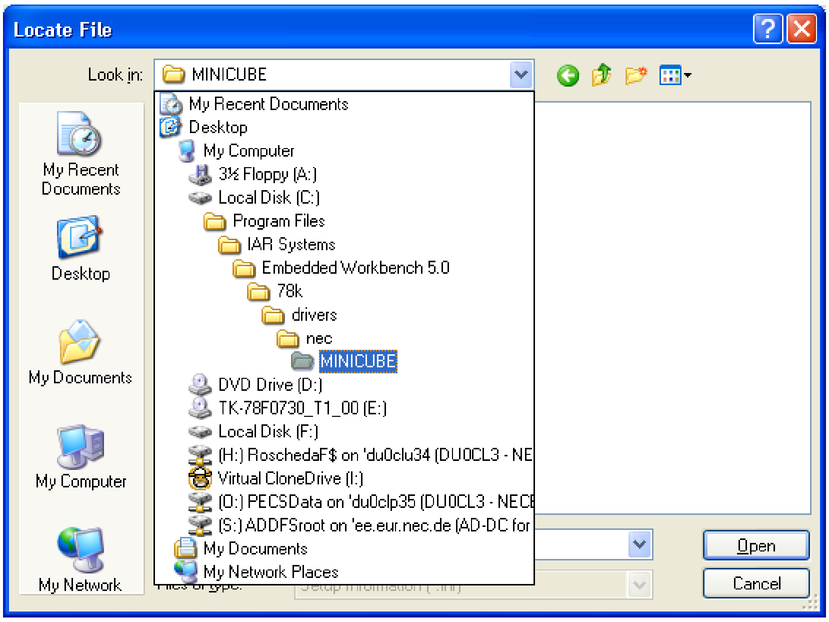
3. Check that "Search for the best driver in these locations." is selected. Select the "Include this location in the search:" check box and then click Browse.

Figure 21: Search Location Specification 1 (Windows XP)



4. Locate the folder "C:\Program Files\IAR Systems\Embedded Workbench 5.0\78K\drivers\nec\MINICUBE" and click OK.

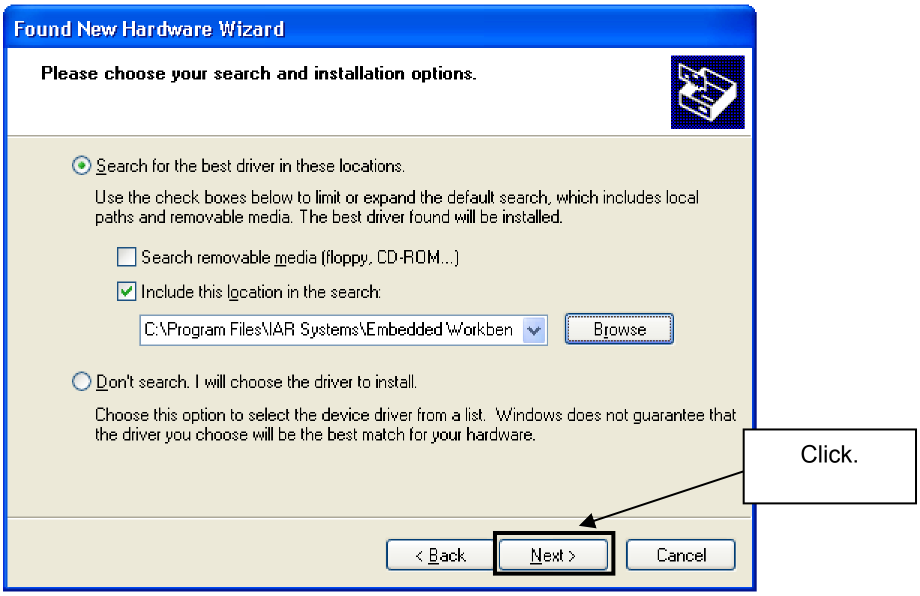
Figure 22: Search Location Specification 2 (Windows XP)



Remark If the installation destination folder is changed at the time of IAR Embedded Workbench installation, enter "new-folder\78K\drivers\NEC\\MINICUBE".

5. After the location of the USB driver has been specified click Next> to continue driver installation.

Figure 23: Search Location Specification 3 (Windows XP)

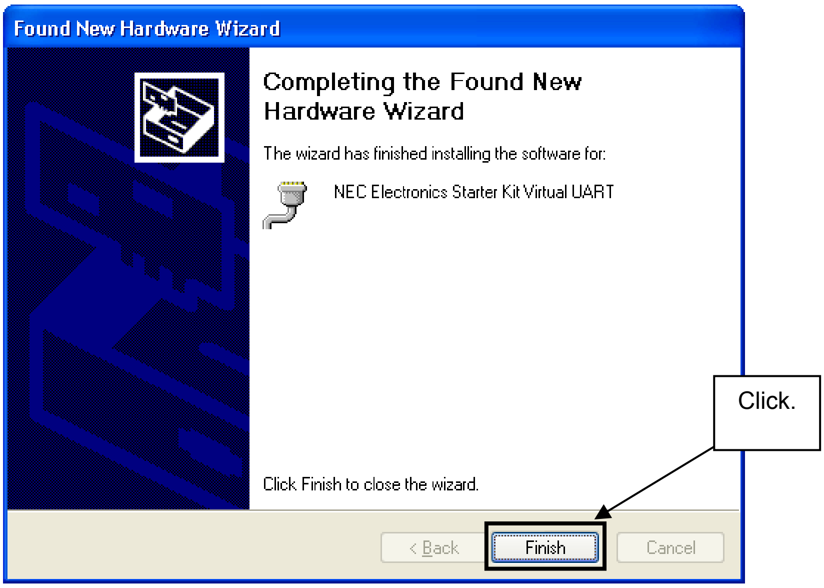


6. As shown below, "NEC Electronics Starter Kit Virtual UART has not passed Windows Logo testing to verify its compatibility with Windows XP." is displayed. Click Continue Anyway.

Figure 24: Windows XP Logo Testing (Windows XP)

7. After the installation of the USB driver is completed the window below is displayed. Click Finish to close the hardware wizard.

Figure 25: USB Driver Installation Completion (Windows XP)



8.4 Confirmation of USB Driver Installation

After installing the USB driver, check that the driver has been installed normally, according to the procedure below. When using the *TK-78F0730 board* in combination with IAR C-SPY debugger the “NEC Electronics Starter Kit Virtual UART” should be present like in the figure below.

Please check in the Windows "Device Manager" within the Windows Properties (“Hardware” tab), that the driver is installed normally.

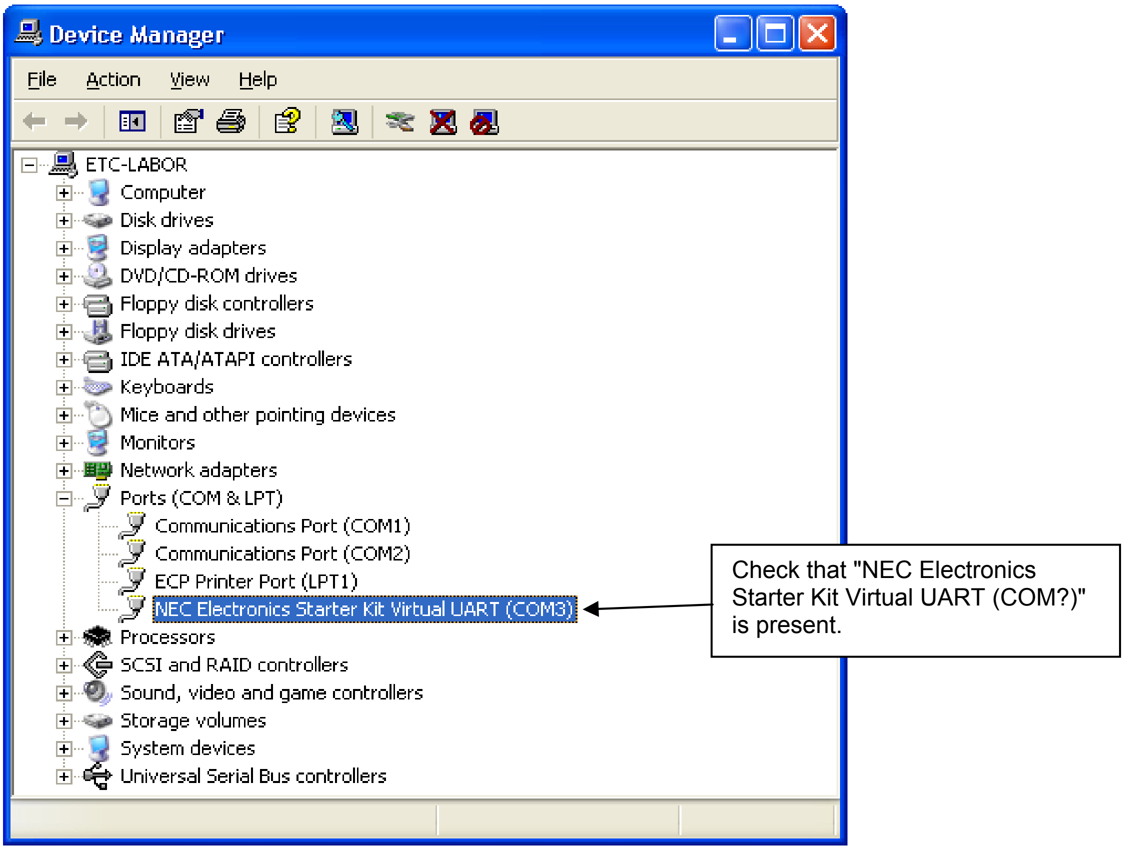


Figure 26: Windows Device Manager

9. IAR sample session

When everything is set up correctly the IAR Embedded Workbench can be started. To do so, start the Embedded Workbench from Windows “Start” menu > “Programs” > folder “IAR Systems” > “IAR Embedded Workbench Kickstart for 78K”. The following screen appears:

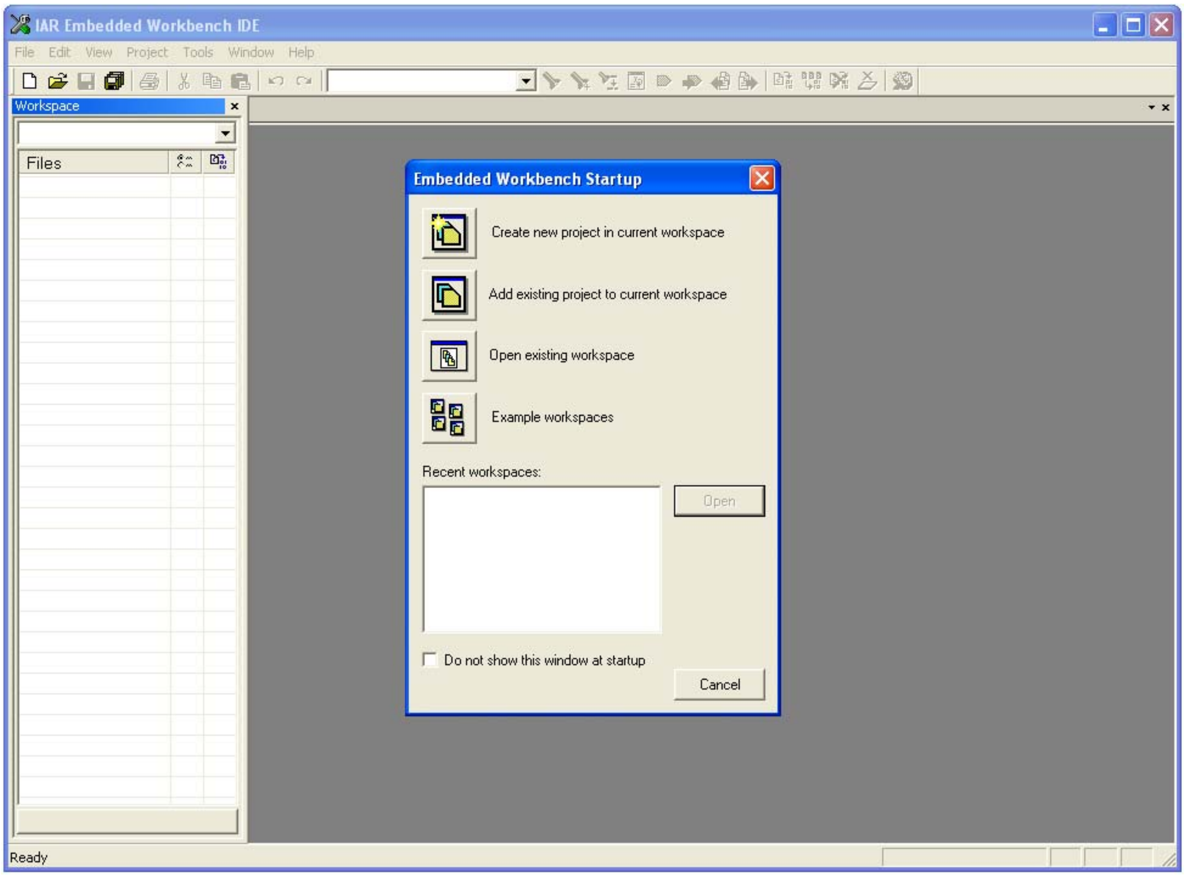


Figure 27: IAR Embedded Workbench

Now select the option “Open exiting workspace” from the “File” menu and locate the sample project folder and open the file “TK-78F0730_Samples.eww”. This is the workspace file that contains general information about all sample projects and corresponding settings.

After the sample workspace has been opened the projects included in the workspace are displayed. Please select the sample project ‘TK-78F0730_Demonstration’.The screen should now look similar to this one:

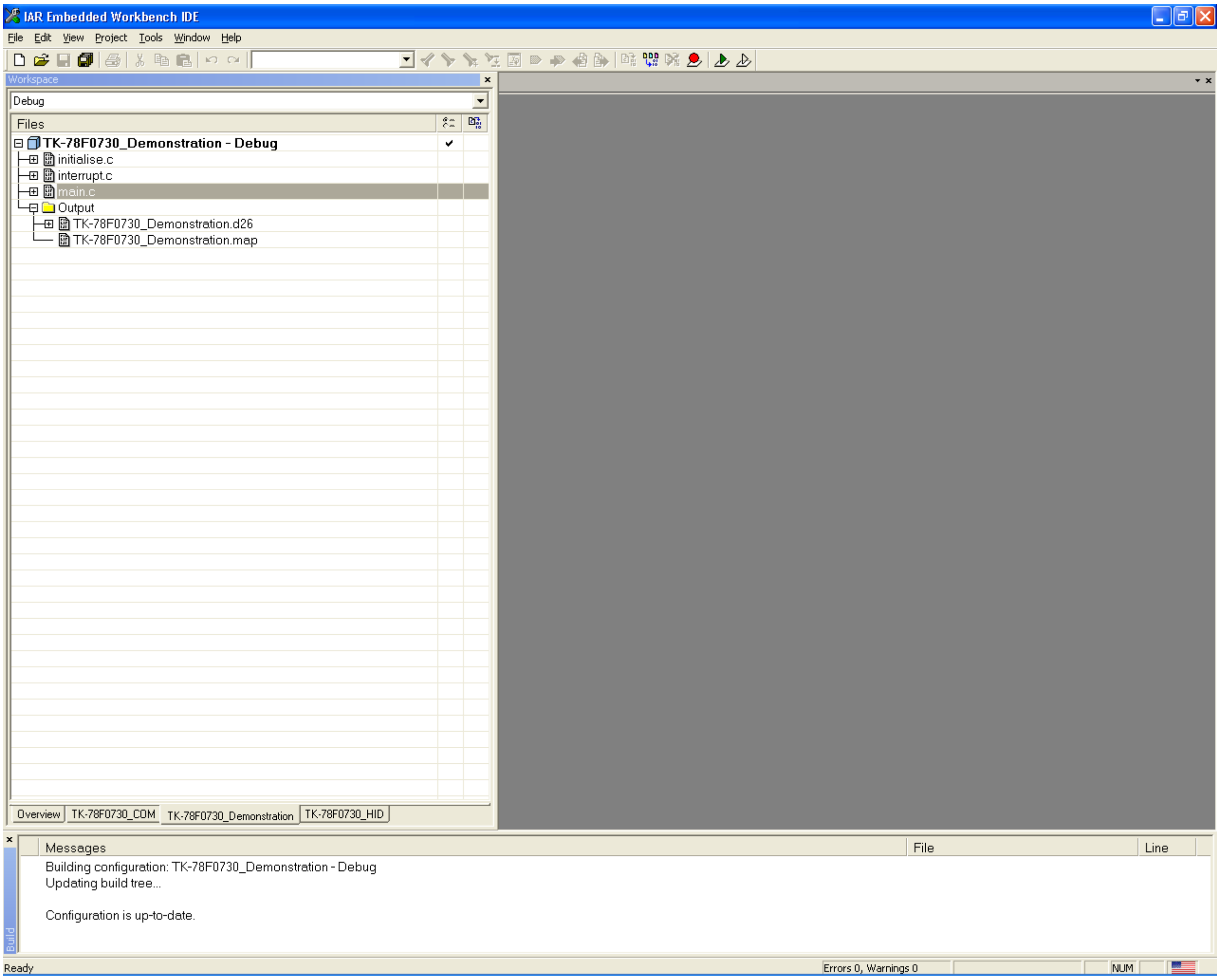


Figure 28: IAR Embedded Workbench Project Workspace

As a next step check some settings of the IAR Embedded Workbench that have to be made for correct operation and usage of the On-Board debug function of the *TK-78F0730 board*. First highlight the upper project folder called “TK-78F0730_Demonstration – Debug” in the workspace window. Then select “Project” > “Options” from the pull-down menus. Next select the category “Debugger”. Make sure that the driver is set to “MINICUBE” in order to use the On-Board debug function of the *TK-78F0730 board*. The device description file must be set to “io78f0730.ddf”. It is recommended to use the default file located in the folder “$TOOLKIT_DIR$\CONFIG\DDF\io78f0730.ddf”.

The corresponding COM port where the *TK-78F0730 board* is connected to the host PC will be detected automatically by the IAR C-SPY debugger.

Note: Although the On-board debug interface is used, the MINICUBE C-Spy driver must be selected instead of the standard driver TK-78K used for other starter kits. If the debug session via MINICUBE2 and the OCD1 connectors shall be started, also the MINICUBE C-Spy driver must be selected.

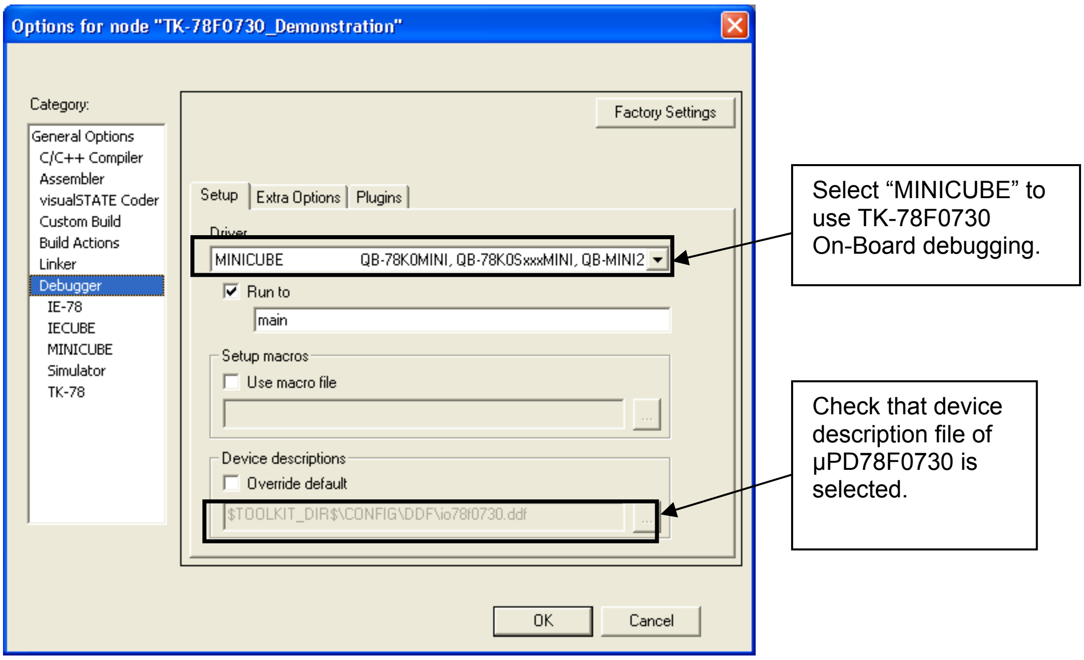


Figure 29: IAR debugger options

Next the correct linker settings of the demo project will be checked. This can be done in the “Linker” category as shown below. Select the “Config” tab and check that the linker command file “lnk78f0730.xcl” is selected. This file is used by the linker and contains information on where to place the different sections of code, data and constants that may be used within the demo project:

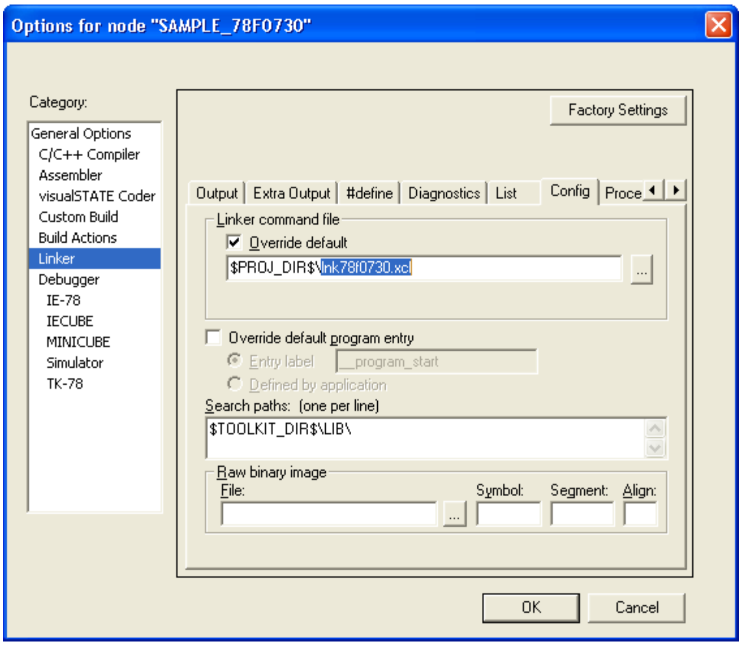


Figure 30: Embedded Workbench Linker Configuration

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 *TK-78F0730 board* and debugged.

To start the IAR C-SPY debugger select the option “Debug” from the “Project” menu or press the () “Debugger” button.

In the next step the TK-78 Emulator has to be configured before downloading a new application. Press the OK button to enter the emulator hardware setup. Set the configuration as show in the figure below and start the download by pressing the OK button.

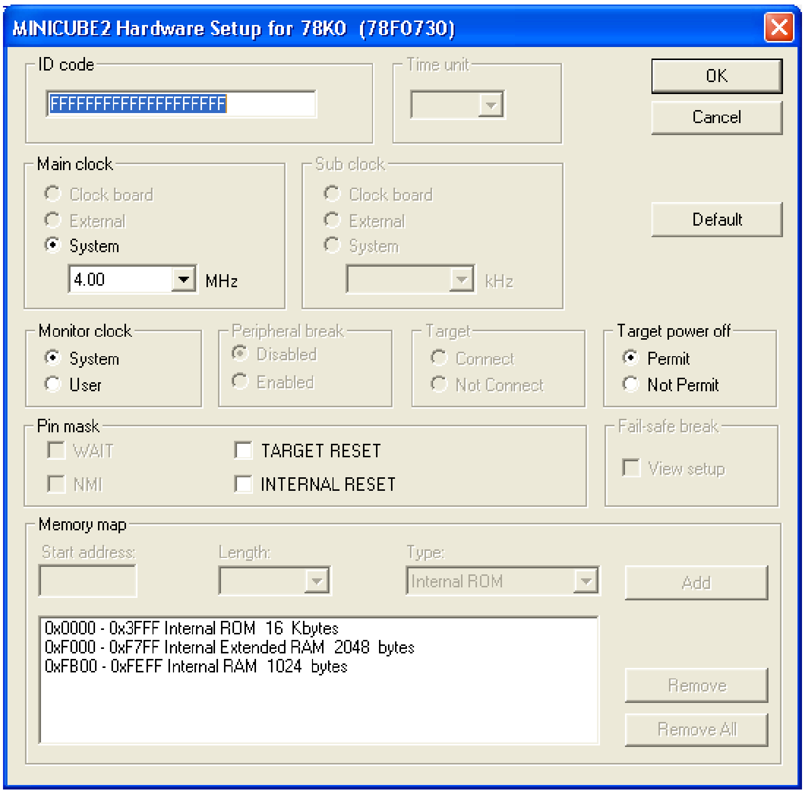


Figure 31: MINICUBE Hardware Setup Dialogue

Now the debugger is started and the demo project is downloaded to the *TK-78F0730 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.

After the download was completed all debug features of IAR C-SPY 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 “78K IAR Embedded Workbench IDE User Guide” of the IAR installation.

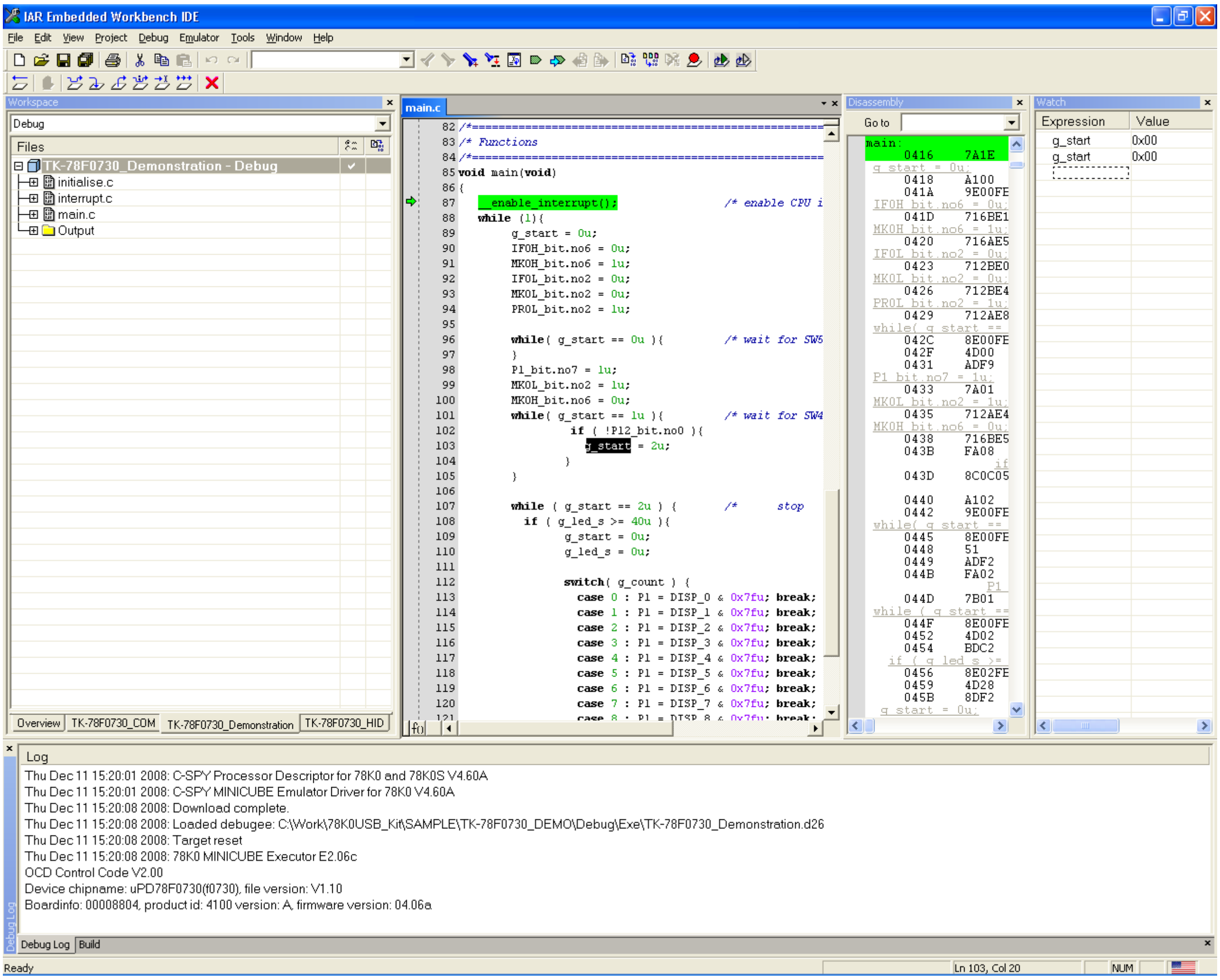


Figure 32: IAR C-SPY debugger

10. Troubleshooting

In some cases it might happen that the connection to the TK-78F0730 can not be established. This can be caused by the following two situations:

- ***Wrong security ID***: The security ID is required to prevent the FLASH memory of the 78F0730 microcontroller from being read by an unauthorized person. The security ID is located in the internal flash memory at addresses 0x0085-0x008E of the 78F0730 microcontroller. The IAR C-SPY debugger starts only when the security ID that is set during debugger start-up and the security ID set at addresses 0x0085 to 0x008E do match.

- ***Disabled On-Chip debug***: The On-Chip debug function of the 78F0730 microcontroller can be controlled by a dedicated Option Byte located at address 0x0084 in the internal flash memory. By disabling the On-Chip debug operation no connection to device can be established neither using the On-board debug interface nor using the MINICUBE2 emulator connector OCD1.

In the above mentioned cases it is necessary to erase the internal flash memory of the 78F0730 microcontroller to restore the security ID and to enable the On-Chip debug function.
Details about erasing and programming the internal flash memory of the 78F0730 are described in chapter 11 “Flash Programmer WriteEZ3”

11. Flash Programmer WriteEZ3

The flash programmer Write EZ3 doesn't need to be installed, but can be directly started from the CDROM.

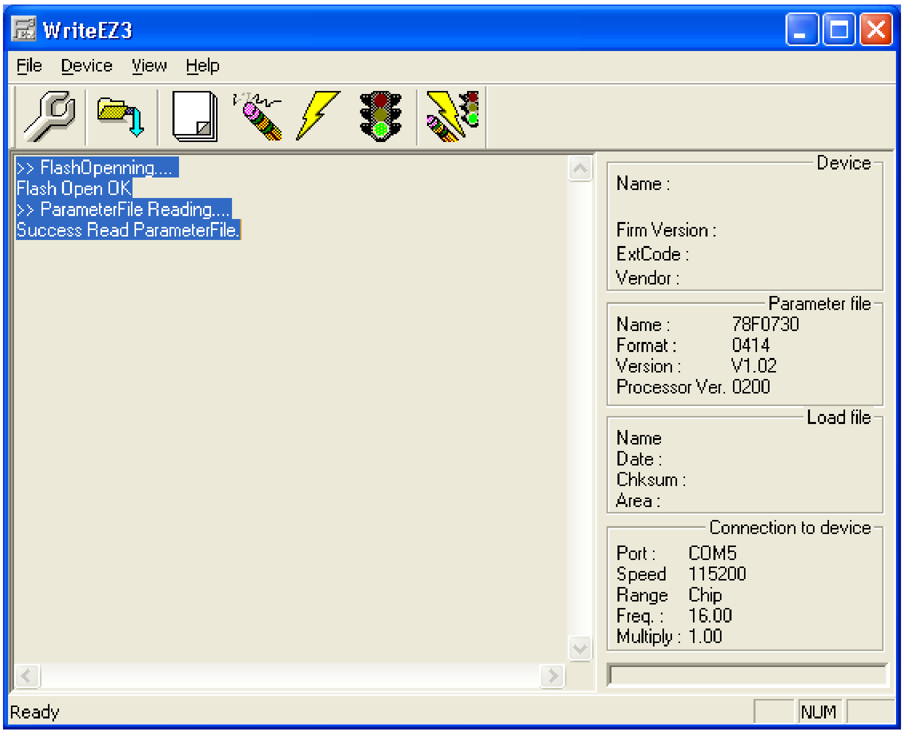


Figure 33: WriteEZ3 Startup

11.1 Device Setup

To provide all necessary information about the device to be programmed, only the corresponding flash parameter file must be loaded. The parameter file (*.prm) for the µPD78F0730 is located on the CDROM in the same folder as the WriteEZ tool. Please use the menu "Device -> Setup… " to open the following dialogue and the button "PRM File Read" to select the parameter file.

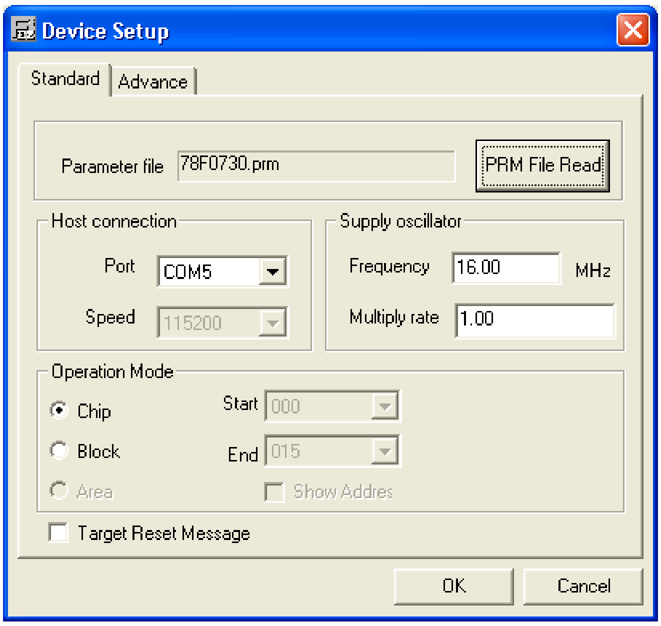


Figure 34: WriteEZ Device Setup Dialogue

Please check that the correct host communication port is selected. The used communication port can be seen in the Windows Device Manager.

11.2 Using WriteEZ

After a successful device selection the internal flash memory can be blank-checked, erased, programmed or verified. WriteEZ can be controlled either by menu or by buttons

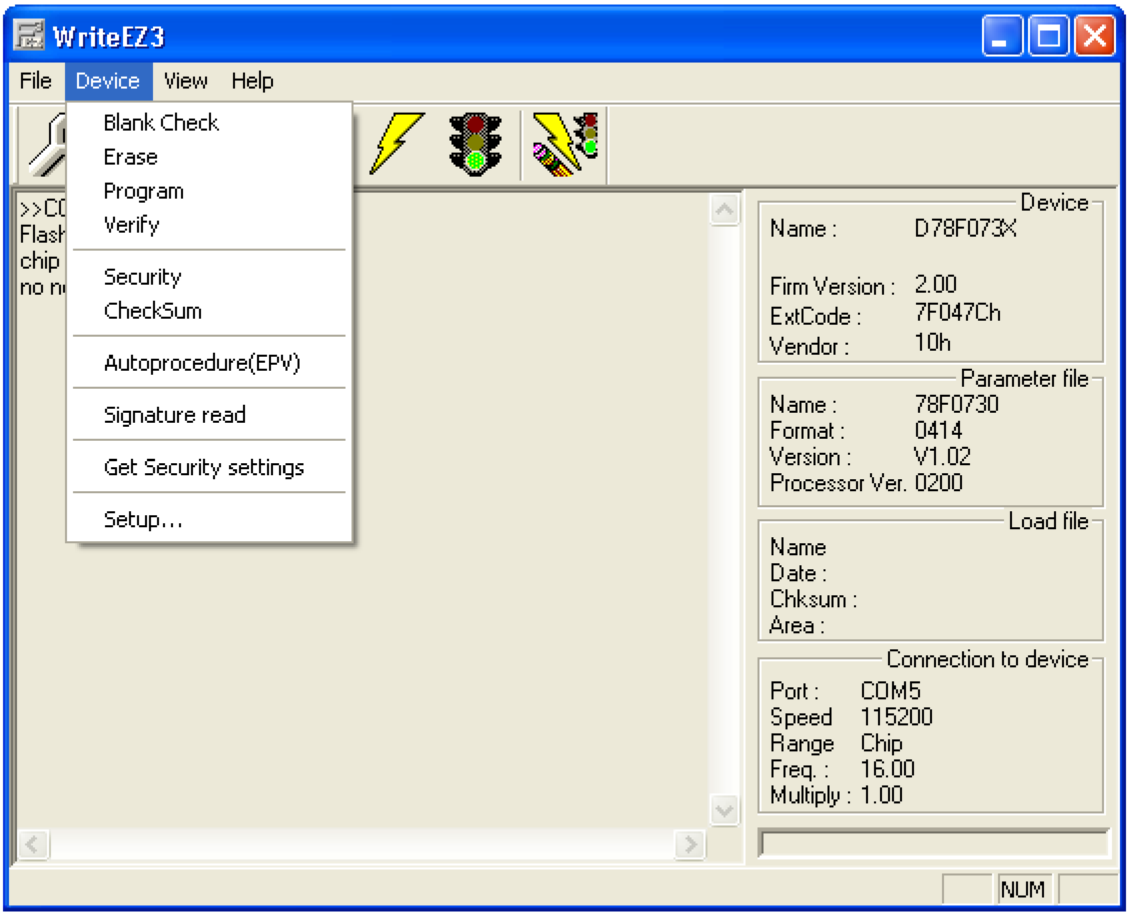


Figure 35: WriteEZ3 Device Menu

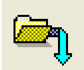
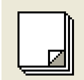
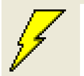
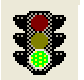
	device setup button
	load file button
	blank check button
	erase button
	program button
	verity button
	erase / program / verify button

Table 13: WriteEZ action buttons

WriteEZ3 supports Intel-Hex and Motorola S-record file formats as input file.

12. Sample programs

12.1 General Introduction

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. In the root directory of all sample programs the workspace file "**TK-78F0730_Samples.eww**" is located. The sample workspace includes all sample projects.

settings	Workspace configuration files, IAR Embedded Workbench
TK-78F0730_COM	Serial Conversion sample project
TK-78F0730_DEMO	Demonstration sample project
TK-78F0730_HID	Human Interface Device sample project
TK-78F0730_Demonstration.ewp	Workspace file, IAR Embedded Workbench

Table 14: Sample directory structure

As an alternative to open the sample-workspace each project file "<name>.ewp" can be added to any user created workspace.

All sample programs use the same directory structure:

TK-78F0730_COM	Serial Conversion sample project
TK-78F0730_DEMO	Demonstration sample project
Debug	debug output files for IAR C-SPY debugger
inc	C header files
settings	configuration files, IAR Embedded Workbench
source	C source files
xcl	Linker control file
TK-78F0730_Demonstration.dep	dependency information file, IAR Embedded Workbench
TK-78F0730_Demonstration.ewd	project setting file, IAR C-SPY debugger
TK-78F0730_Demonstration.ewp	project file, IAR Embedded Workbench
TK-78F0730_HID	Human Interface Device sample project

Table 15: Example directory structure

The main directory contains only the project files for the IAR Systems Embedded Workbench for 78K. All source files are located in the subdirectory `/source`. The `/include` subdirectory contains the header files. The `/xcl` subdirectory contains the linker control file of the 78F0730 device. All output files including the object files, list files, debug information and finally the executable file are stored in the directory `/Debug`.

For details of using the IAR Embedded Workbench and the IAR C-SPY debugger please refer to the "78K IAR Embedded Workbench IDE User Guide".

12.2 "TK-78F0730_DEMO" sample program

This sample program is a simple random number generator between 0 and 9 to demonstrate the usage of the TK-78F0730 starterkit. After starting the generator by pressing SW5 the segments of the seven-segment-LED are flashing until a random number is generated by pressing SW4. The generated number is displayed at the seven-segment-LED U8. To start a new generation loop, please press again SW5

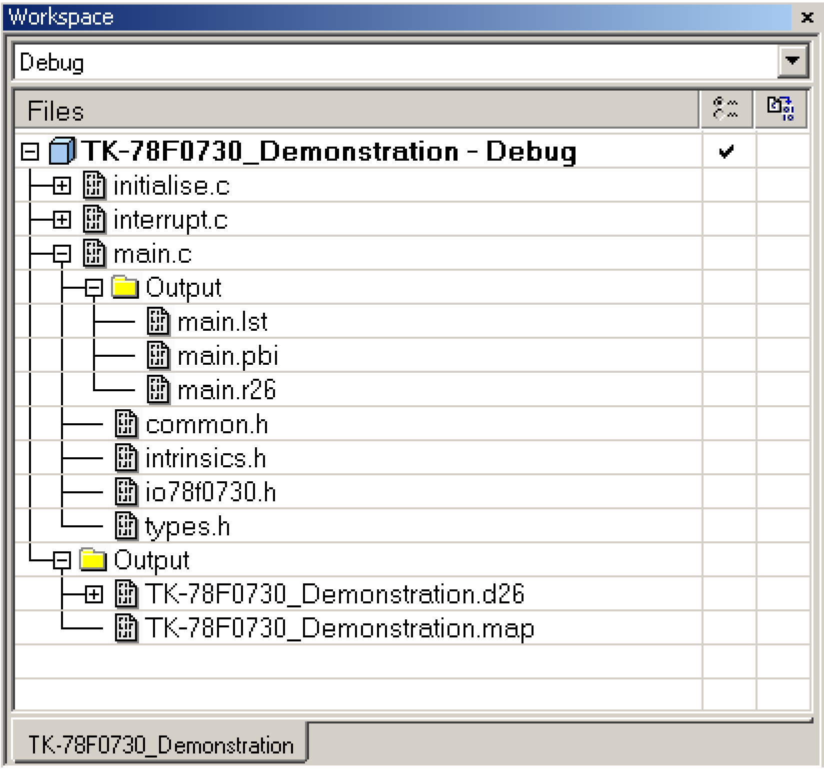


Figure 36: Project Window TK-78F0730_DEMO

The flowchart of the "TK-78F0730_DEMO" is given at the Figure 37.

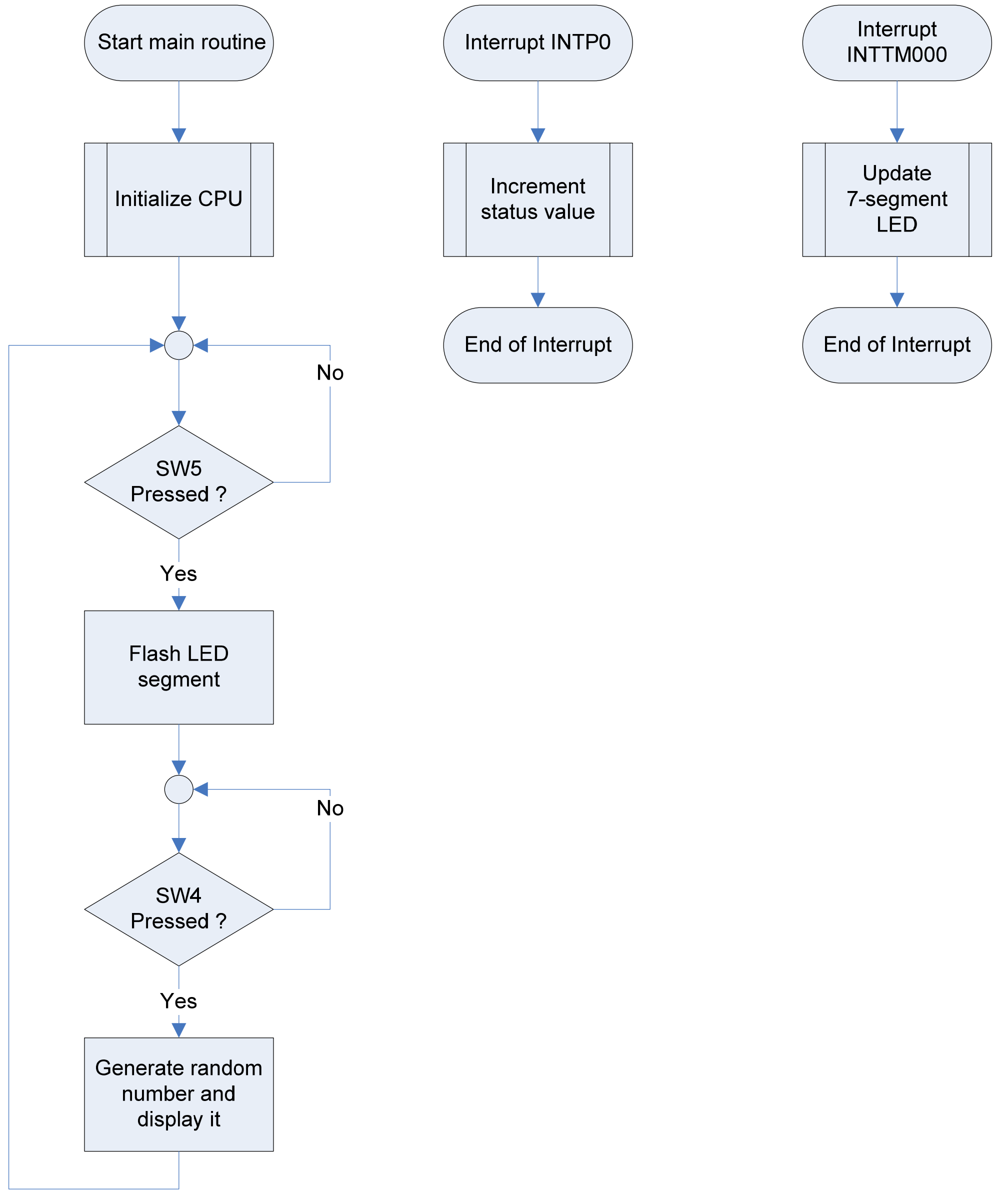


Figure 37: Flowchart TK-78F0730_DEMO

12.2.1 How to run the sample program

After starting the application, the switches SW4, SW5 and the seven-segment LED are the user interface to the sample program.

- Switch SW5

  By pressing this switch, the generation of a new random number is started. In this state the segments of the seven-segment display are flashing.

- Switch SW4

  By pressing this switch, the generation of a new random number is stopped. The newly generated number is displayed at the seven-segment display.

12.3 "TK-78F0730_HID" sample program

This program is a sample driver for the μPF78D0730 acting as human interface device class keyboard. The main functions of the sample driver are as follows:

- Pseudo HID class keyboard. For host machine, it is detected as HID keyboard device.
- Since μPD78F0730 does not support interrupt endpoint, it uses the bulk endpoint for interrupt endpoint.
- Full speed (12 Mbps) device.
- SW4 and SW5 of TK-78F0730 board are used as for key inputs:
  - o Every time SW4 is pressed, it outputs the ASCII code for the alphabet "a" to "z" by toggle action.
  - o Every time SW5 is pressed, it outputs the key code of "Enter".
- Bus powered device.
- Not boot device.
- Remote wake-up function is not supported.

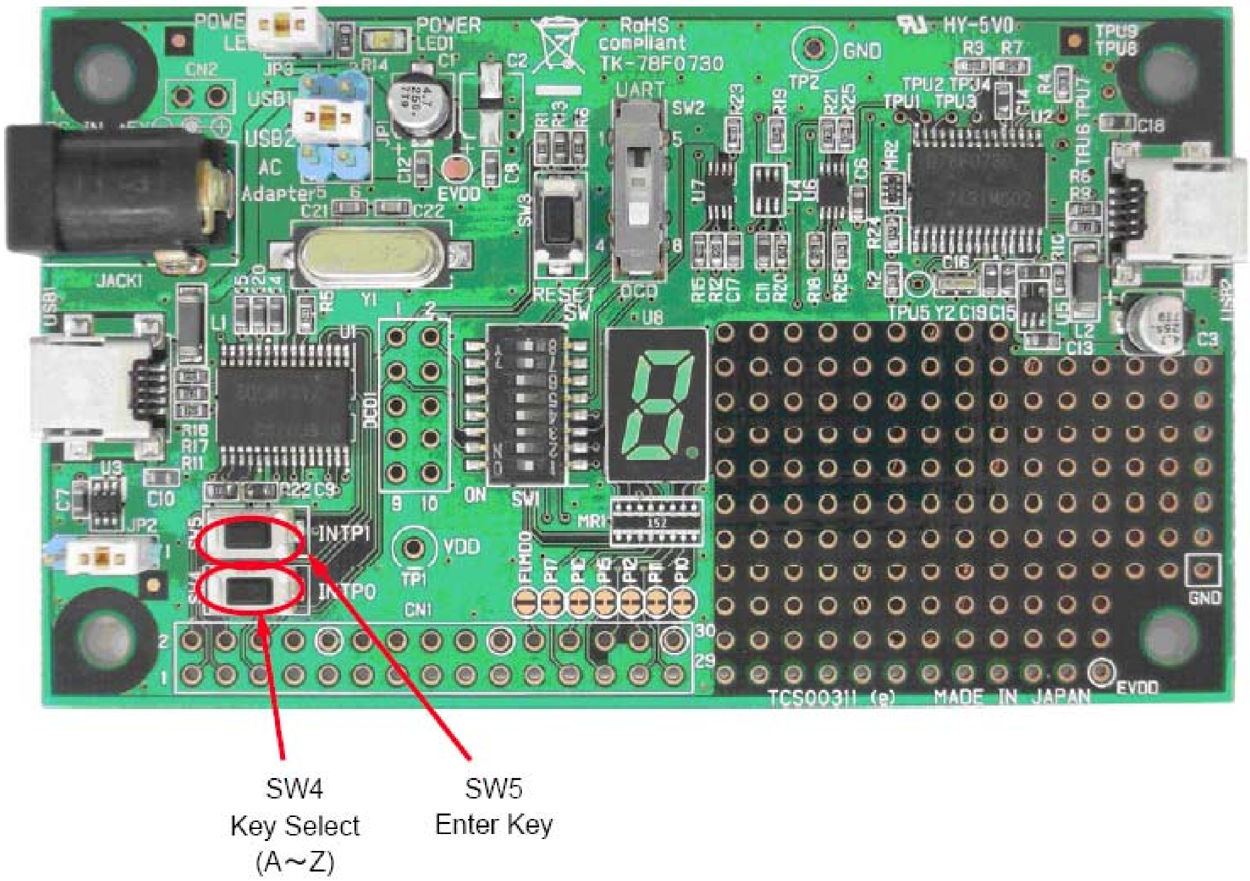


Figure 38: User-Interface HID sample driver

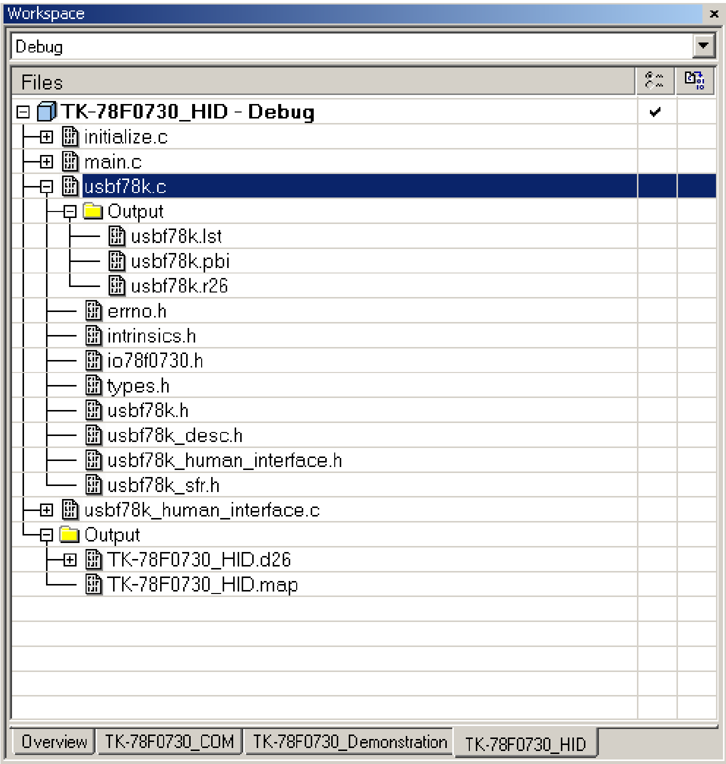


Figure 39: Project Window TK-78F0730_HID

The flowchart of the "TK-78F0730_HID" application is given at the Figure 40.

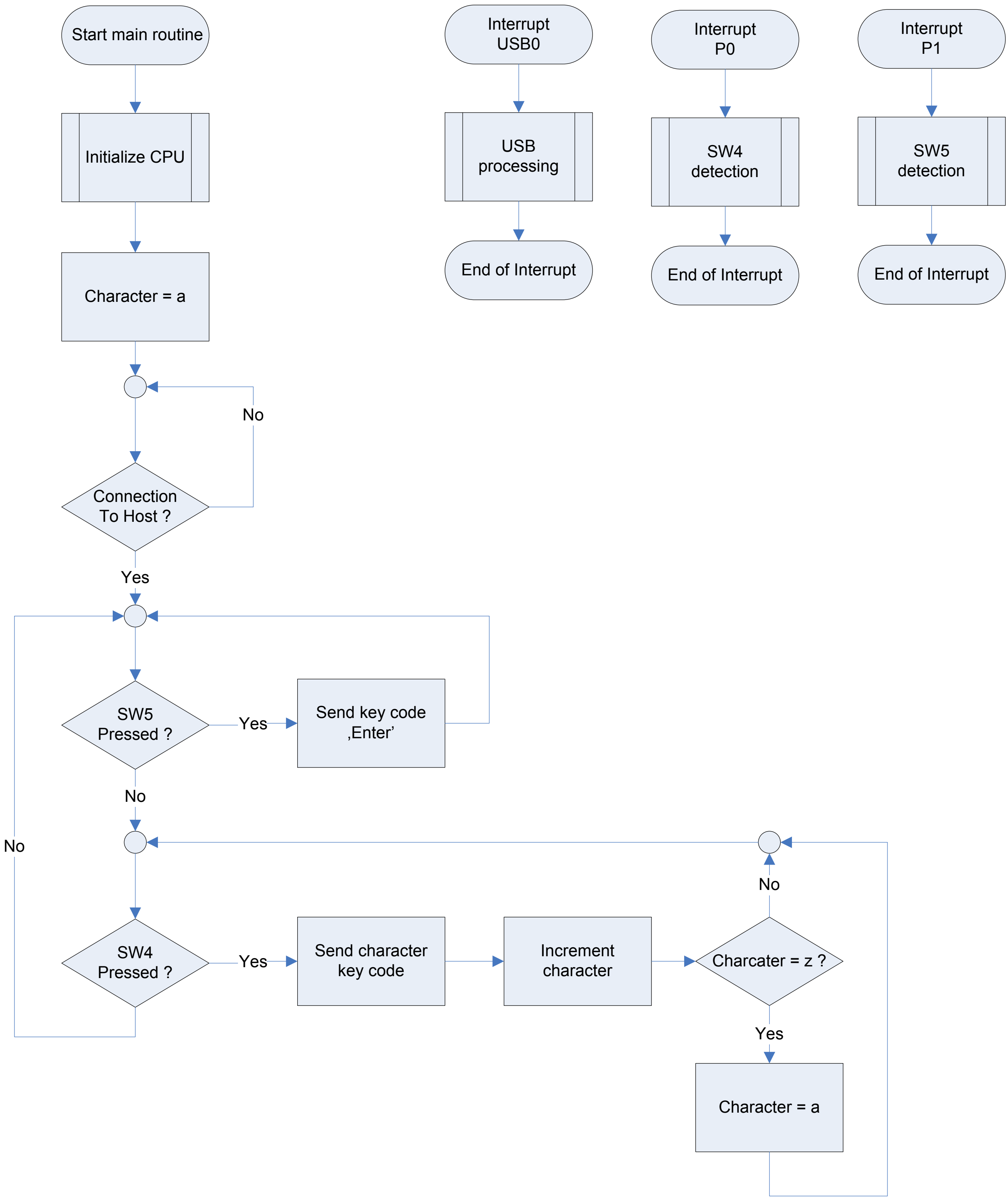


Figure 40: Flowchart TK-78F0730_HID

12.3.1 How to run the sample program

Start the HID sample first, before connecting the starterkit to a host PC via USB connector USB1. Under the above conditions a Windows standard driver is automatically used. It is detected as "HID Keyboard Device".

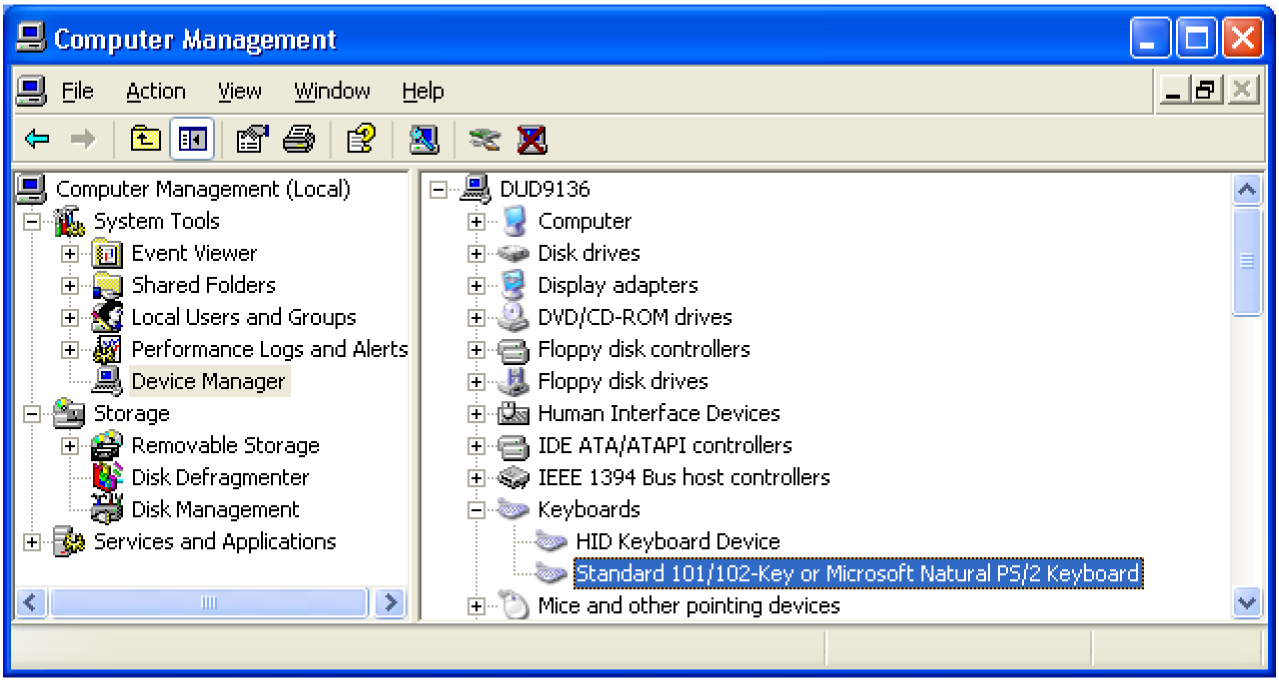


Figure 41: Windows Device Manager HID Sample (new keyboard device)

To display the data entered by pressing SW4 and SW5, please open any ASCII editor.

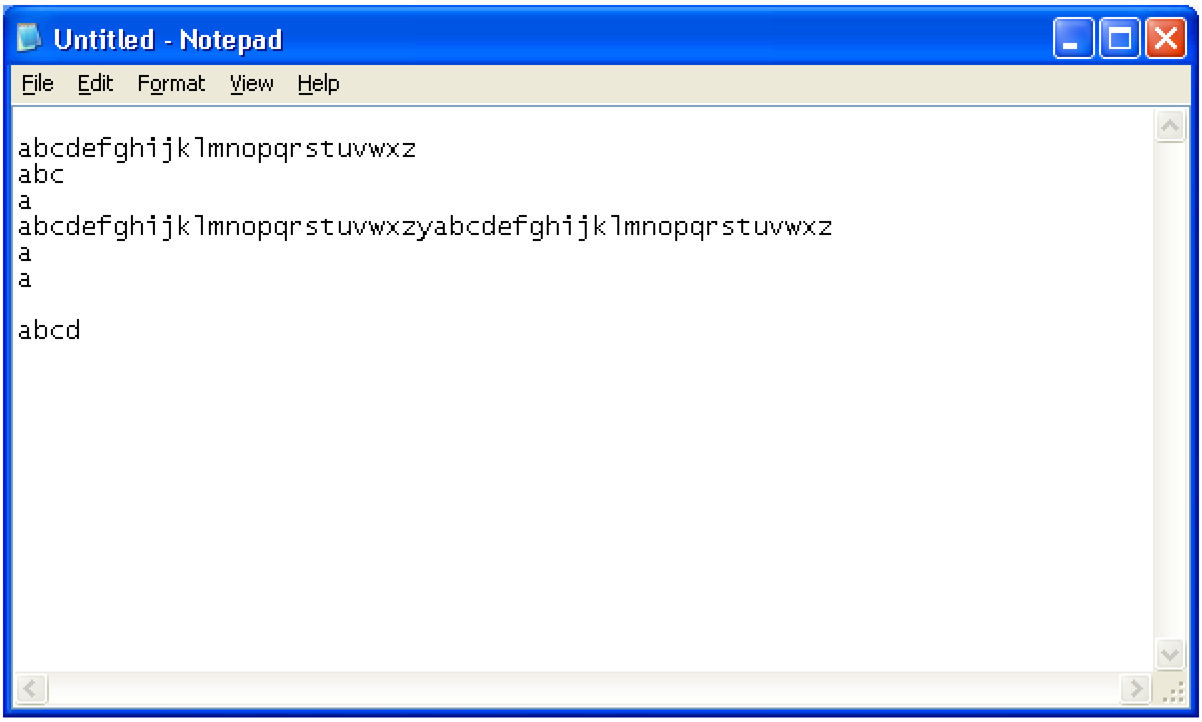


Figure 42: Output HID sample

12.3.2 Development Environment

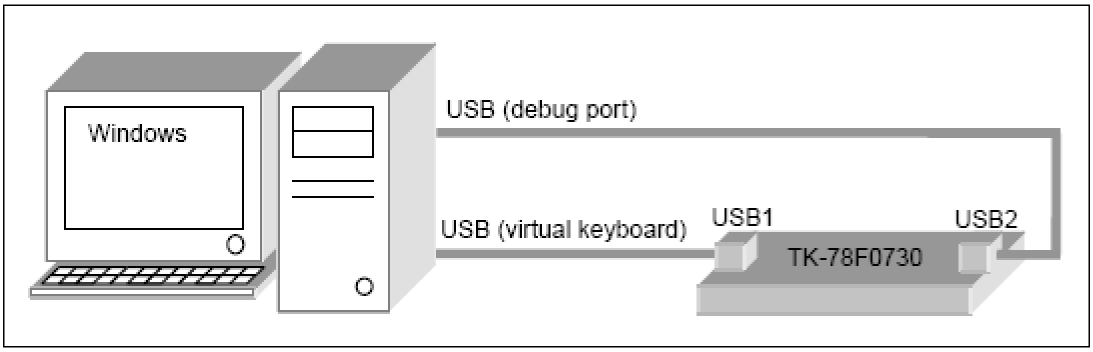


Figure 43: Development Environment HID sample driver

12.3.3 Structure of USB HID class driver

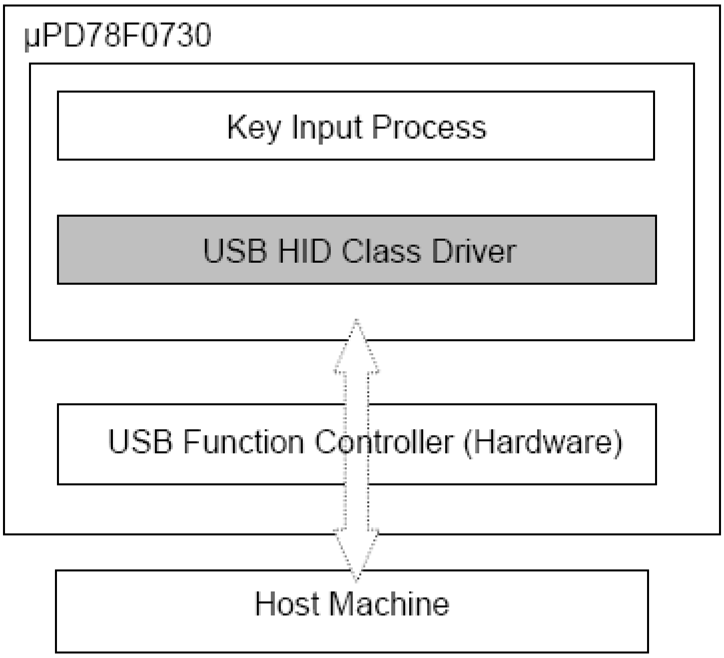


Figure 44: Structure of USB HID class driver

12.3.4 File Structure

File	Description
main.c	Main routine
initialize.c	cpu and board initialization
usbf78k.c	USB related register initialization. Endpoint control. Bulk transfer. Control transfer.
usbf78k_human_interface.c	HID class process

Table 16: Source Modules USB HID Sample

File	Description
main.h	Function prototypes defined in main.c
errno.h	Error code definitions
types.h	Datatype definitions
usbf78k.h	Macro definitions for USB function register setting
usbf78k_desc.h	Descriptor definitions
usbf78k_sfr.h	Macro definitions for USB function register access
usbf78k_human_interface.h	HID class function prototype declaration

Table 17: Header Files USB HID Sample

12.3.5 USB Human Interface Device

For the information about USB HID, please refer to the HID specification "Device Class Definition for Human Interface Devices (HID) Specification Version 1.11".
The sample driver supports following class requests:

- Get_Report: Request to retrieve data from device using control endpoint.
- Get_Idle: Request to retrieve the device idle rate.
- Set_Idle: Request to set the device idle rate.
- Report Descriptor: Special HID class descriptor. It defines the data communication type between host and device.
- HID Descriptor: Special HID class descriptor. It includes description and size of the HID.

12.3.6 Descriptor Information

Field	Size	Description	Value
bLength	1	Descriptor size	0x12
bDescriptor	1	Descriptor type	0x01
bcdUSB	2	BCD format of USB release number	0x0200
bDeviceClass	1	Class code 0x00H: no class 0xFFH: vendor 0x01-0xFEH: specific	0x00
bDeviceSubClass	1	Sub-class code	0x00
bDeviceProtocol	1	Protocol code 0x00: no specific protocol 0xFF: vendor-specific protocol	0x00
bMaxPacketSize0	1	Maximum packet size at endpoint0	0x40
idVendor	2	Vendor ID (USB IF assigns)	0x0409
idProduct	2	Product ID (vendor assigns)	0x01CE
bcdDevice	2	BCD format of device release number	0x0001
iManufacture	1	Index to string descriptor to indicate manufacturer	0x01
iProduct	1	Index to string descriptor to indicate product	0x02
iSerialNumber	1	Index to string descriptor to indicate serial number	0x03
bNumConfigurations	1	Number of devices that can be configured	0x01

Table 18: Device Descriptor HID sample

Field	Size	Description	Value
bLength	1	Descriptor size	0x09
bDescriptor	1	Descriptor type	0x02
wTotalLength	2	Total length of the configuration (configuration, interface, endpoint, and other descriptors)	0x0022
bNumInterfaces	1	Number of interfaces supported in the configuration	0x01
bConfigurationValue	1	Input value (>= 1) for selecting this configuration with SetConfiguration	0x01
iConfiguration	1	Index to string descriptor to indicate descriptor	0x00
bmAttributes	1	Configuration attributes with the unit of bit D7: "1" D6: self-powered D5: remote wake-up D4-D0: reserved (0)	0x80
bMaxPower	1	Maximum power consumption of bus with the unit of 2mA	0x32

Table 19: Configuration Descriptor HID sample

Field	Size	Description	Value
bLength	1	Descriptor size	0x09
bDescriptor	1	Descriptor type	0x04
bInterfaceNumber	1	Index Number (0 based) to indicate this interface in the configuration	0x00
bAlternateSetting	1	Input value to select alternate setting in SetInterface	0x00
bNumEndpoints	1	Interface endpoint number (excluding endpoint 0)	0x01
bInterfaceClass	1	Class Code 0x00: no class 0xFF: vendor 0x01 – 0xFE: specific	0x03
bInterfaceSubclass	1	Subclass code	0x00
bInterfaceProtocol	1	Protocol code 0x00: no specific protocol 0xFF: vendor specific protocol	0x00
iInterface	1	Index to string descriptor to indicate interface	0x00

Table 20: Interface Descriptor HID Sample

Field	Size	Description	Value
bLength	1	Descriptor size	0x09
bDescriptor	1	Descriptor type	0x21
bcdHID	2	BCD format of compliant HID class revision	0x0110
bCountryCode	1	Hardware target country code	0x00
bNumDescriptor	1	Number of class definition descriptors that are depended from this HID	0x01
bDescriptor	1	Type of the first depending class definition descriptor	0x22
bDescriptorLength	2	Size of the first depending class definition descriptor	0x0041

Table 21: Human Interface Device Descriptor

Field	Size	Description	Value
bLength	1	Descriptor size	0x07
bDescriptor	1	Descriptor type	0x05
bEndpointAddress	1	Endpoint address bits: D7: Direction 0: OUT, 1: IN D6-D4: Reserved (0) D4-D0: Endpoint number	0x81
bmAttributes	1	Attribute bits: D1-D0: Transfer type 0: Control 1: Isochronous 2: Bulk 3: Interrupt *D5-D2 is used only by isochronous endpoint D3-D2: Synchronization type 0: No synchronization 1: Asynchronous 2: Adaptive 3: Synchronous D5-D4: Usage type 0: Data endpoint 1: Feedback endpoint 2: Dependant feedback endpoint 3: (reserved)	0x03
wMaxPacketSize	2	Payload size bits: D10-D0: Maximum packet size D12-D11: auditory transaction number per µ frame (only high-speed isochronous and interrupt) 0: No addition (1 transaction / µ frame) 1: 1 (2 transaction / µ frame) 2: 2 (3 transaction / µ frame) 3: Not in use (reserved)	0x0040
bInterval	1	Polling interval for data transfer endpoint Full/low speed interrupt: specify with unit of ms (number of frames) High-speed isochronous/interrupt: specify N for 2 raised to the power of N-1 with unit of µ frame (for example, 1 polling in 8µ frames when bInterval is 4) Full-speed isochronous: specify N for 2 raised to the power of N-1 with unit of 1ms High-speed bulk/control: specify the maximum NAK rate for endpoint with unit of µ frame 0 means that it does not respond NAK on OUT/DATA transaction	0x0A

Table 22: Endpoint Descriptor HID Sample

Field	Size	Description	Value
bLength	1	Descriptor size	0x07
bDescriptor	1	Descriptor type	0x05
bString	41	Language Code: 0x09 0x04	
		Manufacture: “NEC Electronics Co.”	
		Product: “HIDDrv”	
		Serial Number: “0_98765432”	

Table 23: String Descriptor HID Sample

Data	Item
0x05, 0x01	Usage Page(Generic Desktop)
0x09, 0x06	Usage(Keyboard)
0xA1, 0x01	Collection(Application)
0x05, 0x07	Usage Page(Keyboard)
0x19, 0xE0	Usage Minimum(LEFT CTRL)
0x29, 0xE7	Usage Maximum(RIGHT GUI)
0x15, 0x00	Logical Minimum(0)
0x25, 0x01	Logical Maximum(1)
0x75, 0x01	Report Size(1)
0x95, 0x08	Report Count(8)
0x81, 0x02	Input(Variable)
0x95, 0x01	Report Count(1)
0x75, 0x08	Report Size(8)
0x81, 0x01	Input(Constant)
0x95, 0x06	Report Count(6)
0x75, 0x08	Report Size(8)
0x15, 0x00	Logical Minimum(0)
0x26, 0xFF, 0x00	Logical Maximum(255)
0x05, 0x07	Usage Page(Keyboard)
0x19, 0x00	Usage Minimum(0)
0x29, 0x91	Usage Maximum(145)
0x81, 0x00	Input
0xC0	End Collection

Table 24: Report Descriptor HID Sample

12.4 “TK-78F0730_COM” Serial Conversion sample program

USB serial conversion driver is a sample driver for USB function controller that is built in μPD78F0730.
It is compliant with the Universal Serial Bus Specification and its class is the vendor class.
The sample driver uses the control endpoint (endpoint 0), IN and OUT of bulk endpoint (endpoint 1, 2).
Then, it connects with the host driver (vendor class) for USB serial conversion, and functions as virtual COM port. The communication data received by USB function controller is loaded in the sample application, converted to uppercase characters or to lowercase characters (only ASCII character data), and sent from USB. It is also possible baud rate, stop bit, data length, and parity bit from terminal software, but this sample does not use this settings.

* When you use the host drivers for Windows standard communication class, you need 4 endpoints such as control endpoint, IN and OUT of bulk endpoint, and interrupt endpoint. Therefore, the host driver for Windows standard communication class cannot be used, as μPD78F0730 has only 3 endpoints.

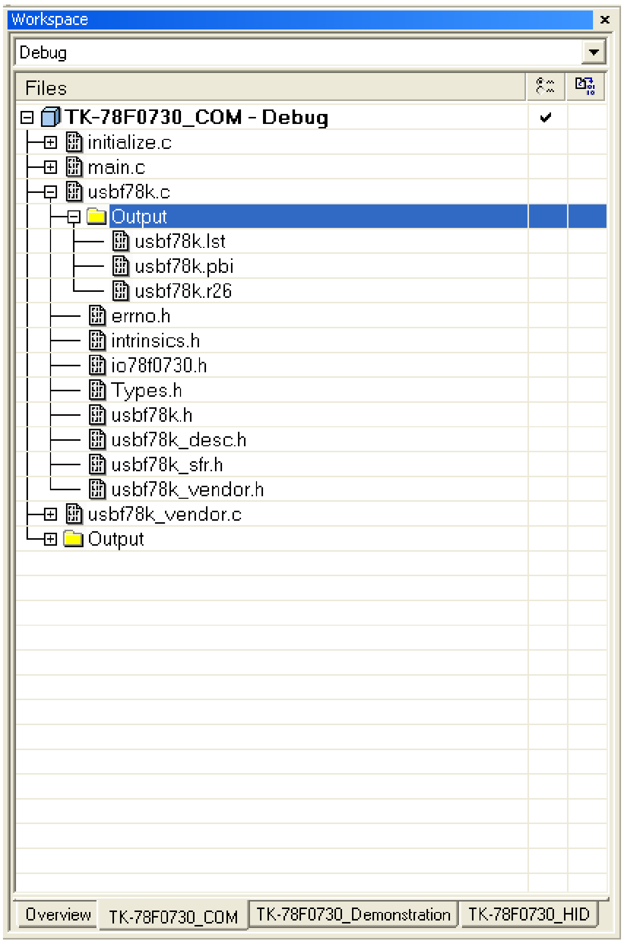
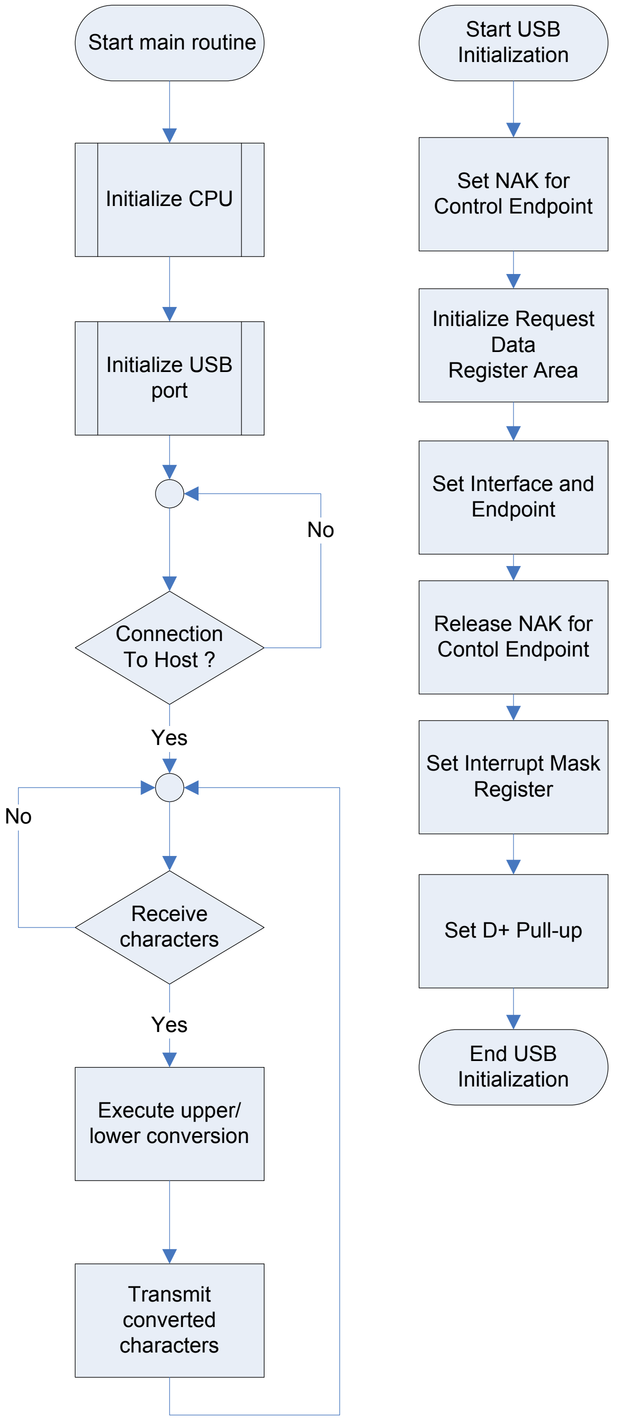


Figure 45: Project Window TK-78F0730_COM

The flowcharts of the “TK-78F0730_COM” sample application are given in the following figures.



Process Description USB Initialization

o Set NAK for Control Endpoint
A NAK response is sent to all the requests including automatic execution requests.
It sets for hardware not to return unexpected data in response to an automatic execution request until registration of data used for the automatic execution request is complete.

o Initialize Request Data Register Area
Descriptor data used to respond to a "Get Descriptor" request is registered in a register. Those data include device status, endpoint 0 status, device descriptor, configuration descriptor, interface descriptor, and endpoint descriptor.

o Set Interface and Endpoint
Set the number of supported interfaces, the status of alternative settings, and the relationship between the interface and endpoints.

o Release NAK for Control Endpoint
The NAK setting at control endpoint is released during registration of data for an automatic execution request is complete.

o Set Interrupt Mask Register
Set the mask for each interrupt source indicated in the interrupt status register of the USB function controller.

o Set D+ Pull-up
Pull-up the D+ signal, and let the host recognize a device is connected.

Figure 46: Flow Chart TK-78F0730_COM application

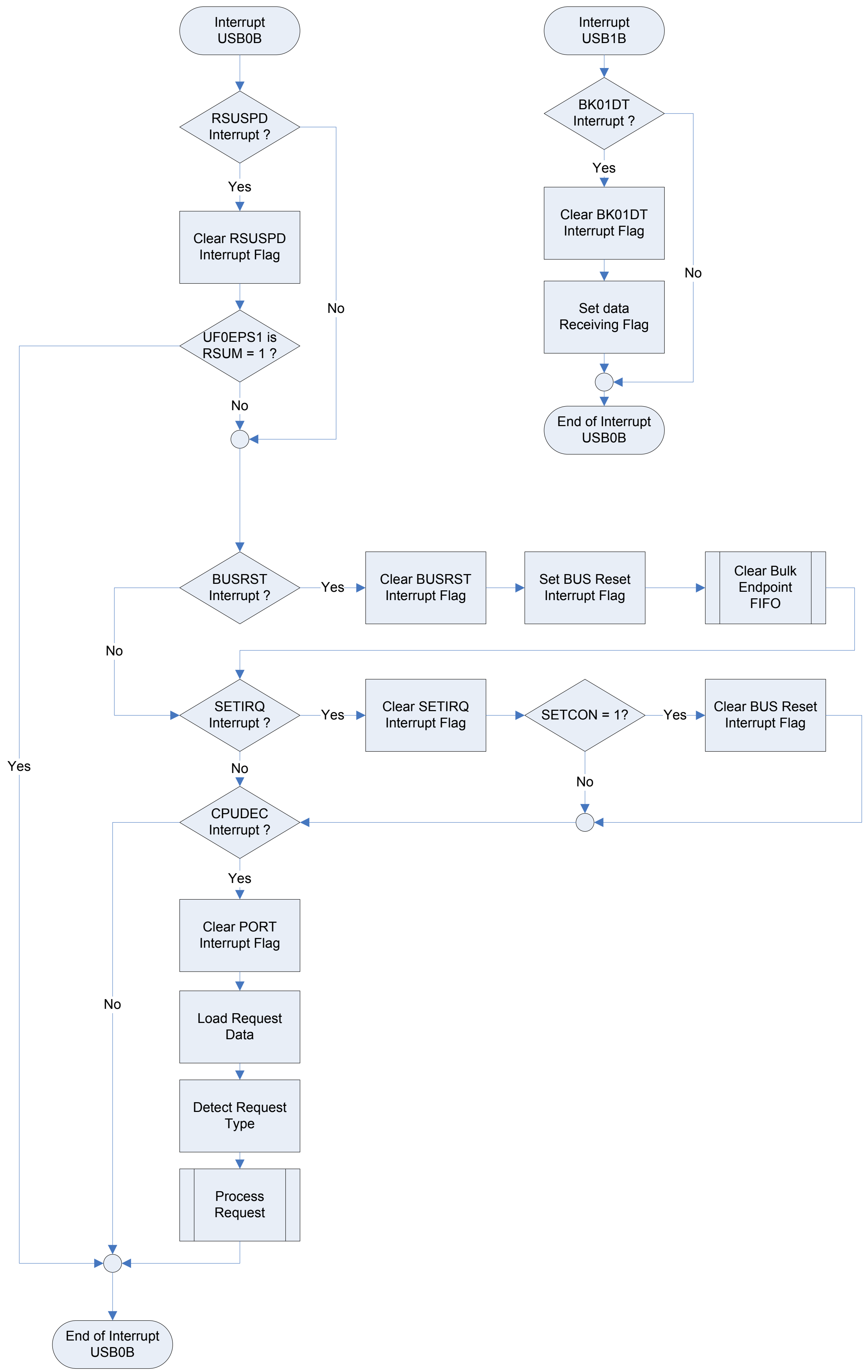


Figure 47: Flow Chart TK-78F0730_COM Interrupts

Process Description Interrupt USB0B

The interrupt USB0B interrupt handler mainly processes CPUDEC interruption.

o Detect RSUSPD Interrupt
It detects the interrupt if RSUSPD bit of UF0IS0 is ON (1).

o Clear RSUSPD Interrupt
Clear the cause of interrupt by setting RSUSPDC bit of UF0IC0 to OFF (0).

o Detect Resume/Suspend
It detects "Suspend" status if RSUM bit of UF0EPS1 is ON (1). If is "Suspend" status, it clear all the causes of interrupt and skip the process.

o Detect BUSRST Interrupt
It detects the interrupt if BUSRST bit of UF0IS0 is ON (1).

o Clear BUSRST Interrupt
Clear the cause of interrupt by setting BUSRST bit of UF0IC0 to OFF (0).

o Set BUS Reset Interrupt Flag
Set BUS Reset interrupt flag (usbf78k_busrst_flg) to 1.

o Clear FIFO for Bulk Endpoint
Clear all the FIFO for bulk endpoint.

o Detect SETRQ Interrupt
It detects the interrupt if SETRQ bit of UF0IS0 is ON (1).

o Clear SETRQ Interrupt
Clear the cause of interrupt by setting SETRQ bit of UF0IC0 to OFF (0).

o Detect Automatic Execution Request (SET_XXXX)
It can detect the status that it receives SET_CONFIGURATION request and processes it, by checking if SETCON bit of UF0SET is ON (1). To check if it is in Configured status, check the value of UF0CNF.

o Clear BUS Reset Interrupt Flag
Clear the BUS Reset interrupt flag (usbf78k_busrst_flg) by setting it to 0.

o Detect CPUDEC Interrupt
It detects the interrupt if CPUDEC bit of UF0IS1 is ON (1).

o Clear PORT Interrupt
Clear the cause of interrupt by setting PORT bit of UF0IC1 to OFF (0).

o Load Request Data
Load receiving data from FIFO, and structure the request data.

o Detect Request Type
Distinguish whether the request data is the standard request that hardware does not respond automatically or vendor request.

o Process Request
Process the request depending on the request type. Endpoint 0 is the endpoint for control transfer. In the enumeration process for plug-in, most standard device requests are automatically processed by hardware. Therefore, in this part, it processes standard requests that are not processed by hardware, class requests, and vendor requests.

Process Description Interrupt USB1B

The interrupt USB1B interrupt handler mainly processes BKO1DT interruption.

o Detect BKO1DT Interrupt
It detects the interrupt if BKO1DT bit of UF0IS3 is ON (1).

o Clear BKO1DT Interrupt
Clear the cause of interrupt by setting BKO1DTC bit of UF0IC3 to OFF (0).

o Set Data Receiving Flag
Set the data receiving flag (usbf78k_rdata_flg) to 1.It detects the interrupt if RSUSPD bit of UF0IS0 is ON (1).

12.4.1 How to run the sample program

The sample application loads the communication data received by USB function controller that is built in μPD78F0730. It converts the data to uppercase characters or to lowercase characters (only ASCII character data), and then executes the sending process by calling USB function controller sending process. The sample application process monitors the data receiving flag (usbf78k_rdata_flg) to start the process. As communication tool running on the host PC any terminal program supporting virtual communication ports can be used. To establish a connection and run the sample please use the following procedure:

- o Install the driver 'necelusbvcom.inf ' according to chapter 12.4.2
- o Connect port USB2 and start a debug session. In the Windows Device Manager the debug port can be seen:

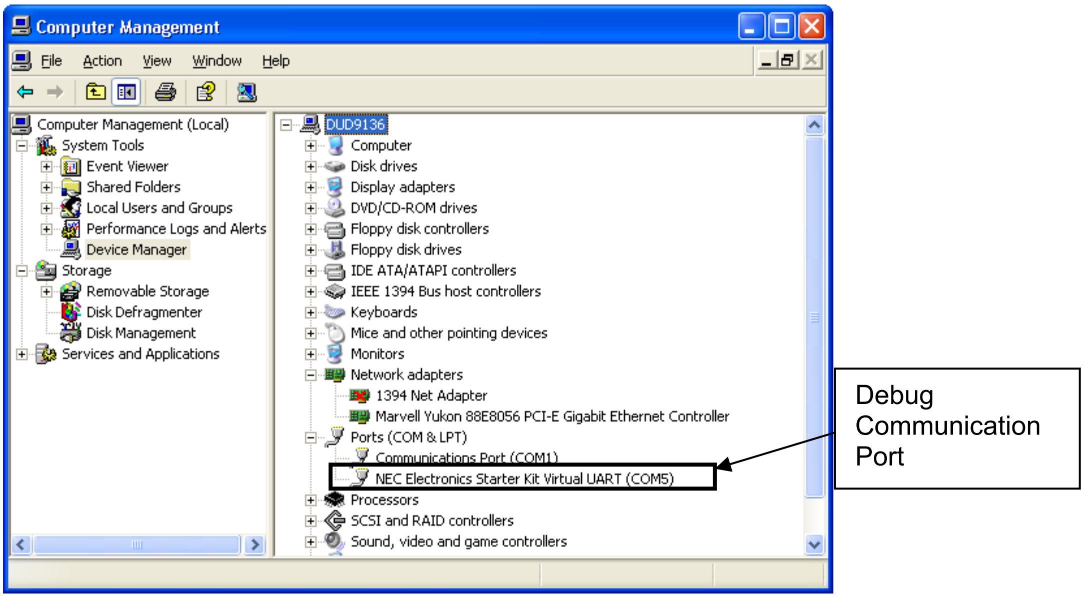


Figure 48: TK-78F0730_COM-sample Debug Communication Port

- o Connect port USB1 and run the sample application. In the Windows Device Manager a new virtual communication port named ‘NEC Electronics Virtual COM Port’ appears:

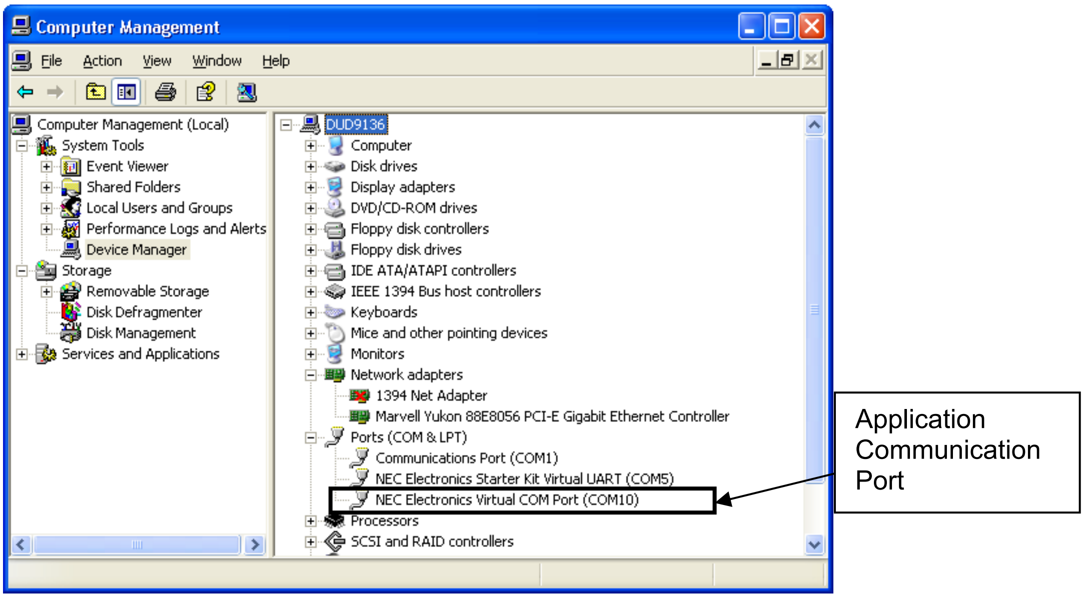


Figure 49: TK-78F0730_COM-sample Application Communication Port

- o Start the terminal program and setup a connection to the application virtual communication port
- o Send some test characters

12.4.2 Driver Installation

µPD78F0730 (USB1) and debug port (USB2) require different driver for connecting to host PC. Those files are described below. Select the appropriate file when you install the drivers.

- [µPD78F0730] (USB1 connector)
  - o necelusbvcom.inf
    This is the driver for USB serial conversion sample program. It is included in the USB serial conversion sample program (folder TK-78F0730_COM\driver).
    When it is connected, it is detected as "NEC Electronics Virtual COM Port".

- [Debug port] (USB2 connector)
  - o MQB2SALL.inf
    This driver is included in the Embedded Workbench for 78K. Depending on the version you find in the following subfolders of the Embedded Workbench installation folder:
    V4.50a or earlier versions: ‘`78k\config\nec\ie_pc_driver\MINICUBE`’
    V4.60a or later versions: ‘`78k\drivers\nec\MINICUBE`’

    It is detected as "NEC Electronics Starter Kit Virtual UART" when it is connected.

12.4.3 Precaution in Sending/Receiving Process

12.4.3.1 Sending/Receiving Process

In the sending process, it is only necessary to process the data size of prepared sending data. From driver point of view, data sending process is completed when the writing process on FIFO is finished. The actual sending process is done by USB function controller with its timing. In the receiving process, it retrieves the receiving data size by one of the function from sample driver, and it process only for the receiving data size. When you call the receiving process function with a data size smaller than the retrieved data size, the remaining data will be discarded. It is assumed that the buffer for the user data is prepared by user. Therefore, the sample program does not buffer the data. The sample application converts the data received from USB to uppercase characters or to lowercase characters and executes the sending process. Process the data receiving from USB and data sending to USB depending on the environment in use.

12.4.3.2 Relationship between Max Packet Size and Transfer Data Size

The sending process can be called regardless of Max Packet Size of endpoint. However, if the process is called with exceeding the Max Packet Size, it has to wait until it finishes writing the data with requested size to FIFO. The receiving process can be called regardless of Max Packet Size of endpoint. However, if the process is called with exceeding the Max Packet Size, it has to wait until it finishes receiving the data with requested size.

12.4.3.3 End of Sending Data

When the sending data size is just the same as the Max Packet Size, it is necessary to inform the host about end of data. To indicate the end of data, NULL packet is automatically sent by the driver and therefore a user does not need to add this process. Due to this NULL packets can be seen while analyzing packets by a protocol analyzer.

12.4.3.4 Plugging / Unplugging USB Cable While Transferring Data

Unplugging the USB cable while transferring data starts looping in the sending/receiving function. The driver can detect this, by monitoring the VDD[2] line of USB connector. However, as TK-78F0730 does not support the VDD line monitoring function, it keeps looping until the cable is re-plugged. It is designed to break out of the loop when the cable is re-plugged.

[2] For more information about VDD line monitoring, refer to "Figure 12-25 Example of Processing After Power Application / Power Failure (3/3)" in "12.7.5 Processing after power application" in the document "µPD78F0730 Preliminary User's Manual (Document No.: U19014EJxxxxxxx.pdf)".
Do not plug/unplug the cables for the appropriate performance. It will not be a problem if it is not during transferring data.

12.4.4 Development Environment

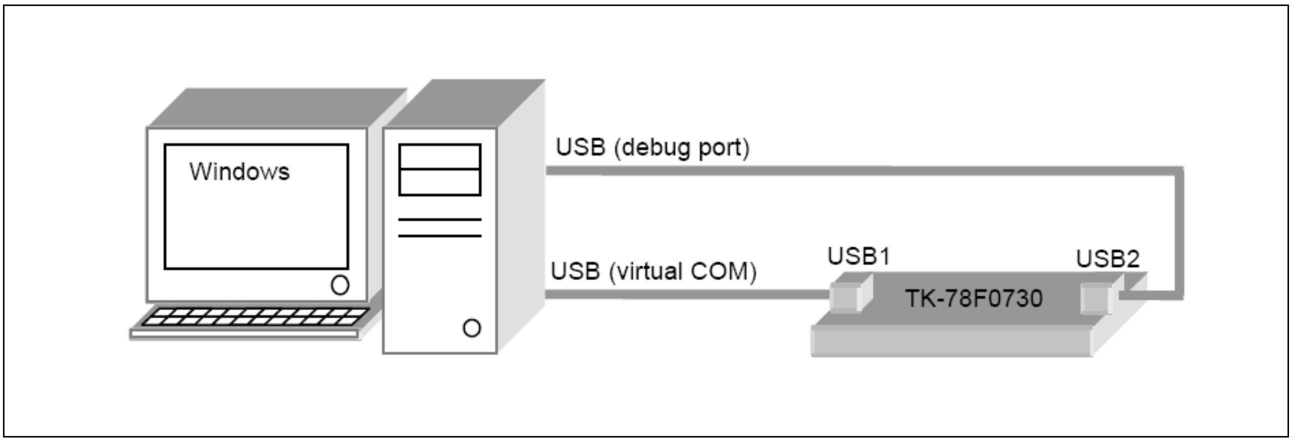


Figure 50: Development Environment Serial Conversion Sample

12.4.5 Structure of Serial Conversion Sample Application

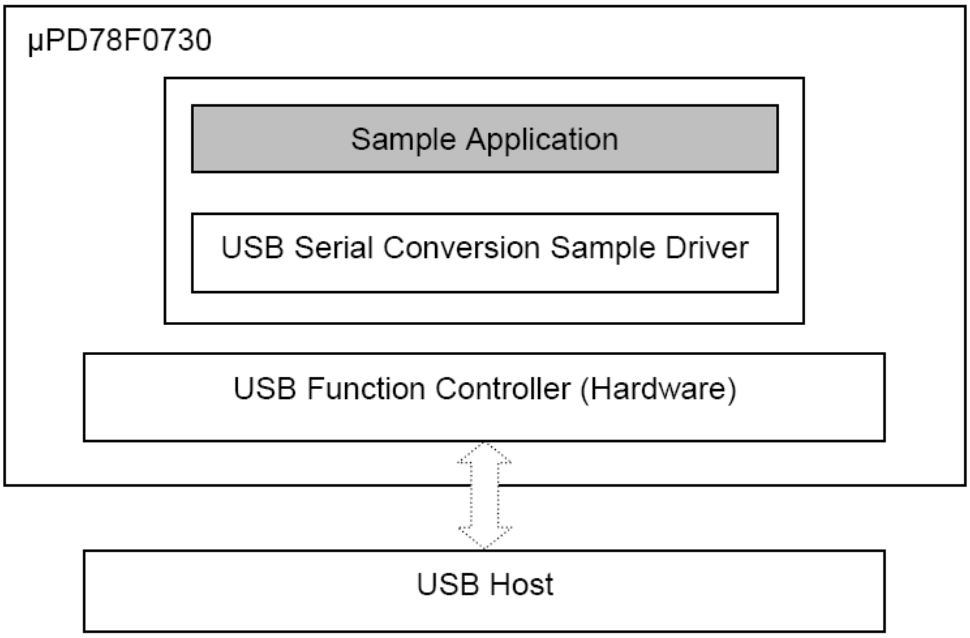


Figure 51: Structure of Serial Conversion Sample Application

12.4.6 File Structure

File	Description
main.c	Main routine
initialize.c	cpu and board initialization
usbf78k.c	USB initialization. Interrupt process. Bulk transfer. Control transfer.
usbf78k_vendor.c	Vendor class process

Table 25: Source Modules USB Serial Conversation Sample

File	Description
main.h	Function prototypes defined in main.c
errno.h	Error code definitions
types.h	Datatype definitions
usbf78k.h	Macro definitions for USB function register setting
usbf78k_desc.h	Descriptor definitions
usbf78k_sfr.h	Macro definitions for USB function register access
usbf78k_vendor.h	Vendor class function prototype declaration

Table 26: Header Files USB Serial Conversation Sample

12.4.7 Descriptor Information

Field	Size	Description	Value
bLength	1	Descriptor size	0x12
bDescriptor	1	Descriptor type	0x01
bcdUSB	2	BCD format of USB release number	0x0200
bDeviceClass	1	Class code 0x00H: no class 0xFFH: vendor 0x01-0xFEH: specific	0xFF
bDeviceSubClass	1	Sub-class code	0x00
bDeviceProtocol	1	Protocol code 0x00: no specific protocol 0xFF: vendor-specific protocol	0x00
bMaxPacketSize0	1	Maximum packet size at endpoint 0	0x40
idVendor	2	Vendor ID (USB IF assigns)	0x0409
idProduct	2	Product ID (vendor assigns)	0x01CD
bcdDevice	2	BCD format of device release number	0x0001
iManufacture	1	Index to string descriptor to indicate manufacturer	0x01
iProduct	1	Index to string descriptor to indicate product	0x02
iSerialNumber	1	Index to string descriptor to indicate serial number	0x03
bNumConfigurations	1	Number of devices that can be configured	0x01

Table 27: Device Descriptor Serial Conversion Sample

Field	Size	Description	Value
bLength	1	Descriptor size	0x09
bDescriptor	1	Descriptor type	0x02
wTotalLength	2	Total length of the configuration (configuration, interface, endpoint, and other descriptors)	0x0020
bNumInterfaces	1	Number of interfaces supported in the configuration	0x01
bConfigurationValue	1	Input value (>= 1) for selecting this configuration with SetConfiguration	0x01
iConfiguration	1	Index to string descriptor to indicate descriptor	0x00
bmAttributes	1	Configuration attributes with the unit of bit D7: "1" D6: self-powered D5: remote wake-up D4-D0: reserved (0)	0x80
bMaxPower	1	Maximum power consumption of bus with the unit of 2mA	0x32

Table 28: Configuration Descriptor Serial Conversion Sample

Field	Size	Description	Value
bLength	1	Descriptor size	0x09
bDescriptor	1	Descriptor type	0x04
bInterfaceNumber	1	Index Number (0 based) to indicate this interface in the configuration	0x00
bAlternateSetting	1	Input value to select alternate setting in SetInterface	0x00
bNumEndpoints	1	Interface endpoint number (excluding endpoint 0)	0x02
bInterfaceClass	1	Class Code 0x00: no class 0xFF: vendor 0x01 – 0xFE: specific	0xFF
bInterfaceSubclass	1	Subclass code	0x00
bInterfaceProtocol	1	Protocol code 0x00: no specific protocol 0xFF: vendor specific protocol	0x00
iInterface	1	Index to string descriptor to indicate interface	0x00

Table 29: Interface Descriptor Serial Conversion Sample

Field	Size	Description	Value
bLength	1	Descriptor size	0x07
bDescriptor	1	Descriptor type	0x05
bEndpointAddress	1	Endpoint address bits: D7: Direction 0: OUT, 1: IN D6-D4: Reserved (0) D4-D0: Endpoint number	0x81
bmAttributes	1	Attribute bits: D1-D0: Transfer type 0: Control 1: Isochronous 2: Bulk 3: Interrupt *D5-D2 is used only by isochronous endpoint D3-D2: Synchronization type 0: No synchronization 1: Asynchronous 2: Adaptive 3: Synchronous D5-D4: Usage type 0: Data endpoint 1: Feedback endpoint 2: Dependant feedback endpoint 3: (reserved)	0x02
wMaxPacketSize	2	Payload size bits: D10-D0: Maximum packet size D12-D11: auditory transaction number per μ frame (only high-speed isochronous and interrupt) 0: No addition (1 transaction / μ frame) 1: 1 (2 transaction / μ frame) 2: 2 (3 transaction / μ frame) 3: Not in use (reserved)	0x0040
bInterval	1	Polling interval for data transfer endpoint Full/low speed interrupt: specify with unit of ms (number of frames) High-speed isochronous/interrupt: specify N for 2 raised to the power of N-1 with unit of μ frame (for example, 1 polling in 8μ frames when bInterval is 4) Full-speed isochronous: specify N for 2 raised to the power of N-1 with unit of 1ms High-speed bulk/control: specify the maximum NAK rate for endpoint with unit of μ frame 0 means that it does not respond NAK on OUT/DATA transaction	0x00

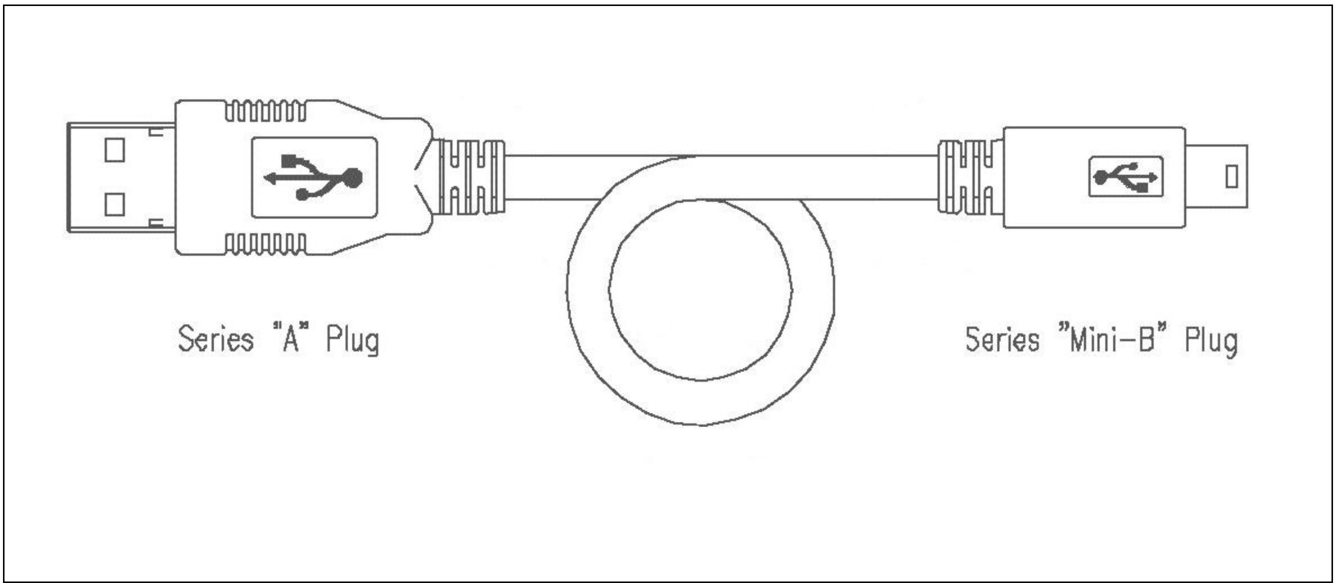
Table 30: Endpoint Descriptor Serial Conversion Sample

Field	Size	Description	Value
bLength	1	Descriptor size	0x07
bDescriptor	1	Descriptor type	0x05
bString	41	Language Code: 0x09 0x04	
		Manufacture: “NEC Electronics Co.”	
		Product: “VirtualCOM”	
		Serial Number: “0_98765432”	

Table 31: String Descriptor Serial Conversion Sample

13. Cables

13.1 USB interface cable (Mini-B type)



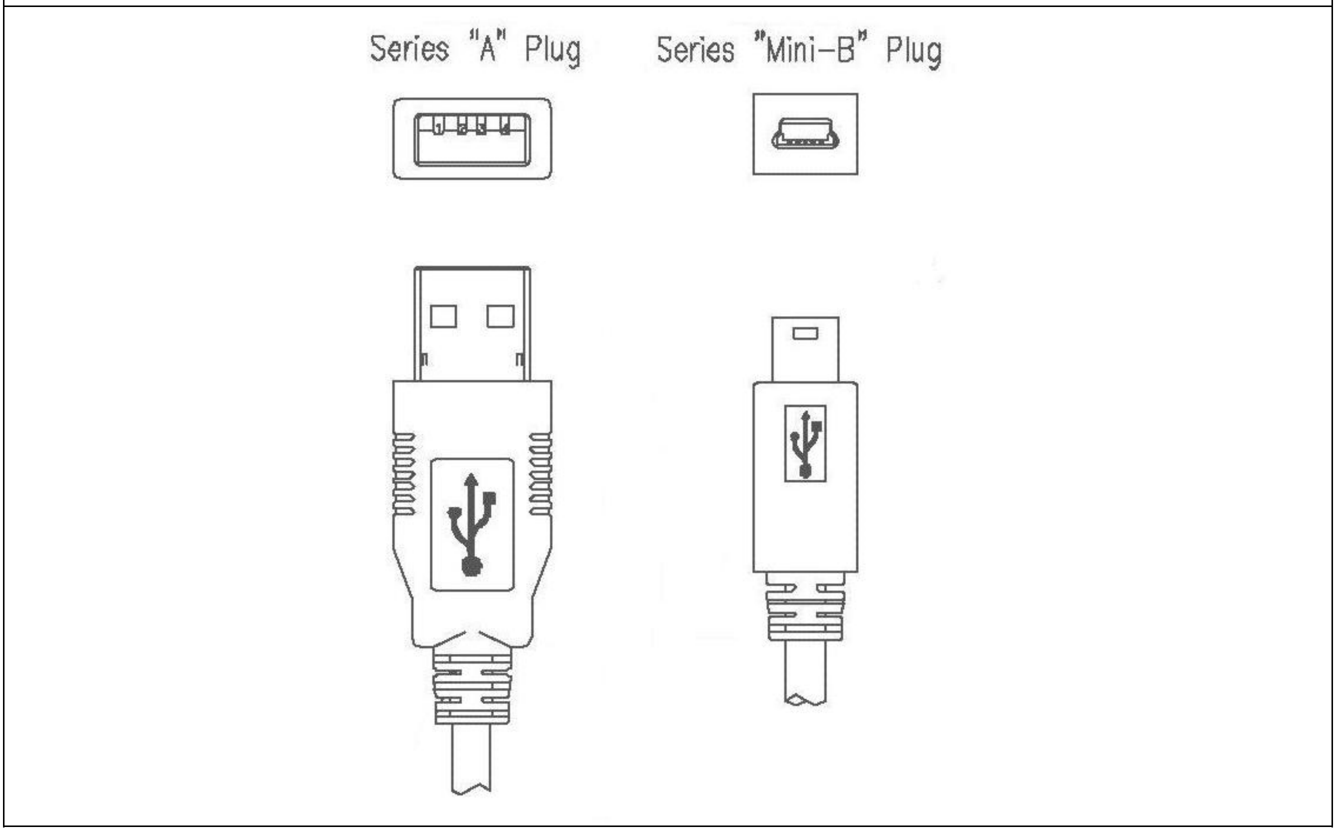


Figure 52: USB interface cable (Mini-B type)

14. Schematics

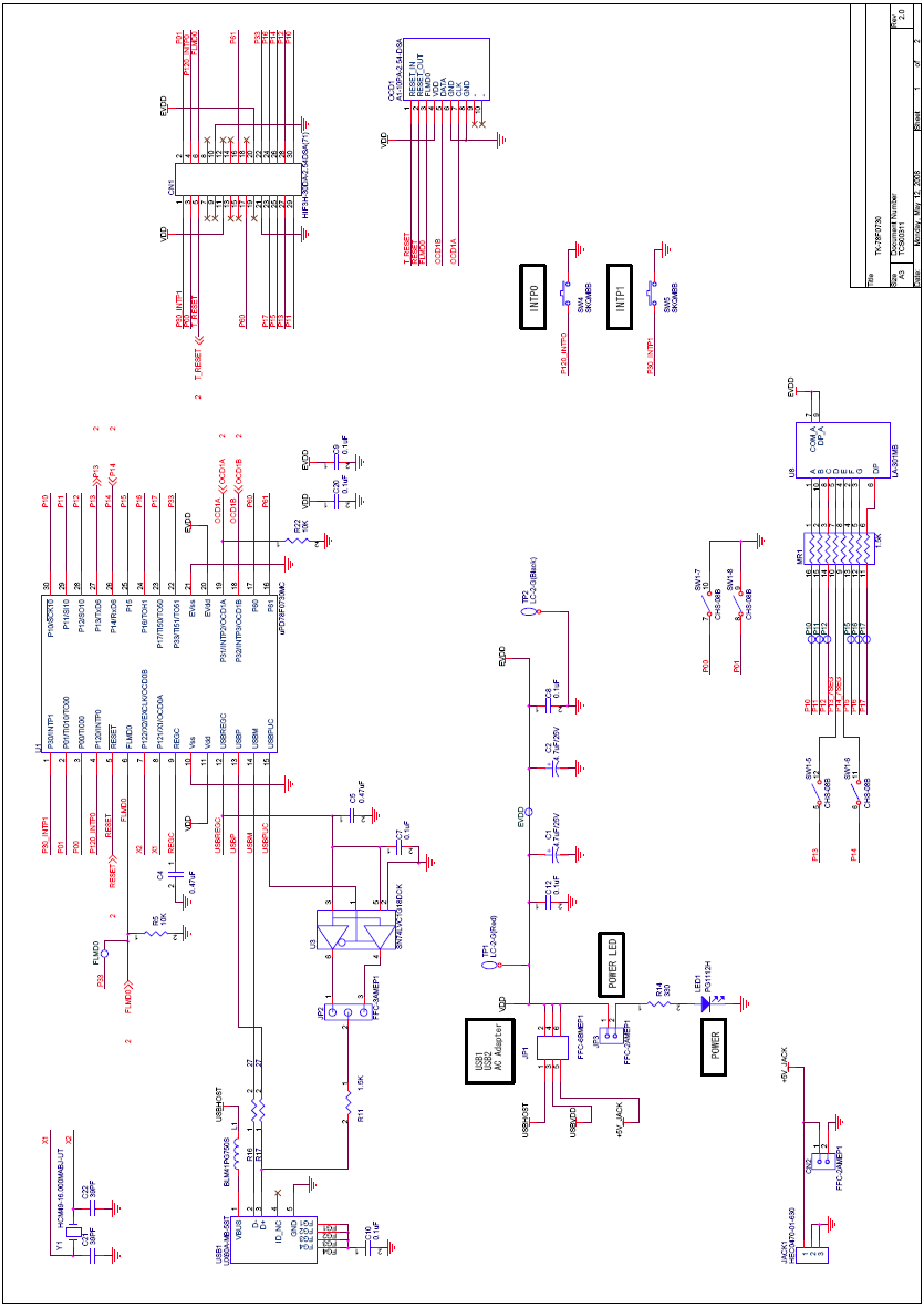


Figure 53: TK-78F0730 schematics 1 of 2

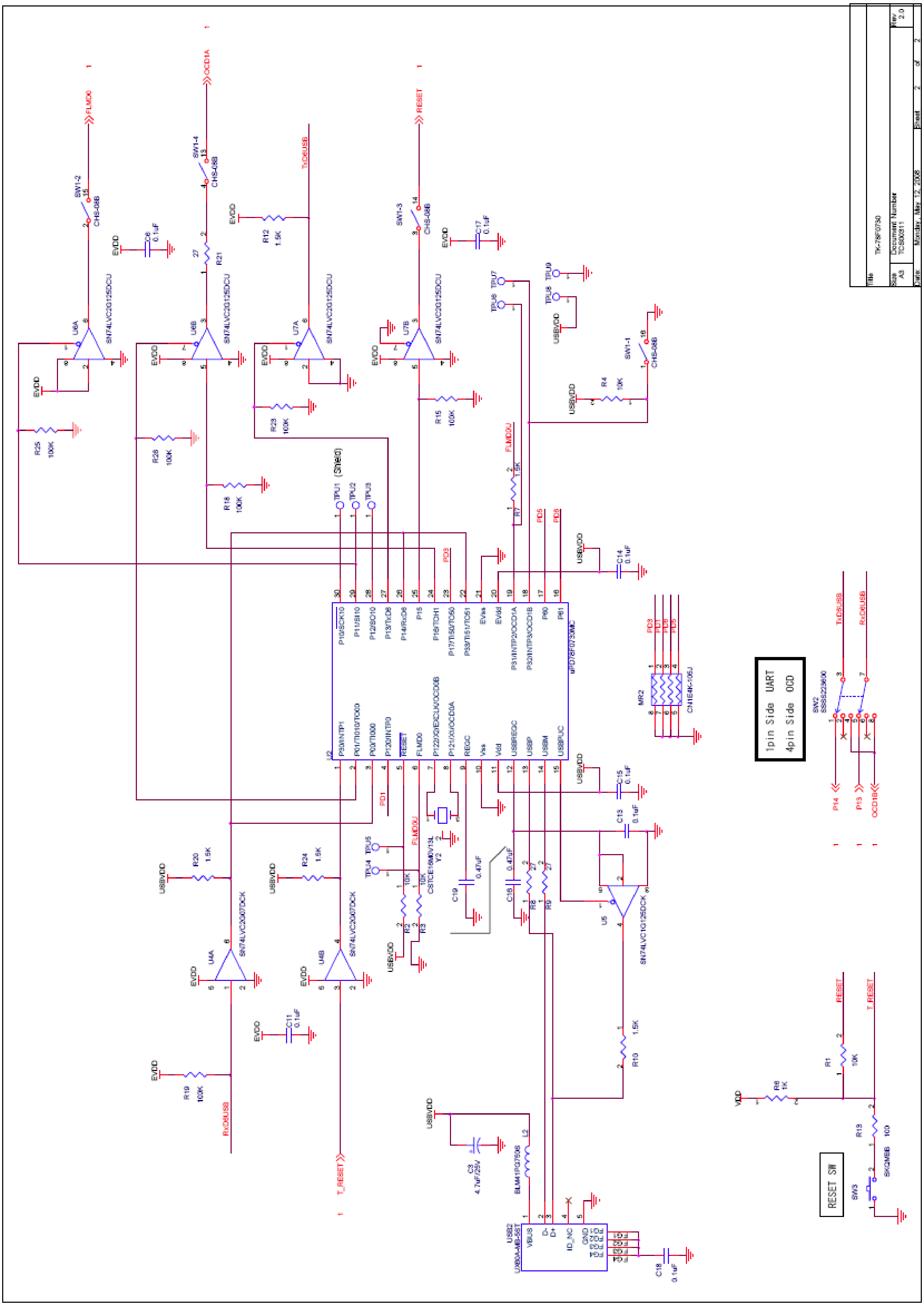


Figure 54: TK-78F0730 schematics 2 of 2

[MEMO]