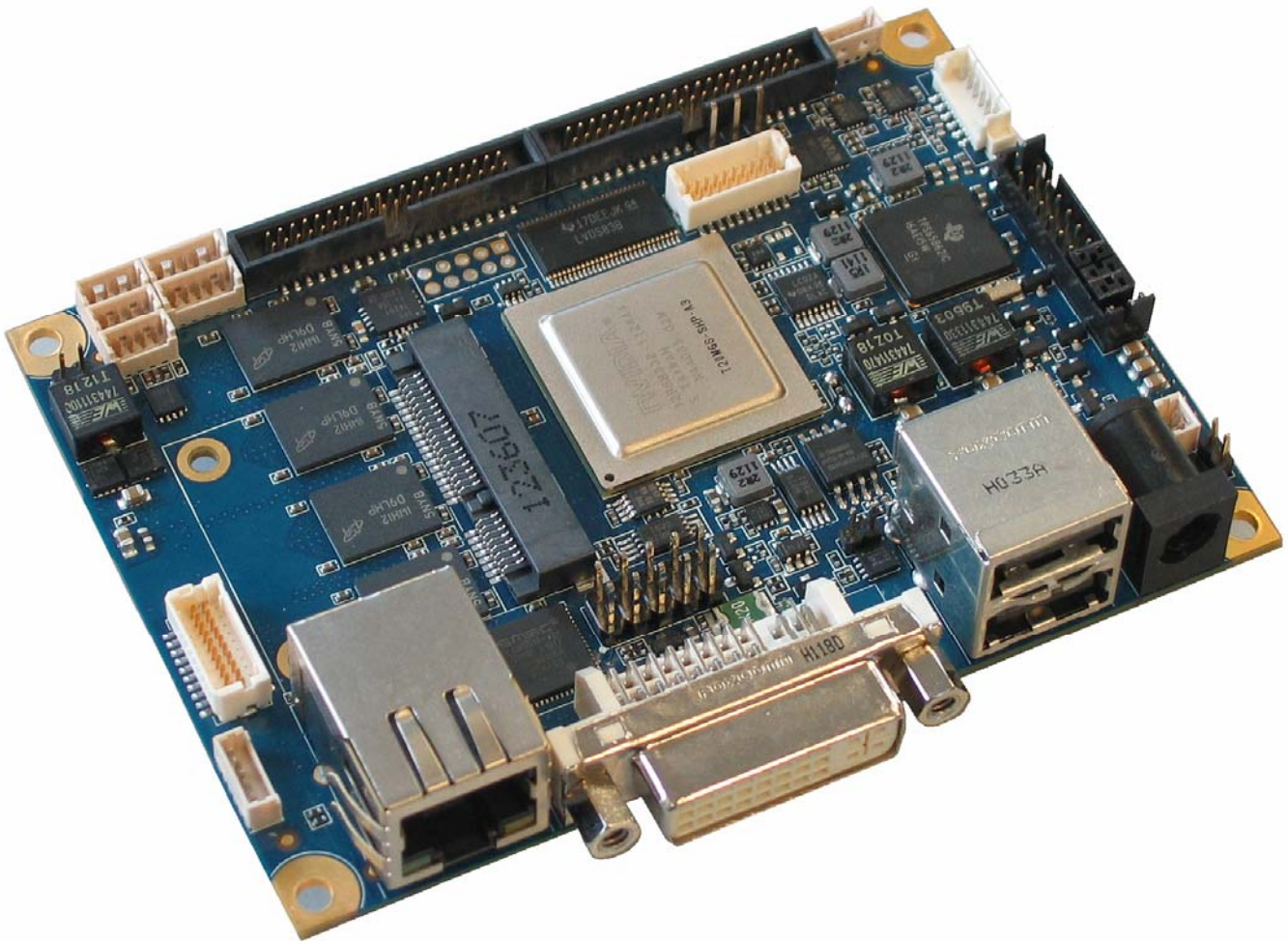


» Kontron Software Guide «



KTT20/pITX

KTD-S0045-D

» Table of Contents «

1	User Information	1
1.1	About This Document.....	1
1.2	Copyright Notice.....	1
1.3	Trademarks.....	1
1.4	Standards.....	1
1.5	Warranty.....	1
1.6	Life Support Policy.....	2
1.7	Technical Support.....	2
2	U-Boot Setup	3
2.1	Setup Command.....	3
2.2	Setup Usage.....	3
2.3	Display Menu.....	4
2.3.1	Boot Display.....	4
2.3.2	LCD Panel Resolution.....	4
2.3.3	Backlight Brightness.....	5
2.3.4	Backlight Output Level.....	5
2.4	Devices Menu.....	5
2.4.1	PCI Express® Interface.....	5
2.4.2	Audio Device.....	5
2.4.3	GPIO Interface.....	6
2.5	Miscellaneous Menu.....	6
2.5.1	Extended Temperature Range.....	6
2.5.2	Temperature Alert Output.....	6
2.5.4	Temperature High Limit.....	6
2.5.5	Temperature Low Limit.....	7
2.6	Password Command.....	7
2.7	Defaults Command.....	7
2.8	Save Command.....	7
2.9	Summary Command.....	8
3	VESA® DisplayID™	9
3.1	LCD/LVDS Technology Overview.....	9
3.1.1	Detailed Timing Descriptor.....	9
3.1.2	24 Bit Color Mapping Tips.....	11
3.2	EDID 1.3 Specification (VESA®).....	12
3.3	DisplayID™ Specification (VESA®).....	12
3.3.1	DisplayID™ Parameter Summary.....	12
3.3.2	DisplayID™ Restrictions.....	13
3.3.3	LCD Panel Selection.....	13

3.3.4	DisplayID™ Windows® Tool	14
3.3.5	Building DisplayID™ File	15
3.3.6	Erasing DisplayID™ Record	15
3.3.7	U-Boot EEPROM Update Tool	15
4	KTT20 Tool Package	17
5	Bootloader Modification and Download	19
5.1	Program IMAGECREATOR	20
5.2	NVFLASH Download Tool	21
5.2.1	Windows® Operation	21
5.2.2	Linux® Operation	23
6	SMSC® USB Hub and LAN Controller	26
7	Alternative Linux® Distributions	26
8	U-Boot Compilation	27
8.1	Hardware Components Compatibility	29
8.2	GPIO Declarations	29
8.3	UART Declarations	30
8.4	Linux® Environment on Windows®	31
9	Linux® BSP	32
9.1	User Login Arguments	32
9.2	Video Decoding	32
9.2.1	Command Line Examples	33
9.2.2	Reencoding Examples	34
9.3	Audio Settings	34
9.4	PCI Express® Interface	36
9.5	CPU Frequency Management	36
9.6	KEAPI Interface	37
9.6.1	KEAPI Command Line Tools	40
10	Android™ BSP	41
10.1	Graphics Interface	41
10.1.1	DVI® Monitor	41
10.1.2	LCD Panel	41
10.2	Video Decoding	41
10.3	Display Density	42
10.4	GPIOs, Temperatures, Backlight and Bootcounter	42
10.4.1	GPIOs	42
10.4.2	Temperatures	43
10.4.3	Backlight	43
10.4.4	Bootcounter	43

11	Windows [®] Embedded Compact 7 (WEC7) BSP.....	44
11.1	U-Boot Settings.....	44
11.1.1	Boot from microSD Card.....	44
11.1.2	Boot from USB key.....	44
11.2	Video Decoding.....	45
11.3	Raster Font Support.....	45
11.4	Graphics Interface.....	46
11.4.1	DVI [®] Monitor.....	46
11.4.2	LCD Panel.....	46
11.5	I ² C [™] Support.....	47
11.6	Watchdog Example.....	47
11.7	GPIO Examples.....	49
	Appendix A: Reference Documents.....	51
	Appendix B: Document Revision History.....	52

1 User Information

1.1 About This Document

This document provides information about products from KONTRON Technology A/S and/or its subsidiaries. No warranty of suitability, purpose or fitness is implied. While every attempt has been made to ensure that the information in this document is accurate the information contained within is supplied “as-is” - no liability is taken for any inaccuracies. Manual is subject to change without prior notice.

KONTRON assumes no responsibility for the circuits, descriptions and tables indicated as far as patents or other rights of third parties are concerned.

1.2 Copyright Notice

Copyright © 2012 - 2013, KONTRON Technology A/S, ALL RIGHTS RESERVED.

No part of this document may be reproduced or transmitted in any form or by any means, electronically or mechanically, for any purpose without the express written permission of KONTRON Technology A/S.

1.3 Trademarks

Brand and product names are trademarks or registered trademarks of their respective owners.

1.4 Standards

KONTRON Technology A/S is certified to ISO 9000 standards.

1.5 Warranty

This product is warranted against defects in material and workmanship for the warranty period from the date of shipment. During the warranty period KONTRON Technology A/S will at its discretion decide to repair or replace defective products.

Within the warranty period the repair of products is free of charge as long as warranty conditions are observed.

The warranty does not apply to defects resulting from improper or inadequate maintenance or handling by the buyer, unauthorized modification or misuse, operation outside of the product's environmental specifications or improper installation or maintenance.

KONTRON Technology A/S will not be responsible for any defects or damages to third party products that are caused by a faulty KONTRON Technology A/S product.

1.6 Life Support Policy

KONTRON Technology's products are not for use as critical components in life support devices or systems without express written approval of the general manager of KONTRON Technology A/S.

As used herein:

Life support devices or systems are devices or systems which

- a) are intended for surgical implant into body or
- b) support or sustain life and whose failure to perform, when properly used in accordance with instructions for use provided in the labelling, can be reasonably expected to result in significant injury to the user.

A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

1.7 Technical Support

Please consult our web site at <http://www.kontron.com/support> for the latest product documentation, utilities, drivers and support contacts or use the special e-mail address sbc-support@kontron.com for a technical problem. In any case you can always contact your board supplier for technical support.

Before contacting support please be prepared to provide as much information as possible:

Board identification:

- Type
- Part number (find PN on label)
- Serial number (find SN on label)

System environment:

- O/S type and version
- Driver origin and version
- Attached hardware (drives, USB devices, LCD panels ...)

2 U-Boot Setup

The sense of a special Setup part is to avoid expendable changes in the proper operating systems. The Setup entries are valid for all supported operating systems (e.g. Linux[®], Android[™] and Windows[®] Embedded Compact). For example if you switch from Linux[®] to Android[™] or vice versa in the ideal case no changes will be necessary. The Setup data records are stored in a non-volatile memory (EEPROM) and not in an erasable script. A possible password input protects from unauthorized access.

The KONTRON Setup provides three configurable main menus:

- ❑ **display**
- ❑ **devices**
- ❑ **misc**

The **display** menu allows to define one or more boot displays, the resolution of a LCD panel and backlight parameters. The **devices** part involves some hardware device settings, e.g. audio controller enabled or disabled. The **misc** menu contains special features, e.g. temperature control.

ATTENTION

Only the original KONTRON BSPs guarantee the realization of the U-Boot Setup features.

2.1 Setup Command

For a help screen type the command without an argument

setup

Now you can see all the supported sub-commands (menus). Normally the following entries are displayed

- ❑ **display** display settings
- ❑ **devices** onboard device configuration
- ❑ **misc** miscellaneous settings
- ❑ **password** password input
- ❑ **defaults** reset all settings to the default values
- ❑ **summary** show all actual settings
- ❑ **save** save all settings to EEPROM

Syntax example: **setup display** <Enter>

2.2 Setup Usage

After the execution of a sub-command (e.g. setup misc) the selection of a submenu requires only a numeric value. Thereafter the real settings are visible. Now you can choose between an alphanumeric (default) or a numeric input. The alphanumeric presentation illustrates intuitive the right choice. If you favor the numeric input delete all chars with the <Backspace> key, type the number and then press the <Enter> key. With the <Cursor up> key previous keyboard inputs are recallable for a quick repetition of often used commands.

After the completion of all changes it is reasonable to control the settings with the summary command.

Syntax: **setup summary** <Enter>

2.3 Display Menu

This menu part includes several display settings

- define the first boot display**
- define the second boot display**
- determine a resolution for the lcd panel**
- define the backlight brightness**
- define the backlight output level**

2.3.1 Boot Display

The NVIDIA® Tegra 250 implies a Graphics Processing Unit (GPU) with two independent display controllers. Without a restriction (except duplicate usage) each controller interface can be configured as

- none** switch off the display controller
- dvi**
- lcd**

Examples:

First boot display	dvi
Second boot display	none
or	
First boot display	none
Second boot display	lcd
or	
First boot display	dvi
Second boot display	lcd
or	
First boot display	lcd
Second boot display	dvi

2.3.2 LCD Panel Resolution

You have the choice to select a panel resolution with a fixed timing or a special setting 'auto' for a free definable timing based on the VESA® DisplayID™ specification. For further details about DisplayID™ see the chapter 'VESA® DisplayID™'. The KTT20/pITX supports following resolutions

- auto** free timing based on VESA® DisplayID™
- vga** fixed timing 640x480 pixel, 18 bit color depth
- wvga** fixed timing 800x480 pixel, 18 bit color depth
- svga** fixed timing 800x600 pixel, 18 bit color depth
- xga** fixed timing 1024x768 pixel, 24 bit color depth

2.3.3 Backlight Brightness

This submenu allows the definition of the analog backlight brightness (voltage range: 0V to +5V). The input format is represented by a decimal number with maximal three digits.

Examples:

Brightness: 0	minimal value = 0V
or	
Brightness: 128	half range = +2.5V
or	
Brightness: 255	maximal value = +5V

2.3.4 Backlight Output Level

Some backlight inverters need a low level for the enable signal, other inverters a high level (normally +5V). Use this submenu to configure the right enable output level; two options are available

- low** voltage = 0V
- high** voltage = +5V

2.4 Devices Menu

This menu part defines several hardware device settings

- PCI Express settings**
- Audio settings**
- GPIO settings**

2.4.1 PCI Express[®] Interface

The selection is limited to the enable respectively disable feature

- disabled**
- enabled**

2.4.2 Audio Device

The selection is also limited to the enable respectively disable feature

- disabled**
- enabled**

2.4.3 GPIO Interface

You can choose between three modes: all interface signals are defined as GPIOs (General Purpose Input Output) or some special signals have another function (I²C™, SPI™ respectively SDIO). For a detailed overview about these signals see the 'KTT20/pITX Users Guide' chapter 'Digital I/O Interface'. The signals are named GPIO16 to GPIO21 respectively GPIO23 to GPIO29, GPIO32 and GPIO36 to GPIO37.

Remark: the operating systems do not support most of these special functions because the possible applications can be too different. Following the possible options

- gpio**
- i2c-spi**
- sdio**

2.5 Miscellaneous Menu

This menu part defines several special features (currently only temperature sensor features)

- Extended temperature range**
- Temperature alert output**
- Temperature high limit**
- Temperature low limit**

2.5.1 Extended Temperature Range

The default temperature range of the sensor is 0°C to +127°C. If you activate the extended range you can measure a temperature between -55°C and +150°C. There are two options

- disabled**
- enabled**

2.5.2 Temperature Alert Output

In some cases it can be useful to disable the alert output. With this submenu is it possible to set this alarm signal to

- disabled**
- enabled**

2.5.4 Temperature High Limit

The temperature high limit controls the alert output. Dependent on the extended temperature setting the limit can be up to +150°C. The input format is represented by a decimal number with maximal three digits.

Example:

Temp high limit: 75

2.5.5 Temperature Low Limit

The temperature low limit controls the alert output. Dependent on the extended temperature setting the limit can be down to -55°C . The input format is represented by a decimal number with maximal three digits.

Example:

Temp low limit: 5

2.6 Password Command

If you want to control the access to the Setup settings it is possible to use a password protection. Maximal eight alphanumeric chars, numbers or special characters are admissible. You can delete an old password respectively cancel the password protection with the input of an empty string.

Syntax: **setup password** <Enter>

Example:

New password: ***** e.g. 12%&fgWQ
Verify password: ***** the same

2.7 Defaults Command

In some cases it can be useful to reset quickly the Setup settings. For an example there is a problem with driving of a single display - preferably a LCD panel - and the connection of a DVI[®] monitor is possible.

Syntax: **setup defaults** <Enter>

The most important default settings are

First boot display	none
Second boot display	dvi
PCI Express settings	enabled
Audio settings	enabled
GPIO settings	gpio
Extended temperature range	disabled

2.8 Save Command

This is one of the most important sub-commands. Without this calling all Setup changes are lost after power off. The save instruction writes the temporary Setup settings into the non-volatile memory device (EEPROM).

Syntax: **setup save** <Enter>

2.9 Summary Command

This Setup command gives a quick overview about all actual settings. An additional feature is the boot-counter report.

Syntax: `setup summary` <Enter>

Example:

DISPLAY PART:

Boot display 1	: dvi
Boot display 2	: lcd
LCD resolution	: wvga
Brightness	: 128
Backlight enable level	: high

DEVICES PART:

PCI Express interface	: enabled
Audio interface	: disabled
I/O interface	: gpio

MISC PART:

Extended temp range	: disabled
Temp alert output	: disabled
Temp high limit	: 80
Temp low limit	: 0
Bootcounter	: 100

3 VESA® DisplayID™

3.1 LCD/LVDS Technology Overview

3.1.1 Detailed Timing Descriptor

The input fields Pixel Clock, Horizontal Active, Horizontal Blank, Horizontal Sync Offset, Horizontal Sync Width, Vertical Active, Vertical Blank, Vertical Sync Offset and Vertical Sync Width must be filled in with the correct values according to the panel's data sheet. In many cases the value for Horizontal/Vertical Blank cannot be read directly from the data sheet. Instead terms such as Display Period (active pixels/lines) or Horizontal/Vertical Total appear.

In this case the following calculation can be made:

⇒ **Blank Value = Total Value – Active Value.**

Sometimes the datasheet does not specify Sync Offset and/or Sync Width. In this case the permissible values can only be determined through testing. However the rule is:

⇒ **The sum of Sync Offset and Sync Width must not exceed the value for Horizontal/Vertical Blank.**

Also datasheets are often different for displays with double pixel clock. If Pixel Clock and Horizontal Values seem to be halved this must be corrected for input:

⇒ **The values must always be entered as though it were a panel with single pixel clock.**

Example 1:

PRIMEVIEW PM07OWL4 (single pixel clock)

Data sheet specifications:

Clock Frequency [typ.]	32 MHz	
HSync Period [typ.]	1056 Clocks	(equivalent to Horizontal Total)
HSync Display Period [typ.]	800 Clocks	(equivalent to Horizontal Active)
HSync Pulse Width [typ.]	128 Clocks	
HSync Front Porch [typ.]	42 Clocks	
HSync Back Porch [typ.]	86 Clocks	
VSyn Period [typ.]	525 Lines	(equivalent to Vertical Total)
VSyn Display Period	480 Lines	(equivalent to Vertical Active)
VSyn Pulse Width [typ.]	2 Lines	
VSyn Front Porch [typ.]	10 Lines	
VSyn Back Porch [typ.]	33 Lines	

Result:

Pixel Clock	32	
Horizontal Active	800	
Horizontal Blank	256	((128 + 42 + 86) → H. Pulse Width + H. Front Porch + H. Back Porch)
Horizontal Sync Offset	42	(H. Front Porch)
Horizontal Sync Width	128	(H. Pulse Width)
Vertical Active	480	
Vertical Blank	45	((2 + 10 + 33) → V. Pulse Width + V. Front Porch + V. Back Porch)
Vertical Sync Offset	10	(V. Front Porch)
Vertical Sync Width	3	(V. Pulse Width)

Example 2 (not useable on KTT20/pITX):**SHARP LQ190E1LW01** (double pixel clock)

Data sheet specifications (no definition of Sync Offset and Sync Width):

Clock Frequency [typ.]	54 MHz	
Horizontal Period (1) [typ.]	844 Clocks	(equivalent to Horizontal Total)
Horizontal Display Period	640 Clocks	(equivalent to Horizontal Active)
Vertical Period [typ.]	1066 Lines	(equivalent to Vertical Total)
Vertical Display Period	1024 Lines	(equivalent to Vertical Active)

Result:

Pixel Clock	108	(2 x 54 MHz)
Horizontal Active	1280	(2 x 640 Clocks)
Horizontal Blank	408	((844 – 640) x 2 Clocks)
Horizontal Sync Offset	45	(normally approx. 10 – 15 % of Horizontal Blank)
Horizontal Sync Width	140	(normally approx. 30 – 70 % of Horizontal Blank)
Vertical Active	1024	
Vertical Blank	42	(1066 – 1024 Lines)
Vertical Sync Offset	1	(normally approx. 1 – 3 Lines)
Vertical Sync Width	3	(normally approx. 1 – 15 Lines)

Example 3 (not useable on KTT20/pITX):**LG-PHILIPS LM170E01-TLA1** (double pixel clock)

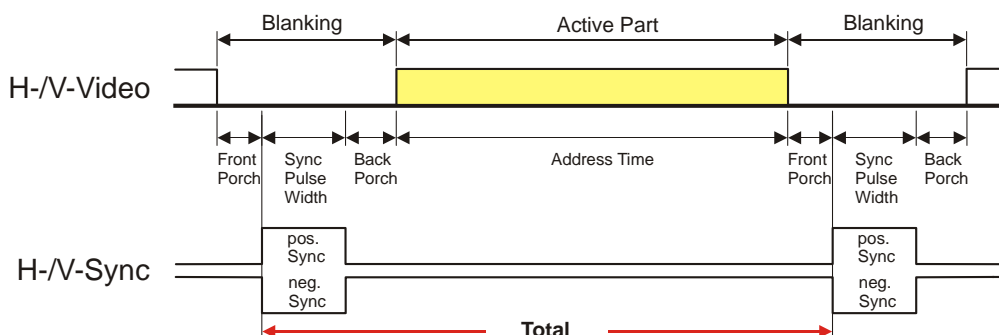
Data sheet specifications:

Clock Frequency [typ.]	54 MHz
Hsync Period [typ.]	844 Clocks
Horiz. Valid [typ.]	640 Clocks
Horiz. Back Porch [typ.]	124 Clocks
Horiz. Front Porch [typ.]	24 Clocks
Vsync Period [typ.]	1066 Lines
Vert. Valid [typ.]	1024 Lines
Vert. Back Porch [typ.]	38 Lines
Vert. Front Porch [typ.]	1 Line

Result:

Pixel Clock	108	(2 x 54 MHz)
Horizontal Active	1280	(2 x 640 Clocks → Horizontal Addr. Time)
Horizontal Blank	408	((844 – 640) x 2 Clocks)
Horizontal Sync Offset	48	(2 x 24 Clocks → Horizontal Front Porch)
Horizontal Sync Width	112	((((408/2 – 124 – 24) x 2) → H. Blank – H. Back Porch – H. Front Porch)
Vertical Active	1024	(Vertical Addr. Time)
Vertical Blank	42	(1066 – 1024 Lines)
Vertical Sync Offset	1	(Vertical Front Porch)
Vertical Sync Width	3	(Vertical Blank – Vertical Back Porch – Vertical Front Porch)

The following picture shows the typical video timing.

Timing Parameter Definitions

3.1.2 24 Bit Color Mapping Tips

The double pixel clock or 24-bit color depth can generally be taken from the datasheet. There are two interface modes existing at 24-bit color depth: **FPDI** (Flat Panel Display Interface) or **LDI** (LVDS Display Interface). Some panels use the line SELL LVDS (SElect Lvds data order). The LVDS data assignment in the datasheet can give you an indication by the last channel (e.g. RX3/TX3 – SELL LVDS = low) whether it is a LDI panel (contains the lowest bits). Most panels have a FPDI interface.

Example:

FPDI data assignment (LVDS channel 3 even or odd):

Tx/Rx27	Red 6 (e.g. even: RE6 or ER6)
Tx/Rx5	Red 7
Tx/Rx10	Green 6 (e.g. even: GE6 or EG6)
Tx/Rx11	Green 7
Tx/Rx16	Blue 6 (e.g. even: BE6 or EB6)
Tx/Rx17	Blue 7
Tx/Rx23	not used

LDI data assignment (LVDS channel 3 even or odd):

Tx/Rx27	Red 0 (e.g. even: RE0 or ER0)
Tx/Rx5	Red 1
Tx/Rx10	Green 0 (e.g. even: GE0 or EG0)
Tx/Rx11	Green 1
Tx/Rx16	Blue 0 (e.g. even: BE0 or EB0)
Tx/Rx17	Blue 1
Tx/Rx23	not used



3.2 EDID 1.3 Specification (VESA®)

The EDID (Extended Display Identification Data) record has a fixed structure. The first 8 bytes contain the distinctive identification 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00. The end of the record is marked by the checksum (1 byte). The result of the addition of all bytes including the checksum has to be zero.

For a comprehensive support of the majority of available panels you don't need all fields of the EDID record. The **Detailed Timing Descriptor** (18 bytes) is the most important field. No 24bit panels (FPDI/LDI) are supported though. This means EDID should only be used for 18bit panels.

For further information please consult the official EDID specification from the VESA® comitee which has to be payed.

3.3 DisplayID™ Specification (VESA®)

Intended as a replacement for all previous EDID versions DisplayID™ contains many new features. It's a structure with several well defined elements (tags). Not every element that is listed in the specification has to be part of the resulting data set (basic section).

KONTRON has decided to use this selection of tags (mandatory presence).

Tag	Description
0x00	Product Identification Data Block (Vendor ID, Product Code, Manufacturing Date ...)
0x03	Type I Detailed Timing Data Block (Pixel Clock, Horizontal/Vertical Data ...)
0x0C	Display Device Data Block (Device Technology, Operating Mode, Color Depth ...)
0x0D	Interface Power Sequencing Data Block (Power On/Off Timing)
0x0F	Display Interface Data Block (Interface Type, Interface Attribute ...)

3.3.1 DisplayID™ Parameter Summary

Only a part of the parameters used in the DisplayID™ Windows® tool are interpreted by a specific board. The following table shows a summary of the used parameters (valid for KTT20/pITX).

Group	Parameter	Comment
Type I Timing	Pixel Clock	
Type I Timing	Horizontal Active	
Type I Timing	Horizontal Blank	
Type I Timing	Horizontal Sync Offset	Front porch
Type I Timing	Horizontal Sync Width	
Type I Timing	Vertical Active	
Type I Timing	Vertical Blank	
Type I Timing	Vertical Sync Offset	Front porch
Type I Timing	Vertical Sync Width	
Display Interface 1	Bits per Pixel	Color depth (18 or 24bit)

3.3.2 DisplayID™ Restrictions

Depending on the graphic controller not all features can be used. The following table shows the most important restrictions.

Restrictions for KTT20/pITX
Panels with dual or quad clock not supported (2 or 4 Pixel per Clock)
Variable power sequencing not supported

3.3.3 LCD Panel Selection

The choice of a LCD display is basically defined by two parameters.

Parameter	Value
Pixel per Clock (Channels)	1
Pixel Clock Range	tdb. MHz

Currently this leads to a maximum resolution of

1280 x 800 Pixel

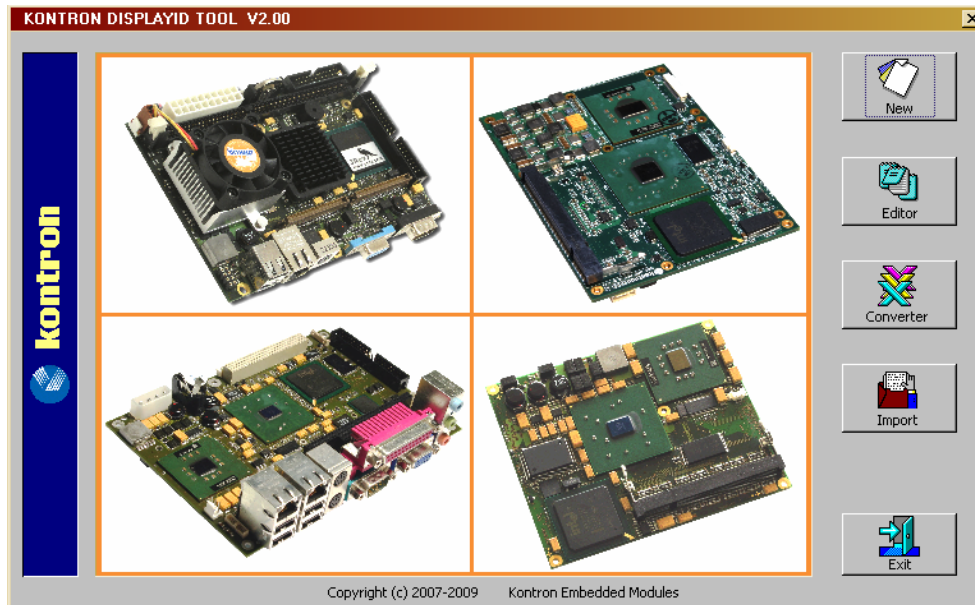
With NVIDIA's® graphic driver it is not guaranteed that every resolution can be achieved. KONTRON does not guarantee the correct function of the board for untypical resolution. In principle the use of DisplayID™ allows realizing every special display resolution. For this a valid DisplayID™ dataset must be written to the onboard EEPROM. Additionally the U-Boot Setup entry

setup display -> submenu [3] -> LCD resolution

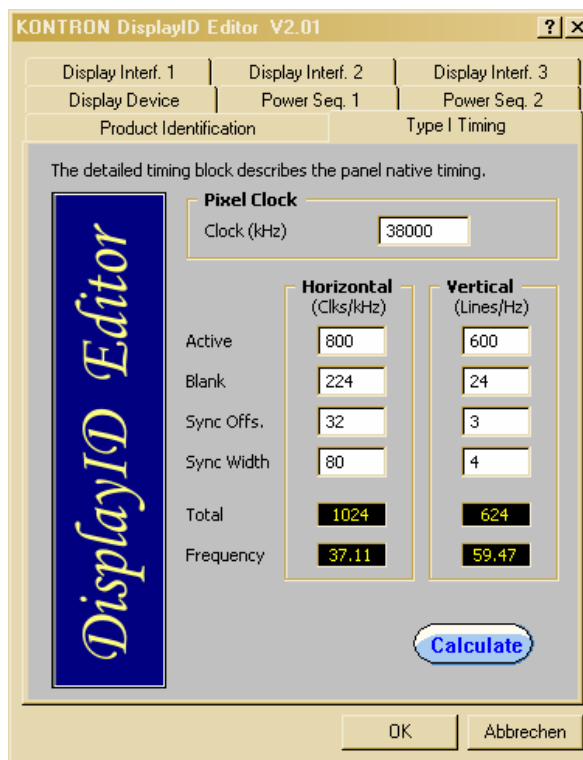
must be set to **auto**.

3.3.4 DisplayID™ Windows® Tool

The DisplayID™ parameter can be modified with the DisplayID™ Windows® tool.



For an example the following picture shows the input fields for the **Detailed Timing** parameters.



For more information see the documentation of the DisplayID™ tool (software can be downloaded from kontron.com).

The DisplayID™ Editor saves the parameters in a intermediate file format. The file extension is 'KDD' (Kontron DisplayID™ Data). This file format cannot be used to program the onboard EEPROM. For transferring this file format into the binary file format for the EEPROM apply the Converter.

3.3.5 Building DisplayID™ File

- ❶ Start the Windows® tool **DisplayID.exe**.
- ❷ Use the **Editor** if you want to modify an existing DisplayID™ file or select **New** to create a complete new record.
- ❸ Change respectively enter new parameters.
- ❹ Save the parameters in a file with the extension 'KDD'.
- ❺ Open the saved 'KDD'-file using the **Converter**.
- ❻ Save the binary file with the extension 'KDB' (Kontron DisplayID™ Binary).
- ❼ Program the onboard EEPROM using the board specific update tool.

3.3.6 Erasing DisplayID™ Record

Create a dummy file with a size of 128 bytes filled with the value 0xFFh and program this file using the U-Boot update tool. This treatment deletes a valid DisplayID™ record.

3.3.7 U-Boot EEPROM Update Tool

The update tool is a new component of the U-Boot bootloader. You need two commands to program a DisplayID™ file into the EEPROM:

- ❑ **ext2load**, **fatload**, **loadb** or **loady**
- ❑ **writedid**

File Operation

The following example gives an overview:

The storage medium is a USB key formatted with a Linux® partition and the DisplayID™ file **wvga.kdb** is located in the root directory. No other USB keys are present. The file size of **wvga.kdb** amounts 81 bytes. For loading the file into memory type the following standard U-Boot command line

```
ext2load usb 0 A00800 wvga.kdb
```

The memory address (0xA00800) is free selectable. With the **md** command you can control the result

```
md.b A00800 80
```

Now you can load the memory content into the EEPROM. Type the new KONTRON U-Boot command

```
writedid A00800 51
```

The 'count' respectively the size argument is a very important parameter. Do not use another value as the file size of your DisplayID™ file.

Serial Download

The following example demonstrates a serial download via the ymodem protocol:

The KTT20/pITX board is connected to a desktop computer with a suitable terminal program (e.g. HyperTerminal or TeraTerm). The file size of **wvga.kdb** amounts 81 bytes.

For downloading the file into memory type the following standard U-Boot command line

```
loady
```

Now U-Boot waits for reply. User input to the desktop terminal program starts the download session. After the download of **wvga.kdb** ends you can control the result with the **md** command (the memory address 0x408000 is fixed)

```
md.b 408000 80
```
























Now you can load the memory content into the EEPROM. Type the new KONTRON U-Boot command

```
writedid 408000 51
```





The 'count' respectively the size argument is a very important parameter. Do not use another value as the file size of your DisplayID™ file.

4 KTT20 Tool Package

The KTT20 Tool Package contains all needed drivers and tools as described below. A short overview:

<input type="checkbox"/>	ICreator	ImageCreator program (Windows®)
	Convert.exe	Console program (auxiliary program)
	ImageCreator.exe	Main program
	README.txt	Important additional information
	SPI.cfg	Configuration file
	SPI_Flasher.bin	Special U-Boot flash version
<input type="checkbox"/>	NVFLASH_Linux	
<input type="checkbox"/>	SDRAM	Executes U-Boot in SDRAM
	ktt20.sh	Shell script file
<input type="checkbox"/>	SPI	Programs U-Boot into SPI™ flash
	spi.sh	Shell script file
	KTT20.bct	Binary configuration table file
	KTT20.cfg	Configuration file
	nvflash	Main program
<input type="checkbox"/>	NVFLASH_Windows	
<input type="checkbox"/>	SDRAM	Executes U-Boot in SDRAM
	KTT20.bat	Batch script file
<input type="checkbox"/>	SPI	Programs U-Boot into SPI™ flash
	SPI.bat	Batch script file
	KTT20.bct	Binary configuration table file
	KTT20.cfg	Configuration file
	lipnv3p.dll	Auxiliary file
	lipnvaes_ref.dll	Auxiliary file
	lipnvapputil.dll	Auxiliary file
	lipnvboothost.dll	Auxiliary file
	lipnvdioconverter.dll	Auxiliary file
	lipnvflash.dll	Auxiliary file
	lipnvos.dll	Auxiliary file
	lipnvusbhost.dll	Auxiliary file
	nvflash.exe	Main program

USB-Driver_Windows **32bit** 32 bit Client Port Driver (CPD)

	NOTICE.txt	Copyright information
	WdfCoInstaller01009.dll	Driver file
	WinUSBCoInstaller2.dll	Driver file
	WUDFUpdate_01009.dll	Driver file

 64bit 64 bit Client Port Driver (CPD)

	NOTICE.txt	Copyright information
	WdfCoInstaller01009.dll	Driver file
	WinUSBCoInstaller2.dll	Driver file
	WUDFUpdate_01009.dll	Driver file
	CPDWinUSB.inf	Driver installation file

5 Bootloader Modification and Download

If you want to create your own bootloader and load it into the SPI™ flash device you must execute several steps or you use the KONTRON Windows® Image Creator. The KONTRON tool generates a special image file which can be downloaded with NVIDIAS® NVFLASH tool into the RAM. Thereafter one part of the image executes a script and program the bootloader into the SPI™ flash device.

ATTENTION

It is impossible to use another boot device as the SPI™ flash (e.g. the NAND flash) because the boot device is hardcoded.

The image file includes three modules:

- ❑ One special U-Boot version with a script (named SPI-FLASHER)
- ❑ One Binary Configuration Table (BCT), content definition by NVIDIA®
- ❑ One custom bootloader for the SPI™ flash device

The first component represents an unchangeable U-Boot version. Do not replace this module with another program.

The Binary Configuration Table is realized as an ASCII file for special usage of the KONTRON Image Creator which contains several parameters for SPI™ flash and SDRAM configuration. An extract:

```
Version=0x00020001;
BlockSize=0x00008000;
PageSize=0x00000800;
PartitionSize=0x01000000;
```

```
DevType[0]=NvBootDevType_Spi;
DeviceParam[0].SpiFlashParams.ReadCommandTypeFast=0;
DeviceParam[0].SpiFlashParams.ClockDivider=12;
DeviceParam[0].SpiFlashParams.ClockSource=NvBootSpiClockSource_PIIPOut0;
```

```
SDRAM[0].MemoryType=NvBootMemoryType_Ddr2;
SDRAM[0].PIIMChargePumpSetupControl=0x00000008;
```

.....

```
SDRAM[0].ApbMiscGpXm2CompPadCtrl=0x01f1f008;
SDRAM[0].ApbMiscGpXm2VttGenPadCtrl=0x07076600;
```

```
BootLoader=u-boot.bin,0x00108000,0x00108000,Complete;
```

There are some important restrictions:

- ❑ Do not change any SPI™ or SDRAM parameter (forbidden areas marked with red color)
- ❑ Do not use a <Space> character, bring all inputs together without a clearance
- ❑ Do not use comments

CAUTION

KONTRON does not repair a board free of charge if the SPI™ flash respectively SDRAM parameters are changed.

The third component implies your own bootloader. Maybe in the ASCII equivalent of the Binary Configuration Table the line labeled **BootLoader** must be changed. The line arguments are defined as follows:

bootloader = <bootloader file name>, <load address>, <entry point>, Complete

The string **Complete** is a fixed designator (not changeable) but the arguments **load address** and **entry point** are bootloader specific. Normally for U-Boot bootloaders both parameters corresponds with the value 0x00108000.

The KONTRON Image Creator package includes the following files:

- IMAGECREATOR.EXE main program
- CONVERT.EXE 32 bit console program (conversion utility)
- SPI-FLASHER.BIN special U-Boot version
- SPI.CFG ASCII equivalent for the Binary Configuration Table

Additional needed file:

- Your own bootloader the file name must be identical with **<bootloader file name>**

NOTICE: Do not use a 64 bit Windows® environment, the package runs only with Windows® XP and 32 bit Windows® 7.

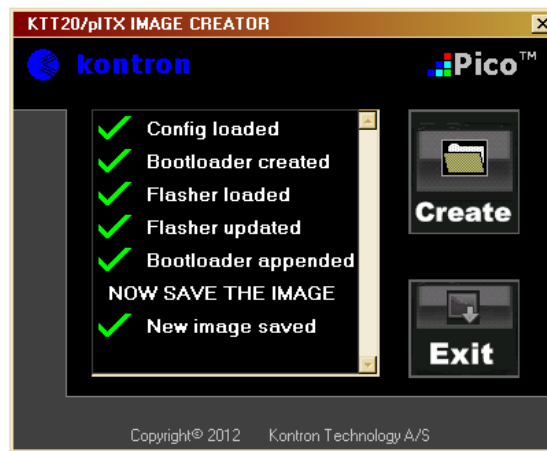
5.1 Program IMAGECREATOR

After the program start you see the following screen

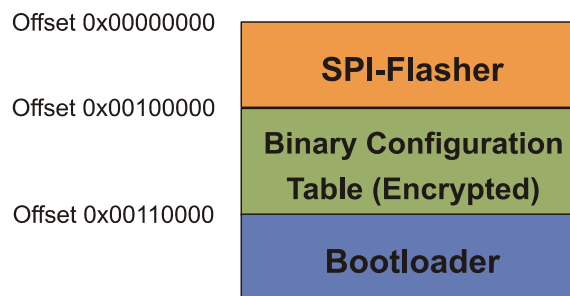


The **Create** button generates the required image. The first step loads the ASCII equivalent for the Binary Configuration Table (e.g. SPI.CFG) which includes the name of your own bootloader. This bootloader must exist in the same directory as the main program. The second step loads the SPI-FLASHER program and combines it with the Binary Configuration Table (now a really binary module) and your own bootloader.

If the execution ends successfully the following screen appears



The image file disposes of the following structure:



5.2 NVFLASH Download Tool

NVIDIA® provides a special tool named NVFLASH to download a bootloader to the target platform. This tool is available for Windows® XP, Windows® 7 and Linux®. The Windows® operating system needs an additional USB client driver (also known as APX) however Linux® comes with a built-in APX driver.

5.2.1 Windows® Operation

- ❶ Install the USB client driver:
 - ❑ For recovery mode first the recovery button J1400 (if you use two buttons) and then the power button J900 must be pressed until the LEDs go on (for details see the 'KTT20/pITX Users Guide' chapter 'Crisis Management').
 - ❑ Windows® reports the message 'New hardware found' and the ordinary hardware assistant appears. Now you can install the driver as usual with the [CPDWinUSB.inf](#) file. After the installation the device manager has a new entry in the USB path named [USB Client Port Driver \(CPD\)](#).

② Download the bootloader into SDRAM:

- If you want to develop your own U-Boot version use the command file from the directory 'SDRAM'. Copy your binary U-Boot file (default name: u-boot.bin), the file from the directory 'SDRAM' and all files from the NVFLASH root directory into the same directory.
- Execute the KTT20.bat file. The following screen output must appear:

```
Nvflash started
rcm version 0X20001
System Information:
  chip name: t20
  chip id: 0x20 major: 1 minor: 3
  chip sku: 0x8
  chip uid: .....
  macrovision: disabled
  hdcp: enabled
  sbk burned: false
  dk burned: false
  boot device: spi
  operating mode: 3
  device config strap: 0
  device config fuse: 0
  sdram config strap: 0

sending file: ktt20.bct

4080/4080 bytes sent
ktt20.bct sent successfully
downloading bootloader -- load address: 0x108000 entry point: 0x108000
sending file: u-boot.bin

..... / ..... bytes sent
u-boot.bin sent successfully
waiting for bootloader to initialize
bootloader failed NvError 0x0
command failure: bootloader download failed
Press enter to continue:
```

- Ignore the error messages. The 'nvflash' program expects some additional operations which are never executed. Now U-Boot is starting and you should see the bootloader serial output on your terminal program.

③ Download the bootloader into SPI™ flash

- If you want to flash a new final bootloader version or to repair a damaged bootloader use the command file from the directory 'SPI'. An important prerequisite is an existing image file as output from the 'ImageCreator' tool (default name: spi-flasher.img) or from a Kontron BSP. Copy your image file, the file from the directory 'SPI' and all files from the NVFLASH root directory into the same directory.

- ❑ Execute the SPI.bat file. The following screen output must appear:

```
Nvflash started
rcm version 0X20001
System Information:
  chip name: t20
  chip id: 0x20 major: 1 minor: 3
  chip sku: 0x8
  chip uid: .....
  macrovision: disabled
  hdcp: enabled
  sbk burned: false
  dk burned: false
  boot device: spi
  operating mode: 3
  device config strap: 0
  device config fuse: 0
  sdram config strap: 0

sending file: spi.bct

4080/4080 bytes sent
spi.bct sent successfully
downloading bootloader -- load address: 0x108000 entry point: 0x108000
sending file: spi-flasher.img

..... / ..... bytes sent
spi-flasher.img sent successfully
waiting for bootloader to initialize
bootloader failed NvError 0x0
command failure: bootloader download failed
Press enter to continue:
```

- ❑ Ignore the error messages. The 'nvflash' program expects some additional operations which are never executed. Now U-Boot is starting and you should see the bootloader serial output on your terminal program.
- ❑ To prevent data loss when switching off the board it is a good solution to control the programming progress over the serial output. You should see as a minimum:

```
SF: Detected SST25VF032B with page size 4096, total 4 MiB
4096 KiB SST25VF032B at 0:0 is now current device
ERASING...
WRITING...
```

5.2.2 Linux® Operation

- ❶ Download the bootloader into SDRAM:
 - ❑ If you want to develop your own U-Boot version use the script file from the directory 'SDRAM'. Copy your binary U-Boot file (default name: u-boot.bin), the file from the directory 'SDRAM' and all files from the NVFLASH root directory into the same directory.

- ❑ Execute the 'ktt20.sh' file. The following screen output must appear:

```
Nvflash started
rcm version 0X20001
System Information:
  chip name: t20
  chip id: 0x20 major: 1 minor: 3
  chip sku: 0x8
  chip uid: .....
  macrovision: disabled
  hdcp: enabled
  sbk burned: false
  dk burned: false
  boot device: spi
  operating mode: 3
  device config strap: 0
  device config fuse: 0
  sdram config strap: 0

sending file: KTT20.bct

4080/4080 bytes sent
KTT20.bct sent successfully
downloading bootloader -- load address: 0x108000 entry point: 0x108000
sending file: u-boot.bin

..... / ..... bytes sent
u-boot.bin sent successfully
waiting for bootloader to initialize
bootloader failed NvError 0x0
command failure: bootloader download failed
```

- ❑ Ignore the error messages. The 'nvflash' program expects some additional operations which are never executed. Now U-Boot is starting and you should see the bootloader serial output on your terminal program.

② Download the bootloader into SPI™ flash

- ❑ If you want to flash a new final bootloader version or to repair a damaged bootloader use the script file from the directory 'SPI'. An important prerequisite is an existing image file as output from the 'ImageCreator' tool (default name: spi-flasher.img) or from a Kontron BSP. Copy your image file, the file from the directory 'SPI' and all files from the NVFLASH root directory into the same directory.

- ❑ Execute the 'spi.sh' file. The following screen output must appear:

```
Nvflash started
rcm version 0X20001
System Information:
  chip name: t20
  chip id: 0x20 major: 1 minor: 3
  chip sku: 0x8
  chip uid: .....
  macrovision: disabled
  hdcp: enabled
  sbk burned: false
  dk burned: false
  boot device: spi
  operating mode: 3
  device config strap: 0
  device config fuse: 0
  sdram config strap: 0

sending file: SPI.bct

4080/4080 bytes sent
SPI.bct sent successfully
downloading bootloader -- load address: 0x108000 entry point: 0x108000
sending file: spi-flasher.img

..... / ..... bytes sent
spi-flasher.img sent successfully
waiting for bootloader to initialize
bootloader failed NvError 0x0
command failure: bootloader download failed
```

- ❑ Ignore the error messages. The 'nvflash' program expects some additional operations which are never executed. Now U-Boot is starting and you should see the bootloader serial output on your terminal program.
- ❑ To prevent data loss when switching off the board it is a good solution to control the programming progress over the serial output. You should see as a minimum:

```
SF: Detected SST25VF032B with page size 4096, total 4 MiB
4096 KiB SST25VF032B at 0:0 is now current device
ERASING...
WRITING...
```

6 SMSC® USB Hub and LAN Controller

The SMSC® controller LAN9514 has its own configuration EEPROM. This allows the automatic loading of static configuration data after reset. The EEPROM opens the ability to disable the onchip LAN controller but this option leads to a non-programmable state. This setting makes it impossible to reprogram the EEPROM. It is strongly recommended that the EEPROM content remains unchanged.

CAUTION

KONTRON does not repair a board free of charge if the LAN controller in the configuration EEPROM is disabled.

7 Alternative Linux® Distributions

If you want to use the installed U-Boot bootloader and only change the Linux® distribution three conditions should be fulfilled

- ❑ format the USB key or microSD™ card with `ext2` or `ext3`
- ❑ rename the kernel to `uImage`
- ❑ store the kernel in the root directory

One way to bypass the last two conditions is to change the U-Boot environment settings. The U-Boot command 'printenv' lists all variables, 'setenv' modifies the values and 'saveenv' stores the new environment. Now the default environment settings:

```
baudrate=115200
bootargs=mem=1024M@0M console=ttyS0,115200n8 console=tty0 lp0_vec=0x2000@0x1C406000 root=/dev/sda1 rootwait
bootargs.base=mem=1024M@0M console=ttyS0,115200n8 console=tty0 lp0_vec=0x2000@0x1C406000
bootcmd=run mmc_boot ; run usb_boot ;
bootdelay=3
bootfile=uImage
ethact=sms0
load_did=loady ; writedid ${loadaddr} ${filesize}
loadaddr=0x408000
mmc_boot=run mmc_setup; mmc rescan ${mmcdev}; ext2load mmc ${mmcdev} ${loadaddr} ${bootfile}; bootm ${loadaddr}
mmc_setup=setenv bootargs ${bootargs.base} root=/dev/mmcblk0p1 rootwait
mmcdev=0
nfs_boot=run nfs_setup; usb start; dhcp; bootm ${loadaddr}
nfs_setup=setenv bootargs ${bootargs.base} root=/dev/nfs ip=dhcp
stderr=serial,lcd
stdin=serial
stdout=serial,lcd
usb_boot=run usb_setup; usb start; ext2load usb ${usbdev} ${loadaddr} ${bootfile}; bootm ${loadaddr}
usb_setup=setenv bootargs ${bootargs.base} root=/dev/sda1 rootwait
usbdev=0
```

8 U-Boot Compilation

Kontron Technology does not provide any support for the free U-Boot version.

You can find a suitable compiler without major effort. One option is to use the Linaro™ ARM® compiler downloadable from the internet address

<http://www.linaro.org/>

You need only four command lines to compile the U-Boot sourcecode. For an example:

```
export PATH=/<compiler path>/arm-none-eabi-gcc-4_6/bin/:$PATH
make distclean CROSS_COMPILE=arm-none-eabi-
make harmony_config CROSS_COMPILE=arm-none-eabi-
make all CROSS_COMPILE=arm-none-eabi-
```

The KTT20/pITX board be based on NVIDIAS® 'Harmony' evaluation board. This explains the compiler switch `harmony_config`. The appendant U-Boot sourcecode is downloadable from the internet address

<http://git.denx.de/?p=u-boot.git;a=summary>

Another possibility:

<http://nv-tegra.nvidia.com/gitweb/?p=3rdparty/u-boot.git;a=summary>

The directory 'include/configs' contains a really important file named 'harmony.h'. For adaption some changes are necessary.

```
/*
 * (C) Copyright 2010, 2011
 * NVIDIA Corporation <www.nvidia.com>
 *
 * See file CREDITS for list of people who contributed to this project.
 *
 * This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public
 * License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any
 * later version.
 *
 * This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the
 * implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public
 * License for more details.
 *
 * You should have received a copy of the GNU General Public License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
 */

#ifdef __CONFIG_H
#define __CONFIG_H

#include <asm/sizes.h>
#include "tegra2-common.h"
```

```
/* High-level configuration options */
#define TEGRA2_SYSMEM          "mem=384M@0M nvmem=128M@384M mem=512M@512M"
#define V_PROMPT              "Tegra2 (Harmony) # "

.....

/* Board-specific serial config */
#define CONFIG_SERIAL_MULTI
#define CONFIG_TEGRA2_ENABLE_UARTC
#define CONFIG_SYS_NS16550_COM1          NV_PA_APB_UARTC_BASE

#define CONFIG_TEGRA2_BOARD_STRING      "NVIDIA Harmony"
#define CONFIG_MACH_TYPE                MACH_TYPE_HARMONY
#define CONFIG_SYS_BOARD_ODMDATA       0x300d8011 /* lp1, 1GB */
#define CONFIG_BOARD_EARLY_INIT_F

/* SD/MMC */
#define CONFIG_MMC
#define CONFIG_GENERIC_MMC
#define CONFIG_TEGRA2_MMC
#define CONFIG_CMD_MMC

#define CONFIG_DOS_PARTITION
#define CONFIG_EFI_PARTITION
#define CONFIG_CMD_EXT2
#define CONFIG_CMD_FAT

/* Environment not stored */
#define CONFIG_ENV_IS_NOWHERE
#endif /* __CONFIG_H */
```


8.1 Hardware Components Compatibility

Between the KTT20/pITX and the 'Harmony' evaluation board there are following conformities/differences:

Component	KTT20/pITX vs. 'Harmony' Evalboard
SDRAM	Compatible
NAND	Compatible
SPI™ Flash	Compatible
PMU	Compatible
LAN	Compatible
DVI®	DVI® vs. HDMI®
CRT	Compatible
LVDS	Compatible
USB Client	Compatible
USB	Nearly compatible
microSD™ (MMC)	Compatible
Audio	Compatible
PCI Express®	Compatible
I²C™	Not compatible
UARTs	Not compatible
GPIOs	Not compatible
Temperature Sensor	Compatible

8.2 GPIO Declarations

The directory 'drivers/gpio' contains a module named 'tegra2_gpio.c'. It disposes of all needed routines to declare a GPIO as input or output respectively to read or write a value. An overview:

```
int gpio_request           // Configure as GPIO (only in older versions)
int gpio_direction_input  // Set GPIO as input
int gpio_direction_output // Set GPIO as output
int gpio_get_value        // Read input value
void gpio_set_value       // Set output value
```

Example (symbolic names see 'arch/arm/include/asm/arch-tegra2/gpio.h'):

```
// gpio_request is a function from older U-Boot versions - possibly not necessary
ret = gpio_request (GPIO_PT1, NULL);           // define GPIO33 as GPIO, also labeled T.01
if (ret) {
    do something;                               // Error
}
gpio_direction_input (GPIO_PT1);               // Set GPIO as input
ret = gpio_get_value (GPIO_PT1);               // Read value, result is always Bit 0
```

```

// gpio_request is a function from older U-Boot versions - possibly not necessary
ret = gpio_request (GPIO_PD6, NULL);           // define GPIO30 as GPIO, also labeled D.06
if (ret) {
    do something;                               // Error
}
gpio_direction_output (GPIO_PD6, 0);           // Set GPIO as output with low level
udelay (1000);                                 // Wait 1 ms
gpio_set_value (GPIO_PD6, 1);                 // Set high level

```

8.3 UART Declarations

For an extensive usage of UARTs the module 'board.c' in the directory 'board/nvidia/common' must be changed. For example:

```

enum {
    // UARTs which we can enable
    UARTA = 1 << 0,
    UARTB = 1 << 1,
    UARTC = 1 << 2,
    UARTD = 1 << 3,
};

static void clock_init_uart (int uart_ids)
{
    if (uart_ids & UARTA)
        enable_uart (PERIPH_ID_UART1);
    if (uart_ids & UARTB)
        enable_uart (PERIPH_ID_UART2);
    if (uart_ids & UARTC)
        enable_uart (PERIPH_ID_UART3);
}

static void pin_mux_uart (int uart_ids)
{
    if (uart_ids & UARTA) {
        pinmux_set_func (PINGRP_UAA, PMUX_FUNC_UARTA);
        pinmux_tristate_disable (PINGRP_UAA);
    }
    if (uart_ids & UARTB) {
        pinmux_set_func (PINGRP_UAD, PMUX_FUNC_IRDA);
        pinmux_set_func (PINGRP_IRRX, PMUX_FUNC_UARTB);
        pinmux_set_func (PINGRP_IRTX, PMUX_FUNC_UARTB);
        pinmux_tristate_disable (PINGRP_UAD);
        pinmux_tristate_disable (PINGRP_IRRX);
        pinmux_tristate_disable (PINGRP_IRTX);
    }
    if (uart_ids & UARTC) {
        pinmux_set_func (PINGRP_UCA, PMUX_FUNC_UARTC);
        pinmux_tristate_disable (PINGRP_UCA);
        pinmux_set_func (PINGRP_UCB, PMUX_FUNC_UARTC);
        pinmux_tristate_disable (PINGRP_UCB);
    }
}

```

```

#ifdef CONFIG_BOARD_EARLY_INIT_F
int board_early_init_f (void)
{
    int uart_ids = 0;           // bit mask of which UART ids to enable

#ifdef CONFIG_TEGRA2_ENABLE_UARTA
    uart_ids |= UARTA;
#endif
#ifdef CONFIG_TEGRA2_ENABLE_UARTB
    uart_ids |= UARTB;
#endif
#ifdef CONFIG_TEGRA2_ENABLE_UARTC
    uart_ids |= UARTC;
#endif
}
.....

```

Add some entries to 'include/configs/harmony.h. For example:

```

#define CONFIG_TEGRA2_ENABLE_UARTA
#define CONFIG_TEGRA2_ENABLE_UARTB
#define CONFIG_TEGRA2_ENABLE_UARTC
#define CONFIG_SYS_NS16550_COM1          NV_PA_APB_UARTC_BASE
#define CONFIG_SYS_NS16550_COM2          NV_PA_APB_UARTA_BASE
#define CONFIG_SYS_NS16550_COM3          NV_PA_APB_UARTB_BASE

```

8.4 Linux® Environment on Windows®

If you want to implement a Linux® environment on a Windows® operating system the ORACLE® VirtualBox® is a good solution. For further information see

<https://www.virtualbox.org/>

VirtualBox® supports all USB interfaces from the host and configures the guest operating system with a virtual USB controller. Likely there are some exceptions, for example the access to the [Client Port Driver \(CPD\)](#).

9 Linux® BSP

9.1 User Login Arguments

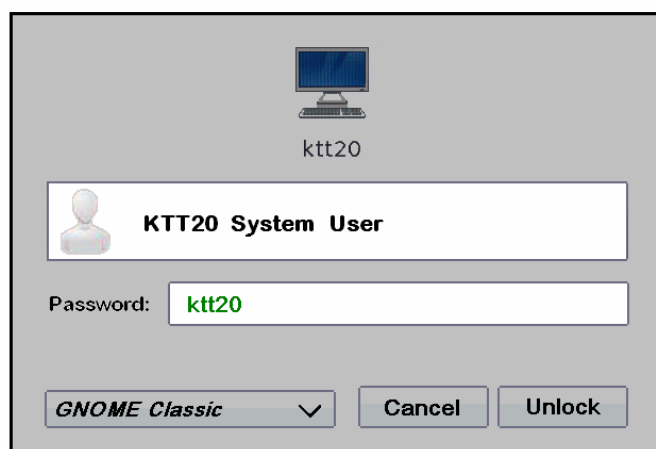
For both, serial remote system or normal display manager login, the arguments are the same.

Serial remote system (necessary input with green color):

ktt20 login: **ktt20**

Password: **ktt20**

Display manager login (necessary input with green color):



Note: there is no 'root' password defined. If you need root privileges use 'sudo' and the user password.

9.2 Video Decoding

Hardware accelerated video decoding requires the [GStreamer/OpenMAX](#) framework and a special NVIDIA® [nvgstplayer](#) application or [gst-launch](#) as a command line tool. KONTRON Technology A/S does not know another programs which can play a H.264 main profile video stream (720p) without dropouts (e.g. Totem Movie Player, MPlayer or VLC).

The following table gives an overview about the limitations:

Video Format	Profile / Level	Max. Resolution / Frame Rate	Max. Throughput	Comment
DivX 4/5/6	1080p HD	1920 x 1080 / 30 fps	10 Mbps	
Xvid	Highdef	1920 x 1080 / 30 fps	10 Mbps	
MPEG-4	Advanced Simple / L4	1920 x 1080 / 30 fps	10 Mbps	
H.264	Baseline (BP) / L4	1920 x 1080 / 30 fps	10 Mbps	
H.264	Main (MP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CAVLC
H.264	Main (MP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CABAC, CAVLC
H.264	High (HiP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CAVLC
H.264	High (HiP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CABAC, CAVLC

Please note that the maximal throughput data are peak values not average values. You may need to re-encode your videos with parameters above to get smooth video output.

ATTENTION

The VC1 video codec and AAC audio codec are not supported (for AAC non-accelerated software decoding can be used).
Exception: The gst-launch tool can handle AAC audio streams.

Only three methods ('sinks' in gstreamer terminology) are supported:

- ❑ `nv_gl_eglimagesink` or `nvximagesink` (**DVI® monitor**)
- ❑ `nv_omx_videosink` (**LVDS panel**)

The operating system contains some installed packages

- ❑ `libgstreamer0.10-0`
- ❑ `gnome-media`

but not the 'gstreamer-tools' (necessary for usage of 'gst-launch'). Add this package with

```
(sudo) apt-get install gstreamer0.10-tools
```

You can get more information with the command line instructions

```
gst-inspect-0.10 | grep sink
gst-inspect-0.10 | grep decode
```

Note: The gst-launch player supports video files with AAC audio content (e.g. `big_buck_bunny_....mov`).

Another command line tool named `gstreamer-properties` allows the video and audio configuration for applications which uses the gstreamer interface (e.g. Totem Movie Player).

9.2.1 Command Line Examples

For smooth video output you must pass several arguments otherwise the program generates dropouts.

- ❑ **Playing H.264 video on a DVI® monitor:**

```
nvgstplayer -i <filename> --svs nv_gl_eglimagesink --sas faad
```
- ❑ **Playing DivX video with MP3 audio stream on a LVDS panel:**

```
nvgstplayer -i <filename> --svs nv_omx_videosink
```
- ❑ **Playing H.264 video with AAC audio stream on a DVI® monitor:**

```
gst-launch-0.10 filesrc location=<filename> ! qtdemux name=q ! queue !
nv_omx_h264dec ! nvximagesink q. ! queue ! faad ! alsasink
```
- ❑ **Playing H.264 video with AAC audio stream on a LVDS panel:**

```
gst-launch-0.10 filesrc location=<filename> ! qtdemux name=q ! queue !
nv_omx_h264dec ! nv_omx_videosink q. ! queue ! faad ! alsasink
```

You can create a shell script for example 'video.sh':

```
#!/bin/sh
gst-launch-0.10 filesrc location=$1 ! qtdemux name=q ! queue ! nv_omx_h264dec ! nvximagesink q. ! queue ! faad ! alsasink
```

and call it with `./video.sh <filename>`

9.2.2 Reencoding Examples

With the `ffmpeg` tool you can reencode video files which do not comply the criteria listed above.

- ❑ **Create H.264 baseline profile, avg bitrate 15 Mbps, peak bitrate 20 Mbps:**

```
ffmpeg -y -i ./video.mov -vcodec libx264 -profile baseline -b 15M-maxrate 20M -bufsize 1830k
-acodec copy ./video_h264_max20_aver15.mov
```

- ❑ **Create Xvid, avg bitrate 8 Mbps, peak bitrate 10 Mbps:**

```
ffmpeg -y -i ./video.mov -vcodec mpeg4 -vtag xvid -b 8M -maxrate 10M-bufsize 1830k
-acodec copy ./video_xvid_max10_aver8.avi
```

9.3 Audio Settings

If you have trouble with audio input/output please check the 'alsamixer' settings.

Examples with the command line tool 'amixer':

- ❑ For an overview type `amixer contents`
- ❑ To change a value type e.g. `amixer cset numid=46 on`

You can find some audio files in the directory `usr/share/sounds/...`. Play these files with the 'aplay' tool, for example `aplay ring.wav`.

The following table gives an indication to solve the problem(s).

Control	Name	Value
1	Left Input PGA Switch	off
2	Left Input PGA Volume	15
3	Left Input PGA Common Mode Switch	off
4	Right Input PGA Switch	off
5	Right Input PGA Volume	15
6	Right Input PGA Common Mode Switch	off
7	ADC OSR	High performance
8	HPF Switch	off
9	HPF Mode	Hi-fi
10	DRC Switch	off
11	DRC Compressor Slope R0	1/16
12	DRC Compressor Slope R1	1
13	DRC Compressor Threshold Volume	124
14	DRC Volume	15
15	DRC Minimum Gain Volume	0
16	DRC Maximum Gain Volume	0
17	DRC Attack Rate	1.45ms
18	DRC Decay Rate	743ms
19	DRC FF Delay	9 samples
20	DRC Anticlip Switch	on
21	DRC QR Switch	on
22	DRC QR Threshold Volume	2

23	DRC QR Decay Rate	0.725ms
24	DRC Smoothing Switch	on
25	DRC Smoothing Hysteresis Switch	on
26	DRC Smoothing Threshold	Medium
27	DRC Startup Volume	6
28	Digital Capture Volume	<both> 60
29	ADC Companding Mode	ulaw
30	ADC Companding Switch	off
31	Digital Sidetone Volume	<both> 0
32	DAC OSR	Low power
33	Digital Playback Volume	<both> 120
34	DAC Soft Mute Rate	Fast (fs/2)
35	DAC Mute Mode	Soft
36	DAC Mono Switch	off
37	DAC Companding Mode	ulaw
38	DAC Companding Switch	off
39	Playback Deemphasis Switch	off
40	Headphone Switch	<both> on
41	Headphone ZC Switch	<both> off
42	Headphone Volume	<both> 0
43	Line Out Switch	<both> on
44	Line Out ZC Switch	<both> off
45	Line Out Volume	<both> 45
46	Speaker Switch	<both> on
47	Speaker ZC Switch	<both> off
48	Speaker Volume	<both> 45
49	Right Speaker Mixer DACL Switch	on
50	Right Speaker Mixer DACR Switch	on
51	Right Speaker Mixer Left Bypass Switch	off
52	Right Speaker Mixer Right Bypass Switch	off
53	Left Speaker Mixer DACL Switch	on
54	Left Speaker Mixer DACR Switch	on
55	Left Speaker Mixer Left Bypass Switch	off
56	Left Speaker Mixer Right Bypass Switch	off
57	Right Output Mixer DACL Switch	on
58	Right Output Mixer DACR Switch	on
59	Right Output Mixer Left Bypass Switch	off
60	Right Output Mixer Right Bypass Switch	off
61	Left Output Mixer DACL Switch	on
62	Left Output Mixer DACR Switch	on
63	Left Output Mixer Left Bypass Switch	off
64	Left Output Mixer Right Bypass Switch	off
65	Right Playback Mux	Right
66	Left Playback Mux	Left

67	DACR Sidetone	None
68	DACL Sidetone	None
69	Right Capture Mux	Right
70	Left Capture Mux	Left
71	ADC Input	ADC
72	Right Input Mode Mux	Single-Ended
73	Right Input Inverting Mux	IN1R
74	Right Input Mux	IN1R
75	Left Input Mode Mux	Single-Ended
76	Left Input Inverting Mux	IN1L
77	Left Input Mux	IN1L
78	Int Spk Switch	on

9.4 PCI Express® Interface

Some mini PCI Express® cards cause a malfunction (e.g. card not detected or interrupt assignment not possible). One way to bypass this issue is to decrease the clock frequency. For this purpose you can expand the U-Boot environment with a new kernel argument `low_pcie_freq`.

Observe the following procedure:

- ❑ Interrupt the kernel boot process in U-Boot.
- ❑ Type `editenv bootargs.base` and press Enter.
- ❑ Add a space and then the new argument. Press Enter.
- ❑ Type `saveenv` and press Enter.

Example:

```
Kontron KTT20 # editenv bootargs.base
edit: mem=1024M@0M console=ttyS0,115200n8 console=tty0 lp0_vec=0x2000@0x1C406000 low_pcie_freq
Kontron KTT20 # saveenv
Saving Environment to SPI Flash...
SF: Detected SST25VF032B with page size 4096, total 4 MiB
Erasing SPI flash...Writing to SPI flash...done
Kontron KTT20 #
```

In the same way you can remove this new argument with the backspace key.

9.5 CPU Frequency Management

Linux® uses as default the frequency setting `Ondemand` which is not optimal for some applications (e.g. video decoding). The Linux® applet `indicator-cpufreq` provides four different CPU frequency modes:

Conservative, Ondemand, Powersave and Performance.

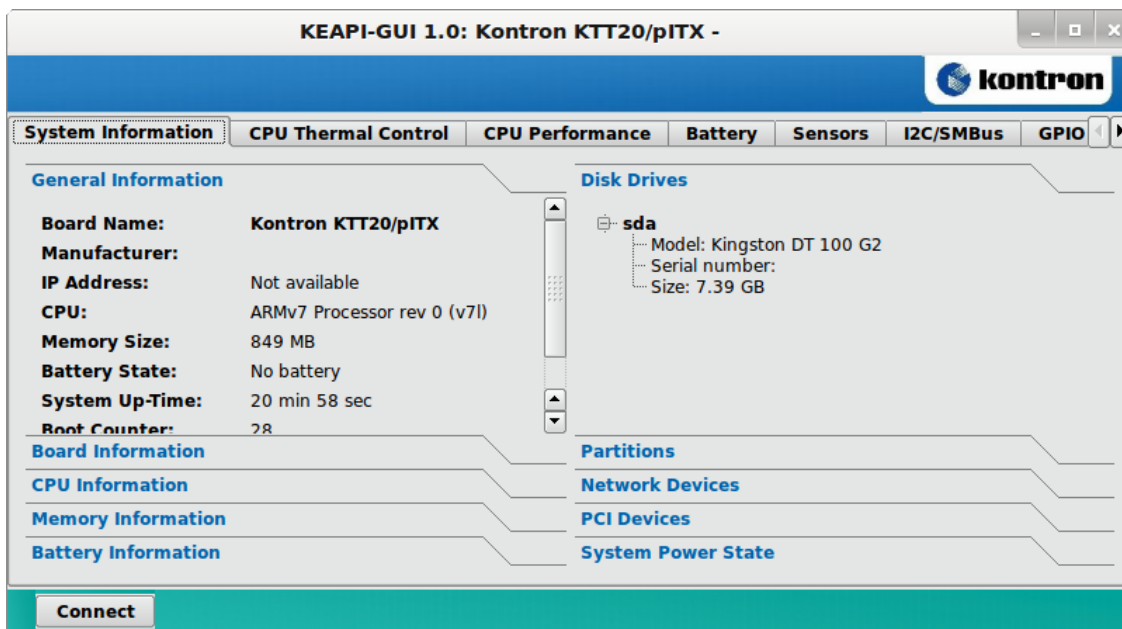
Download this tool with the following terminal commands:

```
(sudo) add-apt-repository ppa:artfwo/ppa
(sudo) apt-get update
(sudo) apt-get install indicator-cpufreq
```

After download you should reboot the operating system.

9.6 KEAPI Interface

For test and demonstration purposes you can find a GUI tool with the menu entries [Accessories - Files - File System](#) in the directory `usr/bin`. Double-click on the `keapi-gui` script, click on the button `Run` and after password input the following screen output appears:

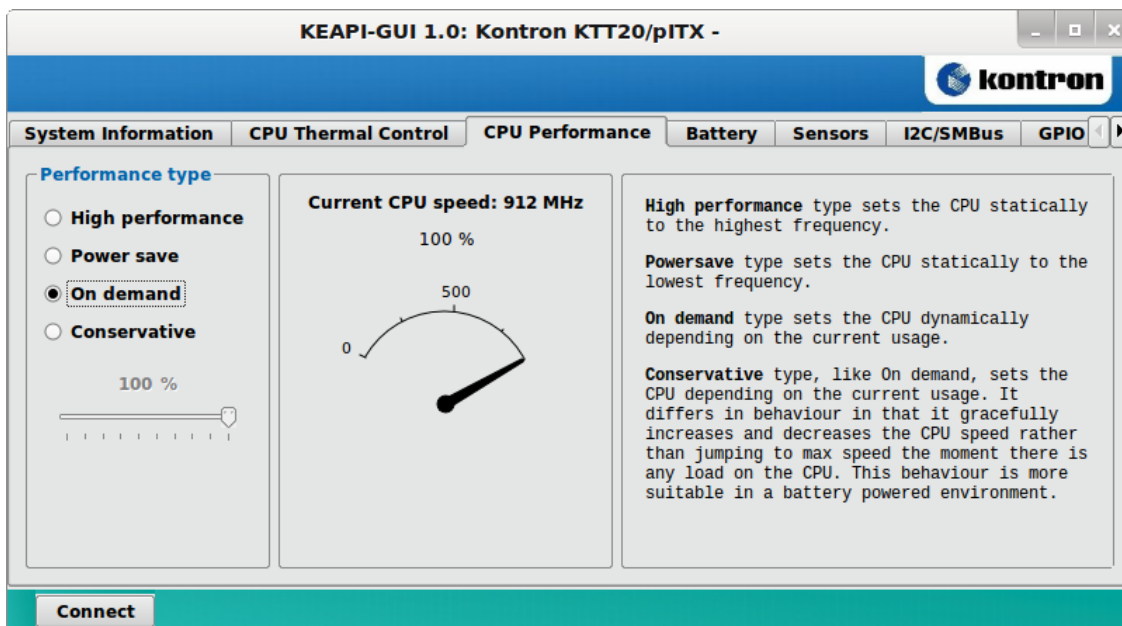


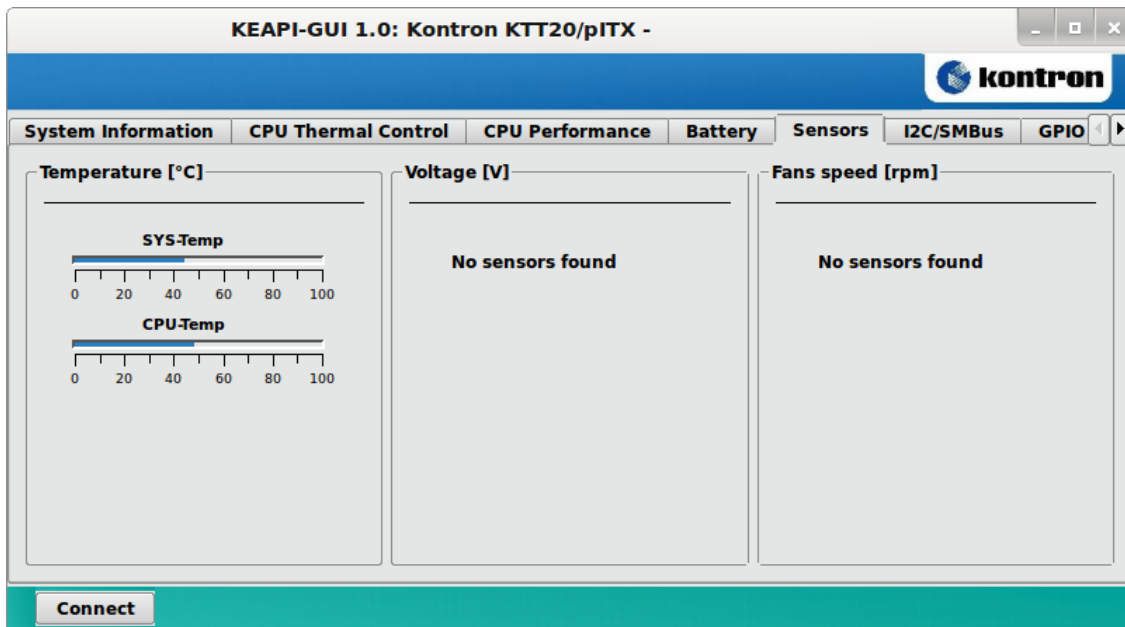
ATTENTION

Do not call directly the `keapi-gui.real` application. Without the root privileges only few keapi functions are ready for use.

In general: all KEAPI functions need root privileges !

The next two pictures demonstrate some features.

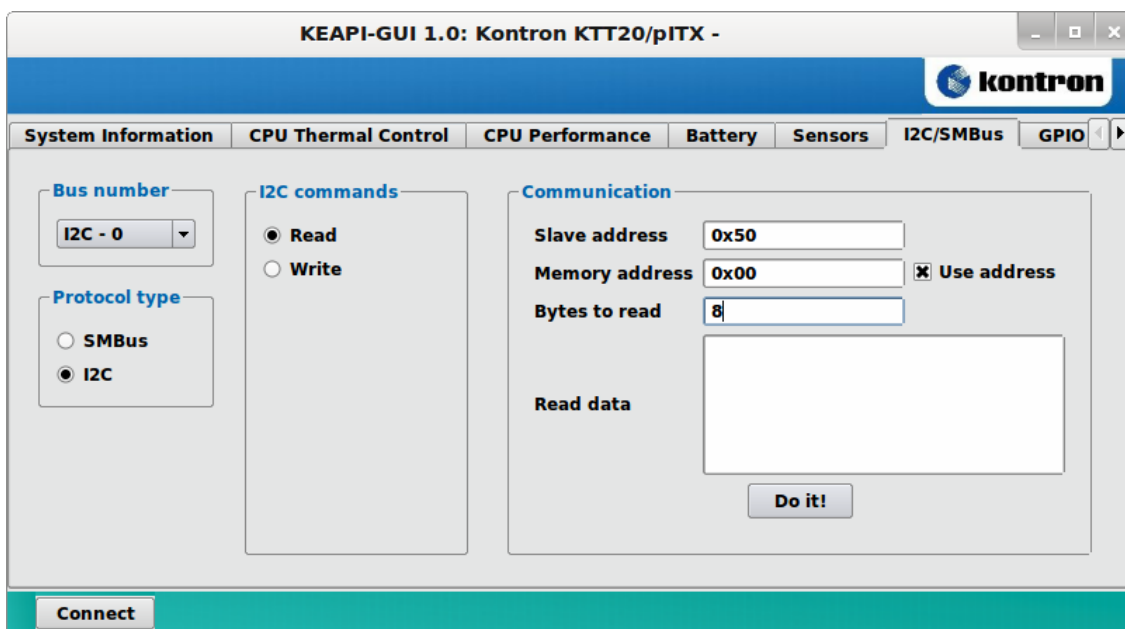




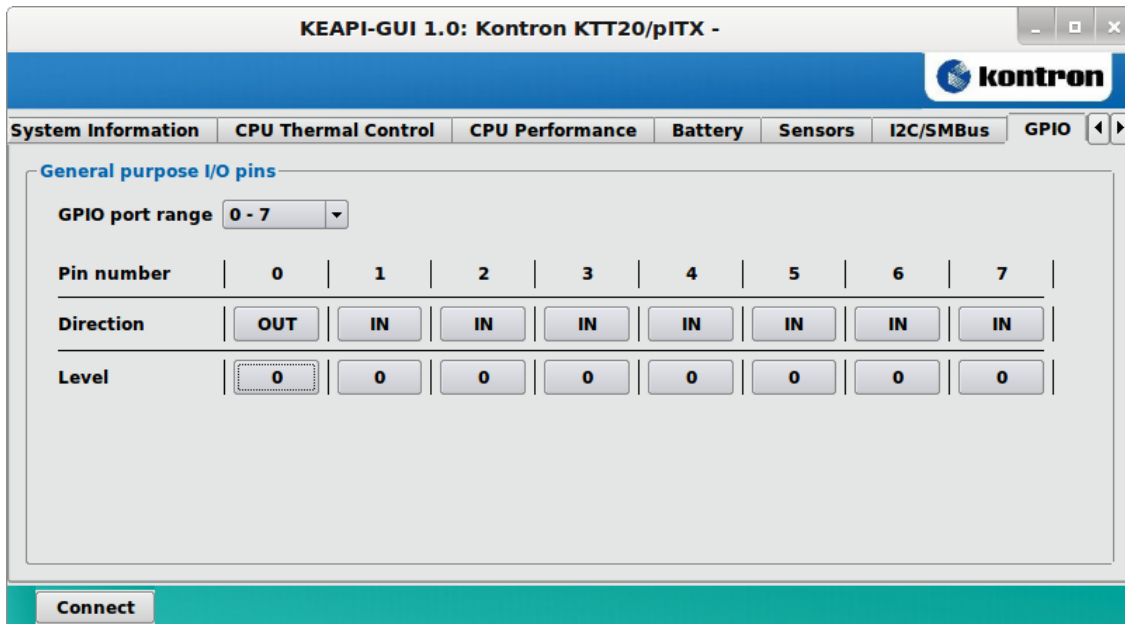
For the I²C™ part some restrictions are valid:

- Do not access board devices (e.g. temperature sensor - this device has an own interface). Especially this part should be used for external components on the digital I/O (GPIO) or mini PCI Express® connector.
- Some datasheets present the device address as a shifted value (e.g. EEPROM address = 0xA0). Use instead the unshifted value (e.g. real EEPROM address = 0x50).
- Bus 1 allows access to the DVI® DDC lines. Do not overwrite the EDID EEPROM.

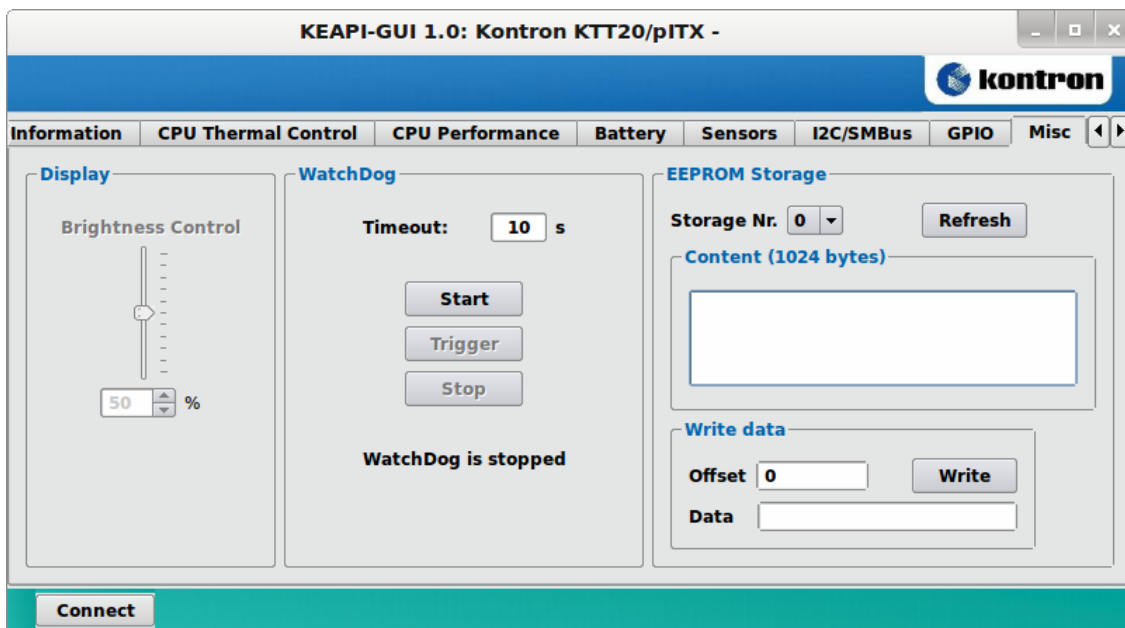
I²C™ bus assignment: **Bus 0** ⇒ mini PCI Express® connector **Bus 1** ⇒ DVI® DDC lines
Bus 2 ⇒ Digital I/O connector **Bus 3** ⇒ Reserved



For digital I/O pin assignment see the 'KTT20/pITX Users Guide' chapter 'Digital I/O Interface'.



The display part is only available if you use a LVDS panel.



9.6.1 KEAPI Command Line Tools

In the same directory where the GUI tool is located you can find the following command line tools (for help screen type the command without arguments):

Note: some modules are pointless (do not use the grayed out tools - they are only available for compatibility purposes).

`keapi-battery`

`keapi-cpu`

`keapi-disk`

`keapi-display`

Only LVDS panel support

`keapi-eeeprom`

`keapi-fan`

`keapi-general`

`keapi-gpio`

`keapi-i2c`

`keapi-memory`

`keapi-netdev`

Network controller support

`keapi-pciddev`

PCI Express® card support

`keapi-smbus`

Same functionality as keapi-i2c

`keapi-temp`

`keapi-voltage`

`keapi-watchdog`

For application programming see the KEAPI interface documentation (KEAPI_spec_v... .pdf).

10 Android™ BSP

10.1 Graphics Interface

10.1.1 DVI® Monitor

The Android™ graphic driver does not offer full EDID support. In most cases two supported resolutions (SXGA and Full-HD) are enough. You need to add the entry '[persist.tegra.hdmi.resolution](#)' in the system file '[build.prop](#)', detectable in the directory `/system`. The driver accepts following values:

Vga, 480p, 576p, 720p, SXGA and 1080p

Example for Full-HD:

```
persist.tegra.hdmi.resolution=1080p
```

10.1.2 LCD Panel

There are no restrictions regarding the supported resolutions. With DisplayID all panels with single channel can be attached.

10.2 Video Decoding

The integrated video player in the operating system allows hardware accelerated video decoding but this player offers only some video formats. Another good choice represents the MX Video Player.

The following table gives an overview about the limitations:

Video Format	Profile / Level	Max. Resolution / Frame Rate	Max. Throughput	Comment
DivX 4/5/6	1080p HD	1920 x 1080 / 30 fps	10 Mbps	
Xvid	Highdef	1920 x 1080 / 30 fps	10 Mbps	
MPEG-4	Advanced Simple / L4	1920 x 1080 / 30 fps	10 Mbps	
H.264	Baseline (BP) / L4	1920 x 1080 / 30 fps	10 Mbps	
H.264	Main (MP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CAVLC
H.264	Main (MP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CABAC, CAVLC
H.264	High (HiP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CAVLC
H.264	High (HiP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CABAC, CAVLC

Please note that the maximal throughput data are peak values not average values. You may need to re-encode your videos with parameters above to get smooth video output.

ATTENTION

The VC1 video codec is not supported

10.3 Display Density

In the system file `'/system/build.prop'` the entry `'ro.sf.lcd_density'` can change the density on both graphic units (DVI[®] and LCD panel) for a better look. The lower the value (e.g. 180, 160 or 130), the higher density and smaller font your screen will have but it is possible that the navigation bar disappears.

10.4 GPIOs, Temperatures, Backlight and Bootcounter

The Android™ BSP does not support the KEAPI interface. Some special functions are realized with the `sysfs` in-memory filesystem which allows the kernel to export information to a user space. All following examples can be executed with the serial remote connection (e.g. with TeraTerm on a desktop PC).

10.4.1 GPIOs

For correlation between GPIO numbers and physical pins refer the 'KTT20/pITX Users Guide' chapter 'Digital I/O Interface'.

- ★ Please use the export function before you access a GPIO pin.

```
echo <gpio num> >/sys/class/gpio/export
```

Example for GPIO16:

```
echo 16 >/sys/class/gpio/export
```

- ★ Check the GPIO pin direction:

```
cat /sys/class/gpio/gpio<gpio num>/direction
```

Example for GPIO16:

```
cat /sys/class/gpio/gpio16/direction
```

- ★ Set the GPIO pin direction:

```
echo in >/sys/class/gpio/gpio<gpio num>/direction      set direction to input
echo out >/sys/class/gpio/gpio<gpio num>/direction     set direction to output
```

Example for GPIO16:

```
echo in >/sys/class/gpio/gpio16/direction
echo out >/sys/class/gpio/gpio16/direction
```

- ★ Get an input value:

```
cat /sys/class/gpio/gpio<gpio num>/value
```

Example for GPIO16:

```
cat /sys/class/gpio/gpio16/value
```

- ★ Set an output value:

```
echo -n 0 >/sys/class/gpio/gpio<gpio num>/value       set output to low level
echo -n 1 >/sys/class/gpio/gpio<gpio num>/value       set output to high level
```

Example for GPIO16:

```
echo -n 0 >/sys/class/gpio/gpio16/value
echo -n 1 >/sys/class/gpio/gpio16/value
```

- ★ Release the GPIO pin:

```
echo <gpio num> >/sys/class/gpio/unexport
```

Example for GPIO16:

```
echo 16 >/sys/class/gpio/unexport
```

10.4.2 Temperatures

The board provides two temperatures: CPU (*temp2*) and sensor onchip (*temp1*). For temperature display in °C divide the returned values by 1000.

- ★ Get the sensor onchip temperature:

```
cat /sys/class/hwmon/hwmon0/device/temp1_input
```

- ★ Get the CPU temperature:

```
cat /sys/class/hwmon/hwmon0/device/temp2_input
```

10.4.3 Backlight

The backlight functionality is only available if you have activated the LCD panel support in U-Boot.

- ★ Get the brightness value:

```
cat /sys/class/backlight/pwm-backlight/brightness
```

- ★ Set a new brightness value (range: 0 to 255):

```
echo <value> >/sys/class/backlight/pwm-backlight/brightness
```

Example:

```
echo 64 >/sys/class/backlight/pwm-backlight/brightness
```

10.4.4 Bootcounter

It can be useful to read the bootcounter.

- ★ Get the bootcounter value:

```
cat /sys/bus/i2c/devices/2-0050/bootcounter
```

11 Windows® Embedded Compact 7 (WEC7) BSP

11.1 U-Boot Settings

The default environment settings intend to load a Linux® or Android™ image from an ext2/ext3 filesystem and not from a FAT/FAT32 partition. Two environment arguments have to be changed: `bootfile` and `mmc_boot` or `usb_boot`. Remove the boot device with the operating system or skip the autoboot operation with any key. Then type '`printenv`' and you see the default arguments:

```
bootfile=uImage
mmc_boot=run mmc_setup; mmc rescan ${mmcdev}; ext2load mmc ${mmcdev} ${loadaddr} ${bootfile}; bootm ${loadaddr}
usb_boot=run usb_setup; usb start; ext2load usb ${usbdev} ${loadaddr} ${bootfile}; bootm ${loadaddr}
```

After the edition you should check the modifications with '`printenv`' but as the most important action U-Boot requires the storage of the environment arguments in the SPI™ flash with '`saveenv`'. The result screen looks as follows:

```
Saving Environment to SPI Flash...
SF: Detected SST25VF032B with page size 4096, total 4 MiB
Erasing SPI flash...Writing to SPI flash...done
```

11.1.1 Boot from microSD Card

Change the arguments with '`editenv`':

```
bootfile=nk.nb0
mmc_boot=dcache off; run mmc_setup; mmc rescan ${mmcdev}; fatload mmc ${mmcdev} ${loadaddr} ${bootfile}; go
${loadaddr}
```

11.1.2 Boot from USB key

Change the arguments with '`editenv`':

```
bootfile=nk.nb0
usb_boot=dcache off; run usb_setup; usb start; fatload usb ${usbdev} ${loadaddr} ${bootfile}; go ${loadaddr}
```


11.2 Video Decoding

The integrated video player in the operating system allows hardware accelerated video decoding. The following table gives an overview about the limitations:

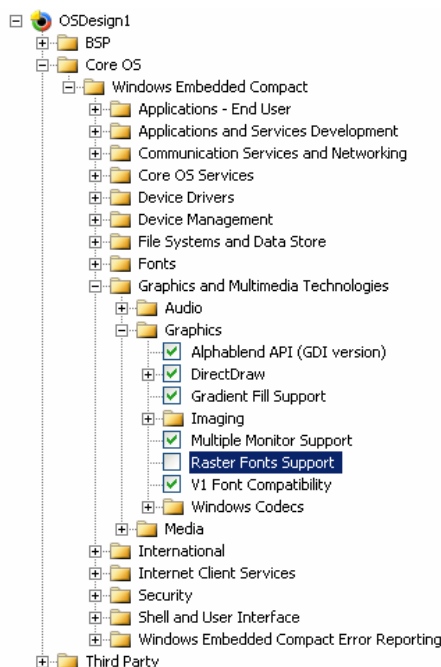
Video Format	Profile / Level	Max. Resolution / Frame Rate	Max. Throughput	Comment
DivX 4/5/6	1080p HD	1920 x 1080 / 30 fps	10 Mbps	
Xvid	Highdef	1920 x 1080 / 30 fps	10 Mbps	
MPEG-4	Advanced Simple / L4	1920 x 1080 / 30 fps	10 Mbps	
H.264	Baseline (BP) / L4	1920 x 1080 / 30 fps	10 Mbps	
H.264	Main (MP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CAVLC
H.264	Main (MP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CABAC, CAVLC
H.264	High (HiP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CAVLC
H.264	High (HiP) / L3.1	1280 x 720 / 30 fps	4 Mbps	CABAC, CAVLC

Please note that the maximal throughput data are peak values not average values. You may need to re-encode your videos with parameters above to get smooth video output.

11.3 Raster Font Support

If you enable the [Raster Fonts Support](#) some applications cause an exception. These include the Internet Explorer, the Music Player and the Video Player. If possible do not enable this setting.

Remark: For Video Player disable also the setting [Window Compositor](#).



11.4 Graphics Interface

You cannot use the U-Boot Setup settings to configure the DVI® or the LCD panel output. Only some entries in 'platform.reg' define for example the boot display or the resolution.

11.4.1 DVI® Monitor

The best way to apply the DVI® monitor consists in the usage of EDID data. Example:

```
IF BSP_NV_DISPLAY
... some settings ...
    "MainPanelBpp"=dword:20
    "DviHotplugBehavior"=dword:1
    "DesktopWidth"=dword:0
    "DesktopHeight"=dword:0
    "EnableEdidMode"=dword:1
    "DesktopScaleMode"=dword:2
    "DefaultDdrawCloneMode"=dword:0
    "AlwaysVSyncExternal"=dword:1
    "FilterScaledDesktops"=dword:1
    "EnableDestAlpha"=dword:1
    "UseStaticResolutionList"=dword:0
ENDIF
```

11.4.2 LCD Panel

For a LCD panel you have to define the entries DesktopWidth and DesktopHeight. Only some fixed resolutions (single channel) are supported. Example:

VGA, WVGA, SVGA and XGA

```
IF BSP_NV_DISPLAY
... some settings ...
    "MainPanelBpp"=dword:20
    "DviHotplugBehavior"=dword:1
; This registry will decide the desktop resolution
; * 800x600 (0x320 x 0x258)
    "DesktopWidth"=dword:320
    "DesktopHeight"=dword:258
    "EnableEdidMode"=dword:0
    "DesktopScaleMode"=dword:2
    "DefaultDdrawCloneMode"=dword:0
    "AlwaysVSyncExternal"=dword:1
    "FilterScaledDesktops"=dword:1
    "EnableDestAlpha"=dword:1
    "UseStaticResolutionList"=dword:0
ENDIF
```

11.5 I²C™ Support

Do not use the BSP I²C™ example. The connector J2000 involves the camera interface and likely this connector is not assembled in the future.

11.6 Watchdog Example

```
#include <windows.h>
#include <pkfuncs.h>

#define WDT_CNT_MAX          10
#define WDT_CNT_TIME        1000           // 1 second
#define WDT_CNT_ABORT       20

int _tmain (int argc, TCHAR *argv[])
{
    HANDLE hWdt;
    LPCWSTR pszWdtName = L"wdtdemo";
    DWORD dwWdtPeriod = 5000;             // 5 seconds
    DWORD dwWdtWait = 1000;              // 1 second
    DWORD dwCount;

    wprintf (TEXT ("Start watchdog demo\r\n"));
    hWdt = CreateWatchDogTimer (pszWdtName, dwWdtPeriod, dwWdtWait, WDOG_RESET_DEVICE, 0, 0);
    if (! hWdt)
    {
        wprintf (TEXT ("Error: invalid handle\r\n"));
        return 1;
    }

    if (GetLastError() == ERROR_ALREADY_EXISTS)
    {
        wprintf (TEXT ("Error: watchdog name already exists\r\n"));
        return FALSE;
    }

    if (! StartWatchDogTimer (hWdt, 0))
    {
        wprintf (TEXT ("Error: StartWatchDogTimer failed\r\n"));
        CloseHandle (hWdt);
        return FALSE;
    }

    dwCount = 0;
    while ((dwCount++) < WDT_CNT_MAX)
    {
        wprintf (TEXT ("Refreshing watchdog timer. Count = %d of %d\r\n"), dwCount, WDT_CNT_MAX);
        if (! RefreshWatchDogTimer (hWdt, 0))
        {
            wprintf (TEXT ("Error: RefreshWatchDogTimer failed\r\n"));
            CloseHandle (hWdt);
            return FALSE;
        }
    }
}
```

```
    Sleep (WDT_CNT_TIME);
}

wprintf (TEXT ("Watchdog timer refresh stopped !\r\n"));
dwCount = 0;
while ((dwCount++) < WDT_CNT_ABORT)
{
    wprintf (TEXT ("Timeout count = %d\r\n"), dwCount);
    Sleep (WDT_CNT_TIME);
}

wprintf (TEXT ("Error: watchdog timeout failed\r\n"));
CloseHandle (hWdt);
return FALSE;
}
```

11.7 GPIO Examples

Defines PORT_C as output and sets each pin to low level (alternative to high level):

```
#include <windows.h>
#include <winioctl.h>
#include <tchar.h>

// define IO controls
#define IOCTL_GPIO_CLEAR_OUTPUT \
    CTL_CODE (FILE_DEVICE_USERDRIVER, 3300, METHOD_BUFFERED, FILE_ANY_ACCESS)

#define IOCTL_GPIO_SET_OUTPUT \
    CTL_CODE (FILE_DEVICE_USERDRIVER, 3301, METHOD_BUFFERED, FILE_ANY_ACCESS)

#define IOCTL_GPIO_CONFIG_OUTPUT \
    CTL_CODE (FILE_DEVICE_USERDRIVER, 3303, METHOD_BUFFERED, FILE_ANY_ACCESS)

// define ports
#define GPIO_PORT_A        0
#define GPIO_PORT_B        1
#define GPIO_PORT_C        2
#define GPIO_PORT_D        3
#define GPIO_PORT_E        4
#define GPIO_PORT_F        5

// index conversion macros
#define GET_INDEX(port, pin) ((port << 3) | (0x07 & pin))

int _tmain (int argc, TCHAR *argv[])
{
    HANDLE hDev;
    UINT    i;
    UCHAR  ucBuffIn, ucBuffOut;

    // Create handle
    hDev = CreateFile (L"PIO1:", GENERIC_READ | GENERIC_WRITE, 0, NULL, OPEN_EXISTING, 0, NULL);
    if (hDev != INVALID_HANDLE_VALUE)
    {
        for (i = 0; i < 8; i++)
        {
            ucBuffIn = GET_INDEX (GPIO_PORT_C, i);
            if (! DeviceIoControl (hDev, IOCTL_GPIO_CONFIG_OUTPUT,
                &ucBuffIn, sizeof (UCHAR), &ucBuffOut, sizeof (UCHAR),
                NULL, NULL))
                wprintf (TEXT ("Error: DeviceIoControl_CONFIG_OUTPUT failed\r\n"));

            // use IOCTL_GPIO_CLEAR_OUTPUT or IOCTL_GPIO_SET_OUTPUT
            if (! DeviceIoControl (hDev, IOCTL_GPIO_CLEAR_OUTPUT,
                &ucBuffIn, sizeof (UCHAR), &ucBuffOut, sizeof (UCHAR),
                NULL, NULL))
                wprintf (TEXT ("Error: DeviceIoControl_CLEAR_SET_OUTPUT failed\r\n"));
        }
    }
}
```

Defines PORT_C as input and reads each pin:

```
#include <windows.h>
#include <winioctl.h>
#include <tchar.h>

// define IO controls
#define IOCTL_GPIO_GET_INPUT \
    CTL_CODE (FILE_DEVICE_USERDRIVER, 3302, METHOD_BUFFERED, FILE_ANY_ACCESS)

#define IOCTL_GPIO_CONFIG_INPUT \
    CTL_CODE (FILE_DEVICE_USERDRIVER, 3304, METHOD_BUFFERED, FILE_ANY_ACCESS)

// define ports
#define GPIO_PORT_A        0
#define GPIO_PORT_B        1
#define GPIO_PORT_C        2
#define GPIO_PORT_D        3
#define GPIO_PORT_E        4
#define GPIO_PORT_F        5

// index conversion macros
#define GET_INDEX(port, pin) ((port << 3) | (0x07 & pin))

int _tmain (int argc, TCHAR *argv[])
{
    HANDLE hDev;
    UINT    i;
    UCHAR  ucBuffIn, ucBuffOut, val = 0;

    // Create handle
    hDev = CreateFile (L"PIO1:", GENERIC_READ | GENERIC_WRITE, 0, NULL, OPEN_EXISTING, 0, NULL);
    if (hDev != INVALID_HANDLE_VALUE)
    {
        for (i = 7; i >= 0; i--)
        {
            ucBuffIn = GET_INDEX (GPIO_PORT_C, i);
            if (! DeviceIoControl (hDev, IOCTL_GPIO_CONFIG_INPUT,
                &ucBuffIn, sizeof (UCHAR), &ucBuffOut, sizeof (UCHAR),
                NULL, NULL))
                wprintf (TEXT ("Error: DeviceIoControl_CONFIG_INPUT failed\r\n"));

            if (! DeviceIoControl (hDev, IOCTL_GPIO_GET_INPUT,
                &ucBuffIn, sizeof (UCHAR), &ucBuffOut, sizeof (UCHAR),
                NULL, NULL))
                wprintf (TEXT ("Error: DeviceIoControl_GET_INPUT failed\r\n"));

            val |= (ucBuffOut & 0x01);
            if (i > 0)
                val <<= 1;
        }

        wprintf (TEXT ("Input value = 0x%02X\r\n"), val);
        Sleep (2000);
    }
}
```

Appendix A: Reference Documents

KONTRON Technology A/S can't guarantee the availability of internet addresses.

Document	Internet Address
NVIDIA® Development	http://developer.nvidia.com/tools/Development
Tegra® 2 Technical Reference Manual	http://developer.nvidia.com/tegra-2-technical-reference-manual
Linux® for Tegra®	http://developer.nvidia.com/linux-tegra
Digital Visual Interface (DVI®)	http://www.ddwg.org
Open LVDS Display Interface Standard Spec. (Open LDI™)	http://www.national.com/analog/displays/open_ldi
IEEE 802.3® Specification (Ethernet)	http://standards.ieee.org/getieee802
Universal Serial Bus Specification (USB)	http://www.usb.org/developers/docs
SD Specification (SD Card)	http://www.sdcard.org/developers/tech/sdio/sdio_spec

Appendix B: Document Revision History

Revision	Date	Author	Changes
S-0045-D	02/08/13	M. Hüttmann	Some minor changes in Android BSP chapter
S-0045-C	02/01/13	M. Hüttmann	Added chapters for Android and WEC7 BSPs, some changes in 'Linux BSP'
S0045-B	12/03/12	M. Hüttmann	Added some subchapter under 'Linux BSP' (Login, PCI Express, KEAPI)
S0045-A	11/19/12	M. Hüttmann	New Kontron design. Added chapter 'Linux BSP'
S0045-0	07/05/12	M. Hüttmann	Created preliminary manual

Corporate Offices

Europe, Middle East & Africa

Oskar-von-Miller-Str. 1
85386 Eching/Munich
Germany
Tel.: +49 (0)8165/ 77 777
Fax: +49 (0)8165/ 77 219
info@kontron.com

North America

14118 Stowe Drive
Poway, CA 92064-7147
USA
Tel.: +1 888 294 4558
Fax: +1 858 677 0898
info@us.kontron.com

Asia Pacific

17 Building, Block #1, ABP
188 Southern West 4th Ring Road
Beijing 100070, P.R.China
Tel.: + 86 10 63751188
Fax: + 86 10 83682438
info@kontron.cn



POWERED BY
NVIDIA® TEGRA®