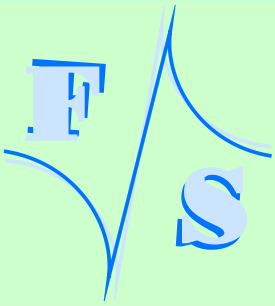
Device Driver documentation

Windows CE 6.0/Compact 7

for FSVYBRID/FSS5PV210

Version: 1.20
Date: 2013-08-23

PicoMOD
PicoCOM
NetDCU
QBliss
armStone
nanoRISC



F&S Elektronik Systeme GmbH
Untere Waldplätze 23
D-70569 Stuttgart
Fon: +49(0)711-123722-0
Fax: +49(0)711-123722-99

History

Date	V	Platform	A,M,R	Chapter	Description	Au
2009-03-12	1.0	3,4	A	*	First version	DK
2009-03-16	1.1	3	A,M	3	IO-Pin table supplemented, IRQCFG table added	MK
2009-04-06	1.2	3	M	5	Configuration example corrected	HF
2009-06-05	1.3	3	M	3	IO-Pin table IRQs. IRQ sample updated.	DK
2009-06-15	1.4	3	M	3	New IOCTLs for interrupt usage.	DK
2010-10-29	1.5	6	M	*	PicoMOD6 added. NI2C added.	HF
2010-03-08	1.5	6	A	13	SD/MMC added	HF
2010-03-09	1.5	3,4,6	A	8	USB function driver added	HF
2010-03-18	1.5	6	M	6	Touch panel registry settings	HF
2010-05-17	1.7	6	M	9	LCD Driver	ZU
2010-09-09	1.8	6	M	6	Touch panel registry settings	ZU
2010-09-22	1.8	6	M	8	USB function driver registry settings	ZU
2010-11-24	1.9	6	M	7	Added description for registry value PhysicalPageSize and DoNotPromptUser (USB host driver).	HF
2011-01-27	1.9	3,4,6,7	A	14	Ethernet added.	HF
2011-01-27	1.9	7	M	7	PicoMOD7 added.	HF
2011-01-27	1.9	7	M	13	Added description of registry value PwrPin and WPPin.	HF
2011-02-21	1.10	6	M	9	LCD Driver	ZU
2011-05-16	1.11	6,7	M	2	Added explanation of how to access mixer from user application.	HF
2011-05-18	1.12		M	*	QBlissA8	HF
2011-08-01	1.13	3,4,6,7,QA8	M	14	Added description for LAN LED.	HF
2011-10-18	1.14	3,4,6,7,QA8	M	13	Correct registry settings	MA
2011-10-26	1.15	7,QA8	M	3	IO-Pin table enhanced, added information if IO could be an interrupt	HF
2012-04-11	1.16	ASA8	A	*	armStoneA8 added	HF
2012-04-18	1.17	6, QA8	A	6	Different touch driver versions.	MR
2012-11-19	1.18	NRA8	A	*	nanoRISC-A8 added	HF
2012-11-22	1.19	ND14	A	*	NetDCU14 added	HF
2012-11-23	1.19	*	A	*	Analogue In driver added	HF
2012-11-23	1.19	*	A	*	PWM driver added	HF
2013-01-24	1.20	*	M	11.2	Added note for LCD output width.	HF
2013-07-24	1.21	ASA5, NDA5, PCA5	A	2	New functions in analogin driver	TM

V	Version
A,M,R	Added, Modified, Removed
Au	Author

About this document

This is the device driver documentation for the F&S platforms FSS5PV210 and FSVYBRID based on Windows Embedded CE 6.0 or Windows Embedded Compact 7. If you need information about older products such as PicoMOD1 (running on Windows CE 5) or NetDCU3 - NetDCU11 please read the corresponding documentation which can be found at: http://www.fs-net.de

For each device driver it is documented on which platform it is implemented. The registry settings, the configuration and programming examples are described in this document. The latest version of this document can be found at: http://www.fs-net.de

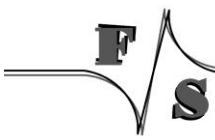
Boards which are using platform FSVYBRID are:

- armStoneA5
- PicoCOMA5
- NetDCUA5

Boards which are using platform FSS5PV210 are:

- armStoneA8
- PicoMOD7A
- NetDCU14
- nanoRISC-A8

Contents



1 Windows CE Stream Interface Driver

All device drivers are implemented as Windows CE Stream Interface Driver. Thus you can access these drivers via the File System and the respective File API (CreateFile, WriteFile, ReadFile, SetFilePointer, DeviceIoControl).

A stream interface driver receives commands from the Device Manager and from applications by means of file system calls. The driver encapsulates all of the information that is necessary to translate those commands into appropriate actions on the devices that it controls. All stream interface drivers, whether they manage built-in devices or installable devices, or whether they are loaded at boot time or loaded dynamically, have similar interactions with other system components. The following illustrations show the interactions between system components for a generic stream interface driver that manages a built-in device.

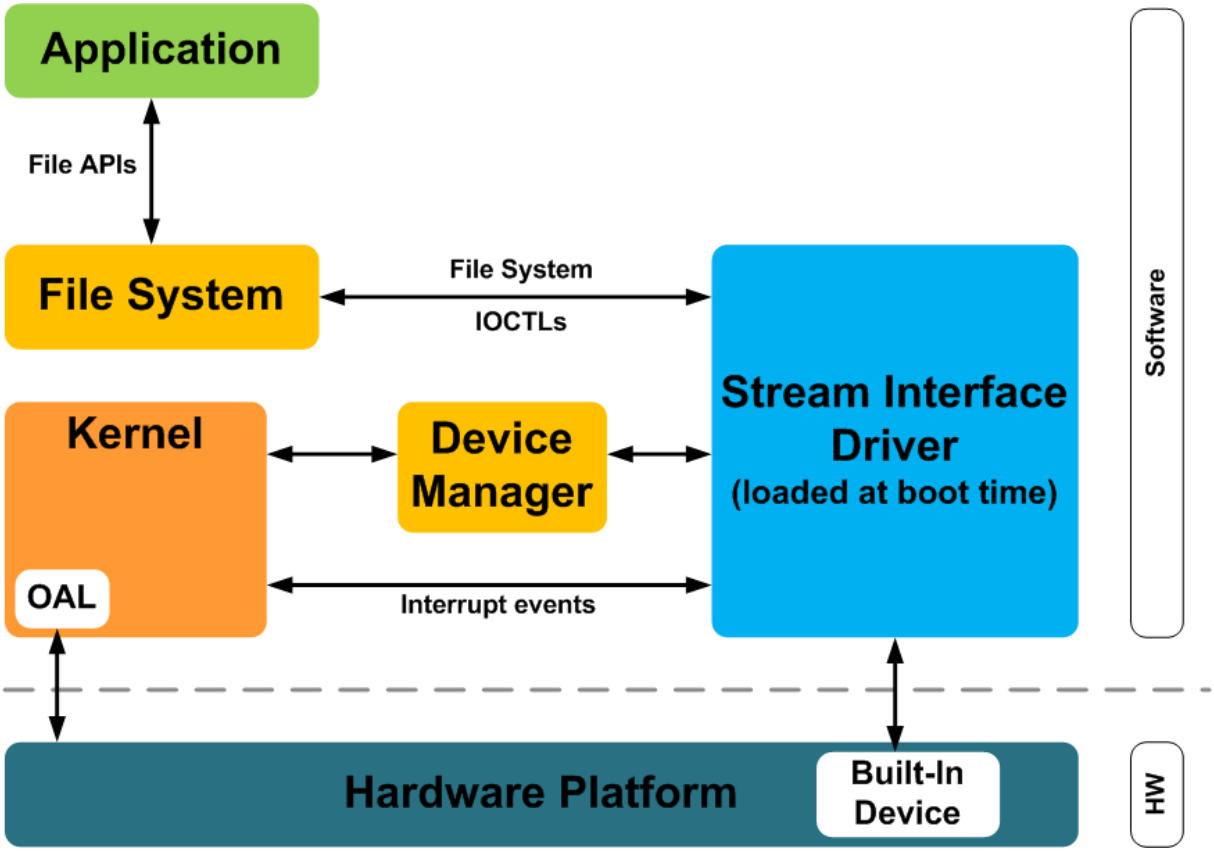


Figure 1: Windows CE: Stream Interface Driver Architecture

2 Analogue Input

2.1 S5PV210

Implemented on: ASA8, ND14, NRA8

Some boards have beside resistive touch interface additional analogue inputs.
These analogue inputs can be read with this driver. You can install one copy of the driver for each input or use the function `SetFilePointer()` to select the channel.
The selection of the channel can be done with the registry key *Channel*.

Installation of the driver is done by setting some registry values under the following registry key:

```
[HKLM\Drivers\BuiltIn\armStoneA8\ANALOGIN]
[HKLM\Drivers\BuiltIn\nanoRISC\ANALOGIN]
[HKLM\Drivers\BuiltIn\NetDCU14\ANALOGIN]
```

Required settings:

Key	Value	Comment
"Prefix"	“AIN“	This required value specifies the driver’s device file name prefix. It is a three-character identifier, such as COM.
“Dll“	“FS_ANALOGIN. DLL“	name of the DLL with the driver
“Order“	Dword:	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
“Index“	Dword:1	This value specifies the device index, a value from 0 through 9.
“Flags”	Dword:0	4: Disabled from loading
“Ioctl“	Dword:4	Call post-initialization function.
“Channel“	Dword:n	Number of the analogue channel. See Table Channel.
“FriendlyName“	“Analogue input driver“	
“Debug”	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 1: Analogue Input: Registry

Table Channel armStoneA8:

Channel	Description
0x02	TOUCH_YM
0x03	TOUCH_YP
0x04	TOUCH_XM
0x05	TOUCH_XP
0x06	Reads value from analogue input 0 (Feature connector pin 29)
0x07	Reads value from analogue input 1 (Feature connector pin 31)
0x08	Reads value from analogue input 2 (Feature connector pin 33)
0x09	Reads value from analogue input 3 (Feature connector pin 35)

Table 2: Analogue Input: armStoneA8 Channel

Table Channel NetDCU14:

Channel	Description
0x02	TOUCH_YM
0x03	TOUCH_YP
0x04	TOUCH_XM
0x05	TOUCH_XP
0x06	Reads value from analogue input 0 (Connector J7, AD0)
0x07	Reads value from analogue input 1 (Connector J7, AD1)
0x08	Reads value from analogue input 2 (Connector J7, AD2)
0x09	Reads value from analogue input 3 (Connector J7, AD3)

Table 3: Analogue Input: NetDCU14 Channel

Programming Example:

A. Open one analogue channel:

```
HANDLE hAIN;
hAIN = CreateFile( _T("AIN1:"),GENERIC_READ, 0, NULL, OPEN_EXISTING
                              ,FILE_ATTRIBUTE_NORMAL, NULL );
if( hAIN == INVALID_HANDLE_VALUE )
{
  ERRORMSG(1,(L"Can not open AIN1. LastError = 0x%x\r\n",GetLastError()));
  return(FALSE);
}
```

Listing 1: Analogue Input: Open channel

B. Read data from previously opened channel:

```
unsigned short data;
DWORD dwSamples = 1;
ReadFile( hAIN, data, dwSamples, &dwSamples, NULL );
if( dwSamples != 1 )
{
  ERRORMSG(1,(L"Can not read from AIN1. LE = 0x%x\r\n",GetLastError()));
}
```

Listing 2: Analogue Input: reading samples

C. Select another channel without changing registry:

```
int nChannel = 0x0;
SetFilePointer( hAIN, nChannel, 0, FILE_BEGIN );
```

Listing 3: Analogue Input: changing channel from application

D. Closing the analogue channel:

```
CloseHandle(hAIN);
```

Listing 4: Analogue Input: closing a channel

2.2 Vybrid

Implemented on: ASA5, NDA5, PCA5

Some boards have beside resistive touch interface additional analogue inputs.
These analogue inputs can be read with this driver. You can install one copy of the driver for each input or use the function `SetFilePointer()` to select the channel.
The selection of the channel can be done with the registry key *Channel*.

Installation of the driver is done by setting some registry values under the following registry key:

```
[HKLM\Drivers\BuiltIn\armStoneA5\ANALOGIN]
[HKLM\Drivers\BuiltIn\NetDCUA5\ANALOGIN]
[HKLM\Drivers\BuiltIn\PicoCOMA5\ANALOGIN]
```

Required settings:

Key	Value	Comment
"Prefix"	“AIN“	This required value specifies the driver’s device file name prefix. It is a three-character identifier, such as COM.
“Dll“	“FS_ANALOGIN.DLL“	name of the DLL with the driver
“Order“	Dword:	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
“Index“	Dword:1	This value specifies the device index, a value from 0 through 9.
“Flags”	Dword:0	4: Disabled from loading
“Ioctl“	Dword:4	Call post-initialization function.
“Channel“	Dword:n	Number of the analogue channel. See Table Channel.
“FriendlyName“	“Analogue input driver“	
"Average"	Dword:n	Number of conversations per sample (4, 8, 16 or 32).
"Offset"	Dword:n	Decimal value to be added or subtracted to raw sample data
"OffsetSign"	Dword:n	0: Offset value is added to raw data 1: Offset is subtracted
"Reference"	Dword:n	0: 3.3V reference voltage 1: 1.2V reference voltage
“Debug”	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 4: Analogue Input: Registry

Table Channel armStoneA5:

Channel	Description
0x00	Reads value from analogue input 0 (Feature connector pin 29)
0x01	Reads value from analogue input 1 (Feature connector pin 31)
0x02	Reads value from analogue input 2 (Feature connector pin 33)
0x03	Reads value from analogue input 3 (Feature connector pin 35)
0x1A	Reads value from temperature sensor

Table 5: Analogue Input: armStoneA5 Channel

Table Channel NetDCUA5:

Channel	Description
0x00	Reads value from analogue input 0 (Connector J7, AD0)
0x01	Reads value from analogue input 1 (Connector J7, AD1)
0x02	Reads value from analogue input 2 (Connector J7, AD2)
0x03	Reads value from analogue input 3 (Connector J7, AD3)
0x1A	Reads value from temperature sensor

Table 6: Analogue Input: NetDCUA5 Channel

Table Channel PicoCOMA5:

Channel	Description
0x1A	Reads value from temperature sensor

Table 7: Analogue Input: PicoCOMA5 Channel

Programming Example:

A. Open one analogue channel:

```
HANDLE hAIN;
hAIN = CreateFile( _T("AIN1:"),GENERIC_READ, 0, NULL, OPEN_EXISTING
                              ,FILE_ATTRIBUTE_NORMAL, NULL );
if( hAIN == INVALID_HANDLE_VALUE )
{
  ERRORMSG(1,(L"Can not open AIN1. LastError = 0x%x\r\n",GetLastError()));
  return(FALSE);
}
```

Listing 5: Analogue Input: Open channel

B. Read data from previously opened channel:

```
unsigned short data;
DWORD dwSamples = 1;
ReadFile( hAIN, data, dwSamples, &dwSamples, NULL );
if( dwSamples != 1 )
{
  ERRORMSG(1,(L"Can not read from AIN1. LE = 0x%x\r\n",GetLastError()));
}
```

Listing 6: Analogue Input: reading samples

C. Select another channel without changing registry:

```
int nChannel = 0x0;
SetFilePointer( hAIN, nChannel, 0, FILE_BEGIN );
```

Listing 7: Analogue Input: changing channel from application

D. Closing the analogue channel:

```
CloseHandle(hAIN);
```

Listing 8: Analogue Input: closing a channel

E. Get adc settings

```
#include "fs_analogin_sdk.h"

DWORD dwBytesReturned;
AIN_INFO cAIN_INFO;

DeviceIoControl(hADC, IOCTL_AIN_GETINFO, NULL, 0, &cAIN_INFO, sizeof(AIN_INFO),
                &dwBytesReturned, NULL);
```

Listing 9: Get adc settings

F. Set adc settings

```
#include "fs analogin sdk.h"

DWORD dwBytesReturned;
AIN INFO cAIN INFO;

DeviceIoControl(hADC, IOCTL_AIN_SETINFO, NULL, 0, &cAIN_INFO, sizeof(AIN_INFO),
                &dwBytesReturned, NULL);
```

Listing 10: Set adc settings

G. Read temperature

```
#include "fs analogin sdk.h"

FLOAT fTemperatur;

DeviceIoControl(hADC, IOCTL_AIN_GETTEMPERATUR, &fTemperatur, 1, NULL, 0, &dwBytesReturned,
NULL);
```

Listing 11: Read temperature

3 Audio Driver

Implemented on: PM3,PM4,PM6,PM7,PM7A,QA8,ASA8,ND14

Audio driver for is implemented as wavedev2 driver and can be configured under the following registry key:

```
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\Audio]
```

The mixer line settings are compatible across all Windows CE 6.0 based platforms. Possible settings:

Key	Value	Comment
“Prefix”	“WAV”	This required value specifies the driver’s device file name prefix. It is a three-character identifier, such as COM.
“DLL”		Name of the driver file.
“Index”	Dword:1	This value specifies the device index, a value from 0 through 9.
“InChannel”	Dword:n	This value selects the input channel. 2 = Line-In 3 = Microphone
“InMute”	Dword:0|1	Set this 1 to mute input channel. Default: 0
“MicBoost”	Dword:0|1	Set this 1 to boost microphone input by 20dB. Default: 0
“BypassMute”	Dword:0|1	Set this to 0 to route Line-In directly to Line-Out. Default: 1
“SidetoneMute”	Dword:0|1	Set this to 0 to route Mic-In directly to Line-Out. Default: 1
“SidetoneVol”	Dword:n	Volume for Sidetone effect. Default: 0
“LineInVolLeft”	Dword:n	Volume for Line-In left. Default: 0x27
“LineInVolRight”	Dword:n	Volume for Line-In right. Default: 0x27
“HeadphoneVolLeft”	Dword:n	Volume for Headphone left. Default: 0x39
“HeadphoneVolRight”	Dword:n	Volume for Headphone right. Default: 0x39
“OutMute”	Dword:0|1	Set this 1 to mute output channel. Default: 0

Key	Value	Comment
“Debug”	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 8: Audio: Registry settings

Audio driver supports mixer. You can use the command line tool SoundInfo.exe to get an overview of mixer interface and current state of controls. With the control panel applet “Audio Mixer” you can also change the current settings of Audio Mixer. Any mixer changes automatically adapt the registry settings. To store the current configuration permanently you just have to save the registry.

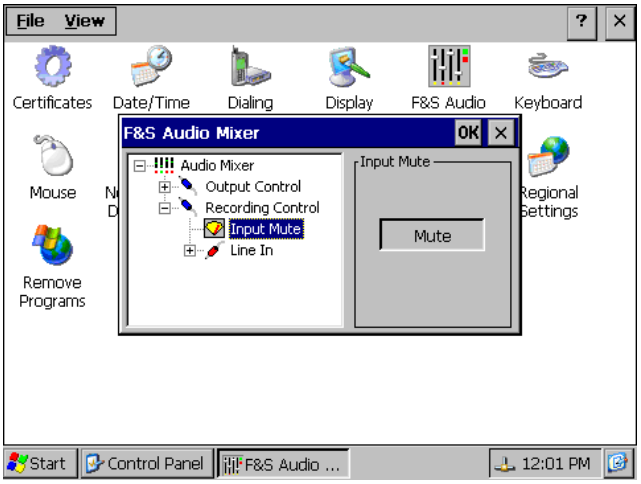


Figure 2: F&S Audio Mixer control

3.1 Mixer Programming Example

Sometimes it is necessary to change a mixer control from your application. In this case you must know the *LineID* and the *type* of control you want to change.

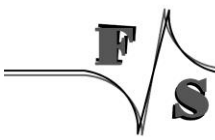
The *LineID* is a combination of the following values;

LINE_OUT	0x80
PCM_IN	0x81
HPHONE	0x82
LINE_IN	0x83
MIC	0x84
OUT2	0x85
OUT3	0x86
MONOOUT	0x87
ALC	0x88
NOLINE	0xFF

Use the following macro to generate the *LineID*:

```
/* mixer line ID are 16-bit values formed by concatenating the source and destination line
indices */
#define MXLINEID(dst,src) ((USHORT) ((USHORT)(dst) | (((USHORT) (src)) << 8)))
```

Listing 12: Audio: Macro for LineID

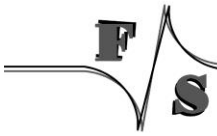


The following table lists the available combination of *LineID* and control *type*. Replace xx at the beginning of the control type with `MIXERCONTROL_CONTROLTYPE`.

MXLINEID(dst,src)	Control Type	Registry Name	Control Name
(LINE_OUT,NOLINE)		MasterOutVol	Master Volume
(LINE_OUT,OUT2)		LineOut2Vol	LineOut Volume
(LINE_OUT,HPHONE)		HeadphoneVol	Headphone Volume
(LINE_OUT,HPHONE)	xx_MUTE	HeadphoneMute	Headphone Mute
(LINE_OUT,MIC)		SidetoneVol	Sidetone Volume
(LINE_OUT,ALC)		ALCSidetoneVol	Sidetone Volume
(PCM_IN,NOLINE)		MasterInVol	Master Volume
(LINE_OUT,NOLINE)	*xx*_BASS	BassBoost	Master Bass
(LINE_OUT,NOLINE)	*xx*_TREBLE	TrebleBoost	Treble Boost
(PCM_IN,LINE_IN)		LineInVol	LineIn Volume
(LINE_OUT,NOLINE)	*xx*_MUTE	MasterOutMute	Master Mute
(PCM_IN,NOLINE)	*xx*_MUTE	MasterInMute	Master Mute
(LINE_OUT,MIC)	*xx*_MUTE	SidetoneMute	Sidetone Mute
(LINE_OUT,ALC)	*xx*_MUTE	ALCSidetoneMute	Sidetone Mute
(LINE_OUT,NOLINE)	*xx*_MONO	OutputRenderMonoOnly	Mono
(PCM_IN,NOLINE)	*xx*_MUTE	MasterInMute	Rec Mute
(LINE_OUT,OUT2)	*xx*_MUTE	LineOut2Mute	LineOut Mute
(PCM_IN,MIC)	*xx*_ONOFF	MicBoost	Mic Boost
(PCM_IN,NOLINE)	*xx*_ONOFF	RecBoost	Boost
(LINE_OUT,LINE_IN)	*xx*_MUTE	BypassMute	Line In BYPASS
(PCM_IN,MIC)	*xx*_MUX	MicMode	Mic Mode
(PCM_IN,NOLINE)	*xx*_MUX	InChannel	Input Select
(LINE_OUT,NOLINE)	*xx*_EQPRESET	SoundMode	Eq Preset

Remark:

Not all controls are available on every platform. Use soundinfo.exe to get a list of the available controls.



With the above information it's now easy to manipulate control state from your application.

```
/* mixer line ID are 16-bit values formed by concatenating the source and destination line
indices */
#define MXLINEID(dst,src) ((USHORT) ((USHORT)(dst) | (((USHORT) (src)) << 8)))

HMIXER m HMixer;
MIXERLINECONTROLS cMixCtrls;
MIXERCONTROL cMyCtrl;

if( mixerOpen( &m HMixer, 0, ( DWORD )hwnd, 0, CALLBACK WINDOW ) != MMSYSERR NOERROR )
{
    PrintMessage( "CMixerBase::Init", "Could not open mixer device" );
    return -1;
}

memset( &cMixCtrls, 0, sizeof(cMixCtrls) );
cMixCtrls.cbStruct        = sizeof(MIXERLINECONTROLS);
cMixCtrls.dwLineID        = line.dwLineID;
cMixCtrls.dwControlType   = MIXERCONTROL_CONTROLTYPE_MUX;
cMixCtrls.cControls       = 1;
cMixCtrls.cbmxctrl        = sizeof(MIXERCONTROL);
mixerLineControl.pamxctrl = &cMyCtrl;

if( mixerGetLineControls( ( HMIXEROBJ )m HMixer, &cMixCtrls,
        MIXER_GETLINECONTROLSF_ONEBYTYPE ) != MMSYSERR_NOERROR )
{
    PrintMessage( "CMixerBase::Init", "Could not find specified mixer control." );
    CloseMixer();
    return 0;
}

MIXERCONTROLDETAILS mcd;
MIXERCONTROLDETAILS_UNSIGNED* pData = NULL;

pData = (MIXERCONTROLDETAILS UNSIGNED*)malloc(
                        sizeof(MIXERCONTROLDETAILS UNSIGNED)* cMyCtrl.cMultipleItems );

mcd.cbStruct       = sizeof( MIXERCONTROLDETAILS );
mcd.dwControlID    = cMyCtrl.dwControlID;
mcd.cMultipleItems = cMyCtrl.cMultipleItems;
mcd.cChannels = 1;

mcd.cbDetails = sizeof(MIXERCONTROLDETAILS UNSIGNED);
mcd.paDetails = pData;

result = mixerGetControlDetails( ( HMIXEROBJ )hMixer, &mcd, MIXER GETCONTROLDETAILSF VALUE );
```

Listing 13: Audio: Access mixer from user application

4 Digital I/O

Implemented on: PM3,PM4,PM6,PM7,PM7A,QA8,ASA8,NRA8

armStone, PicoMOD and QBliss has programmable I/O lines. You have to use this driver to configure and access the I/O lines.

Installation of the driver is done by setting some registry values under the following registry key:

```
[HKLM\Drivers\BuiltIn\DIGITALIO]
```

Required settings:

Key	Value	Comment
Prefix	"DIO"	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Dll		Name of the DLL with the driver
Order	Dword:97	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
Ioctl	Dword:4	Call post-initialization function.
Port	Dword:n	0..15
UseAsIO - or - UseAsIOA UseAsIOB UseAsIOC UseAsIOD *UseAsIOx*	Dword:n	1 = The corresponding pin is used as general purpose I/O. One bit for each I/O pin.
DataDir - or - DataDirA DataDirB DataDirC DataDirD *DataDirx*	Dword:n	Data Direction. 0 = The corresponding pin is an input. 1 = The corresponding pin is an output. One bit for each I/O pin.
DataInit - or - DataInitA DataInitB DataInitC DataInitD *DataInitx*	Dword:n	Default value of the output pin after driver initialization.

Key	Value	Comment
IRQCfg0 - or - IRQCfg0A IRQCfg0B IRQCfg0C IRQCfg0D *IRQCfg0x*	Dword:n	Interrupt configuration register 0.
IRQCfg1 - or - IRQCfg1A IRQCfg1B IRQCfg1C IRQCfg1D *IRQCfg1x*	Dword:n	Interrupt configuration register 1.
IRQCfg2 - or - IRQCfg2A IRQCfg2B IRQCfg2C IRQCfg2D *IRQCfg2x*	Dword:n	Interrupt configuration register 2.
PullUp - or - PullUpA PullUpB PullUpC PullUpD PullUpx	Dword:n	Set to 1 to enable internal pull-up.
PullDownp - or - PullDownA PullDownB PullUDownC PullUDownD PullUDownx	Dword:n	Set to 1 to enable internal pull-down
FriendlyName	Digital I/O driver	
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 9: Digital I/O: Registry settings

4.1 Port description PicoMOD

The following table is useful if you want to use **UseAsIO*x*/DataDir*x*/DataInit*x***. These values are 32 bit DWORD registry values. Each value (x=A..x=D) configures 4 ports. In contrast to this, you can also use registry values **UseAsIO/DataDir/DataInit** with data type HEX.

Port 0

Bit	7	6	5	4	3	2	1	0
Pin	23	24	21	22	19	20	17	18
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
UseAsIOA Bit	7	6	5	4	3	2	1	0
DataDirA Bit	7	6	5	4	3	2	1	0
DataInitA Bit	7	6	5	4	3	2	1	0
IRQCfg0A IRQCfg1A IRQCfg2A	---	---	---	---	---	---	---	---

Port1

Bit	7	6	5	4	3	2	1	0
Pin	44	41	42	34	31	32	29	30
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
UseAsIOA Bit	15	14	13	12	11	10	9	8
DataDirA Bit	15	14	13	12	11	10	9	8
DataInitA Bit	15	14	13	12	11	10	9	8
IRQCfg0A IRQCfg1A IRQCfg2A	15	14	13	---	---	10	9	8

Port 2

Bit	7	6	5	4	3	2	1	0
Pin	52	49	50	47	48	45	46	43
R/W	R	R	R	R	R	R	R	R
UseAsIOA Bit	23	22	21	20	19	18	17	16
DataDirA Bit	23	22	21	20	19	18	17	16
DataInitA Bit	23	22	21	20	19	18	17	16
IRQCfg0A IRQCfg1A IRQCfg2A	---	---	---	---	---	---	---	16

Port 3

Bit	7	6	5	4	3	2	1	0
Pin	60	57	58	55	56	53	54	51
R/W	R	R	R	R	R	R	R	R
UseAsIOA Bit	31	30	29	28	27	26	25	24
DataDirA Bit	31	30	29	28	27	26	25	24
DataInitA Bit	31	30	29	28	27	26	25	24
IRQCfg0A IRQCfg1A IRQCfg2A	---	---	---	---	---	26	25	24

Port 4

Bit	7	6	5	4	3	2	1	0
Pin	70	67	68	65	66	63	64	61
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
UseAsIOB Bit	7	6	5	4	3	2	1	0
DataDirB Bit	7	6	5	4	3	2	1	0
DataInitB Bit	7	6	5	4	3	2	1	0
IRQCfg0B IRQCfg1B IRQCfg2B	---	---	---	---	---	---	---	---

Port 5

Bit	7	6	5	4	3	2	1	0
Pin	78	75	76	73	74	71	72	69
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
UseAsIOB Bit	15	14	13	12	11	10	9	8
DataDirB Bit	15	14	13	12	11	10	9	8
DataInitB Bit	15	14	13	12	11	10	9	8
IRQCfg0B IRQCfg1B IRQCfg2B	---	---	---	---	---	---	---	---

Port 6

Bit	7	6	5	4	3	2	1	0
Pin	---	---	86	81	82	79	80	77
R/W	R	R	R	R	R	R	R	R
UseAsIOB Bit	23	22	21	20	19	18	17	16
DataDirB Bit	23	22	21	20	19	18	17	16
DataInitB Bit	23	22	21	20	19	19	17	16
IRQCfg0B IRQCfg1B IRQCfg2B	---	---	---	---	---	---	---	---

Port 7

Bit	7	6	5	4	3	2	1	0
Pin	---	---	---	---	---	---	---	---
R/W	R	R	R	R	R	R	R	R
UseAsIOB Bit	31	30	29	28	27	26	25	24
DataDirB Bit	31	30	29	28	27	26	25	24
DataInitB Bit	31	30	29	28	27	26	25	24
IRQCfg0B IRQCfg1B IRQCfg2B	---	---	---	---	---	---	---	---

Port 8

Bit	7	6	5	4	3	2	1	0
Pin	88	87	---	---	4	3	2	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
UseAsIOC Bit	7	6	5	4	3	2	1	0
DataDirC Bit	7	6	5	4	3	2	1	0
DataInitC Bit	7	6	5	4	3	2	1	0
IRQCfg0C IRQCfg1C IRQCfg2C	7	6	---	---	---	---	---	---

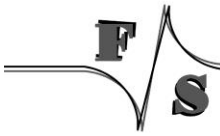
Port 9

Bit	7	6	5	4	3	2	1	0
Pin	---	---	---	---	126	98	93	90
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
UseAsIOC Bit	15	14	13	12	11	10	9	8
DataDirC Bit	15	14	13	12	11	10	9	8
DataInitC Bit	15	14	13	12	11	10	9	8
IRQCfg0C IRQCfg1C IRQCfg2C	---	---	---	---	11	10	9	8

Table 10: Digital I/O - PicoMOD Port 0 – 9

Digital-IO					PicoMOD6										
					PM6-Pin	Startintf	PIO-Pin	Function							
IO-Pin	Port		Registry settings					*COM*	*I2C*	*SPI*	*USB*	*SD/ MMC*	*LCD*	*CF*	*sonst.*
0	**0**	**Port 0**	**0**	**UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1**	**18**	J2-3	GPA0	RXD0							
1	**1**		**1**		**17**	J2-5	GPA1	TXD0							
2	**2**		**2**		**20**	J2-6	GPA2	CTS0							
3	**3**		**3**		**19**	J2-4	GPA3	RTS0							
4	**4**		**4**		**22**	J7-9	GPA4	RXD1							
5	**5**		**5**		**21**	J7-10	GPA5	TXD1							
6	**6**		**6**		**24**		GPB0	RXD2							
7	**7**		**7**		**23**		GPB1	TXD2							
8	**0**	**Port 1**	**8**		**30**		GPK7				USB-PWR1				
9	**1**		**9**		**29**	**J5-9**	GPN5								GPIO5
10	**2**		**10**		**32**		GPK8				USB-PWR2				
11	**3**		**11**		**31**	**J5-10**	GPB6		SDA						
12	**4**		**12**		**34**	**J5-11**	GPB5		SCL						
13	**5**		**13**		**42**	J4-13	GPN0								GPIO0
14	**6**		**14**		**41**	**J5-1**	GPN1								GPIO1
15	**7**		**15**		**44**	**J5-5**	GPN2								GPIO2
16	**0**	**Port 2**	**16**		**43**	**J5-7**	GPN3								GPIO3
17	**1**		**17**		**46**	**J5-8**	GPN4								GPIO4
18	**2**		**18**		**45**		GPG0					SDCLK			
19	**3**		**19**		**48**		GPG1					SDCMD			
20	**4**		**20**		**47**		GPG2					SDDAT0			
21	**5**		**21**		**50**		GPG3					SDDAT1			
22	**6**		**22**		**49**		GPG4					SDDAT2			
23	**7**		**23**		**52**		GPG5					SDDAT3			
24	**0**	**Port 3**	**24**		**51**		GPG6					SD-CD			
25	**1**		**25**		**54**		GPN6					SD-PWR			GPIO6
26	**2**		**26**		**53**		GPL12					SD-WP			
27	**3**		**27**		**56**		GPK3						LCD-ENA		
28	**4**		**28**		**55**		GPK2						LCD-DEN		
29	**5**		**29**		**58**		GPK0						VLCD-ON		
30	**6**		**30**		**57**		GPK1						VCFL-ON		
31	**7**		**31**		**60**		GPF15						VEEK		
32	**0**	**Port**	**32**		**61**		GPI10						VD0		

33	**1**	**4**	**33**		**64**		GPI11						VD1		
34	**2**		**34**		**63**		GPI7						VD7		
35	**3**		**35**		**66**		GPI3						VD3		
36	**4**		**36**		**65**		GPI4						VD4		
37	**5**		**37**		**68**		GPI5						VD5		
38	**6**		**38**		**67**		GPI6						VD6		
39	**7**		**39**		**70**		GPI7						VD7		
40	**0**	**Port 5**	**40**		**69**		GPI12						VD12		
41	**1**		**41**		**72**		GPI13						VD13		
42	**2**		**42**		**71**		GPI14						VD14		
43	**3**		**43**		**74**		GPI15						VD15		
44	**4**		**44**		**73**		GPJ7						VD23		
45	**5**		**45**		**76**		GPJ3						VD19		
46	**6**		**46**		**75**		GPJ4						VD20		
47	**7**		**47**		**78**		GPJ5						VD21		
48	**0**	**Port 6**	**48**		**77**		GPJ6						VD22		
49	**1**		**49**		**80**		GPJ7						VD23		
50	**2**		**50**		**79**		GPJ8						VLINE		
51	**3**		**51**		**82**		GPJ9						VFRAME		
52	**4**		**52**		**81**		GPJ10						VM		
53	**5**		**53**		**86**		GPJ11						VCLK		
54	**6**		**54**		**-**		-	-	-	-	-	-	-	-	-
55	**7**		**55**		**-**		-	-	-	-	-	-	-	-	-
56	**0**	**Port 7**	**56**		**-**		-	-	-	-	-	-	-	-	-
57	**1**		**57**		**-**		-	-	-	-	-	-	-	-	-
58	**2**		**58**		**-**		-	-	-	-	-	-	-	-	-
59	**3**		**59**		**-**		-	-	-	-	-	-	-	-	-
60	**4**		**60**		**-**		-	-	-	-	-	-	-	-	-
61	**5**		**61**		**-**		-	-	-	-	-	-	-	-	-
62	**6**		**62**		**-**		-	-	-	-	-	-	-	-	-
63	**7**		**63**		**-**		-	-	-	-	-	-	-	-	-
64	**0**	**Port 8**	**64**		**1**	**J5-6**	GPC3			SPICS0					
65	**1**		**65**		**2**	**J5-2**	GPC1			SPICLK0					
66	**2**		**66**		**3**	**J5-4**	GPC0			SPIMISO 0					
67	**3**		**67**		**4**	**J5-3**	GPC2			SPIMOSI 0					
68	**4**		**68**		**-**		-	-	-	-	-	-	-	-	-
69	**5**		**69**		**-**		-	-	-	-	-	-	-	-	-
70	**6**		**70**		**87**		GPP14							CD_CF	
71	**7**		**71**		**88**		GPP8							IRQ_CF	
72	**0**	**Port 9**	**72**		**90**		GPP10							INPACK	
73	**1**		**73**		**93**		GPP11							REG_CF	
74	**2**		**74**		**98**		GPP9							RESET_ CF	



75	**3**		**75**		**126**		GPK10							CD_PWREN	
76	**4**		**76**		**33**		GPK12								GPIO7
77	**5**		**77**		**36**		GPK13								GPIO8
78	**6**		**78**		**-**		-	-	-	-	-	-	-	-	-
79	**7**		**79**		**-**		-	-	-	-	-	-	-	-	-

					PicoMOD7										
Digital-IO					**PM7-Pin**	**Startintf**	**PIO-Pin**	**Function**							
IO-Pin	**Port**		**Registry settings**					*COM*	*I2C*	*SPI*	*USB*	*SD/ MMC*	*LCD*	*CF*	*sonst.*
0	**0**	**Port 0**	**0**	**UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1**	**18**	J2-3	GPA0_0	RXD0							
1	**1**		**1**		**17**	J2-5	GPA0_1	TXD0							
2	**2**		**2**		**20**	J2-6	GPA1_2	CTS0							
3	**3**		**3**		**19**	J2-4	GPA1_3	RTS0							
4	**4**		**4**		**22**	J7-9	GPA0_4	RXD1							
5	**5**		**5**		**21**	J7-10	GPA0_5	TXD1							
6	**6**		**6**		**24**		GPA1_0	RXD2							
7	**7**		**7**		**23**		GPA1_1	TXD2							
8	**0**	**Port 1**	**8**		**30**		GPH2_7				USB-PWR1				IRQ
9	**1**		**9**		**29**	**J5-9**	GPH3_5								GPIO5, IRQ
10	**2**		**10**		**32**		GPH2_6				USB-PWR2				IRQ
11	**3**		**11**		**31**	**J5-10**	GPD5		SDA						
12	**4**		**12**		**34**	**J5-11**	GPD6		SCL						
13	**5**		**13**		**42**	J4-13	GPH3_0								GPIO0, IRQ
14	**6**		**14**		**41**	**J5-1**	GPH3_1								GPIO1, IRQ
15	**7**		**15**		**44**	**J5-5**	GPH3_2								GPIO2, IRQ
16	**0**	**Port 2**	**16**		**43**	**J5-7**	GPH3_3								GPIO3, IRQ
17	**1**		**17**		**46**	**J5-8**	GPH3_4								GPIO4, IRQ
18	**2**		**18**		**45**		GPG0_0					SDCLK			
19	**3**		**19**		**48**		GPG0_1					SDCMD			
20	**4**		**20**		**47**		GPG0_2					SDDAT0			
21	**5**		**21**		**50**		GPG0_3					SDDAT1			
22	**6**		**22**		**49**		GPG0_4					SDDAT2			
23	**7**		**23**		**52**		GPG0_5					SDDAT3			
24	**0**	**Port**	**24**		**51**		GPG1_2					SD-CD			

25	**1**	**3**	**25**		**54**		GPH3_6					SD-PWR			GPIO6, IRQ
26	**2**		**26**		**53**		GPG1_0					SD-WP			
27	**3**		**27**		**56**		GPH0_3						LCD-ENA		IRQ
28	**4**		**28**		**55**		GPH0_4						LCD-DEN		IRQ
29	**5**		**29**		**58**		GPH0_6						VLCD-ON		IRQ
30	**6**		**30**		**57**		GPH0_5						VCFL-ON		IRQ
31	**7**		**31**		**60**		GPD0						VEEK		
32	**0**	**Port 4**	**32**		**61**		GPF0_4						VD0		
33	**1**		**33**		**64**		GPF0_5						VD1		
34	**2**		**34**		**63**		GPF0_6						VD7		
35	**3**		**35**		**66**		GPF0_7						VD3		
36	**4**		**36**		**65**		GPF1_0						VD4		
37	**5**		**37**		**68**		GPF1_1						VD5		
38	**6**		**38**		**67**		GPF1_2						VD6		
39	**7**		**39**		**70**		GPF1_3						VD7		
40	**0**	**Port 5**	**40**		**69**		GPF2_0						VD12		
41	**1**		**41**		**72**		GPF2_1						VD13		
42	**2**		**42**		**71**		GPF2_2						VD14		
43	**3**		**43**		**74**		GPF2_3						VD15		
44	**4**		**44**		**73**		GPF2_6						VD23		
45	**5**		**45**		**76**		GPF2_7						VD19		
46	**6**		**46**		**75**		GPF3_0						VD20		
47	**7**		**47**		**78**		GPF3_1						VD21		
48	**0**	**Port 6**	**48**		**77**		GPF3_2						VD22		
49	**1**		**49**		**80**		GPF3_3						VD23		
50	**2**		**50**		**79**		GPF0_0						VLINE		
51	**3**		**51**		**82**		GPF0_1						VFRAME		
52	**4**		**52**		**81**		GPF0_2						VM		
53	**5**		**53**		**86**		GPF0_3						VCLK		
54	**6**		**54**		**-**		-	-	-	-	-	-	-	-	-
55	**7**		**55**		**-**		-	-	-	-	-	-	-	-	-
56	**0**	**Port 7**	**56**		**-**		-	-	-	-	-	-	-	-	-
57	**1**		**57**		**-**		-	-	-	-	-	-	-	-	-
58	**2**		**58**		**-**		-	-	-	-	-	-	-	-	-
59	**3**		**59**		**-**		-	-	-	-	-	-	-	-	-
60	**4**		**60**		**-**		-	-	-	-	-	-	-	-	-
61	**5**		**61**		**-**		-	-	-	-	-	-	-	-	-
62	**6**		**62**		**-**		-	-	-	-	-	-	-	-	-
63	**7**		**63**		**-**		-	-	-	-	-	-	-	-	-
64	**0**	**Port 8**	**64**		**1**	**J5-6**	GPB3			SPICS0					
65	**1**		**65**		**2**	**J5-2**	GPB1			SPICLK0					

66	**2**		**66**		**3**	**J5-4**	GPB0			SPIMISO 0					
67	**3**		**67**		**4**	**J5-3**	GPB2			SPIMOSI 0					
68	**4**		**68**		**-**		-	-	-	-	-	-	-	-	-
69	**5**		**69**		**-**		-	-	-	-	-	-	-	-	-
70	**6**		**70**		**87**		GPK3_5							CD_CF	
71	**7**		**71**		**88**		GPK3_1							IRQ_CF	
72	**0**		**72**		**90**		GPK3_3							INPACK	
73	**1**		**73**		**93**		GPK3_4							REG_CF	
74	**2**		**74**		**98**		GPK3_2							RESET_ CF	
75	**3**	**Port 9**	**75**		**126**		GPH2_5							CD_PW REN	IRQ
76	**4**		**76**		**33**		GPH3_7								GPIO7, IRQ
77	**5**		**77**		**36**		GPH2_0								GPIO8, IRQ
78	**6**		**78**		**-**		-	-	-	-	-	-	-	-	-
79	**7**		**79**		**-**		-	-	-	-	-	-	-	-	-

4.2 Port description QBliss

The port numbering of QBliss is much more easier compared to PicoMOD. On QBliss port number is equal to pin number. That means if you want to use pin 196 (FAN_PWMOUT) as I/O, port number is 196.
The QBliss connector X1 has a total of 230 pins.
For configuration you can use registry values **UseAsIO*x*/DataDir*x*/DataInit*x***. These values are 32 bit DWORD registry values. Each value (x=A..x=H) configures 4 ports. In contrast to this, you can also use registry values **UseAsIO/DataDir/DataInit** with data type HEX.

Digital-IO					**X1-Pin**	**picoITX**	**PIO-Pin**						
IO-Pin	**Port**		**Registry settings**					*COM*	*I2C*	*SPI*	*SD/ MMC*	*LCD*	*sonst.*
0	**0**	**Port 0**	**0**	**UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1**	**--**		-						
1	**1**		**1**		**1**		-						
2	**2**		**2**		**2**		-						
3	**3**		**3**		**3**		-						
4	**4**		**4**		**4**		-						
5	**5**		**5**		**5**		-						
6	**6**		**6**		**6**		-						
7	**7**		**7**		**7**		-						
8	**0**	**Port 1**	**8**		**8**		-						
9	**1**		**9**		**9**		-						

10	2		10		10		-						
11	3		11		11		-						
12	4		12		12		-						
13	5		13		13		-						
14	6		14		14		-						
15	7		15		15		-						
16	0	Port 2	16		16		-						
17	1		17		17	X21	GPH2_0						WAKE#, IRQ
18	2		18		18		-						
19	3		19		19		-						
20	4		20		20	X21	GPH1_7						PWR_BTN#, IRQ
21	5		21		21	X21	GPH2_1						SLP_BTN#, IRQ
22	6		22		22	X21	GPG2_2						LID_BTN#, IRQ
23	7		23		23		-						
24	0	Port 3	24		24		-						
25	1		25		25		-						
26	2		26		26		-						
27	3		27		27		-						
28	4		28		28	X21	GPH1_6						RST_BTN#, IRQ
29	5		29		29		-						
30	6		30		30		-						
31	7		31		31		-						
32	0	Port 4	32		32		-						
33	1		33		33		-						
34	2		34		34		-						
35	3		35		35		-						
36	4		36		36		-						
37	5		37		37		-						
38	6		38		38		-						
39	7		39		39		-						
40	0	Port 5	40		40		-						
41	1		41		41		-						
42	2		42		42	X17	GPG0_0				SDIO_ CLK#		
43	3		43		43	X17	GPG1_2				SDIO_ CD#		
44	4		44		44		-						
45	5		45		45	X17	GPG0_1				SDIO_ CMD		
46	6		46		46	X17	GPH1_0				SDIO_ WP		IRQ
47	7		47		47		-						
48	0	Port 6	48		48	X17	GPG0_3				SDIO_ DAT1		
49	1		49		49	X17	GPG0_2				SDIO_ DAT0		

50	**2**		**50**	**50**	X17	GPG0_5				SDIO_DAT3		
51	**3**		**51**	**51**	X17	GPG0_4				SDIO_DAT2		
52	**4**		**52**	**52**	X17	GPG0_7				SDIO_DAT5		
53	**5**		**53**	**53**	X17	GPG0_6				SDIO_DAT4		
54	**6**		**54**	**54**	X17	GPG1_1	-	-	-	SDIO_DAT7	-	-
55	**7**		**55**	**55**	X17	GPG1_0	-	-	-	SDIO_DAT6	-	-
56	**0**	**Port 7**	**56**	**56**		-	-	-	-	-	-	-
57	**1**		**57**	**57**		-	-	-	-	-	-	-
58	**2**		**58**	**58**		-	-	-	-	-	-	-
59	**3**		**59**	**59**		GPC2	-	-	-	-	-	AC97_SYNC
60	**4**		**60**	**60**		-	-	-	-	-	-	-
61	**5**		**61**	**61**		GPC1	-	-	-	-	-	AC97_RST#
62	**6**		**62**	**62**		-	-	-	-	-	-	-
63	**7**		**63**	**63**		GPC0	-	-	-	-	-	AC97_BITCLK
64	**0**	**Port 8**	**64**	**64**	X21	GPH3_3						SMB_ALERT#, IRQ
65	**1**		**65**	**65**		GPC3						AC97_SDI
66	**2**		**66**	**66**	X19	GPD6		I2C_CLK				
67	**3**		**67**	**67**		GPC4						AC97_SDO
68	**4**		**68**	**68**	X19	GPD5	-	I2C_DAT	-	-	-	
69	**5**		**69**	**69**	TP13	GPH0_7	-	-	-	-	-	THRM#, IRQ
70	**6**		**70**	**70**	X21	GPH1_5						WDTRIG#, IRQ
71	**7**		**71**	**71**		-						
72	**0**	**Port 9**	**72**	**72**	X21	GPH3_1						WDOUT, IRQ
73	**1**		**73**	**73**		-						
74	**2**		**74**	**74**		-						
75	**3**		**75**	**75**		-						
76	**4**		**76**	**76**		-						
77	**5**		**77**	**77**		-						
78	**6**		**78**	**78**		-	-	-	-	-	-	-
79	**7**		**79**	**79**		-	-	-	-	-	-	-
80	**0**	**Port 10**	80	**80**		-						
81	**1**		81	**81**		-						
82	**2**		82	**82**		-						
82	**3**		83	**83**		-						
84	**4**		84	**84**		-						
85	**5**		85	**85**		-						
86	**6**		86	**86**		-						
87	**7**		87	**87**		-						

88	**0**	**Port 11**	88		**88**		-						
89	**1**		89		**89**		-						
90	**2**		90		**90**		-						
91	**3**		91		**91**		-						
92	**4**		92		**92**		-						
93	**5**		93		**93**		-						
94	**6**		94		**94**		-						
95	**7**		95		**95**		-						
96	**0**	**Port 12**	96		**96**		-						
97	**1**		97		**97**		-						
98	**2**		98		**98**		-						
99	**3**		99		**99**		-						
100	**4**		100		**100**		-						
101	**5**		101		**101**		-						
102	**6**		102		**102**		-						
103	**7**		103		**103**		-						
104	**0**	**Port 13**	104		**104**		-						
105	**1**		105		**105**		-						
106	**2**		106		**106**		-						
107	**3**		107		**107**		-						
108	**4**		108		**108**		-						
109	**5**		109		**109**		-						
110	**6**		110		**110**		-						
111	**7**		111		**111**		GPH0_6					LVDS_PPEN	IRQ
112	**0**	**Port 14**	112		**112**		GPH0_4					LVDS_BLEN	IRQ
113	**1**		113		**113**		-						
114	**2**		114		**114**		-						
115	**3**		115		**115**		-						
116	**4**		116		**116**		-						
117	**5**		117		**117**		-						
118	**6**		118		**118**		-						
119	**7**		119		**119**		-						
120	**0**	**Port 15**	120		**120**		-						
121	**1**		121		**121**		-						
122	**2**		122		**122**		-						
123	**3**		123		**123**		GPD0					LVDS_BLT_CTRL	
124	**4**		124		**124**		-						
125	**5**		125		**125**		GPG2_0					LVDS_DID_DAT	
126	**6**		126		**126**		GPG2_4					LCDS_BLC_DAT	
127	**7**		127		**127**		GPG2_1					LVDS_DID_CLK	
128	**0**	**Port 16**	128		**128**		GPG2_5					LVDS_BLC_CLK	
129	**1**		129		**129**		-						
130	**2**		130		**130**		-						

131	3		131	**131**		-						
132	4		132	**132**		-						
133	5		133	**133**		-						
134	6		134	**134**		-						
135	7		135	**135**		-						
136	0	Port 17	136	**136**		-						
137	1		137	**137**		-						
138	2		138	**138**		-						
139	3		139	**139**		-						
140	4		140	**140**		-						
141	5		141	**141**		-						
142	6		142	**142**		-						
143	7		143	**143**		-						
144	0	Port 18	144	**144**		-						
145	1		145	**145**		-						
146	2		146	**146**		-						
147	3		147	**147**		-						
148	4		148	**148**		-						
149	5		149	**149**		-						
150	6		150	**150**		GPG2_2						HDMI_CTRL_DAT
151	7		151	**151**		-						
152	0	Port 19	152	**152**		GPG2_3						HDMI_CTRL_CLK
153	1		153	**153**		GPH0_5						HDMI_PD#, IRQ
154	2		154	**154**		-						
155	3		155	**155**		-						
156	4		156	**156**		-						
157	5		157	**157**		-						
158	6		158	**158**		-						
159	7		159	**159**		-						
160	0	Port 20	160	**160**		-						
161	1		161	**161**	X19	GPA0_4	RXD1					(PCIE3_TX+)
162	2		162	**162**	X19	GPA0_5	TXD1					(PCIE3_TX-)
163	3		163	**163**	X19	GPA0_6	CTS1					(PCIE3_RX+)
164	4		164	**164**	X19	GPA0_7	RTS1					(PCIE3_RX-)
165	5		165	**165**		-						
166	6		166	**166**		-						
167	7		167	**167**		-						
168	0	Port 21	168	**168**		-						
169	1		169	**169**		-						
170	2		170	**170**		-						
171	3		171	**171**		-						
172	4		172	**172**		-						

173	5		173		**173**		-						
174	6		174		**174**		-						
175	7		175		**175**		-						
176	0	Port 22	176		**176**		-						
177	1		177		**177**		-						
178	2		178		**178**		-						
179	3		179		**179**		-						
180	4		180		**180**		-						
181	5		181		**181**		-						
182	6		182		**182**		-						
183	7		183		**183**		-						
184	0	Port 23	184		**184**		-						
185	1		185		**185**	X5	GPA0_0	RXD0					(LPC_AD0)
186	2		186		**186**	X5	GPA0_1	RXD1					(LPC_AD1)
187	3		187		**187**	X5	GPA0_2	CTS1					(LPC_AD2)
188	4		188		**188**	X5	GPA0_3	RTS1					(LPC_AD3)
189	5		189		**189**		-						
190	6		190		**190**		-						
191	7		191		**191**		-						
192	0	Port 24	192		**192**		-						
193	1		193		**193**		-						
194	2		194		**194**		GPH3_0						SPKR, IRQ
195	3		195		**195**	X21	GPH0_1						FAN_TACHOI, IRQ
196	4		196		**196**	X21	GPH0_0						FAN_PWMOT, IRQ
197	5		197		**197**		-						
198	6		198		**198**		-						
199	7		199		**199**	X19	GPB2			SPI_MOSI			
200	0	Port 25	200		**200**		-						
201	1		201		**201**	X19	GPB0			SPI_MISO			
202	2		202		**202**		-						
203	3		203		**203**	X19	GPB1			SPI_CLK			
204	4		204		**204**		-						
205	5		205		**205**		-						
206	6		206		**206**		-						
207	7		207		**207**		-						
208	0	Port 26	208		**208**		-						
209	1		209		**209**		-						
210	2		210		**210**		-						
211	3		211		**211**		-						
212	4		212		**212**		-						
213	5		213		**213**		-						
214	6		214		**214**		-						

215	7		215		215		-						
216	0	Port 27	216		216		-						
217	1		217		217		-						
218	2		218		218		-						
219	3		219		219		-						
220	4		220		220		-						
221	5		221		221		-						
222	6		222		222		-						
223	7		223		223		-						
224	0	Port 28	224		224		-						
225	1		225		225		-						
226	2		226		226		-						
227	3		227		227		-						
228	4		228		228		-						
229	5		229		229		-						
230	6		230		230		-						

4.3 Port description armStone

The port numbering of armStone is similar to QBliss. That means port number is equal to pin number of connector “feature connector”. That means if you want to use pin 1 as I/O, port number is 1.
The armStone feature connector has a total of 66 pins.
For configuration you can use registry values **UseAsIO*x*/DataDir*x*/DataInit*x***. These values are 32 bit DWORD registry values. Each value (x=A..x=H) configures 4 ports. In contrast to this, you can also use registry values **UseAsIO/DataDir/DataInit** with data type HEX.

Digital-IO				Pin	PIO-Pin	armStoneA8 Board Revision: 1.10					
IO-Pin	Port		Registry settings			*COM*	*I2C*	*SPI*	*SD/ MMC*	*LCD*	*sonst.*
0	0	Port 0	0	---	-						
1	1		1	1	GPH1_7						IRQ
2	2		2	2	GPH2_7						IRQ
3	3		3	3	GPH2_6						IRQ
4	4		4	4	GPH2_5						IRQ
5	5		5	5	GPH2_4						IRQ
6	6		6	6	GPH2_3						IRQ
7	7		7	7	GPH2_2						IRQ
8	0	Port 1	8	8	GPH2_1						IRQ
9	1		9	9	GPH2_0						IRQ

UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1

10	2		10	10	GPB2			MISO			
11	3		11	11	GPB3			MOSI			
12	4		12	12	GPA0_4	RxD1					
13	5		13	13	GPH1_6						IRQ
14	6		14	14	GPA0_5	TxD1					
15	7		15	15	GPH1_0						IRQ
16	0	Port 2	16	16	-						
17	1		17	17	GPH3_7						IRQ
18	2		18	18	GPH3_6						IRQ
19	3		19	19	GPH3_5						IRQ
20	4		20	20	GPH3_4						IRQ
21	5		21	21	GPH3_3						IRQ
22	6		22	22	GPH3_2						IRQ
23	7		23	23	GPH3_1						IRQ
24	0	Port 3	24	24	GPH3_0						IRQ
25	1		25	25	-						
26	2		26	26	-						
27	3		27	27	-						
28	4		28	28	GPD0_1						PWM
29	5		29	29	-						
30	6		30	30	GPD0_2						PWM
31	7		31	31	-						
32	0	Port 4	32	32	GPD0_3						PWM
33	1		33	33	-						
34	2		34	34	GPH0_5						IRQ
35	3		35	35	-						
36	4		36	36	GPA1_0	RxD2					
37	5		37	37	-						
38	6		38	38	GPA1_1	TxD2					
39	7		39	39	-						
40	0	Port 5	40	40	GPD0_1						PWM
41	1		41	41	-						
42	2		42	42	-						
43	3		43	43	-						
44	4		44	44	-						
45	5		45	45	-						
46	6		46	46	-						
47	7		47	47	-						
48	0	Port 6	48	48	-						
49	1		49	49	-						
50	2		50	50	-						
51	3		51	51	-						
52	4		52	52	-						

53	5		53		53	-						
54	6		54		54	-						
55	7		55		55	-						
56	0		56		56	-						
57	1		57		57	-						
58	2		58		58	-						
59	3	Port 7	59		59	-						
60	4		60		60	-						
61	5		61		61	-						
62	6		62		62	-						
63	7		63		63	-						
64	0		64		64	-						
65	1		65		65	-						
66	2		66		66	-						
	3	Port 8	67			-						
	4		68			-						
	5		69			-						
	6		70			-						
	7		71			-						

4.4 Port description nanoRISC-A8

The port numbering of nanoRISC is is the same as for QBlissA8. On nanoRISC port number is equal to pin number. That means if you want to use pin 196 (FAN_PWMOUT) as I/O, port number is 196.
The nanoRISC connector X1 has a total of 230 pins.
For configuration you can use registry values **UseAsIO*x*/DataDir*x*/DataInit*x***. These values are 32 bit DWORD registry values. Each value (x=A..x=H) configures 4 ports. In contrast to this, you can also use registry values **UseAsIO/DataDir/DataInit** with data type HEX.

Digital-IO				X1-Pin		PIO-Pin						
IO-Pin	Port		Registry settings				*COM*	*I2C*	*SPI*	*SD/ MMC*	*LCD*	*sonst.*
0	0	Port 0	0	DataDir / DataInit / IRQCfg0 /	--		-						
1	1		1		1		-						
2	2		2		2		-						
3	3		3		3		-						
4	4		4		4		-						

5	5		5		5		-					
6	6		6		6		-					
7	7		7		7		-					
8	0	Port 1	8		8		-					
9	1		9		9		-					
10	2		10		10		-					
11	3		11		11		-					
12	4		12		12		-					
13	5		13		13		-					
14	6		14		14		-					
15	7		15		15		-					
16	0	Port 2	16		16		-					
17	1		17		17							
18	2		18		18							
19	3		19		19							
20	4		20		20							
21	5		21		21							
22	6		22		22							
23	7		23		23							
24	0	Port 3	24		24							
25	1		25		25							
26	2		26		26							
27	3		27		27							
28	4		28		28							
29	5		29		29		-					
30	6		30		30		-					
31	7		31		31		-					
32	0	Port 4	32		32		-					
33	1		33		33		-					
34	2		34		34		GPA0_2					UART0_CTS#
35	3		35		35		GPA0_3					UART0_RTS#
36	4		36		36		GPA0_0					UART0_RxD
37	5		37		37		GPA0_1					UART0_TxD
38	6		38		38		GPJ2_0					GPIO/ UART_CLK
39	7		39		39		GPA1_2					UART1_CTS#/ UART2_RxD
40	0	Port 5	40		40		GPA1_3					UART1_RTS#/ UART2_TxD
41	1		41		41		GPA1_0					UART1_RxD
42	2		42		42		GPA1_1					UART1_TxD
43	3		43		43		GPG0_2					SD0_CD#
44	4		44		44		GPG0_0					SD0_CLK
45	5		45		45		GPG0_1					SD0_CMD
46	6		46		46		GPG0_2					SD0_D0

47	7		47		47		GPG0_3						SD0_D1
48	0		48		48		GPG0_4						SD0_D2
49	1		49		49		GPG0_5						SD0_D3
50	2		50		50		GPH1_3						IRQ/ SD0_WP#
51	3	Port 6	51		51								
52	4		52		52								
53	5		53		53		GPB0			SPI0_CLK			
54	6		54		54			-	-	SP1_CLK-		-	-
55	7		55		55			-	-	-		-	-
56	0		56		56		-	-	-	-	-	-	-
57	1		57		57		GPB1	-	-	SPI1_CS#	-	-	-
58	2		58		58		GPB2	-	-	- SPI0_MISO	-	-	-
59	3	Port 7	59		59		GPB3	-	-	SPI0_MOSI	-	-	
60	4		60		60		-	-	-	SPI1_CS#	-	-	
61	5		61		61		-	-	-	SPI1_MISO	-	-	
62	6		62		62		-	-	-	SPI1_MOSI	-	-	
63	7		63		63		GPC0_1	-	-	-	-	-	AC97_RESET#
64	0		64		64		-						
65	1		65		65		GPC0_0						AC97_BITCLK
66	2		66		66		-						
67	3	Port 8	67		67		GPC0_3						AC97_SDI
68	4		68		68		GPC0_4	-		-	-	-	AC97_SDO
69	5		69		69		GPC0_2	-	-	-	-	-	AC97_SYNC
70	6		70		70		-						
71	7		71		71		-						
72	0		72		72		-						
73	1		73		73		GPH1_1						IRQ/ BL_POW_EN#
74	2		74		74		-						
75	3	Port 9	75		75		-						
76	4		76		76		-						
77	5		77		77		-						
78	6		78		78		GPE1_3	-	-	-	-	-	CAM_CLKOUT
79	7		79		79		GPE0_3	-	-	-	-	-	CAM_D0
80	0		80		80		GPE0_4						CAM_D1
81	1		81		81		GPE0_5						CAM_D2
82	2	Port 10	82		82		GPE0_6						CAM_D3
82	3		83		83		GPE0_7						CAM_D4
84	4		84		84		GPE1_0						CAM_D5
85	5		85		85		GPE1_1						CAM_D6

86	**6**		86		**86**		GPE1_2						CAM_D7
87	**7**		87		**87**		GPE1_4						CAM_FIELD
88	**0**		88		**88**		GPE0_2						CAM_HREF
89	**1**		89		**89**		GPE0_0						CAM_PCLK
90	**2**		90		**90**		GPJ0_0						CAM_RESET
91	**3**	**Port 11**	91		**91**		GPE0_1						CAM_VSYNC
92	**4**		92		**92**		-						
93	**5**		93		**93**		-						
94	**6**		94		**94**		-						
95	**7**		95		**95**		-						
96	**0**		96		**96**		-						
97	**1**		97		**97**		-						
98	**2**		98		**98**		-						
99	**3**	**Port 12**	99		**99**		-						
100	**4**		100		**100**		-						
101	**5**		101		**101**		-						
102	**6**		102		**102**		-						
103	**7**		103		**103**		-						
104	**0**		104		**104**		-						
105	**1**		105		**105**		-						
106	**2**		106		**106**		-						
107	**3**	**Port 13**	107		**107**		-						
108	**4**		108		**108**		-						
109	**5**		109		**109**		-						
110	**6**		110		**110**		-						
111	**7**		111		**111**								
112	**0**		112		**112**								
113	**1**		113		**113**								
114	**2**		114		**114**		-						
115	**3**	**Port 14**	115		**115**		-						
116	**4**		116		**116**		-						
117	**5**		117		**117**		-						
118	**6**		118		**118**		-						
119	**7**		119		**119**		-						
120	**0**		120		**120**		-						
121	**1**		121		**121**		-						
122	**2**		122		**122**		-						
123	**3**	**Port 15**	123		**123**								
124	**4**		124		**124**								
125	**5**		125		**125**								
126	**6**		126		**126**								
127	**7**		127		**127**								
128	**0**	**Port**	128		**128**								

129	1	16	129		129		-						
130	2		130		130		-						
131	3		131		131		-						
132	4		132		132		-						
133	5		133		133		-						
134	6		134		134		-						
135	7		135		135		-						
136	0	Port 17	136		136		-						
137	1		137		137		-						
138	2		138		138		-						
139	3		139		139		-						
140	4		140		140		-						
141	5		141		141		-						
142	6		142		142		GPH3_4						IRQ
143	7		143		143		GPH3_3						IRQ
144	0	Port 18	144		144		GPH0_0						IRQ
145	1		145		145		GPH0_1						IRQ
146	2		146		146		GPH0_2						IRQ
147	3		147		147		GPH0_3						IRQ
148	4		148		148		GPH0_4						IRQ
149	5		149		149		GPH0_5						IRQ
150	6		150		150		GPH0_6						IRQ
151	7		151		151		GPD0_0						
152	0	Port 19	152		152		GPH2_0						IRQ
153	1		153		153		GPH2_1						IRQ
154	2		154		154		-						USBH_PWEN
155	3		155		155		GPH2_2						IRQ
156	4		156		156		-						USBH_OC#
157	5		157		157		GPH1_4						IRQ
158	6		158		158		GPG3_0-						USBOTG_OC#
159	7		159		159		GPH1_4						IRQ
160	0	Port 20	160		160		-						
161	1		161		161		GPJ0_2						TS_CLK
162	2		162		162		GPJ0_3						TS_SYNC
163	3		163		163		GPJ0_4						TS_VAL/ HDMI_TX1N
164	4		164		164		GPJ0_5						TS_DATA/ HDMI_TX0N
165	5		165		165		GPJ0_6						TS_ERROR/ HDMI_TX1P
166	6		166		166		GPJ2_2						HDMI_TX0P
167	7		167		167		GPJ2_3						HDMI_TXCN
168	0	Port 21	168		168		GPJ2_4						HDMI_TX2N
169	1		169		169		GPJ2_5						HDMI_TXCP
170	2		170		170		GPJ2_6						HDMI_TX2P

171	3		171		**171**		-						
172	4		172		**172**		-						
173	5		173		**173**		GPD1_1		I2C0_SCL				
174	6		174		**174**		GPD1_0		I2C0_SDA				
175	7		175		**175**		-						
176	0		176		**176**		-						
177	1		177		**177**		GPD1_3		I2C1_SCL				
178	2		178		**178**		GPD1_2		I2C1_SDA				
179	3	**Port 22**	179		**179**		-						
180	4		180		**180**		-						
181	5		181		**181**		-						
182	6		182		**182**		-						
183	7		183		**183**		-						
184	0		184		**184**		GPF0_0					LCD_HSYNC	
185	1		185		**185**		GPH1_2					LCD_POW_EN#	IRQ
186	2		186		**186**		GPF0_3					LCD_VCLK	
187	3	**Port 23**	187		**187**		GPF0_4					LCD_VD0	
188	4		188		**188**		GPF0_5					LCD_VD1	
189	5		189		**189**		GPF0_6					LCD_VD2	
190	6		190		**190**		GPF0_7					LCD_VD3	
191	7		191		**191**		GPF1_0					LCD_VD4	
192	0		192		**192**		GPF1_1					LCD_VD5	
193	1		193		**193**		GPF1_2					LCD_VD6	
194	2		194		**194**		GPF1_3					LCD_VD7	
195	3	**Port 24**	195		**195**		GPF1_4					LCD_VD8	
196	4		196		**196**		GPF1_5					LCD_VD9	
197	5		197		**197**		GPF1_6					LCD_VD10	
198	6		198		**198**		GPF1_7					LCD_VD11	
199	7		199		**199**		GPF2_0					LCD_VD12	
200	0		200		**200**		GPF2_1					LCD_VD13	
201	1		201		**201**		GPF2_2					LCD_VD14	
202	2		202		**202**		GPF2_3					LCD_VD15	
203	3	**Port 25**	203		**203**		GPF2_4					LCD_VD16	
204	4		204		**204**		GPF2_5					LCD_VD17	
205	5		205		**205**		GPF2_6					LCD_VD18	
206	6		206		**206**		GPF2_7					LCD_VD19	
207	7		207		**207**		GPF3_0					LCD_VD20	
208	0		208		**208**		GPF3_1					LCD_VD21	
209	1	**Port 26**	209		**209**		GPF3_2					LCD_VD22	
210	2		210		**210**		GPF3_3					LCD_VD23	
211	3		211		**211**		GPF0_2					LCD_VDEN	

212	4		212		212		GPF0_1					LCD_VSYNC	
213	5		213		213		-						
214	6		214		214		-						
215	7		215		215		-						
216	0	Port 27	216		216		-						
217	1		217		217		-						
218	2		218		218		-						
219	3		219		219		-						
220	4		220		220		-						
221	5		221		221		-						
222	6		222		222		-						
223	7		223		223		-						
224	0	Port 28	224		224		-						
225	1		225		225		-						
226	2		226		226		-						
227	3		227		227		-						
228	4		228		228		-						
229	5		229		229		-						
230	6		230		230		-						

4.5 Port description NetDCU14

The following table is useful if you want to use **UseAsIO*x*/DataDir*x*/DataInit*x***. These values are 32 bit DWORD registry values. Each value (x=A) configures 4 ports. In contrast to this, you can also use registry values **UseAsIO/DataDir/DataInit** with data type HEX.

Port 0

Bit	7	6	5	4	3	2	1	0
Pin	2	3	4	5	6	7	8	9
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
UseAsIOA Bit	7	6	5	4	3	2	1	0
DataDirA Bit	7	6	5	4	3	2	1	0
DataInitA Bit	7	6	5	4	3	2	1	0
IRQCfg0A IRQCfg1A IRQCfg2A	7	6	5	4	3	2	1	0

Port1

Bit	7	6	5	4	3	2	1	0
Pin	---	---	---	---	10	11	13	15
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
UseAsIOA Bit	15	14	13	12	11	10	9	8
DataDirA Bit	15	14	13	12	11	10	9	8
DataInitA Bit	---	---	---	---	11	10	9	8
IRQCfg0A IRQCfg1A IRQCfg2A	---	---	---	---	---	---	---	---

Port 2

Bit	7	6	5	4	3	2	1	0
Pin	17	18	19	20	21	22	23	24
R/W	R	R	R	R	R	R	R	R
UseAsIOA Bit	23	22	21	20	19	18	17	16
DataDirA Bit	23	22	21	20	19	18	17	16
DataInitA Bit	23	22	21	20	19	18	17	16
IRQCfg0A IRQCfg1A IRQCfg2A	---	---	---	---	---	---	---	---

Table 11: Digital I/O - NetDCU14 Port0 - 2

Digital-IO					J5-Pin	PIO-Pin			
IO-Pin	Port		Registry settings				*I2C*	*SPI*	
0	0	Port 0	0	UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1	9	GPH2_0			IRQ
1	1		1		8	GPH2_1			IRQ
2	2		2		7	GPH2_2			IRQ
3	3		3		6	GPH2_3			IRQ
4	4		4		5	GPH2_4			IRQ
5	5		5		4	GPH2_5			IRQ
6	6		6		3	GPH2_6			IRQ
7	7		7		2	GPH2_7			IRQ
8	0	Port 1	8		15	GPH1_0		CLK	IRQ
9	1		9		13	GPH1_6		CS#	
10	2		10		11	GPB3	SCL	MOSI	
11	3		11		10	GPB2	SDA	MISO	
12	4		12		---	---			
13	5		13		---	---			
14	6		14		---	---			
15	7		15		---	---			
16	0	Port 2	16		24	GPH3_0			IRQ
17	1		17		23	GPH3_1			IRQ
18	2		18		22	GPH3_2			IRQ
19	3		19		21	GPH3_3			IRQ
20	4		20		20	GPH3_4			IRQ
21	5		21		19	GPH3_5			IRQ
22	6		22		18	GPH3_6			IRQ
23	7		23		17	GPH3_7			IRQ
24	0	Port 3	24		---	---			
25	1		25		---	---			
26	2		26		---	---			
27	3		27		---	---			
28	4		28		---	---			
29	5		29		---	---			
30	6		30		---	---			
31	7		31		---	---			

4.6 Interrupt configuration

IRQCfg2	IRQCfg1	IRQCfg0	Function
0	0	0	Interrupt Disabled
0	0	1	Rising Edge Enabled
0	1	0	Falling Edge Enabled
0	1	1	Rising and Falling Edge Enabled
1	0	0	Interrupts Disabled
1	0	1	High Level Enabled
1	1	0	Low Level Enabled

Table 12: Digital I/O - Interrupt configuration

4.7 Programming example

Headerfile:

```
#include <dio_sdk.h>
```

Listing 14: Digital I/O: Headerfile

A. Opening a digital port

```
HANDLE hDIO;
hDIO = CreateFile( _T("DIO1:"), GENERIC_READ|GENERIC_WRITE, 0, NULL, OPEN_EXISTING,
                   FILE ATTRIBUTE NORMAL, NULL );

if( INVALID_HANDLE_VALUE == hDIO )
{
    ERRORMSG(1,(L"INVALID HANDLE VALUE\r\n"));
    return(FALSE);
}
```

Listing 15: Digital I/O: Open a port

H. Write data to port

```
unsigned char data = 0xAA;
DWORD dwBytesWrite = 1;
WriteFile( hDIO, &data, dwBytesWrite, &dwBytesWrite, NULL );
if( dwBytesWrite != 1 )
{
  ERRORMSG(1,(L"Can not write to DIO1. LE = 0x%x\r\n",GetLastError()));
}
```

Listing 16: Digital I/O: write data to port

I. Change port

```
/* The following code sets file pointer to
 * Port 1. After this function you can use
 * ReadFile() or Write File() to access Port 1
 */
LONG lDistance = 1;
SetFilePointer( hDIO, lDistance, NULL, FILE_BEGIN);
```

Listing 17: Digital I/O: changing the port

J. Get / Set / Clear individual pin

```
DWORD dwOutCount = 0;
DWORD dwPin = 7;
BYTE byPinLevel = 0xAA;
/*
 *      Get level of pin.
 *      dwPin = pin of interest (7 for GPIO7 which is Pin#2 on J5) = input parameter.
 *      byPinLevel = level of pin = output parameter. 0 = 0V, 1 = 3.3V
 */
DeviceIoControl(g_hDio, IOCTL_DIO_GET_PIN, &dwPin, sizeof(BYTE), &byPinLevel, sizeof(BYTE),
                &dwOutCount, NULL);
DeviceIoControl(g hDio, IOCTL DIO SET PIN, &dwPin, sizeof(BYTE), NULL, 0, &dwOutCount, NULL);
DeviceIoControl(g_hDio, IOCTL_DIO_CLR_PIN, &dwPin, sizeof(BYTE), NULL, 0, &dwOutCount, NULL);
```

Listing 18: Digital I/O: Access individual pin

K. Using Interrupts (use dio_sdk.h):

```
/* Open the digitalio port */
HANDLE hDIO = CreateFile(_T("DIO1:"), GENERIC_WRITE|GENERIC_READ, 0, NULL, OPEN_EXISTING
                                    , FILE_ATTRIBUTE_NORMAL, NULL );
//Add error handling here

/*
 *  WAITIRQ.dwPin = pin number to use as irq.
 *  I.e.: GPIO2 = PIN44 = IO15, dwPin must set to 15
 *  WAITIRQ.dwTimeout = Timeout in ms to wait for irq.
 *  Used for IOCTL_DIO_WAIT_IRQ.
 */
WAITIRQ cWaitIrq[2];
cWaitIrq[0].dwPin = 15;
cWaitIrq[0].dwTimeout = 20000;
cWaitIrq[1].dwPin = 16;
cWaitIrq[1].dwTimeout = 20000;

/* Request a sysintr  */
DeviceIoControl(hDIO,IOCTL DIO REQUEST IRQ, &cWaitIrq[0].dwPin, sizeof(DWORD), NULL
                                            , 0, NULL, NULL);

/* Wait for a sysintr */
DWORD dwWaitRes = -1;         /* Return value that
                               * indicates the event result.
                               * WAIT_OBJECT_0,
                               * WAIT ABANDONED,
                               * WAIT TIMEOUT */

DeviceIoControl(hDIO,IOCTL_DIO_WAIT_IRQ, &cWaitIrq[0], sizeof(WAITIRQ), &dwWaitRes
                                        , sizeof(DWORD), NULL, NULL );


/* Call InterruptDone on a sysintr */
DeviceIoControl(hDIO,IOCTL DIO DONE IRQ, &cWaitIrq[0].dwPin, sizeof(DWORD), NULL, 0
                                        , NULL, NULL );







/* Release a sysintr */
DeviceIoControl(hDIO,IOCTL_DIO_RELEASE_IRQ, &cWaitIrq[0].dwPin, sizeof(DWORD), NULL, 0
                                            , NULL, NULL );

/* Close the digitalio port */
CloseHandle(hDIO);
```

Listing 19: Digital I/O: Using Interrupts

L. Closing port

```
CloseHandle(hDIO);
```

Listing 20: Digital I/O: Closing port

5 Driver for Serial I/O (UART)

Implemented on: PM6,PM7A,QA8,ASA8,ND14,NRA8

PicoMOD has a maximum of four serial ports (UART). QBliss has a maximum of two serial ports (UART). armStone has a maximum of 3 serial ports. Following there is an explanation of settings not available in the standard Windows CE driver.

Installation of the driver is done by setting some registry values under the following registry key:

```
[HKLM\Drivers\BuiltIn\Serial1]
[HKLM\Drivers\BuiltIn\Serial2]
[HKLM\Drivers\BuiltIn\Serial3]
[HKLM\Drivers\BuiltIn\Serial4]
```

Settings:

Key	Value	Comment
Priority256	dword:	Priority for the serial interface thread. Default: 159
DIOBitRTS	dword:	This optional value specifies the bit number of the DIO ports which will be used for the RTS function of the driver.
AFCEnable	dword:	Auto Flow Control Default: 0 Platform: ASA8

Table 13: UART - Registry settings

Remark:

The driver support RTS_CONTROL_TOGGLE. This function and the RTS pin can be used for RS485 interface.

5.1 Auto Flow Control (AFC)

In AFC, the nRTS signal depends on the condition of the receiver, whereas the nCTS signals control the operation of transmitter. The UART's transmitter transfers the data to FIFO if nCTS signals are activated (in AFC, nCTS signals means that other UART's FIFO is ready to receive data). Before UART receives data, the nRTS signals must be activated if its receive FIFO has more than 2-byte as spare. The nRTS signals must be inactivated if its receive FIFO has less than 1-byte as spare (in AFC, the nRTS signals means that its own receive FIFO is ready to receive data).

6 Matrix-Keyboard

Implemented on: PM6,PM7,PM7A,QA8,ASA8,ND14,NRA8

It is possible to connect a matrix keyboard to the board. Matrix keyboard could be also an easy way to configure a pin as input and get a key down event when the pin toggles from high to low. The organization of this keyboard is very flexible. You can use a maximum of 16 (rows) * 16 (columns) + 32 (static keys). So you can connect 256+32 keys. All inputs must connect with resistors to 3.3 Volt. The driver polls the keyboard every 20 ms. In the case a key is pressed, the driver reads the scan code and saves the value. After additional 20 ms it checks the scan code. If the scan code is unchanged the scan code will be transformed with the information stored in the mapping table in a PS2 keyboard scan code. The routing of this keyboard code is the same as the one from a PS2 keyboard. The mapping table for converting a scan code in an PS2 keyboard code is stored in the registry.

The settings which influence the driver are stored under key:

```
[HKLM\HARDWARE\DEVICEMAP\KEYBD\MATRIX]
```

Key	Value	Comment
Type	dword:1	See Table 15: Martix Keyboard: Type registry value
RowReverse	dword:0	Reverse all bits of the row. Bit 0 to Bit 7, Bit 1 to Bit6
ColReverse	dword:0	Reverse all bits of the column. Bit 0 to Bit 7, Bit 1 to Bit6
ChangeRowCol	dword:0	Exchange the scan-value of row and column.
AutoKeyUp	dword:0	If a matrix key is pressed and the previous key is not released, this value sends the KEYUP message to the system.
OutputScanCode	dword:0	Set this value to 1 to output the scan-code of the currently pressed key as a debug message on the serial debug line.

Table 14: Matrix Keyboard: Registry settings

Type	Function
0	Matrix keyboard driver OFF
1	Matrix keyboard 16x16+32, 16 rows, 16 cols, 32 static keys, single key detection
3	Matrix keyboard 16x16, 16 rows, 16 cols, 0 static keys, single key detection
16	Matrix keyboard 16x16, 16 rows, 16 cols, 0 static keys, multiple key detection
17	Matrix keyboard 16x16+32, 16 rows, 16 cols, 32 static keys, multiple key detection

Table 15: Martix Keyboard: Type registry value

The organization of the columns is done under the following registry key:

```
[HKLM\HARDWARE\DEVICEMAP\KEYBD\MATRIX\COLS]
```

Key	Value	Comment
IOCol0	Dword:	Number of IO-Pin Pin (see Chapter 4 Digital I/O) you want use for column 0. See Table 23: Matrix Keyboard: Connector J1
…		
IOCol*n*	Dword:	Number of IO you want use for last column. See Table 23: Matrix Keyboard: Connector J1

Table 16: Matrix Keyboard: Cols registry values

Please do not add other registry values to this key, because amount of values is directly used for amount of columns.

The organization of the rows is done under the following registry key:

```
[HKLM\HARDWARE\DEVICEMAP\KEYBD\MATRIX\ROWS]
```

Key	Value	Comment
IORow0	Dword:	Number of IO-Pin (see Chapter 4 Digital I/O) you want use for row 0. See Table 23: Matrix Keyboard: Connector J1
…		
IORow*n*	Dword:	Number of IO you want use for last row. See Table 23: Matrix Keyboard: Connector J1

Table 17: Matrix Keyboard: Rows registry values

Please do not add other registry values to this key, because amount of values is directly used for amount of rows.

The organization of the static keys is done under the following registry key:

```
[HKLM\HARDWARE\DEVICEMAP\KEYBD\MATRIX\STATIC]
```

Key	Value	Comment
IOStaticKey0	Dword:	Number of IO you want use for static key 0. See Table 23: Matrix Keyboard: Connector J1
StaticKey0	Dword:	PS2 code for static key 0. See Table 20: Matrix Keyboard: PS2 Scan Codes
…		
IOStaticKey*n*	Dword:	Number of IO you want use for last static key. See Table 23: Matrix Keyboard: Connector J1
StaticKey*n*	Dword:	PS2 code for last static key. See Table 20: Matrix Keyboard: PS2 Scan Codes

Table 18: Matrix Keyboard: Static registry values

You have to add two registry values for each static key. Please do not add other registry values to this key, because amount of values is directly used for amount of static keys. It's also possible to use this driver without matrix keys. I.e. if you have only a small number of keys you can configure the driver like shown in *Example2*. This could be also a good alternative to using digital IO driver. Especially with .NET framework because you get changes to the IO in the way of key strokes and have not poll to driver.
Mapping of matrix keys to PS2 values are stored under

```
[HKLM\HARDWARE\DEVICEMAP\KEYBD\MATRIX\MAP]
```

Under `\MAP` you can make settings in the following form:

Key	Value
"1"	Dword:2
"2"	Dword:3
"3"	Dword:4
"4"	Dword:5

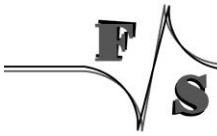
Table 19: Matrix Keyboard: Map registry value

The value under Key (string!) is the scan code from the matrix keyboard. The range of this value is from 1 to 127 and must be given in decimal format. The value must be in hexadecimal form. In the above example you send the PS2-Code 2 if you press the matrix key 1.

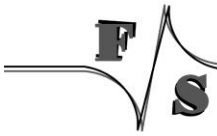
PS2 Scan Codes:

V-KEY	**PS2-Scan-Code**
0	// Scan Code 0x0
VK_ESCAPE	// Scan Code 0x1
'1'	// Scan Code 0x2
'2'	// Scan Code 0x3

V-KEY	PS2-Scan-Code
'3'	// Scan Code 0x4
'4'	// Scan Code 0x5
'5'	// Scan Code 0x6
'6'	// Scan Code 0x7
'7'	// Scan Code 0x8
'8'	// Scan Code 0x9
'9'	// Scan Code 0xA
'0'	// Scan Code 0xB
VK_HYPHEN	// Scan Code 0xC
VK_EQUAL	// Scan Code 0xD
VK_BACK	// Scan Code 0xE
VK_TAB	// Scan Code 0xF
'Q'	// Scan Code 0x10
'W'	// Scan Code 0x11
'E'	// Scan Code 0x12
'R'	// Scan Code 0x13
'T'	// Scan Code 0x14
'Y'	// Scan Code 0x15
'U'	// Scan Code 0x16
'I'	// Scan Code 0x17
'O'	// Scan Code 0x18
'P'	// Scan Code 0x19
VK_LBRACKET	// Scan Code 0x1A
VK_RBRACKET	// Scan Code 0x1B
VK_RETURN	// Scan Code 0x1C
VK_LCONTROL	// Scan Code 0x1D
'A'	// Scan Code 0x1E
'S'	// Scan Code 0x1F
'D'	// Scan Code 0x20
'F'	// Scan Code 0x21
'G'	// Scan Code 0x22
'H'	// Scan Code 0x23
'J'	// Scan Code 0x24
'K'	// Scan Code 0x25
'L'	// Scan Code 0x26
VK_SEMICOLON	// Scan Code 0x27
VK_APOSTROPH	// Scan Code 0x28
VK_BACKQUOTE	// Scan Code 0x29
VK_LSHIFT	// Scan Code 0x2A
VK_BACKSLASH	// Scan Code 0x2B
'Z'	// Scan Code 0x2C
'X'	// Scan Code 0x2D
'C'	// Scan Code 0x2E
'V'	// Scan Code 0x2F
'B'	// Scan Code 0x30
'N'	// Scan Code 0x31



V-KEY	PS2-Scan-Code
'M'	// Scan Code 0x32
VK_COMMA	// Scan Code 0x33
VK_PERIOD	// Scan Code 0x34
VK_SLASH	// Scan Code 0x35
VK_RSHIFT	// Scan Code 0x36
VK_MULTIPLY	// Scan Code 0x37
VK_LMENU	// Scan Code 0x38
VK_SPACE	// Scan Code 0x39
VK_CAPITAL	// Scan Code 0x3A
VK_F1	// Scan Code 0x3B
VK_F2	// Scan Code 0x3C
VK_F3	// Scan Code 0x3D
VK_F4	// Scan Code 0x3E
VK_F5	// Scan Code 0x3F
VK_F6	// Scan Code 0x40
VK_F7	// Scan Code 0x41
VK_F8	// Scan Code 0x42
VK_F9	// Scan Code 0x43
VK_F10	// Scan Code 0x44
VK_NUMLOCK	// Scan Code 0x45
VK_SCROLL	// Scan Code 0x46
VK_NUMPAD7	// Scan Code 0x47
VK_NUMPAD8	// Scan Code 0x48
VK_NUMPAD9	// Scan Code 0x49
VK_SUBTRACT	// Scan Code 0x4A
VK_NUMPAD4	// Scan Code 0x4B
VK_NUMPAD5	// Scan Code 0x4C
VK_NUMPAD6	// Scan Code 0x4D
VK_ADD	// Scan Code 0x4E
VK_NUMPAD1	// Scan Code 0x4F
VK_NUMPAD2	// Scan Code 0x50
VK_NUMPAD3	// Scan Code 0x51
VK_NUMPAD0	// Scan Code 0x52
VK_DECIMAL	// Scan Code 0x53
VK_SNAPSHOT	// Scan Code 0x54
VK_F11	// Scan Code 0x57
VK_F12	// Scan Code 0x58
VK_LWIN	// Scan Code 0x5B
VK_RWIN	// Scan Code 0x5C
VK_APPS	// Scan Code 0x5D
VK_HELP	// Scan Code 0x63
VK_F13	// Scan Code 0x64
VK_F14	// Scan Code 0x65
VK_F15	// Scan Code 0x66
VK_F16	// Scan Code 0x67
VK_F17	// Scan Code 0x68
VK_F18	// Scan Code 0x69
VK_F19	// Scan Code 0x6A



V-KEY	PS2-Scan-Code
VK_F20	// Scan Code 0x6B
VK_F21	// Scan Code 0x6C
VK_F22	// Scan Code 0x6D
VK_F23	// Scan Code 0x6E
VK_F24	// Scan Code 0x76
VK_DIVIDE	// Scan Code 0xE035
VK_SNAPSHOT	// Scan Code 0xE037
VK_RMENU	// Scan Code 0xE038
VK_HOME	// Scan Code 0xE047
VK_UP	// Scan Code 0xE048
VK_PRIOR	// Scan Code 0xE049
VK_LEFT	// Scan Code 0xE04B
VK_RIGHT	// Scan Code 0xE04D
VK_END	// Scan Code 0xE04F
VK_DOWN	// Scan Code 0xE050
VK_NEXT	// Scan Code 0xE051
VK_INSERT	// Scan Code 0xE052
VK_DELETE	// Scan Code 0xE053
VK_LWIN	// Scan Code 0xE05B
VK_RWIN	// Scan Code 0xE05C
VK_APPS	// Scan Code 0xE05D

Table 20: Matrix Keyboard: PS2 Scan Codes

Scan codes matrix 8x8:

	C0	C1	C2	C3
R0	0x01	0x02	0x03	0x04
R1	0x11	0x12	0x13	0x14
R2	0x21	0x22	0x23	0x24
R3	0x31	0x32	0x33	0x34
R4	0x41	0x42	0x43	0x44
R5	0x51	0x52	0x53	0x54
R6	0x61	0x62	0x63	0x64
R7	0x71	0x72	0x73	0x74

Table 21: Matrix Keyboard: Scan Codes matrix 8x8 C0 – C3

	C4	C5	C6	C7
R0	0x05	0x06	0x07	0x08
R1	0x15	0x16	0x17	0x18
R2	0x25	0x26	0x27	0x28

R3	0x35	0x36	0x37	0x38
R4	0x45	0x46	0x47	0x48
R5	0x55	0x56	0x57	0x58
R6	0x65	0x66	0x67	0x68
R7	0x75	0x76	0x77	0x78

Table 22: Matrix Keyboard: Scan Codes matrix 8x8 C4 – C7

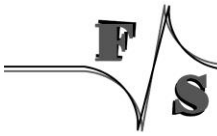
Note:

This is an example configuration. The amount of columns and rows is not fixed.

PicoMOD Connector J1:

Pin	IO	Default Interface	Starter-Kit Interface
1	64	I/O-Pin 64	SPI CS
2	65	I/O-Pin 65	SPI CLK
3	66	I/O-Pin 66	SPI MISO
4	67	I/O-Pin 67	SPI MOSI
17	1	I/O-Pin 1	COM2 TXD
18	0	I/O-Pin 0	COM 2 RXD
19	3	I/O-Pin 3	COM2 RTS
20	2	I/O-Pin 2	COM2 CTS
21	5	COM1 TXD	COM1 TXD
22	4	COM1 RXD	COM1 RXD
23	7	I/O-Pin 7	COM3 TXD
24	6	I/O-Pin 6	COM3 RXD
29	9	I/O-Pin 9	GPIO5
30	8	I/O-Pin 8	USB Host Power
31	11	I/O-Pin 11	I2C SDA
32	10	I/O-Pin 10	USB Device Detect
34	12	I/O-Pin 12	I2C SCL
41	14	I/O-Pin 14	GPIO1
42	13	I/O-Pin 13	GPIO0
43	16	I/O-Pin 16	GPIO3

Pin	IO	Default Interface	Starter-Kit Interface
44	15	I/O-Pin 15	GPIO2
45	18	I/O-Pin 18	SD-CARD CLK
46	17	I/O-Pin 17	GPIO4
47	20	I/O-Pin 20	SD-CARD DAT0
48	19	I/O-Pin 19	SD-CARD CMD
49	22	I/O-Pin 22	SD-CARD DAT2
50	21	I/O-Pin 21	SD-CARD DAT1
51	24	I/O-Pin 24	SD-CARD Detect
52	23	I/O-Pin 23	SD-CARD DAT3
53	26	I/O-Pin 26	SD-CARD Write Protect
54	25	I/O-Pin 25	SD-CARD Power Enable
55	28	I/O-Pin 28	LCD DEN
56	27	I/O-Pin 27	LCD Enable
57	30	I/O-Pin 30	VCFL On
58	29	I/O-Pin 29	VLCD On
60	31	I/O-Pin 31	LCD VEEK
61	32	I/O-Pin 32	LCD
63	34	I/O-Pin 34	LCD
64	33	I/O-Pin 33	LCD
65	36	I/O-Pin 36	LCD
66	35	I/O-Pin 35	LCD
67	38	I/O-Pin 38	LCD
68	37	I/O-Pin 37	LCD
69	40	I/O-Pin 40	LCD
70	39	I/O-Pin 39	LCD
71	42	I/O-Pin 42	LCD
72	41	I/O-Pin 41	LCD
73	44	I/O-Pin 44	LCD
74	43	I/O-Pin 43	LCD
75	46	I/O-Pin 46	LCD
76	45	I/O-Pin 45	LCD



Pin	IO	Default Interface	Starter-Kit Interface
77	48	I/O-Pin 48	LCD
78	47	I/O-Pin 47	LCD
79	50	I/O-Pin 50	LCD
80	49	I/O-Pin 49	LCD
81	52	I/O-Pin 52	LCD
82	51	I/O-Pin 51	LCD
86	53	I/O-Pin 53	LCD
87	70	I/O-Pin 70	CF /CD
88	71	I/O-Pin 71	CF /IRQ
90	72	I/O-Pin 72	CF INPACK
93	73	I/O-Pin 73	CF REG
98	74	I/O-Pin 74	CF RESET
126	75	I/O-Pin 75	CF Card Power Enable

Table 23: Matrix Keyboard: Connector J1

Please note, that you must be very careful with your configuration. If you want to use i.e. IO 1 (pin 17) for keyboard, you must disable serial driver for this port.

Configuration Example:

B. Create matrix keyboard with matrix 2x2 and no static keys. We use pins at connector J1 of PicoMOD which are routed to starter kit connector J5.

```
HKLM\hardware\devicemap\keybd\matrix]
    "Type"=dword:10 ; multi
    "OutputSCanCode"=dword:1
    "Debug"=dword:4

[HKLM\hardware\devicemap\keybd\matrix\Cols]
    "IOCol0"=dword:E      ; IO 14 (pin 41)
    "IOCol1"=dword:F      ; IO 15 (pin 44)

[HKLM\hardware\devicemap\keybd\matrix\Rows]
    "IORow0"=dword:10      ; IO 16 (pin 43)
    "IORow1"=dword:11      ; IO 17 (pin 46)


[HKLM\hardware\devicemap\keybd\matrix\map]
    "1"=dword:1E      ; r0,c0 -> 'A'
    "2"=dword:30      ; r0,c1 -> 'B'
    "17"=dword:2E     ; r1,c0 -> 'C'
    "18"=dword:20     ; r1,c1 -> 'D'
```

Listing 21: Matrix Keyboard: Example 1

Create keyboard with two static keys and no matrix. We use pins at connector of PicoMOD which are routed to starter kit connector J5.

```
[HKLM\hardware\devicemap\keybd\matrix]
    "Type"=dword:11 ; multi with static keys
    "OutputSCanCode"=dword:1
    "Debug"=dword:4

[HKLM\hardware\devicemap\keybd\matrix\Static]
    "IOStaticKey0"=dword:E    ; IO 14 (pin 41)
    "StaticKey0"=dword:1E     ; PS2 code 'A'
    "IOSTaticKey1"=dword:F    ; IO 15 (pin 44)
    "StaticKey1"=dword:30     ; PS2 code 'B'

; remove this key or delete all values
[HKLM\hardware\devicemap\keybd\matrix\Cols]

; remove this key or delete all values
[HKLM\hardware\devicemap\keybd\matrix\Rows]

; remove this key or delete all values
[HKLM\hardware\devicemap\keybd\matrix\map]
```

Listing 22: Matrix Keyboard: Example 2

7 Touchpanel Driver

Implemented on: PM3,PM4,PM6,PM7,PM7A,ASA8,ND14,NRA8

[HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\TOUCH]

Possible settings for PicoMOD:

Key	Value	Comment
CalibrationData	"0,0,0,0,0,"	Set this value to the given string to avoid the calibration screen after restart.
TouchSamples	Dword:3..20	With this value you can adjust the amount of samples that are used to create the position value. As more samples as longer the time you have to press on the same place. Default: 7
SamplePeriodLowHns	Dword	Sample period settings in 100 ns units for low sample periods. Default: 20ms (200000)
SamplePeriodHighHns	Dword	Sample period settings in 100 ns units for high sample periods. Default: 10ms (100000)
DeltaXCoordTolerance	Dword:0..0x3FF	This value is used by the touch sample filter routine to accept and reject points. Increasing the tolerance will generally allow faster pen movements to be detected. This will also increase noise and tend to cause erratic touch behaviour. Default: 20
DeltaYCoordTolerance	Dword:0..0x3FF	This value is used by the touch sample filter routine to accept and reject points. Increasing the tolerance will generally allow faster pen movements to be detected. This will also increase noise and tend to cause erratic touch behavior. Default: 16
AdcReadHoldoffHns	Dword:	Amount of time (in 100 ns units) to wait after biasing the plates before starting an ADC read to determine an X or Y coordinate. This allows the voltage at the ADC input to settle. More time may be needed if large capacitors or other filtering devices are used. Wait times that are too small will result in poor touch performance (unstable pen position). Wait times that are too long will cause poor system performance and may reduce the touch sampling frequency. Default: 2000 = 200us

Key	Value	Comment
PenDownHoldoffHns	Dword:	Amount of time (in 100 ns units) to wait before reading the state of the plates when determining if the pen is up or down. Too small of a wait will make it impossible for the driver to tell the true state of the plates as the inputs will not have enough time to settle. This can cause the pen to get stuck in the down position. As with the AdcReadHoldOffDelay, this value may need to be increased if large capacitors or other hardware filtering is present. Too high a value will cause poor system performance and may reduce the touch sampling frequency. Default: 50000 = 5ms
MinMove	Dword:1..0x3FF	Minimum move (A/D resolution) before MouseMove is signalled. MinMove: 5
MaxMove	Dword:1..0x3FF	Maximum move (A/D resolution) which is recognized and send to application layer. MaxMove: 50
AutoCalib	Dword:0..10000	Time in ms before event 20 is signalled to application layer when touch is pressed. Can be used for automatic touch calibration. AutoCalib=0 disables this function. Default: 0
UseStandardDeviation	Dword:0…100	Value in percentage of the touch controler resolution. If standard deviation of the "TouchSamples" values exceeds this value the data colletion is marked as corrupt. In contrast to the "DeltaX/YCoordTolerance" we will include several runaway values but exclude hight statistical spreaded values. Standard deviation is checked before "DeltaX/YCoordTolerance". Default: 0, means standard deviation is not used. Supported only on PicoMOD6.
CheckDownWhileSample	Dword:0|1	1 = Marking data collection as corrupt when "touch up" is detected while sampling "TouchSamples". Value supported only on PicoMOD6 Default: 0
ADCConvFreq	Dword:	Default: 1000000

Key	Value	Comment
DiscardStartEvents	Dword: 0…4	Touchpanel devices may be inaccurate at the beginning and the end of a measurement time slot. If this inaccuracy is in the order of the time for create a positions value the evaluation of the samples per position may also deliver inaccurate results. To prevent this behavior use “DiscardStartEvents” to discard the starting position value(s). Default: 0 Available at PicoMOD6, touchpanel driver version 1.5 and higher.
DiscardTailEvents	Dword: 0…8	Touchpanel devices may be inaccurate at the beginning and the end of a measurement time slot. If this inaccuracy is in the order of the time for create a positions value the evaluation of the samples per position may also deliver inaccurate results. To prevent this behavior use “DiscardTailEvents” to discard the tailing positions value(s). Default: 0 Available at PicoMOD6, touchpanel driver version 1.5 and higher.
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 24: Touch: Registry settings

```
[HKEY_LOCAL_MACHINE\SYSTEM\CALIBRUI]
```

Possible settings:

Key	Value	Comment
NoKeyboard	Dword:1	This parameter tells touch panel calibration to not wait for a keystroke at the end of calibration.

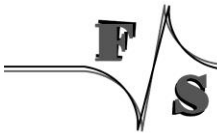
Table 25: Touch: Calibration

```
[HKEY_LOCAL_MACHINE\DRIVERS\BUILTIN\TOUCH]
```

Possible settings:

Key	Value	Comment
Priority256 HighPriority256	Dword:10 9	Set this value to adjust the priority of the touch panel driver.

Table 26: Touch: Adjusting the priority



Meanwhile, most kernel images include additional driver for capacitive touch controllers, which can be connected to PicoCOM via I2C. These drivers are deactivated by default.

7.1 MXT224 Touch Driver

To activate the MXT touch driver there is a corresponding `ndcucfg` script available. If you are connected to the board via telnet you just need to type the following command:

```
ndcucfg –B\Windows\fs_touch_mxt224.txt
```

This script sets all required registry settings. Here is a list of the meaning of these values located at:

```
[HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\TOUCH]
```

Key	Value Type	Default Value	Comment
ChangeIO	DWORD	20	Touch interrupt IO-Pin number.
ResetIO	DWORD	-1	IO-Pin used to trigger controller reset during initialization. A value of -1 disables this functionality.
I2CDevAddr	DWORD	0x96	I2C Device address of the touch controller.
InvertX	DWORD 0/1	0	Invert all X-coordinates.
InvertY	DWORD 0/1	0	Invert all Y-coordinates.
SWCalibration	DWORD	0	Enable SW touch calibration which is only required if the touch area is different to the display size.

Table 27: Capactive touch driver registry settings.

Note:

A touch calibration is not required as the touch controller automatically scales the touch sample to the screen size.

7.2 EDT Touch Driver

If you need to use the EDT touch driver there also is a corresponding `ndcucfg` script available. After you are connected to the board via telnet you just need to call the following command:

```
ndcucfg -B\Windows\fs_touch_edt.txt
```

This script sets all required registry settings.
Here is a list of the meaning of these values located at:

```
[HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\TOUCH]
```

Key	Value Type	Default Value	Comment
ChangeIO	DWORD	20	Touch interrupt IO-Pin number.
ResetIO	DWORD	21	IO-Pin used to trigger controller reset during initialization. A value of -1 disables this functionality.
I2CDevAddr	DWORD	0x70	I2C Device address of the touch controller.

Table 28: Capactive touch driver registry settings.

After you activated this touch driver you should call the `touch calibrate` command, to use the touch panel correctly. Don't forget to save the registry settings with the `reg save` command.

8 USB Host Driver

Implemented on: 3,4,6,7,QA8

PicoMOD/QBliss supports USB Host and USB Device. If customer doesn't need USB Device, USB Device can be configured for USB Host.

The registry key for the driver is:

```
[HKLM\Drivers\Builtin\OHCI]
```

Use the following parameters to configure the driver:

Key	Value	Comment
Prefix	"HCD"	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Dll		Name of the DLL with the driver.
Order	Dword:1	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
PortCount	Dword:1	Possible values are 1 or 2. When you set this value to 2 , you have to disable USB Device driver.
PhysicalPageSize	Dword:	Size of physical memory used for USB buffers. Increase this value if you use many devices and one of the devices will not recognized. I.e. if you connect four devices increase to 0x40000. Default: 0x10000

Table 29: USB Host: Registry settings

Use the following key to configure some important Windows CE USB host controller settings:

```
[HKLM\Drivers\Drivers\USB\LoadClients]
```

Use the following parameters to configure the driver:

Key	Value	Comment
DoNotPromptUser	Dword	Allows to disable the USB driver dialog. Default: 0

Table 30: Windows CE USB Host: Controller Registry settings

Note:

When using PortCount = 2 (configuration for 2 USB host, no USB device) and using the StarterKit you need to modify your hardware. Please contact the hardware department of F&S for detailed information.
You also need to disable the USB Function driver. You can do that by setting the ‘Flags’ value in [HKEY_LOCAL_MACHINE\Drivers\Builtin\USBFN] to ‘4’.

9 USB Device 2.0 Driver

Implemented on: PM3,PM4,PM6,PM7,PM7A,QA8,ASA8,NRA8

armStone/PicoMOD/QBliss/nanoRISC supports USB Host and USB Device. If customer doesn't need USB Device, USB Device can be configured for USB Host.

The registry key for the USB device driver is:

```
[HKLM\Drivers\Builtin\USBFN]
```

Use the following parameters to configure the driver:

Key	Value	Comment
Prefix	"UFN"	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Dll		Name of the DLL with the driver.
Order	Dword:32	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
Flags	Dword:<0|4>	Set this value to 4 to disable USB device driver.
Speed	Dword: 0, 1, 3	0: High speed (USB 2.0) 1: Full speed (USB 2.0) 3: Full speed (USB 1.1) Default: 0 Available at PicoMOD6/7, USBFN driver version 1.3 and higher.

Table 31: USB Device: Registry settings

The USB device interface can be configured for the following functionality:

- Serial
- Mass Storage
- RNDIS

The selection of the function is done under following registry key:

```
[HKEY_LOCAL_MACHINE\Drivers\USB\FunctionDrivers]
```

Use the following parameters to configure the driver:

Key	Value	Comment
DefaultClientDriver	“USBSER_Class “ “Mass_Storage_Class” “RNDIS”	Select function class of USB device interface.

Table 32: USB Device: Registry settings

10 LCD Driver

Implemented on: PM3,PM4,PM6,PM7,QA8

PicoMOD/QBliss has a very flexible and powerful interface for LCD and EL displays. The driver is fully configurable over the Window CE registry. The user has the possibility to adjust the driver to a new display by himself.

The registry key for the PicoMOD3 and PicoMOD4 driver is:

```
[HKLM\Drivers\Display\SAMSUNG]
```

The registry key for the PicoMOD6, PicoMOD7 and QBlissA8 driver is:

```
[HKLM\Drivers\Display\LCD]
```

Use the following parameters to configure the driver:

Key	Value	Meaning
Mode	Dword:	Number of the predefined configuration or new user configuration.
UseBootMem	Dword:	Use memory provided by bootloader for frame buffer
Verbose	Dword:	Enables additional output at serial debug port.

Table 33: LCD: Registry settings

With parameter Mode you have the possibility to use one of the fixed configurations stored in the kernel or to define a new configuration in registry. Values between 0 and 99 are reserved for fixed configurations. For your own configuration you have to use values between 100 and 199.

The following configurations are predefined in kernel:

Mode	Name	XxY	Type
0	TFT, 60 Hz, 16Bpp	640x480	Active
1	TFT 16Bpp	800x600	Active
2	TFT 16Bpp	1024x768	Active
3			
4	TFT 16Bpp	320x240	Active

Table 34: LCD: Modes

For configurations with Mode higher than 99 you have to create a new sub-key with the Name ModeXXX. Detailed information how to perform these settings and a series of display drivers adjustments described in the documentation “NetDCU Display”.

For adjust PWM frequency (ASA8, ND14,NRA8,PM6,PM7,PM7A and QA8 only) you can set:

```
[HKLM\Drivers\Display\LCD\ModeXXX\ContrastFreq=DWORD:<clock in HZ>]
```

For adjust LCD port drive strength (ASA8, PM6, PM7 and QA8 only) you can set:

```
[HKLM\Drivers\Display\LCD\ModeXXX\LCDPortDriveStrength=DWORD:<value>
]
```

Following values can set:

Value	LCD Port Drive Strength
0	2 mA
1	4 mA
2	7 mA
3 (default)	9 mA

Table 35: LCD: Port Drive Strength

11 LCD Driver for FSS5PV210

Implemented on: PM7A, ASA8, NRA8, ND14

armStoneA8/NanoRISC-A8/NetDCU14/PicoMOD7A has a very flexible and powerful interface for LCD TFT displays and DVI-D (HDMI) monitors . The driver is fully configurable over the Window CE/Compact 7 registry. Some display types are already predefined, so that a simple choice from a list is all that is required. If the display is not already predefined, the user has the possibility to adjust the driver to a new display by himself by setting a few parameters or download a new display-driver

The display driver supports the following features:

- Interface for digital LCD TFT (analog RGB or LVDS)
- Interface for DVI-D (HDMI) or analog VGA
- Adjustable frame buffer depth 16/24/32 BPP
- Adjustable output depth 16/18/24 BPP
- Overlays
- DirectDraw
- OpenGL ES 1.1 and 2.0
- MultiMonitor support (same/different resolutions)

The registry key for the driver is:

```
[HKLM\Drivers\Display\LCD]
```

Use the following parameters to configure the driver:

Key	Value	Meaning
Mode	Dword:	Number of the predefined configuration or new user configuration.
UseBootMem	Dword:	Use memory provided by boot loader for frame buffer.
VidMemCache	Dword:	Use cached video memory for display frame buffer. Default: 0
AccelLevel	Dword:	See control utility FS 2D acceleration.
Win0QOS Win1QOS Win2QOS Win3QOS Win4QOS	Dword:	Quality of servcie control for memory access.
Verbose	Dword:	Enables additional output at serial debug port.

Table 36: LCD - Registry settings

With parameter Mode you have the possibility to use one of the fixed configurations stored in the kernel or to define a new configuration in registry. Values between 0 and 99 are reserved for fixed configurations. For your own configuration you have to use values between 100 and 199.

The following configurations are predefined in kernel:

Mode	Name	XxY	BPP	VCLK
0	VGA standard display	640x480	16	25MHz
1	SVGA standard display	800x600	16	38MHz
2	XGA standard display	1024x768	16	65MHz
3				
4	QVGA standard display	320x240	16	6MHz
5	XGA standard display 56MHz	1024x78	16	56MHz
6	EDT ET070080	800x480	16	33MHz
7	EDT ET035080	320x240	16	10MHz
8	Hitachi TX09	240x320	16	6MHz
9	EDT ET043080	480x272	16	9MHz
10	NEC NL6448BC	640x480	16	25MHz
11	Sharp LQ104	640x480	16	25MHz
12	AOU G104SN03	640x480	16	25MHz
13	EDT ET057090DH	640x480	16	25MHz
14	AOU G104SN02	800x600	16	38MHz
15	Hitachi TX18D35	800x480	16	33MHz
16	WXGA standard display	1280x800	16	90MHz
17	WVGA standard display	1024x600	16	51MHz
18	CHIMEI G070Y	800x480	16	auto

Table 37: LCD - Modes

If you select one of the above configurations, automatically a sub-key with name Mode0 or Mode1 or ModeX is created. It is possible to adjust the predefined configuration by writing special values to this sub-key. For configurations with Mode higher than 99 you have to create a new sub-key with the Name ModeXXX. Detailed information how to perform these settings and a series of display driver's adjustments described in the documentation "NetDCU Display".

For adjust PWM frequency you can set:

```
[HKLM\Drivers\Display\LCD\ModeXXX\ContrastFreq=DWORD:<clock in HZ>]
```

For adjust LCD port drive strength you can set:

```
[HKLM\Drivers\Display\LCD\ModeXXX\LCDPortDriveStrength=DWORD:<val>]
```

Following values can set:

Value	LCD Port Drive Strength
0	2 mA
1	4 mA
2	7 mA
3 (default)	9 mA

Table 38: LCD - Port Drive Strength

11.1 Default Display Mode

	Digital RGB	LVDS
NetDCU14	6 = 800x480 (ET070080)	18 = 800x480 (CHIMEI G070Y)
PicoMOD7A	6 = 800x480 (ET070080)	18 = 800x480 (CHIMEI G070Y)
armStoneA8	--	18 = 800x480 (CHIMEI G070Y)
nanoRISC-A8	6 = 800x480 (ET070080)	--

Table 39: LCD - Default Display Mode

11.2 Default LCD Output Width

Output width of LCD controller is automatically adjusted depending on the board.

	Digital RGB	LVDS
NetDCU14	LCD_CONFIG_OUT24BIT	LCD_CONFIG_OUT18BIT

	Digital RGB	LVDS
PicoMOD7A	LCD_CONFIG_OUT18BIT	L1: LCD_CONFIG_OUT18BIT L2: LCD_CONFIG_OUT24BIT
armStoneA8	--	LCD_CONFIG_OUT18BIT
nanoRISC-A8	LCD_CONFIG_OUT24BIT	--

The configuration can be changed with registry parameter CONFIG.

Note:

Don't configure LCD output width different to the above values for LVDS !

11.3 Display Mode Registry Settings

The following settings can be made to define a display mode. Settings are placed in the registry under key

```
[HKLM\Drivers\Display\LCD\ModeX]

[HKLM\Drivers\Display\LCD\HDMI\ModeX]
```

Key	Type	Meaning
"name"	sz:	Name of the driver as a text string. Only for information purposes.
Type	Dword:	See „*Registry Value Type*“
Config	Dword:	See „*Registry Value Config*“
Columns	Dword:	Amount of visible pixels in X-direction.
PPL	Dword:	Amount of clocks in X-direction before the HSYNC signal. This value is optional and normally the same as Columns.
BLW	Dword:	Beginning-of-line-wait: Value (0-255) specifies the number of VCLK periods between the falling edge of HSYNC and the start of active data.
HSW	Dword:	Horiz-sync-pulse-width: Value (0-255) specifies the number of pixel clock periods to pulse the line clock at the end of each line.
ELW:	Dword:	End-of-line-wait: Value (0-255) specifies the number of VCLK periods between the end of active data and the rising edge of HSYNC.
Rows	Dword:	Amount of visible pixels in Y-direction.
LPP	Dword:	Lines per panel: This is an optional parameter and in most cases it is the same as Rows.

Key	Type	Meaning
BFW	Dword:	Beginning-of-frame wait: Value (0–255) specifies the number of inactive lines at the start of a frame, after vertical synchronization period.
VSW	Dword:	Vertical sync pulse width: Value (0–255) specifies the number of line clock periods to pulse the FRP pin at the end of each frame after the end-of-frame wait (EFW) period elapses. Frame clock used as VSYNC signal in active mode.
EFW	Dword:	End-of-frame line clock wait count: Value (0–255) specifies the number of inactive lines at the end of a frame, before vertical synchronization period.
Width	Dword:	Physical width of the display
Height	Dword:	Physical height of the display
Bpp	Dword:	Bits per Pixel. The number of bits that represents one pixel in display memory.
ContrastEnable	Dword:	Switch on/off contrast voltage generation.
ContrastValue	Dword:	Initial value for contrast voltage.
LCDClk	Dword:	LCD pixel clock in MHz
EnableCursor	Dword:	1: show cursor on screen.
Rotate	Dword:	0, 90, 180, 270
Msignal	Dword:	0: output low 1: output high 2: toggle Default: 2
HVSync	Dword:	0: output low 1: output high 2: toggle Default: 2
LCDPortDriveStrength	Dword:	See Table 35: LCD: Port Drive Strength
PONLcdPow	Dword:	Delay in ms before LCD power is switched on.
PONLcdEna	Dword:	Delay in ms before display enable signal is switched on.
PONLcdBufEna	Dword:	Delay in ms before buffers are switched on.
PONVeeOn	Dword:	Delay in ms before Vee is switched on.
PONCflPow	Dword:	Delay in ms before CFL is switched on.

11.3.1 Registry Value Type

Value	Meaning
0x0000	Default
0x0002	TFT-Display
0x0004	Colour-Display

0x0100	Enable contrast voltage VEE
0x0200	Output more information to serial debug line

Table 40: LCD - Display Driver Registry Value Type

11.3.2 Registry Value Config

Symb. Name	Value	Meaning
LCD_USE_PON_REGS	0x00010000	Default case. Same result as if no bit is set.
LCD_USE_PON_MODE2	0x00020000	VLCD->VCLK->Vee->DEN->CFL
LCD_USE_PON_MODE3	0x00040000	Vee->all OFF->VLCD->VBUF->DEN->CFL
LCD_USE_PON_MODE4	0x00080000	
LCD_USE_PON_CUSTOM	0x000F0000	PON (PowerOn) sequencing can be specified in detail with registry values PONLcdPow, PONLcdEna, PONLcdBufEna, PONVeeOn and PONCflPow.
LCD_VSP	0x00100000	Vertical sync polarity: active low
LCD_HSP	0x00200000	Horizontal sync polarity: active low
LCD_CLKP	0x00400000	Clock polarity: active low
LCD_OEP	0x00800000	Output enable polarity: active low
LCD_OUTDEF	0x00000000	Use default output width. See *Table 39: LCD - Default Display Mode*
LCD_OUT16BIT	0x01000000	RGB565
LCD_OU18BIT	0x02000000	RGB666
LCD_OUT24BIT	0x03000000	RGB888
LCD_DEMODE	0x10000000	Use signal DE/M for timing. Drive HSync and VSync low.

Table 41: LCD - Display Driver Registry Value Config

11.4 Multiple Monitor Feature

This feature allows connection of one digital RGB display or LVDS display (PANEL) and one analog RGB display or HDMI monitor (HDMI) to the board. You can use these multiple screens as one large combined screen to create more screen space for applications. That means you can show different content on the two screens. This extra space is useful whenever you need to maximize your on-screen workspace.

The support for multiple screens does not affect the performance of applications when those applications run in a single screen environment. In other words, when an application runs on a system with a single screen, no additional overhead is present in the high-performance graphics operations code. On a multiple screen system, however, performance is slightly affected if an application runs only on one of the graphics devices. Also, performance can be greatly affected if an application spans multiple screens, especially for graphics-intensive operations.

11.4.1 Registry Settings

To specify the number of screens present in a multiple screen system, set the `HKLM\SYSTEM\GDI\MONITORS\TOTAL MONITORS` registry entry equal to the number of screens. You should only set this registry entry to a value between one and four because Windows Embedded CE supports a maximum of four screens. The default value is one. The following code example shows how to specify that the system has two screens.

```
[HKLM\SYSTEM\GDI\MONITORS]
```

Required Settings:

Key	Type	Meaning
Total Monitors	Dword:1	Amount of monitors connected to the board. Possible values 1 or 2. Default: 1

Specify settings for digital panel under the following key:

```
[HKLM\Drivers\Display\LCD]
```

are taken.

Specify settings for analog CRT under the following key:

```
[HKLM\Drivers\Display\LCD\HDMI]
```

If you don’t create the key and don’t create value MODE default mode 0 is used.

11.4.2 Default Modes HDMI Interface

Mode	Resolution
0	HDMI_720x480_RGB565
1	HDMI_720x576_RGB565
2	HDMI_1280x720_RGB565
3	HDMI_1920x1080_RGB565

Example:

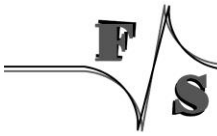
Following registry values for digital panel with VGA resolution and VGA/HDMI with 480p (720x480) resolution.

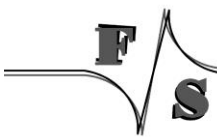
```
reg open \SYSTEM\GDI\MONITORS
reg set value "Total Monitors" dword 2
reg open \Drivers\Display\LCD
reg set val Mode dword 0
reg create key HDMI
reg set value Mode dword 0
reg save
```

11.4.3 Application Development

The following table shows the functions that Windows Embedded CE provides for working with multiple screens.

Function	Description
EnumDisplayMonitors	Enumerates screens that intersect a region formed by the intersection of a specified clipping rectangle and the visible region of a specified device context.
GetMonitorInfo	Retrieves information about a screen.
MonitorEnumProc	An application-defined callback function that is called by the EnumDisplayMonitors function.
MonitorFromPoint	Retrieves a handle to the screen that contains a specified point.
MonitorFromRect	Retrieves a handle to the screen that has the largest area of intersection with a specified rectangle.
MonitorFromWindow	Retrieves a handle to the screen that has the largest area of intersection with the bounding rectangle of a specified window.





12 Soft-Keyboard

Sometimes it is useful to have a virtual keyboard on your display which can be controlled by using the touch panel.

To do this you must copy the file SOFTKB.DLL to the folder FFSDISK. The configuration program NDCUCFG (version 012 and higher) has a command to show the input panel on the screen (sip on).

Installation of the driver softkb.dll is done by setting some registry values under the following registry key:

```
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\SIP]
```

Required settings:

Key	Value	Comment
Prefix	“SIP“	This required value specifies the driver’s device file name prefix. It is a three-character identifier, such as COM.
Dll	“SOFTKB.DLL“	name of the driver file
Order	Dword:50	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
Index	Dword:0	This value specifies the device index, a value from 0 through 9.

Table 42: Softkeybd: Registrysettings

13 CAN

The CAN interface driver is described in a separated documentation, that can be download from http://www.fs-net.de.

14 I2C Driver

Implemented on: PM3,PM4,PM6,PM7,PM7A,QA8,ASA8,ND14,NRA8

armStone/PicoMOD/QBliss/nanoRISC supports GPIO I2C driver.

The registry key for the driver is:

```
[HKLM\Drivers\Drivers\Builtin\I2C1]
```

Use the following parameters to configure the driver:

Key	Value	Comment
Prefix	"I2C"	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Dll	"pmX_ni2c.dll", "fs_ni2c.dll"	Name of the DLL with the driver
Order	Dword:0x101	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
ClockFreq	Dword:	Clock speed in Hz
Priority256	Dword:	
PinSDA	Dword:	Pin number (see *Digital I/O*) of SDA signal
PinSCL	Dword:	Pin number (see *Digital I/O*) of SCL signal
IntPullUp	Dword:	Enable Internal pull-up for SDA/SCL.
DrvStrength	Dword:	Set drive strength control for SDA/SCL.

Table 43: I2C: Registry settings

The full documentation of the driver can be found in document "WinCE-I2C+NI2C_eng.pdf". For a first test, you can use the dialog based tool FS_ScanI2C.exe. This program lists the available I2C ports and scans the port for devices.

15 Native I2C Driver

Implemented on: PM3,PM4,PM6,PM7,PM7A,QA8,ASA8,NRA8

armStone/PicoMOD/QBliss/nanoRISC supports native I2C driver.

The registry key for the driver is:

```
[HKLM\Drivers\Drivers\Builtin\I2C1]
```

Use the following parameters to configure the driver:

Key	Value	Comment
Prefix	"I2C"	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Dll	"pmX_ni2c.dll", "fs_ni2c.dll"	Name of the DLL with the driver
Order	Dword:0x101	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.

Table 44: Native I2C: Registry settings

The full documentation of the driver can be found in document "WinCE-I2C+NI2C_eng.pdf". For a first test, you can use the dialog based tool FS_ScanI2C.exe. This program lists the available I2C ports and scans the port for devices.

QBlissA8:
At QBlissA8 we have two NI2C drivers and one I2C driver. The usage is as follows:

Connector	driver
J1: camera interface	I2C1: (native I2C driver)
X1: SMB_DAT, SMB_CLK	I2C1: (native I2C driver)
X1: I2DAT, I2CLK	I2C2: (native I2C driver)
X1: HDMI_CTRL_DAT, HDMI_CTRL_CLK	I2C9: (I2C driver)

Table 45: QBlissA8 I2C driver usage

armStoneA8:
At armStoneA8 we have two NI2C drivers and two I2C drivers.
The usage is as follows:

Connector	driver
Feature connector	I2C1: (native I2C driver)
Cap. Touch connector	I2C2: (native I2C driver)
Audio	I2C8: (I2C driver)
HDMI_CTRL_DAT, HDMI_CTRL_CLK	I2C9: (I2C driver)

Table 46: armStoneA8 I2C driver usage

PicoMOD7A:
At PicoMOD7A we have one NI2C drivers and two I2C drivers.
The usage is as follows:

Connector	driver
Main connector	I2C1: (native I2C driver)
Audio	I2C8: (I2C driver), on-board
HDMI_CTRL_DAT, HDMI_CTRL_CLK	I2C9: (I2C driver), on-board

Table 47: PicoMOD7A I2C driver usage

nanoRISC-A8:
At nanoRISC-A8 we have one NI2C driver and one I2C driver.
The usage is as follows:

Connector	driver
I2C0_SCL, I2C0_SDA	I2C1: (native I2C driver)
HDMI_CTRL_DAT, HDMI_CTRL_CLK	I2C9: (I2C driver)

Table 48: nanoRISC-A8 I2C driver usage

16 PWM Driver

Implemented on: ASA8, ND14

armStoneA8 has 4 PWM outputs. First is controlled by the display driver (contrast voltage), second to fourth can be controlled by the PWM driver. Usage of fourth PWM is limited to the case when resistive touch driver is disabled.
NetDCU14 has 2 PWM outputs. One is controlled by the display driver (contrast voltage) and one can be controlled by the PWM driver.

Installation of the driver is done by setting some registry values under the following registry key:

```
[HKLM\Drivers\BuiltIn\armStoneA8\PWM]
[HKLM\Drivers\BuiltIn\NetDCU14\PWM]
```

Required settings:

Key	Value	Comment
"Prefix"	“PWM“	This required value specifies the driver’s device file name prefix. It is a three-character identifier, such as COM.
“Dll“	“FS_PWM.DLL“	Name of the DLL with the driver
“Order“	Dword:0x97	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
“Index“	Dword:1	This value specifies the device index, a value from 0 through 9.
“Channel”	Dword:	See table channel.
“Mode”	Dword:0|1	0: Absolute mode. Values range between 0 and “Steps” 1: Percent mode Values between 0 and 100%. Default: 1
“Steps”	Dword:0..0xFFFF	Amount of clocks in one frame. Default: 0xFFF
“Freq”	Dword:	Clock frequency Default: 300000Hz
“Default”	Dword:	PWM value after loading of the driver. Default: 0
“FriendlyName“	“PWM driver for NetDCU“	
“Flags”	Dword:0	4: Disabled from loading Default: 4

Key	Value	Comment
“Debug”	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 49: PWM: Registy

Note:

After opening the channel you can call WriteFile() to set the high phase. Use ReadFile() to read back the current value. The type of pointer is BYTE for Mode 1 and WORD for Mode 0. Please take a look at file pwm_sdk.h for additional IOCTL’s.

Note:

This driver is disabled by default. Enable this driver by setting registry value Flags to 0.

Table Channel armStoneA8:

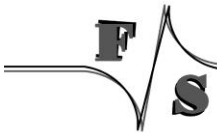
Channel	Description
0x00	Do not use! Backlight control. Use contrast control of display driver. (Display connector pin 25)
0x01	TOUT1 (Feature connector pin 28)
0x02	TOUT2 (Feature connector pin 30)
0x03	Disable resistive touch driver before using! TOUT3 (Feature connector pin 32)

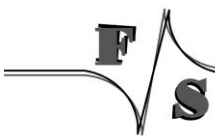
Table 50: PWM - armStoneA8 Channel

Table Channel NetDCU14:

Channel	Description
0x00	Do not use! Backlight control. Use contrast control of display driver. (Display connector pin 25)
0x01	PIFPWM (Connector J4, PARINTF, pin 15)

Table 51: PWM - NetDCU14 Channel





17 SD/MMC Driver

Implemented on: PM3,PM4,PM6,PM7,PM7A,QA8,ASA8,ND14,NRA8

Platform supports SD/MMC driver. There will be a driver for external SD slot and one for internal (only PM6/PM7A/ND14) SD slot.

The registry key for the external slot (PicoMOD/QBliss) is:

```
[HKLM\Drivers\Builtin\HSMMC]
```

The registry key for the on-board slot (PicoMOD6/7) is:

```
[HKLM\Drivers\Builtin\HSMMC1]
```

The registry key for the on-board slot (armStoneA8) is:

```
[HKLM\Drivers\Builtin\SDMMC_CH0]
```

The registry key for the external slot (PicoMOD7A,nanoRISC-A8) is:

```
[HKLM\Drivers\Builtin\SDMMC_CH0]
```

The registry key for the on-board slot (PicoMOD7A,nanoRISC-A8) is:

```
[HKLM\Drivers\Builtin\SDMMC_CH2]
```

Use the following parameters to configure the driver:

Key	Value	Comment
Prefix	“HSC“	This required value specifies the driver’s device file name prefix. It is a three-character identifier, such as COM.
Dll	“xxx.dll“	Name of the DLL with the driver
Order	Dword:0x15 Dword:0x16	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
IRQ		Don’t change.
PwrPin	Dword:	Number of the I/O pin used as power on pin. See documentation of digital I/O driver for possible values. In case you don’t use MOSFET to switch card voltage, set this value to -1 (0xffffffff) to free pin for other purposes,. Default: 25
WP	Dword:	Number of the I/O pin used as write protect pin. See documentation of digital I/O driver for possible values. In case you don’t want to use this hardware switch, set this value to -1 (0xffffffff) to free pin for other purposes,. Default: 26
WriteProtect	Dword:<0|1>	Enable disable write protection. This value will be ored with the hardware WP pin.
CardAvailable	Dword:<0|1>	Only for internal SD slot.
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 52: SD/MMC Driver Registry Settings

18 Ethernet Driver

Implemented on: PM3,PM4,PM6,PM7,PM7A,QA8,ASA8,ND14

The Ethernet-Interface features a small set of additional configurations:

```
[HKEY_LOCAL_MACHINE\Comm\ETHNETA1\Parms]
[HKEY_LOCAL_MACHINE\Comm\ETHNETB1\Parms]
```

Use the following parameters to configure the driver:

Key	Value	Comment
SpeedDuplex	Dword:	Enable/disable auto negotiation and select link speed 0x3100: AutoDetect 0: 10Mb-Half-Duplex 0x100: 10Mb-Full-Duplex 0x2000: 100Mb-Half-Duplex 0x2100: 100Mb-Full-Duplex Default: 0x3100
TxQueue	Dword:	Send Packet Mode. 0=OFF 1=ON Default: 1
VLAN	Dword:	VLAN on or off. 0=disable 1=enable Default: 0
VLAN_ID	Dword:	VLAN ID, set the value is between 0 to 4095. Default: 0
WakeUpFromLinkChange	Dword:	Wake-Up When Link Change. 0=disable 1=enable Default:0
WakeUpFromPacket	Dword:	Wake-Up when receive ARP/PING or MAGIC packet. 0=disable 1=Magic Packet 2=PING/ARP 3=Magic Packet/PING/ARP Default: 0
BackPressure	Dword:	Back Pressure Function. 0=disable 1=enable Default:1
FlowControl	Dword:	Flow Control Function. 0=disable 1=enable Default:1
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 53: Ethernet Driver Registry Settings

Signal-Name	PinNo	Signal-Type	Comment
GBE_ACT#	Q7:.pin44	I_FULL/COL	Full-Duplex/Collision Status. If this signal is low, it indicates full-duplex link established, and if it is high, then the link is in half-duplex mode. When in half-duplex and collision occurrence, the output will be driven low for 80ms and driven high at minimum 80ms.
GBE_LINK100#	Q7.pin7	I_SPEED	Speed Status: If this signal is low, it indicates 100Mbps, and if it is high, then the speed is 10Mbps.
GBE_LINK#, PM: ETH	Q7.pin13 PM: pin128	I_LK/ACT	Link Status/Active: If this signal is low, it indicates link, and if it is high, then the link is fail. When in link status and line activity occurrence, this signal is pulsed high (LED off) for 80ms whenever transmit or receive activity is detected. This signal is then driven low again for a minimum of 80ms, after which time it will repeat the process if TX or RX activity is detected.

Table 54: Ethernet - meaning of LEDs

19 Screen Saver Driver

Implemented on: PM3,PM4,PM6,PM7,PM7A,QA8,ASA8,ND14,NRA8

F&S Screen Saver driver works in combination with Microsoft power management driver pm.dll. Purpose of the driver is to avoid unwanted clicks when display is in screen-off state and touch is used to bring display back in run state.

The registry key for the driver is:

```
[HKLM\Drivers\Drivers\Builtin\PSS1]
```

Use the following parameters to configure the driver:

Key	Value	Comment
Prefix	“PSS“	This required value specifies the driver’s device file name prefix. It is a three-character identifier, such as COM.
Dll	"FSPMScreenSaver.dll"	Name of the DLL with the driver
Order	Dword:0x1	This value specifies the load order for the driver. If two drivers have the same load order value, the drivers load in the order that they occur in the registry.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
DxOn	Dword:	0
DxOff	Dword:	4
Flags	Dword:	0x10: User mode driver

Table 55: PSS: Registry settings

20 Appendix

Important Notice

The information in this publication has been carefully checked and is believed to be entirely accurate at the time of publication. F&S Elektronik Systeme assumes no responsibility, however, for possible errors or omissions, or for any consequences resulting from the use of the information contained in this documentation.

F&S Elektronik Systeme reserves the right to make changes in its products or product specifications or product documentation with the intent to improve function or design at any time and without notice and is not required to update this documentation to reflect such changes.

F&S Elektronik Systeme makes no warranty or guarantee regarding the suitability of its products for any particular purpose, nor does F&S Elektronik Systeme assume any liability arising out of the documentation or use of any product and specifically disclaims any and all liability, including without limitation any consequential or incidental damages.

Products are not designed, intended, or authorized for use as components in systems intended for applications intended to support or sustain life, or for any other application in which the failure of the product from F&S Elektronik Systeme could create a situation where personal injury or death may occur. Should the Buyer purchase or use a F&S Elektronik Systeme product for any such unintended or unauthorized application, the Buyer shall indemnify and hold F&S Elektronik Systeme and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, expenses, and reasonable attorney fees arising out of, either directly or indirectly, any claim of personal injury or death that may be associated with such unintended or unauthorized use, even if such claim alleges that F&S Elektronik Systeme was negligent regarding the design or manufacture of said product.

Listings

Figures

Tables