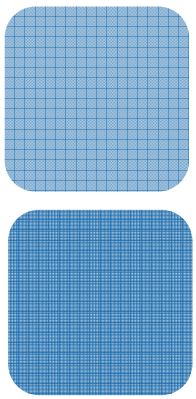
Device Driver Documentation

Windows Embedded Compact for FSVYBRID

Version 1.17
(2016-09-26)



Elektronik Systeme

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	1.00	All	A	*	First version	HF
2014-05-01	1.01	FSVYBRID	A	2	Added description for registry value Resolution	HF
2014-05-22	1.02	FSVYBRID	A,M	13,14,17	Correct I2C and SPI registry settings	MA
2014-05-22	1.02	FSVYBRID	M	8	Correct description for 2nd USB Host.	HF
2014-05-22	1.03	FSVYBRID	M	5	Correct description for use serial control lines.	ZU
2014-06-28	1.04	FSVYBRID	A, M	7.1.2	Added new registry value for EDT touch	AD
2014-07-21	1.05	FSVYBRID	A	1.1	Description for registry key HKLM\Platform in chapter Boot Information	HF
2014-07-22	1.05	FSVYBRID	A	20	Description of Broadcast Driver	HF
2014-07-21	1.05	FSVYBRID	A	21	Description for File System Filter FSDFilter	HF
2014-07-23	1.05	FSVYBRID	M	*	Changed to new CI	JG
2014-07-30	1.06	FSVYBRID	A	10.4	Added chapter XP Skin	HF
2014-08-25	1.07	FSVYBRID	M	10.3.3	Added chapter for table LCDPortDriveStrength.	HF
2014-09-02	1.08	FSVYBRID	M	4.1, 4.2, 4.3	Correct and format IO Tables	AD
2014-09-10	1.09	FSVYBRID	M	7.1.3	Added new registry values for res. Touch driver	ZU
2014-09-16	1.09	FSVYBRID	A	22	Description for File System Redirector	HF
2014-10-01	1.09	FSVYBRID	M	16	Card detection modes	ZU
2014-10-27	1.10	FSVYBRID	A	13, 14	Added drive strength table for I2C, NI2C	AD
2014-10-30	1.10	FSVYBRID	M	7	5-wire touch support	ZU
2014-11-05	1.10	FSVYBRID	M	14, 7.1.1, 7.1.3	Added new registry values for NI2C and figure for FS I2C bus test tool. Correct description of I2CdevAddr.	AD
2014-12-01	1.10	FSVYBRID	M	5	Added registry values for UART configuration	TM
2014-12-05	1.11	FSVYBRID	A	8	Adding information about ForceFullSpeed on USB Host	MA
2015-01-16	1.12	FSVYBRID	M	4.1	Added corrected IO table ASA5	AD
2015-02-09	1.13	FSVYBRID	M	14	Added “I2C1:” driver activation note for NDA5.	AD
2015-06-12	1.14	FSVYBRID	M	9	Remove wrong description for “ForceFullSpeed”=1	ZU
2015-06-12	1.14	FSVYBRID	M	8	Add default setting for “ForceFullSpeed”	ZU
2015-09-17	1.15	FSVYBRID	A	7.1.3	Added description for TouchSamples, AdcReadHoldoffHns, SetDelay, TouchRate	AD
2016-03-21	1.16	FSVYBRID	M	7.1.3	Modified description for TouchSamples, AdcReadHoldoffHns, SetDelay, TouchRate	AD
2016-04-21	1.16	FSVYBRID	M	4.1	Correct NetDCUA5 IO Table	ZU
2016-06-29	1.16	FSVYBRID	A	5	Added UART Overview	JG
2016-09-29	1.17	FSVYBRID	A, M	7.1.3	Added description for for PNDTPullUp, OPMode, ChannelsNr, Z1MinBound, Z1MaxBound, Z2MinBound, Z2MaxBound. Modified description for Threshold.	AD

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

About this document

This is the device driver documentation for the F&S platform FSVYBRID based on Windows Embedded CE 6.0 or Windows Embedded Compact 7/2013. 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
- NetDCUA5
- PicoCOMA5
- PicoMOD1.2

- PicoMODA5

Table of Contents

1 Boot Process

After power up, the internal ROM Loader is started. Depending from configuration, ROM loader trys to load NBoot from NAND flash memory. If successful NBoot starts and loads EBoot.
To increase reliability of boot process, we have installed NBoot two times. If ROM loader could not load first copy of NBoot for any reason, it loads backup copy of NBoot. Please take a look to following diagram about boot process.

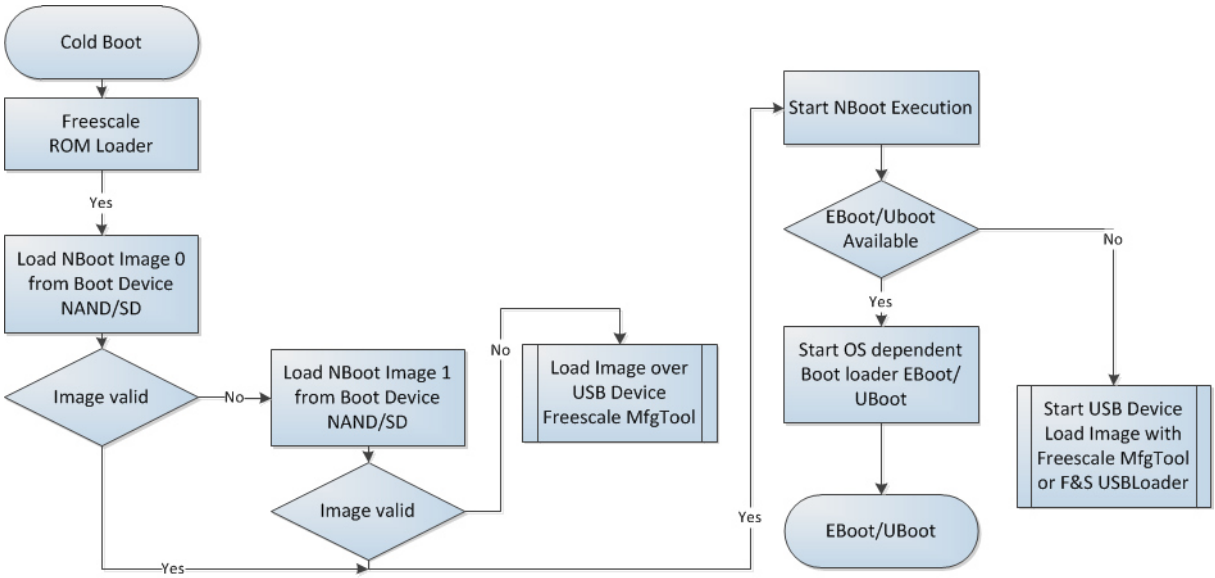


Figure 1: Boot Process

1.1 Boot Information

During start of Windows CE, kernel writes some information to registry. This information can be found under following registry key:

```
[HKLM\Platform]
```

BoardType	Dword	i.e. 0=armStoneA5, 1=PicoCOMA5, 2=NetDCUA5, 4=PicoMODA5, 5=PicoMOD1.2
BoardName	String	i.e. PicoCOMA5, NetDCUA5
BoardRevision	Dword	i.e. 100
BootVerMajor	Dword	Major version of EBoot loader
BootVerMinor	Dword	Minor version of EBoot loader
KernelVersion	String	Version of Windows CE Kernel
KernelVersionDate	String	Build date of Windows CE Kernel
KernelVersionTime	String	Build time of Windows CE Kernel
RestartReason	String	i.e. <UNKNOWN>, Power On Reset, CA5 Watchdog, RESETB, Software
StepStone Loader, Version	Dword	Version of installed NBoot

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.

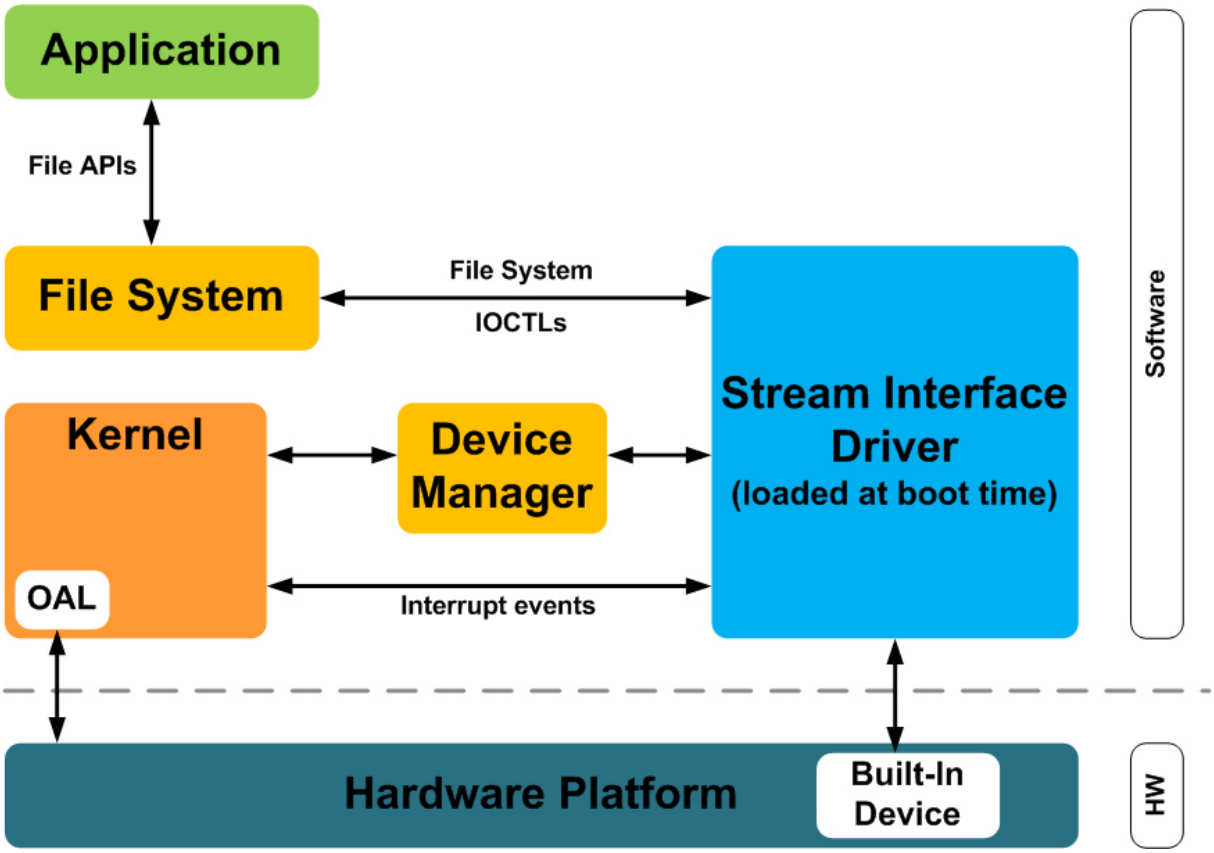


Figure 2: Windows CE: Stream Interface Driver Architecture

2 Analogue Input

Implemented on: ASA5, NDA5, PCA5, PMA5

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.
Average	Dword:n	Number of conversations per sample (4, 8, 16 or 32).
Channel	Dword:n	Number of the analogue channel. See Table Channel.
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
Resolution	Dword:n	8, 10 or 12 bit resolution.
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0
FriendlyName	"Analogue input driver"	

Table 1: 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 2: 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 3: Analogue Input: NetDCUA5 Channel

Table Channel PicoCOMA5:

Channel	Description
0x01	Reads value from analogue input 1 (Pin 41)
0x1A	Reads value from temperature sensor

Table 4: 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 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

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 5: 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 6: 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 7: Read temperature

3 Audio Driver

Implemented on: ASA5, PCA5, NDA5, PMA5

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

```
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\armStoneA5\Audio]
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\NetDCUA5\Audio]
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\PicoCOMA5\Audio]
```

The mixer line settings are compatible across all Windows Embedded 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
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 5: 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.

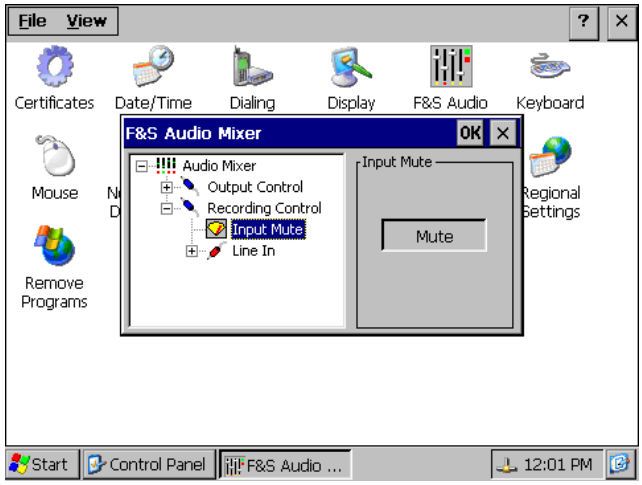


Figure 3: F&S Audio Mixer control

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 8: 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 9: Audio: Access mixer from user application

4 Digital I/O

Implemented on: ASA5, PCA5, NDA5, PM1.2, PMA5

Boards have programmable I/O lines. You have to use this driver to configure and access these 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 6: Digital I/O: Registry settings

4.1 Port description armStone

The port numbering of armStone 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.

					armStoneA5							
Digital-IO					Pin	Function					capabilities	NetDCU J5
IO-Pin	Port		Registry settings			*COM*	*I2C*	*SPI1 CAN*	*LCD*	*other*		
0	0	Port 0	0	UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1								
1	1		1		1					3.3V		26
2	2		2		2					5.0V		25
3	3		3		3					COL0	I/O/IRQ*	24
4	4		4		4					COL1	I/O/IRQ*	23
5	5		5		5					COL2	I/O/IRQ*	22
6	6		6		6					COL3	I/O/IRQ*	21
7	7		7		7					COL4	I/O/IRQ*	20
8	0	Port 1	8		8					COL5	I/O/IRQ*	19
9	1		9		9					COL6	I/O/IRQ*	18
10	2		10		10					COL7	I/O/IRQ*	17
11	3		11		11					GND		16
12	4		12		12			SPI_CLK		COL11	I/O/IRQ*	15
13	5		13		13	TXD2					I/O/IRQ	14
14	6		14		14			SPI_CS		COL10	I/O/IRQ*	13
15	7		15		15	RXD2					I/O/IRQ	12
16	0	Port 2	16		16		CLK	SPI_MOSI		COL9	I/O/IRQ*	11
17	1		17		17		DAT	SPI_MISO		COL8	I/O/IRQ*	10
18	2		18		18					ROW0	I/O/IRQ	9
19	3		19		19					ROW1	I/O/IRQ	8
20	4		20		20					ROW2	I/O/IRQ	7
21	5		21		21					ROW3	I/O/IRQ	6
22	6		22		22					ROW4	I/O/IRQ	5
23	7		23		23					ROW5	I/O/IRQ	4
24	0	Port 3	24		24					ROW6	I/O/IRQ	3
25	1		25		25					ROW7	I/O/IRQ	2
26	2		26		26					KBINT	I/O/IRQ	1
27	3		27		27					GND		
28	4		28		28					PWM1	I/O/IRQ	
29	5		29		29					ANALOGIN0		
30	6		30		30					PWM2	I/O/IRQ	
31	7		31		31					ANALOGIN1		
32	0	Port 4	32		32					PWM3	I/O/IRQ	
33	1		33		33					ANALOGIN2		
34	2		34		34				VCFL_ON		I/O/IRQ	
35	3		35		35					ANALOGIN3		
36	4		36		36					RXD3		
37	5		37		37					GND		
38	6		38		38					TXD3		
39	7		39		39					3.3V		
40	0	Port 5	40		40					5.0V		
41	1		41		41					MICIN		
42	2		42		42					GND		
43	3		43		43					N/C		

44	4		44		44					LINEINR		
45	5		45		45					LINEOUTR		
46	6		46		46					GND		
47	7		47		47					GND		
48	0	Port 6	48		48					LINEINL		
49	1		49		49					LINEOUTL		
50	2		50		50					GND		
51	3		51		51					XRESETBTN		
52	4		52		52					3.3V		
53	5		53		53					N/U		
54	6		54		54					N/C		
55	7		55		55					RXA		
56	0	Port 7	56		56					RTSA		
57	1		57		57					TXA		
58	2		58		58					CTSA		
59	3		59		59					N/C		
60	4		60		60					N/C		
61	5		61		61					GND		
62	6		62		62					5.0V		
63	7		63		63					CANRX		
64	0	Port 8	64		64					CANTX		
65	1		65		65					BOOTSEL		
66	2		66		66					3.3V		
67	3		67									
68	4		68						* - 3.3V on 4,7kΩ - PullUp			
69	5		69						N/U – not used			
70	6		70						N/C – not connected			
71	7		71									

Table 7: Digital I/O pins – armStoneA5

4.2 Port description NetDCUA5

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	15	14	13	12	11	10	9	8
IRQCfg0A IRQCfg1A IRQCfg2A	15	14	13	12	11	10	9	8

Port 2

Bit	7	6	5	4	3	2	1	0
Pin	17	18	19	20	21	22	23	24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
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	23	22	21	20	19	18	17	16

Port 12

Bit	7	6	5	4	3	2	1	0
Pin	---	---	---	1	---	---	---	---
R/W	--	--	--	R/W	--	--	--	--
UseAsIOD Bit	---	---	---	100	---	---	---	---
DataDirD Bit	---	---	---	100	---	---	---	---
DataInitD Bit	---	---	---	100	---	---	---	---
IRQCfg0D IRQCfg1D IRQCfg2D	---	---	---	100	---	---	---	---

Table 8: Digital I/O – NetDCUA5 Port0 - 2

Digital-IO					NetDCUA5				
						Function			capabilities
IO-Pin	Port		Registry settings		J5-Pin	*I2C*	*SPI1 CAN*	*other*	
0	0	Port 0	0	UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1	9			ROW0	I/O/IRQ
1	1		1		8			ROW1	I/O/IRQ
2	2		2		7			ROW2	I/O/IRQ
3	3		3		6			ROW3	I/O/IRQ
4	4		4		5			ROW4	I/O/IRQ
5	5		5		4			ROW5	I/O/IRQ
6	6		6		3			ROW6	I/O/IRQ
7	7		7		2			ROW7	I/O/IRQ
8	0	Port 1	8		15		CLK		I/O/IRQ
9	1		9		13		CS		I/O/IRQ
10	2		10		11	SCL	MOSI		I/O/IRQ
11	3		11		10	SDA	MISO		I/O/IRQ
12	4		12						
13	5		13						
14	6		14						
15	7		15						
16	0	Port 2	16		24			COL0	I/O/IRQ
17	1		17		23			COL1	I/O/IRQ
18	2		18		22			COL2	I/O/IRQ
19	3		19		21			COL3	I/O/IRQ
20	4		20		20			COL4	I/O/IRQ
21	5		21		19			COL5	I/O/IRQ
22	6		22		18			COL6	I/O/IRQ
23	7		23		17			COL7	I/O/IRQ
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				
…									
96	0	Port 12	96						
97	1		97						
98	2		98						
99	3		99						
100	4		100		1				I/O/IRQ
101	5		101						
102	6		102						
103	7		103						

Table 9: Digital I/O pins – NetDCUA5

4.3 Port description PicoCOMA5

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					PCA5-Pin	PicoCOMA5 Function							capabilities	SKIT-Pin (J10)
IO-Pin	Port		Registry settings			*COM*	*I2C*	*SPI1 CAN*	*USB*	*SD/ MMC*	*LCD*	*other*		
0	0	Port 0	0	UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1	13	TXD2							I/O/IRQ	
1	1		1		14	RXD2							I/O/IRQ	
2	2		2		15	RTS2/TXD3							I/O/IRQ	
3	3		3		16	CTS2/RXD3							I/O/IRQ	
4	4		4		17	TXD1							I/O/IRQ	
5	5		5		18	RXD1							I/O/IRQ	
6	6		6		23				OTGVBUS			~~EINT4~~	No I/O	
7	7		7		24				PWR			~~EINT8~~	I/O/IRQ	
8	0	Port 1	8		26			MISO0					I/O/IRQ	3
9	1		9		27			MOSI0					I/O/IRQ	4
10	2		10		28			SPCK0					I/O/IRQ	5
11	3		11		29			PCS0					I/O/IRQ	6
12	4		12		32		SDA	CANRX1					I/O/IRQ	9
13	5		13		33		SCL	CANTX1					I/O/IRQ	10
14	6		14		34					DAT0			I/O/IRQ	
15	7		15		35					DAT1			I/O/IRQ	
16	0	Port 2	16		36					DAT2			I/O/IRQ	
17	1		17		37					DAT3			I/O/IRQ	
18	2		18		38					CLK			I/O/IRQ	
19	3		19		39					CMD			I/O/IRQ	
20	4		20		40							EINT2	I/O/IRQ	11
21	5		21		41							PWM	I/O/IRQ	12
22	6		22		43						R1 (R3)		I/O/IRQ	
23	7		23		44						R2 (R4)		I/O/IRQ	
24	0	Port 3	24		45						R3 (R5)		I/O/IRQ	
25	1		25		46						R4 (R6)		I/O/IRQ	
26	2		26		47						R5 (R7)		I/O/IRQ	
27	3		27		48						G0 (G2)		I/O/IRQ	
28	4		28		49						G1 (G3)		I/O/IRQ	
29	5		29		50						G2 (G4)		I/O/IRQ	
30	6		30		51						G3 (G5)		I/O/IRQ	
31	7		31		52						G4 (G6)		I/O/IRQ	
32	0	Port 4	32		53						G5 (G7)		I/O/IRQ	7
33	1		33		54						B1 (B3)		I/O/IRQ	8
34	2		34		55						B2 (B4)		I/O/IRQ	
35	3		35		56						B3 (B5)		I/O/IRQ	
36	4		36		57						B4 (B6)		I/O/IRQ	
37	5		37		58						B5 (B7)		I/O/IRQ	
38	6		38		59						VCLK		I/O/IRQ	
39	7		39		60						VM		I/O/IRQ	
40	0	Port 5	40		63						HSYNC / B0 (B2)		I/O/IRQ	
41	1		41		64						VSYNC / R0 (R2)		I/O/IRQ	
42	2		42		65						~~VEEK~~	BL_CTRL	I/O/IRQ	
43	3		43		66						VLCD-ON		I/O/IRQ	
44	4		44		67						VCFL-ON		I/O/IRQ	
45	5		45		68						VCD-DEN		I/O/IRQ	
46	6		46		69	RTS1							I/O/IRQ	13
47	7		47		11	CTS1						EINT1	I/O/IRQ	1

48	0	Port 6	48	**12**					CD		~~EINT0~~	I/O/IRQ	2
49	1		49	**30**		SDA	CANTX					I/O/IRQ	
50	2		50	**31**		SCL	CANRX					I/O/IRQ	
51	3		51										
52	4		52										
53	5		53										
46	6		54										
47	7		55										

Table 10: Digital I/O pins – PicoCOMA5

4.4 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 11: Digital I/O - PicoMOD Port 0 – 9

4.5 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

4.6 Programming example

Headerfile:

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

Listing 10: 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 11: 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 12: 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 13: 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 14: 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 15: Digital I/O: Using Interrupts

L. Closing port

```
CloseHandle(hDIO);
```

Listing 16: Digital I/O: Closing port

5 Driver for Serial I/O (UART)

Implemented on: ASA5, PCA5, NDA5, PM1.2, PMA5

Vybrid boards have a maximum of four serial ports (UART). Following communication settings are supported by the driver (data bits, parity, stop bit(s)):
(7,O,1), (7,E,1), (8,N,1), (8,N,2), (8,O,1), (8,E,1).
Installation of the driver is done by setting some registry values under the following registry key:

```
[HKLM\Drivers\BuiltIn\<Board Type>\UART<n>]
```

Settings:

Key	Value	Comment
Priority256	dword:	Priority for the serial interface thread. Default: 159
Invert	dword:	This optional value specifies the polarity of Tx, Rx and RTS lines. 0x0 -> no polarity inversion at all (default) 0x1 -> Tx polarity inversion 0x2 -> Rx polarity inversion 0x4 -> RTS polarity inversion These values can be combined bitwise.
UseRTSCTS	dword:	Set this value to 1 to enable access to RTS/CTS lines Default : 0
RS485	dword:	Set this value to 1 to enable RS485 mode.
RXDMAEnable	dword:	Set this value to 1 to enable the RX DMA Default: 0
TXDMAEnable	dword:	Set this value to 1 to enable the TX DMA Default: 0

Table 9: UART - Registry settings

Remark:

The driver support RTS_CONTROL_TOGGLE. This function and the RTS pin can be used for RS485 interface. No additional registry setting is required.
Alternative you can set the registry value RS485 to 1. This does the same as setting RTS_CONTROL_TOGGLE in the DCB.

The driver (since V1.10) supports access to RTS/CTS via “EscapeCommFunction”. This function and the RTS/CTS pins can be used for RS232 interfaces with RTS/CTS lines.

5.1 UART Overview armStoneA5

armStone										
	COM1 (Debug)				COM2					
Signal	*RXA/RXD1*	*TXA/TXD1*	*CTSA/CTS1*	*RTSA/RTS1*	*RXD2*	*TXD2*			RXC/RXD3	TXC/TXD3
Connector	J12-55	J12-57	J12-58	J12-56	J12-15	J12-13			J12-36	J12-38
armStoneA5	UART1				UART2				UART3	

5.2 UART Overview PicoCOMA5

PicoCOM										
	COM1 (Debug)				COM2				COM3	
SKIT-Signal	*RXD1*	*TXD1*	*CTS1*	*RTS1*	*RXD2*	*TXD2*	*CTS2*	*RTS2*	*RXD3*	*TXD3*
SKIT-Connector	J7-3	J7-5	J10-1	J10-13	J6-3	J6-5	J6-6	J6-4	J7-3/5 (RS485)	
Connector-Pin	18	17	55 (PC2)	69	14	13	16	15	52 (PC2)	51 (PC2)
			11 (PC3/4)						16 (PC 3,4,A5)	15 (PC 3,4,A5)
PicoCOMA5 V1.00	UART5				UART1				UART2	
PicoCOMA5 V1.10	UART1				UART2				UART3	

5.3 UART Overview PicoMODA5

PicoMOD										
	COM2				COM1(Debug)		COM3		COM4	
SKIT-Signal	*RXD2*	*TXD2*	*CTS2*	*RTS2*	*RXD2*	*TXD2*	*RXD3*	*TXD3*	*RXD3*	*TXD3*
SKIT-Connector	J2-3	J2-5	J2-6	J2-4	J4-3	J4-5	J6-3	J6-5	J2-6	J2-4
Connector-Pin	18	17	20	19	24	23	22	21	**20/6**	**19/5**
PicoMODA9	UART1				UART2		UART4		UART5	

6 Matrix-Keyboard

Implemented on: all

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 14: 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 13: 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 14: 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 22: Matrix Keyboard: Connector J1 (example for PicoCOM only)
…		
IOCol*n*	Dword:	Number of IO you want use for last column. See Table 22: Matrix Keyboard: Connector J1 (example for PicoCOM only)

Table 15: 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 22: Matrix Keyboard: Connector J1 (example for PicoCOM only)
…		
IORow*n*	Dword:	Number of IO you want use for last row. See Table 22: Matrix Keyboard: Connector J1 (example for PicoCOM only)

Table 16: 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 22: Matrix Keyboard: Connector J1 (example for PicoCOM only)
StaticKey0	Dword :	PS2 code for static key 0. See Table 19: Matrix Keyboard: PS2 Scan Codes
…		
IOStaticKey*n*	Dword :	Number of IO you want use for last static key. See Table 22: Matrix Keyboard: Connector J1 (example for PicoCOM only)
StaticKey*n*	Dword :	PS2 code for last static key. See Table 19: Matrix Keyboard: PS2 Scan Codes

Table 17: 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 18: 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
'3'	// Scan Code 0x4
'4'	// Scan Code 0x5

V-KEY	PS2-Scan-Code
'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
'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

V-KEY	PS2-Scan-Code
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
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

V-KEY	PS2-Scan-Code
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 19: 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 20: 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 21: 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
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

Pin	IO	Default Interface	Starter-Kit Interface
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
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 22: 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 17: 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 18: Matrix Keyboard: Example 2

7 Touchpanel Driver

Implemented on: ASA5, PCA5, NDA5, PMA5

From Windows Embedded 7 Compact we support native driver interface as well as stream driver interface. Both driver models use different software components for managing driver access and different registry configuration. The registry keys for the values are

`[HKEY_LOCAL_MACHINE\DRIVERS\BUILTIN\TOUCH_(NAME)] and`
`[HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\TOUCH]`

With stream driver interface we use the touch screen proxy driver to provide access to GWES functions. The table 20 shows registry values for the proxy driver:

`[HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\TOUCH]`

Key	Value Type	Default Value	Comment
DriverName	String	fs_tchproxy.dll	Name of driver that GWES loads.
MaxCalError	DWORD	10	The maximum error distance permitted in a touch screen calibration, in screen unit.

Table 23: Touch screen proxy driver settings

Additional to proxy driver followed registry key have to be configured:

`[HKEY_LOCAL_MACHINE\SYSTEM\GWE\TOUCHPROXY]`

Key	Value Type	Default Value	Comment
TchCaldll	String	fs_tchcaldll.dll	Indicates the touch calibration DLL that the touch proxy DLL uses. You can omit this value if the touch driver has its own calibration routines.
DriverLoadTimeoutMs	DWORD	100	Indicates how long touch proxy will wait for touch driver to load in ms.

Table 24: Touch screen proxy driver - GWE settings

Note:

We ported stream driver interface to Windows CE 6, so that same registry configuration can be used.

7.1.1 MXT224 Touch Driver

The MXT224 highly configurable touchscreen controller that is part of the Atmel maXTouch product platform. This driver can be activated by setting the registry value Flags.

Here is a list of registry entries for the touch driver, including values you might set to use the touch screen driver:

```
[HKEY_LOCAL_MACHINE\DRIVERS\BUILTIN\TOUCH_MXT224]
```

Key	Value Type	Default Value	Comment
ChangeIO	DWORD	20	Touches 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 = (0x4B<<1)	I2C Device address of the touch controller with write command flag
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.
LogFileDebug	DWORD	0	0: No log messages. 3: Write log messages to file *LogFile*.
LogFile	SZ		Name of file for log messages..
Flags	DWORD	4	4: Driver is not loaded. 8: Driver will be loaded.

Table 25: Capacitive touch driver registry settings

Note:

A touch calibration is not required as the touch controller automatically scales the touch samples to the screen size. Other touch drivers have to be deactivated.

7.1.2 EDT Touch Driver

The FT5406 are single-chip capacitive touch panel controller. They adopt the mutual capacitance approach, which supports true multi-touch capability. This driver can be activated by setting the registry value Flags.

Here is a list of registry entries for the touch driver, including values you might set to use the touch screen driver:

`[HKEY_LOCAL_MACHINE\DRIVERS\BUILTIN\TOUCH_EDT]`

Key	Value Type	Default Value	Comment
ChangeIO	DWORD	20	Touches 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 = (0x38<<1)	I2C Device address of the touch controller with write command flag.
Flags	DWORD	4	4: Driver is not loaded. 8: Driver will be loaded.
Threshold	DWORD	-1	Value to determine when a signal is a valid touch. Default value means that register default value will be used.
Gain	DWORD	-1	Adjusts the sensitivity of the sensing circuit. Lower value increases the sensitivity. Default value means that register default value will be used.
Offset	DWORD	-1	Adjusts the behavior of the touch close to the edge. Default value means that register default value will be used.

Table 26: Capacitive 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.

Note:

Touch proxy have to wait longer for the touch driver as default value 100 ms.

7.1.3 SX865x Touch Driver

The SX8654 is haptics enabled 4/5-wire resistive touch screen controller with proximity detection, optimized for hand held applications. The driver can be activated by setting the registry value Flags.

Here is a list of registry entries for the touch driver, including values you might set to use the touch screen driver:

```
[HKEY_LOCAL_MACHINE\DRIVERS\BUILTIN\TOUCH_SX865x]
```

Key	Value Type	Default Value	Comment
ChangeIO	DWORD	0x63	Touches interrupt IO-Pin number.
ResetIO	DWORD	0x62	IO-Pin used to trigger controller reset during initialization. A value of -1 disables this functionality.
I2CDevAddr	DWORD	0x90 = (0x48<<1)	I2C Device address of the touch controller with write command flag.
Threshold	DWORD	800	Normalized threshold value. Represent a ratio between the touch resistance Rt and the total resistance for the Y plate Rytot. Is checked as condition: Threshold > [Ypos/4095 * (Z2/Z1 -1) * 100]
Flags	DWORD	8	4: Driver is not loaded. 8: Driver will be loaded.
I2CDevice	SZ	“I2C3:”	I2C Device name.
MinMove	DWORD	0	Minimum move before Mouse Move is signaled
MaxMove	DWORD	0x3FF	Maximum move for which a Mouse Move is signaled
TouchType	DWORD	0	0: 4-wire touch panel 1: 5-wire touch panel
TouchSamples	DWORD	3	Amount of samples that are used to create the position value. Possible values are 1, 3, 5, 7 samples. Corresponds to the FILT bits of RegTouch1 register.

Key	Value Type	Default Value	Comment
AdcReadHoldoffHns	DWORD	5680	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. SeeTable 28 for possible values. Corresponds to the POWDLY bits of RegTouch0 register.
SetDelay	DWORD	5	Amount of time (in 100 ns units) to wait between the consecutive conversions of the same channel. See Table 28 for possible values. Corresponds to the SETDLY bits of the RegTouch2 register.
TouchRate	DWORD	200	Touch coordinates acquisition rate in count per seconds (cps). Corresponds to the TOUCHRATE bits of the RegTouch0 register. Min: 10 cps Max: 5 kcps
PNDTPullUp	DWORD	1	Pen detection pull-up: 0: 114 kOhms 1: 228 kOhms 2: 57 kOhms 3: 28 kOhms
OPMode	DWORD	0	Pen trigger mode: 0: without release irq 1: with release irq
ChannelsNr	DWORD	4	Channels number: 2: X,Y channels 4: X,Y,Z1,Z2 channels
Z1MinBound	DWORD	1	Minimum bound value for pressure Z1
Z1MaxBound	DWORD	4095	Maximum bound value for pressure Z1
Z2MinBound	DWORD	1	Minimum bound value for pressure Z2
Z2MaxBound	DWORD	4095	Maximum bound value for pressure Z2

Table 27: Resistive touch driver registry settings

Amount of time
5

Amount of time
11
22
44
89
178
710
1420
2480
5680
11400
22700
45500
90900
181900

Table 28: Possible values in 100 ns units

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: ASA5, NDA5, PCA5, PM1.2, PMA5

Board 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\USB1]
```

The registry key for the 2nd , optional USB host is:
```
[HKLM\Drivers\Builtin\USB0]
```

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	fs_hcd_hsotg0.dll fs_hcd_hsotg.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.
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
ForceFullSpeed	Dword:	1: Force the port to use only full speed Default: 0
Flags	Dword:	4: Disable driver from loading. Default for USB0.

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 disabling the USB driver dialog. Default: 0

Table 30: Windows CE USB Host: Controller Registry settings

Note:

When using 2x 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: ASA5, NDA5, PCA5, PM1.2, PMA5

Board 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	"fs_usbfn.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.
ForceFullSpeed	Dword: 0, 1	0: High speed (USB 2.0) 1: Full speed (USB 1.1) Default: 0
Priority256	Dword:	Default: 100

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 for FSVYBRID

Implemented on: ASA5, PCA5, NDA5, PM1.2, PMA5

Boards have a very flexible and powerful interface for LCD TFT displays. The driver is fully configurable over the Window CE/Compact 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)
- Adjustable frame buffer depth 8/16/24/32 BPP
- Adjustable output depth 16/18/24 BPP
- Overlays
- DirectDraw

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
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	BPP	DE	HVSync	VCLK
0	VGA standard display	640x480	16	On	On	25MHz
1	SVGA standard display	800x600	16	On	Low	40MHz
2	XGA standard display	1024x768	16	On	On	65MHz
3						
4	QVGA standard display	320x240	16	On	On	6MHz
5	XGA standard display 56MHz	1024x78	16	On	On	56MHz
6	EDT ET070080	800x480	16	On	On	33MHz
7	EDT ET035080	320x240	16	On	On	10MHz
8	Hitachi TX09	240x320	16	On	On	6MHz
9	EDT ET043080	480x272	16	On	On	9MHz
10	NEC NL6448BC	640x480	16	On	On	25MHz

Mode	Name	XxY	BPP	DE	HVSync	VCLK
11	Sharp LQ104	640x480	16	On	On	25MHz
12	AOU G104SN03	640x480	16	On	On	25MHz
13	EDT ET057090DH	640x480	16	On	On	25MHz
14	AOU G104SN02	800x600	16	On	On	38MHz
15	Hitachi TX18D35	800x480	16	On	On	33MHz
16	WXGA standard display	1280x800	16	On	On	90MHz
17	WVGA standard display	1024x600	16	On	On	51MHz
18	CHIMEI G070Y	800x480	16	On	Low	auto
19	ET070080 ASA5	800x480	16	On	On	33MHz

Table 34: 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".

10.1 Default Display Mode

	Digital RGB	LVDS
armStoneA5	19 = 800x480 (ET070080)	19 = 800x480 (ET070080)
PicoCOMA5	6 = 800x480 (ET070080)	--
NetDCUA5	0 = 640x480 (VGA standard display)	--

Table 35: LCD - Default Display Mode

10.2 Default LCD Output Width

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

	Digital RGB	LVDS
armStoneA5	LCD_CONFIG_OUT18BIT	LCD_CONFIG_OUT18BIT
PicoCOMA5	LCD_CONFIG_OUT16BIT	--
NetDCUA5	LCD_CONFIG_OUT18BIT	--

The configuration can be changed with registry parameter CONFIG.

Note:

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

10.3 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]
```

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.

Key	Type	Meaning
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.
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.
ContrastFreq	Dword:	Frequency for PWM in Hz.
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 *„10.3.3* Registry Value LCDPortDriveStrength“
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.

10.3.1 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 36: LCD - Display Driver Registry Value Type

10.3.2 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 35: LCD - Default Display Mode”*
LCD_OUT16BIT	0x01000000	RGB565
LCD_OUT18BIT	0x02000000	RGB666
LCD_OUT24BIT	0x03000000	RGB888
LCD_DEMODE	0x10000000	Use signal DE/M for timing. Drive HSync and VSync low.

Table 37: LCD - Display Driver Registry Value Config

10.3.3 Registry Value LCDPortDriveStrength

Adjust LCD port drive strength with following parameter:

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

Following values can set:

Value	LCD Port Drive Strength
1 (default)	150 Ohm
2	75 Ohm
3	50 Ohm
4	37 Ohm
5	30 Ohm
6	25 Ohm
7	20 Ohm

Table 38: LCD - Port Drive Strength

10.4 UI Skin / XP Mode

Most pre-configured images are built with SYSGN_XPSKIN set. Advantage is, that the standard explorer and dialog boxes look more modern compared to the old Win2K skin.
Disadvantage is that with XP skin you can't set background color of a button control.
To overcome this we have modified the skin and it's now possible to activate old drawing with a registry value.

```
[HKLM\System\GWE]
```

Key	Type	Meaning
BtnOldSkin	dword	Set this value to 1 to enable old button drawing. This also speeds up drawing at all. Default: 0

11 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 39: Softkeybd: Registrysettings

12 CAN

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

13 I2C Driver

Implemented on: ASA5, PCA5, NDA5, PM1.2, PMA5

Board supports GPIO (bit banging) I2C driver.

The registry key for the driver is:

```
[HKLM\Drivers\Builtin\<board>\I2C<n>]
```

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	fs_i2c.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. 0: output driver disabled 1: 150 Ohm 2: 75 Ohm 3: 50 Ohm 4: 37 Ohm 5: 30 Ohm 6: 25 Ohm 7: 20 Ohm (default)
Flags	Dword:	4: Disabled from loading

Table 40: 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_I2CScan.exe. (s. Fig. Figure 4) This program lists the available I2C ports and scans the port for devices.

14 Native I2C Driver

Implemented on: ASA5, PCA5, NDA5, PM1.2, PMA5

Board supports native I2C driver.

The registry key for the driver is:

```
[HKLM\Drivers\Builtin\<board>\I2C<n>]
[HKLM\Drivers\Builtin\I2C3]
```

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	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:0x30d40	200000 kBit/s (default)
DrvStrength	Dword:	Set drive strength control for SDA/SCL. 0: output driver disabled 1: 150 Ohm 2: 75 Ohm 3: 50 Ohm 4: 37 Ohm 5: 30 Ohm 6: 25 Ohm 7: 20 Ohm (default)
IntPullUp	Dword:n	0: Disable internal pull-up (default) 1: 100k pull-down 2: 47k pull-up 3: 100k pull-up 4: 22k pull-up
Priority256	Dword:103	Priority for the transmit/receive thread. Default: 103
RepeatedStarts	Dword:0|1	0: disabled 1: enabled (default)
Flags	Dword:	4: Disable driver from loading
ClkToggleNumber	Dword:n	Toggles number of SCL line Default: 8
I2CBusCheck	Dword:n	Recovery strategy for different I2C functions. 0: disabled 1: by initialization (default) 2: by each transfer

Table 41: 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_I2CScan.exe. (s. Fig. Figure 4) This program lists the available I2C ports and scans the port for devices.

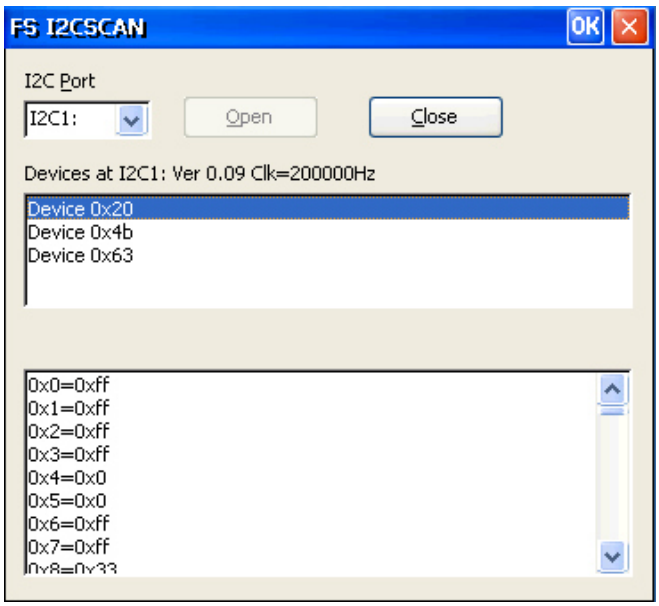


Figure 4: F&S I2C Bus test tool

armStoneA5:
At armStoneA5 we have two NI2C drivers and one I2C drivers.
The usage is as follows:

Connector	driver
Feature connector	I2C1: (native I2C driver)
Touch	I2C3: (native I2C driver), on-board, and connector
Audio	I2C8: (I2C driver)

Table 42: armStoneA5 I2C driver usage

PicoCOMA5:
At PicoCOMA5 we have two NI2C drivers and one I2C drivers.
The usage is as follows:

Connector	driver
Main connector	I2C1: (native I2C driver)
Touch	I2C3: (native I2C driver), on-board
Audio	I2C8: (I2C driver), on-board

Table 43: PicoCOMA5 I2C driver usage

NetDCUA5:
At NetDCUA5 we have two NI2C drivers and one I2C driver.
The usage is as follows:

Connector	driver
I2C0_SCL, I2C0_SDA	I2C1: (native I2C driver), connector J5
Touch	I2C3: (native I2C driver), on-board
Audio	I2C8: (I2C driver), on-board

Table 44: NetDCUA5 I2C driver usage

Default is “I2C1:” driver deactivated because J5 connector pins for NIC2 device are used by “SPI1:” driver.

15 PWM Driver

Implemented on: ASA5, PCA5, NDA5

armStoneA5 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.
NetDCUA5 and PicoCOMA5 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\armStoneA5\PWM]
[HKLM\Drivers\BuiltIn\NetDCUA5\PWM]
[HKLM\Drivers\BuiltIn\PicoCOMA5\PWM]
```

Required settings:

Key	Value	Comment
Prefix	String	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM. Default: PWM
Dll	String	Name of the DLL with the driver Default: fs_pwm.dll
Order	Dword:n	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…9	This value specifies the device index, a value from 0 through 9. Default: 1
Channel	Dword:n	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:n	Clock frequency Default: 3000Hz
Default	Dword:n	PWM value after loading of the driver. Default: 0
FriendlyName	String	"PWM driver for NetDCU"
Flags	Dword:4	4: Disabled from loading 0: Driver enabled Default: 4
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 45: PWM: Regisity

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:

On NetDCUA5 and armStoneA5 all PWM channels use the same frequency. The last loaded driver set the resulting value.

Table Channel armStoneA5:

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 46: PWM – armStoneA5 Channel

Table Channel NetDCUA5:

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 47: PWM – NetDCUA5 Channel

16 SD/MMC Driver

Implemented on: ASA5, NDA5, PCA5, PM1.2, PMA5

Platform supports SD/MMC driver. There will be a driver for external SD slot and one for internal (only ASA5/NDA5) SD slot.

The registry key for the on-board slot (armStoneA5, NetDCUA5) is:

```
[HKLM\Drivers\Builtin\SDMMC_CH1]
```

The registry key for the external slot (PicoCOMA5) is:

```
[HKLM\Drivers\Builtin\SDMMC_CH1]
```

Use the following parameters to configure the driver:

Key	Value	Comment
Prefix	SHC	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Dll	fs_sdhc.dll	Name of the DLL with the driver
Order	Dword:0x21	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.
UseCardAvailabel	Dword:<0|1>	Disables/enables the SW mode for card detection.If SW mode is disabled the card is detected by CD pin
CardAvailable	Dword:<0|1>	Disables/enables the card by SW
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 48: SD/MMC Driver Registry Settings

17 Native SPI Driver

Implemented on: ASA5, PCA5, NDA5, PM1.2, PMA5

Board supports native SPI driver.

The registry key for the driver is:

`[HKLM\Drivers\Builtin\SPI1<n>]`

Use the following parameters to configure the driver:

Key	Value	Comment
Prefix	SPI	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Dll	fs_nspi.dll	Name of the DLL with the driver
Order	Dword:0x41	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.
DriverMethod	Dword:1	0: IRQ 1: Polling 2+3: DMA Up to now, only Polling is implemented.
Flags	Dword:	4: Disabled from loading

Table 49: Native I2C: Registry settings

The full documentation of the driver can be found in document "WinCE_NSPI_eng.pdf".

18 Ethernet Driver

Implemented on: ASA5, NDA5, PCA5, PM1.2, PMA5

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 !! Not implemented !!
TxQueue	Dword:	Send Packet Mode. 0=OFF 1=ON Default: 1 !! Not implemented !!
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 !! Not implemented !!
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 !! Not implemented !!
BackPressure	Dword:	Back Pressure Function. 0=disable 1=enable Default:1 !! Not implemented !!
FlowControl	Dword:	Flow Control Function. 0=disable 1=enable Default:1 !! Not implemented !!
IPv4MulticastEnableAll	Dword:0|1	Set this value to 1 to enable receive of all multicast messages. Default: 0
CFHMaxCRCError	Dword:0	Maximal amount of CRC errors before adapter is reset. 0 means disabled. Default: 0
CFHMaxOverflowError	Dword:0	Maximal amount of Overrun errors before adapter is reset. 0 means disabled. Default: 0
CFHMaxCollisionError	Dword:0	Maximal amount of late collisions before adapter is reset. 0 means disabled. Default: 0
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 50: Ethernet Driver Registry Settings

19 Screen Saver Driver

Implemented on: ASA5, NDA5, PCA5, PM1.2, PMA5

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 51: PSS: Registry settings

20 Broadcast Driver

This driver is loaded during system start and sends a broadcast to the network including some device information. The broadcast message can be catched by the F&S tool FSDevcieSpy or by your own management application.

The registry key for the driver is:

```
[HKLM\Drivers\Drivers\Builtin\BCSend]
```

Use the following parameters to configure the driver:

Key	Value	Comment
Prefix	BCS	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Dll	fs_bcsend.dll	Name of the DLL with the driver
BroadcastCount		Amount of retries. Default: 20
BroadcastDelay		Time between two broadcasts in seconds. Default:6
DeviceName		Set this value to get a unique device name. Default: "Device"
DeviceInfo		Set this value to get unique device info. Default: "Info"

The structure of the broadcast package is as follow:

```
/* Broadcast information */
typedef struct tagBcastInfo
{
    char devname[64];
    char devinfo[64];
    char ident[64];
    USHORT wMAC[3];
} BCASTINFO, * PBCASTINFO;
```

Listing 19: Broadcast Driver: broadcast message

Note:
The driver sends the message to port 4242.

Beside the automatic send function during startup of the driver it is also possible to create a handle to the driver and call the IOCTL IOCTL_BCS_SEND_BROADCAST. The declaration of the IOCTL is in file fs_bcsend_sdk.h.

Example usage of IOCTL_BCS_SEND_BROADCAST:

```
#include "stdafx.h"

#include "fs_bcsend_sdk.h"


int _tmain(int argc, TCHAR *argv[], TCHAR *envp[])
{
    HANDLE hBCS;
    BCIOCTLINFO BCInfo;
    int arg, error;
    PTCHAR pszDevName = NULL;
    PTCHAR pszDevInfo = NULL;
    PTCHAR pszCount = NULL;
    PTCHAR pszDelay = NULL;

    error = FALSE;
    for (arg = 1; arg < argc; arg++)
    {
        if ((argv[arg][0] == '-') || (argv[arg][0] == '/'))
        {
            switch(toupper(argv[arg][1]))
            {
            case 'N':
                pszDevName = argv[++arg];
                break;
            case 'I':
                pszDevInfo = argv[++arg];
                break;
            case 'D':
                pszDelay = argv[++arg];
                break;
            case 'C':
                pszCount = argv[++arg];
                break;
            default:
                error = TRUE;
                break;
            }
        }
        else
        {
            error = TRUE;
        }

        if (error)
        {
            _ftprintf(stderr, _T("Illegal argument: \"%s\"\r\n"), argv[arg]);
            error = FALSE;
        }
    }

    memset(&BCInfo, 0, sizeof(BCIOCTLINFO));

    if (pszDevName)
    {
        if (!WideCharToMultiByte(CP_ACP, 0, pszDevName, -1, BCInfo.devname, 63,
                                 NULL, NULL))
            strcpy(BCInfo.devname, "Default Name");
    }
    else
    {
        strcpy(BCInfo.devname, "Default Name");
    }

    if (pszDevInfo)
    {
```

```
        if (!WideCharToMultiByte(CP_ACP, 0, pszDevInfo, -1, BCInfo.devinfo, 63,
                                 NULL, NULL))
            strcpy(BCInfo.devinfo, "Default Info");
    }
    else
    {
        strcpy(BCInfo.devinfo, "Default Info");
    }

    if (pszDelay)
    {
        if (!_stscanf(pszDelay, _T("%d"), &BCInfo.dwBCDelay))
            BCInfo.dwBCDelay = 5;
        if (BCInfo.dwBCDelay < 1)
            BCInfo.dwBCDelay = 1;
        if (BCInfo.dwBCDelay > 60)
            BCInfo.dwBCDelay = 60;
    }
    else
    {
            BCInfo.dwBCDelay = 5;
    }

    if (pszCount)
    {
        if (!_stscanf(pszCount, _T("%d"), &BCInfo.dwBCCount))
            BCInfo.dwBCCount = 10;
    }
    else
    {
        BCInfo.dwBCCount = 10;
    }

    hBCS = CreateFile(_T("BCS1:"),GENERIC_READ,0,NULL, OPEN_EXISTING,
                      FILE_ATTRIBUTE_NORMAL, NULL);
    if (hBCS != INVALID_HANDLE_VALUE)
    {
        DeviceIoControl(hBCS,IOCTL_BCS_SEND_BROADCAST, &BCInfo, sizeof(BCIOCTLINFO),
                        NULL, 0, NULL, NULL);
        CloseHandle(hBCS);
    }
    else
    {
        DWORD dwError;
        dwError = GetLastError();
    }
    return 0;
}
```

Listing 20: Broadcast Driver: Example BCIoctl

21 File System Filter

Purpose of this file system filter is to limit access to files or directories.

The registry key for the driver is:

```
[HKLM\System\StorageManager\Profiles\NANDFMD\Filters\FSDFilter]
```

For each filter rule you have to define a sub key under FSDFilter (i.e.FSDFIlter\1). There is a maximum of 10 filter rules.

Use the following parameters to configure one filter rule:

Key	Value	Comment
Path	String	Path of the file system object which should be protected.
Protect	Dword	Bit combined value of protection. Bit 0: FSDF_PROTECT_DELETE Bit 1: FSDF_PROTECT_RENAME Bit 2: FSDF_PROTECT_MOVE

Example:

```
[HKEY_LOCAL_MACHINE\System\StorageManager\Profiles\FFSDISK\Filters\FSDFilter]
       "Dll"="fsdfilter.dll"
       "Order"=dword:2
;       "Debug"=dword:fff

; These definition eliminates the possibility to reset the user hive by
; renaming its parent directory
[HKEY_LOCAL_MACHINE\System\StorageManager\Profiles\FFSDISK\Filters\FSDFilter\1]
        "Path"="\\documents and settings\\default"
        "Protect"=dword:7
```

Listing 21: File System Filter: Example

Note:
This filter is not included by default. It is also not possible to simply add it. It must be included in hive registry of a customer specific image. Therefore documentation is added for reference only.

22 File System Redirector

Purpose of this file system driver is to redirect access to files or directories. Enabling drive redirection lets you view and manage folders on local drives in a remote session.

The registry key for the driver is:

```
[HKLM\System\StorageManager\Autoload\FSDFilterRedir]
```

Use the following parameters to configure one filter rule:

Key	Value	Comment
Dll	FSDFilterRedir.dll	Name of the DLL with the driver
IsEnabled	Dword	Set to 0 to disable file system driver. Default: 0
FolderName	String	Name of the folder which will be created under root directory.
RootPath	String	This registry entry also determines the root of the filter driver. If you set RootPath to "\\", the whole file system comes under the scope of this filter. You can change this registry entry if you want to reduce the scope of the redirection filter.
MountFlags	Dword	Bit combined value: Bit 0: Specifies a hidden file system For more info read MSDN documentation.

Note:
This driver is not included by default. It is also not possible to simply add it. It must be included in hive registry of a customer specific image. Therefore documentation is added for reference only.

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.
Specific testing of all parameters of each device is not necessarily performed unless required by law or regulation.
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.
Specifications are subject to change without notice.

Warranty Terms

Hardware Warranties
F&S guarantees hardware products against defects in workmanship and material for a period of two (2) years from the date of shipment. Your sole remedy and F&S's sole liability shall be for F&S, at its sole discretion, to either repair or replace the defective hardware product at no charge or to refund the purchase price. Shipment costs in both directions are the responsibility of the customer. This warranty is void if the hardware product has been altered or damaged by accident, misuse or abuse.

Software Warranties
Software is provided "AS IS". F&S makes no warranties, either express or implied, with regard to the software object code or software source code either or with respect to any third party materials or intellectual property obtained from third parties. F&S makes no warranty that the software is useable or fit for any particular purpose. This warranty replaces all other warranties written or unwritten. F&S expressly disclaims any such warranties. In no case shall F&S be liable for any consequential damages.

Disclaimer of Warranty
THIS WARRANTY IS MADE IN PLACE OF ANY OTHER WARRANTY, WHETHER EXPRESSED, OR IMPLIED, OF MERCHANTABILITY, FITNESS FOR A SPECIFIC PURPOSE, NON-INFRINGEMENT OR THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION, EXCEPT THE WARRANTY EXPRESSLY STATED HEREIN. THE REMEDIES SET FORTH HEREIN SHALL BE THE SOLE AND EXCLUSIVE REMEDIES OF ANY PURCHASER WITH RESPECT TO ANY DEFECTIVE PRODUCT.

Limitation on Liability
UNDER NO CIRCUMSTANCES SHALL F&S BE LIABLE FOR ANY LOSS, DAMAGE OR EXPENSE SUFFERED OR INCURRED WITH RESPECT TO ANY DEFECTIVE PRODUCT. IN NO EVENT SHALL F&S BE LIABLE FOR ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES THAT YOU MAY SUFFER DIRECTLY OR INDIRECTLY FROM USE OF ANY PRODUCT. BY ORDERING THE PRODUCT, THE CUSTOMER APPROVES THAT THE F&S PRODUCT, HARDWARE AND SOFTWARE, WAS THOROUGHLY TESTED AND HAS MET THE CUSTOMER'S REQUIREMETS AND SPECIFICATIONS

Listings

Figures

Tables