Device Driver Documentation

Windows Embedded Compact for FSiMX6

Version 1.13
(2021-03-02)

Preliminary

This Document Is Subject to Change without Notice

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	All	A	3	Added description for registry value Resolution	WY
2015-02-05	1.02	efus	A	4.2	Added port description table for efus	HF
2015-02-05	1.02	All	M	10	Added output device values to LCD registry value Type	HF
2015-02-05	1.02	All	M	10	Added pre-defined modes for HDMI	HF
2015-02-05	1.02	All	M	10	Clarify usage of registry value OutputDevice	HF
2015-02-05	1.02	All	M	13	Additional information about I2C interfaces	HF
2015-03-05	1.03	All	A	10	Added description for parameter AccelLevel	HF
2015-03-15	1.03	All	A	2.1	Debug Message System	HF
2015-04-09	1.04	All	M	16	Added registry value IntPullUp and DrvStrength	HF
2015-04-09	1.04	All	M	13.2	Added picture of F&S I2C Bus test tool	HF
2015-06-18	1.05	Efus	A	16	Added Pin relation table for efus	MA
2015-06-18	1.05	All	A	22	Add FSStartup	MA
2015-06-26	1.05	All	A	23	CPU Core Temperature	ZU
2015-09-11	1.05	All	A,M	23	CEDDK Functions	MA
2015-09-28	1.06	All	M	*, 13	Add documentation for armStoneA9R2	HF
2015-10-05	1.06	All	A	23	Add CEDDK function to get board name	HF
2015-10-12	1.06	All	A	10	Add parameter VidMemLen and VideMemBase	HF
2016-01-20	1.06	All	A	24	Add documentation for High Resolution Timer	ZU
2016-03-21	1.06	All	A	7.3	Added description for TouchSamples, AdcReadHoldoffHns, SetDelay, TouchRate	AD
2016-03-22	1.06	All	A	14	Add description for i.MX6Solox PWM channels	HF
2016-06-29	1.06	QBlissA9, armStoneA9, PicoMODA9	A	5	Added UART Overviews	JG
2016-06-30	1.06	All	M	10	Add parameter LCD_BACKLIGHT	ZU
2016-09-26	1.07	All	A, M	7.3	Added description for PNDTPullUp, OPMode, ChannelsNr, Z1MinBound, Z1MaxBound, Z2MinBound, Z2MaxBound. Modified description for Threshold.	AD
2017-01-11	1.07	All	A	26	Added description for FCR	AD
2017-01-20	1.07	All	A	23	Added description for FSMinShell	HF
	1.08	All	M	25		
2017-03-27	1.09	efusA7UL, PicoCOM1.2	A	1, 4, 5, 13	Add documentation for PicoCOM1.2 and efus™A7UL	TM
2017-09-14	1.10	efusA9	A	7.4, 7.5	Add description for cap. touches SiS92XX, EXC3000	AD
2017-11-03	1.11	QBlissA9r2, NetDCUA9	A,M	1.1, 4.8, 4.9, 5.7, 5.3	Add documentation for QBliss™A9r2 and NetDCUA9	TM
2019-04-10	1.12	NetDCUA9	M	1	Add AIN channnels for NetDCUA9	
2019-05-02	1.12	NetDCUA9	A	10	Added link to NetDCUA9 - Display	JG
2021-03-02	1.13	PicoCOMA9X	A	4	Added PicoCOMA9X I/O Pin Table	JG

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

About this document

This is the device driver documentation for the F&S platform FSiMX6 based on 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 FSiMX6 are:

- armStone™A9
- armStone™A9R2
- efus™A9
- efus™A7UL
- NetDCUA9
- PicoCOM1.2
- PicoMODA9
- QBliss™A9
- QBliss™A9r2

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.

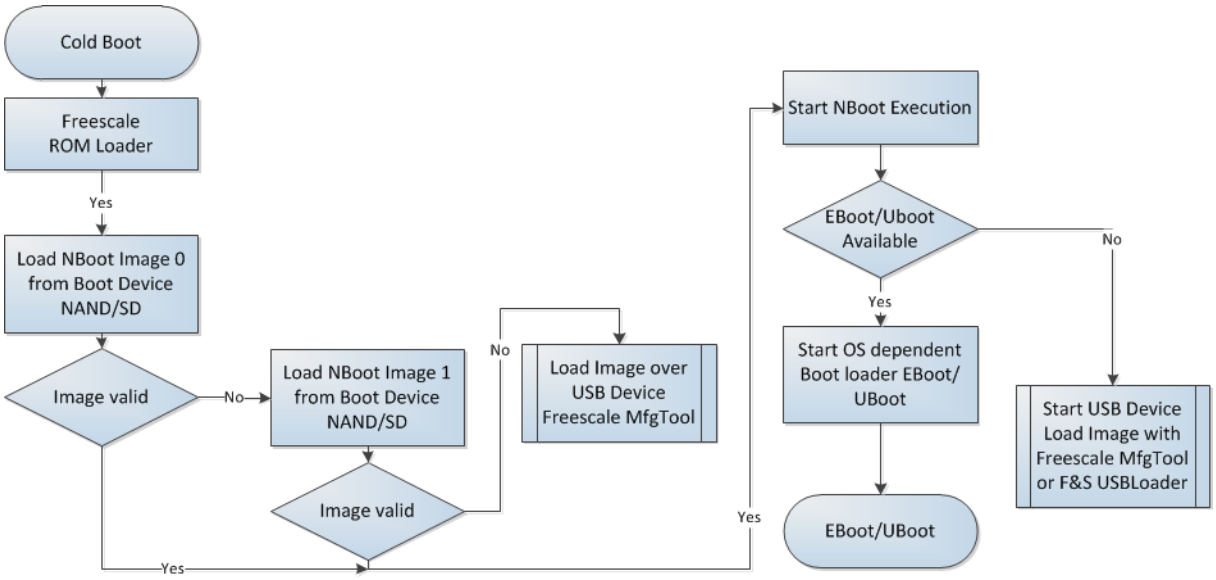

Figure 1: Boot Process

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=efusA9, 1=armStoneA9, 2=PicoMODA9, 3=QBlissA9, 4=armStoneA9R2, 6=QBlissA9r2, 7=NetDCUA9, 8=efusA9X, 9=PicoCOMA9X, 16=efusA7UL, 18=PicoCOM1.2
BoardName	String	i.e. efusA9, armStoneA9
BoardRevision	Dword	i.e. 100
BootVerMajor	Dword	Major version of EBoot loader
BootVerMinor	Dword	Minor version of EBoot loader
F3SSerial	Sting	Serial number for F3S
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 (IPP), CA9 Watchdog (WDOG), RESET, Software, JTAG
StepStone Loader, Version	Dword	Version of installed NBoot

2 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.

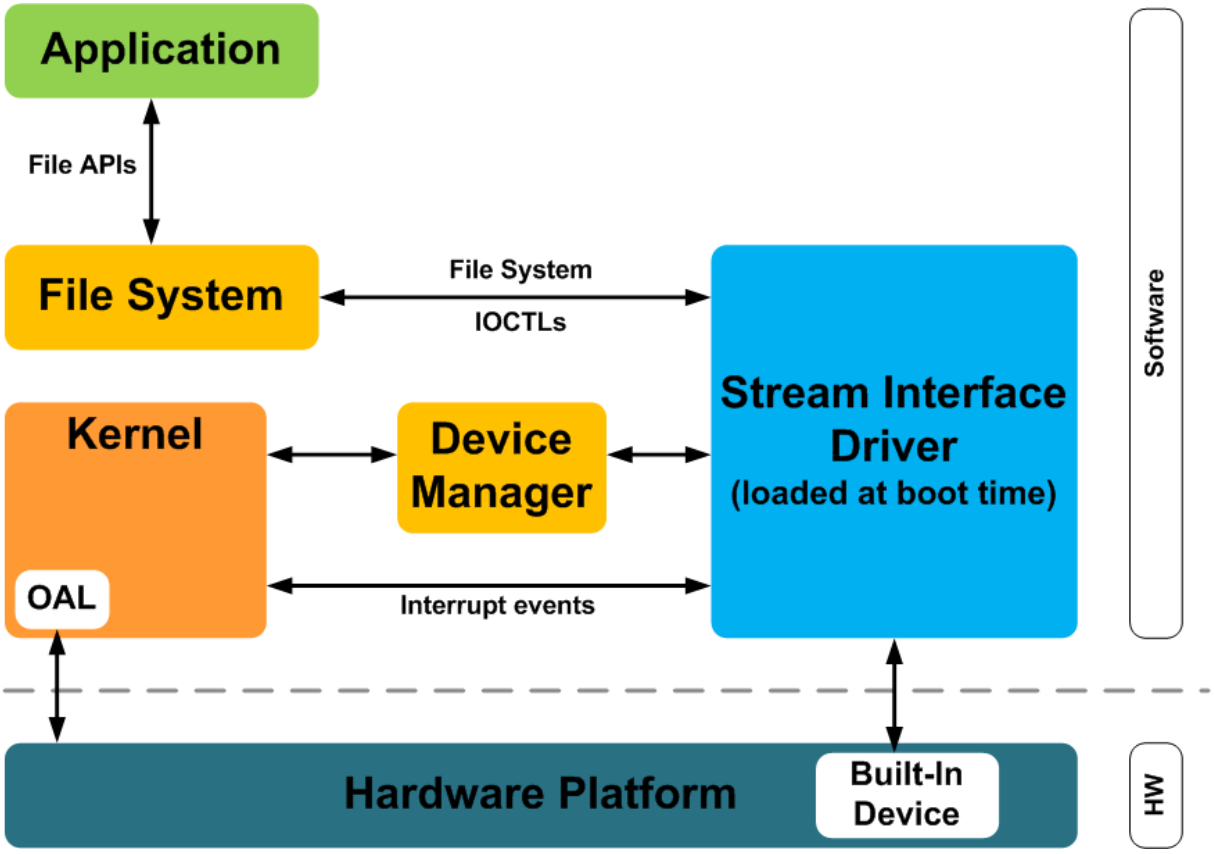

Figure 2: Windows CE: Stream Interface Driver Architecture

2.1 Debug Message System

Most of the drivers implement Windows Embedded Compact debug message system. You can control which debug messages are sent to the output stream by enabling or disabling a debug zone. Each driver has its own separate debug zones. If a zone is active, messages for that zone are sent to the output stream. If a zone is inactive, messages for that zone are suppressed. There are two ways manipulate debug zones:

2.1.1 Registry

1.) Start ndcucfg.exe
2.) Type: reg open \DebugZones
3.) Example DIO : reg set val DIO dword 0x000F
4.) Reg save

2.1.2 Target Control “shell.exe”

1.) Open a command window (cmd.exe). I.e. connect to the device by TELNET.
2.) Start target control : shell -c
3.) Get module index : gi mod
4.) Use the zo command to show or change debug zones

3 Analogue Input

Implemented on: ASA9, ASA9R2, NDA9

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\armStoneA9\ANALOGIN]
[HKLM\Drivers\BuiltIn\armStoneA9R2\ANALOGIN]
[HKLM\Drivers\BuiltIn\NetDCUA9\ANALOGIN]
```

Required settings:

Entry	Value	Description
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…9	This value specifies the device index. Default: 1
Flags	Dword:0	4: Disabled from loading
Rate	Dword:0…6	This value controls the data rate setting. See Table 4: Analogue Input: Registry parameter data rate. ASA9, ASA9R2 only
Channel	Dword:4…7	Number of the analogue channel. See Table Channel.
FullScale	Dword:0…5	This value configures the programmable gain amplifier. See Table 5: Analogue Input: Registry parameter FullScale. ASA9, ASA9R2 only
I2CDev	String	I2C device name.
I2CDevAddr	Dword:n	I2C device address of the external Analog-to-Digital Converter.
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 armStoneA9/armStoneA9R2:

Channel	Description
0x04	Reads value from analogue input 0 (Feature connector pin 29)
0x05	Reads value from analogue input 1 (Feature connector pin 31)
0x06	Reads value from analogue input 2 (Feature connector pin 33)
0x07	Reads value from analogue input 3 (Feature connector pin 35)

Table 2: Analogue Input: armStoneA9 Channel

Table Channel NetDCUA9

Channel	Description
0x10	Reads value from analogue input 0 (J7.11)
0x14	Reads value from analogue input 1 (J7.12)
0x18	Reads value from analogue input 2 (J7.9, optional, refer HW documentation)
0x1C	Reads value from analogue input 3 (J7.10, optional, refer HW documentation)

Table 3: Analogue Input: NetDCUA9 Channel

Registry parameter Rate (data rate):

Data Rate	Description
0x00	128 Samples per Second
0x01	250 Samples per Second
0x02	490 Samples per Second
0x03	920 Samples per Second
0x04	1600 Samples per Second
0x05	2400 Samples per Second
0x06	3300 Samples per Second

Table 4: Analogue Input: Registry parameter data rate

Registry parameter FullScale (gain):

Gain	Description
0x00	Full Scale = ±6.144V
0x01	Full Scale = ±4.096V
0x02	Full Scale = ±2.048V
0x03	Full Scale = ±1.204V
0x04	Full Scale = ±0.512V
0x05	Full Scale = ±0.256V

Table 5: Analogue Input: Registry parameter FullScale

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't 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 driver 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 settings

F. Set driver 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 settings

4 Digital I/O

Implemented on: ASA9, PMA9, QBA9, EFA9 EFA7UL, PCA9X, PC1.2, QBA9R2, NDCUA9

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:

Entry	Value	Description
Dll		Name of the DLL with the driver
Prefix	"DIO"	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
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.
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.
IRQCfg0 - or - IRQCfg0A IRQCfg0B IRQCfg0C IRQCfg0D *IRQCfg0x*	Dword:n	Interrupt configuration register 0.

Entry	Value	Description
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.

					armStoneA9							
Digital-IO				Pin	Function					capabilities	66 pin connector
IO-Pin	Port		Registry settings		*COM*	*I2C*	*SPI1 CAN*	*LCD*	*other*		
0	0	Port 0	0	UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1	0							
1	1		1		1							1
2	2		2		2							2
3	3		3		3					COL0	I/O/IRQ	3
4	4		4		4					COL1	I/O/IRQ	4
5	5		5		5					COL2	I/O/IRQ	5
6	6		6		6					COL3	I/O/IRQ	6
7	7		7		7					COL4	I/O/IRQ	7
8	0	Port 1	8		8					COL5	I/O/IRQ	8
9	1		9		9					COL6	I/O/IRQ	9
10	2		10		10					COL7	I/O/IRQ	10
11	3		11		11							11
12	4		12		12		SPI_CLK			COL11	I/O/IRQ	12
13	5		13		13	TXD1					I/O/IRQ	13
14	6		14		14		SPI_CS			COL10	I/O/IRQ	14
15	7		15		15	RXD1					I/O/IRQ	15
16	0	Port 2	16		16		CLK[*1)]	SPI_MOSI		COL9	I/O/IRQ	16
17	1		17		17		DAT[*1)]	SPI_MISO		COL8	I/O/IRQ	17
18	2		18		18		CLK1			ROW0	I/O/IRQ	18
19	3		19		19					ROW1	I/O/IRQ	19
20	4		20		20					ROW2	I/O/IRQ	20
21	5		21		21					ROW3	I/O/IRQ	21
22	6		22		22					ROW4	I/O/IRQ	22
23	7		23		23					ROW5	I/O/IRQ	23
24	0	Port 3	24		24					ROW6	I/O/IRQ	24
25	1		25		25					ROW7	I/O/IRQ	25
26	2		26		26		DAT1			XGPIO18	I/O/IRQ	26
27	3		27		27							27
28	4		28		28					PWM2	I/O/IRQ	28
29	5		29		29							29
30	6		30		30					PWM4	I/O/IRQ	30
31	7		31		31							31
32	0	Port 4	32		32					PWM1	I/O/IRQ	32
33	1		33		33							33
34	2		34		34				VCFL_ON		I/O/IRQ	34
35	3		35		35							35
36	4		36		36	RXD2						36
37	5		37		37							37
38	6		38		38	TXD2						38
39	7		39		39							39
40	0	Port 5	40		40							40
41	1		41		41							41
42	2		42		42							42
43	3		43		43							43
44	4		44		44							44
45	5		45		45							45
46	6		46		46							46

47	7		47		**47**							47
48	0	Port 6	48		**48**							48
49	1		49		**49**							49
50	2		50		**50**							50
51	3		51		**51**							51
52	4		52		**52**							52
53	5		53		**53**							53
54	6		54		**54**							54
55	7		55		**55**							55
56	0	Port 7	56		**56**							56
57	1		57		**57**							57
58	2		58		**58**							58
59	3		59		**59**							59
60	4		60		**60**							60
61	5		61		**61**							61
62	6		62		**62**							62
63	7		63		**63**			CANRX*2)				63
64	0	Port 8	64		**64**			CANTX*2)				64
65	1		65		**65**							65
66	2		66		**66**							66
67	3		67		**67**							
68	4		68		**68**							
69	5		69		**69**							
70	6		70		**70**							
71	7		71		**71**							

Table 7: Digital I/O pins – armStoneA9

*1): Soft I2C

*2): External Phy needed

4.2 Port description efus

On efus port number is equal to pin number. That means if you want to use pin 16 (SD_A_WP) as I/O, port number is 16.
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			Pin	Function				capabilities	SKIT connector J22
IO-Pin	Port	Registry settings		*COM / I2C / SPI / CAN*	*SD / MMC*	*LCD*	*other*		
0		0					N/A		
1	Port 0 0	1	1				5.0V		
2	Port 0 1	2	2				5.0V		
3	Port 0 2	3	3				5.0V		
4	Port 0 3	4	4				5.0V		
5	Port 0 4	5	5				5.0V		
6	Port 0 5	6	6				5.0V		
7	Port 0 6	7	7				GND		
8	Port 0 7	8	8				GND		
9	Port 1 0	9	9				VBAT		
10	Port 1 1	10	10				V33		
11	Port 1 2	11	11				ACOK		
12	Port 1 3	12	12				RESET_INn		
13	Port 1 4	13	13				N/C		
14	Port 1 5	14	14				RESET_OUTn		
15	Port 1 6	15	15				RXD_C		
16	Port 1 7	16	16		SD_A_WP			I/O/IRQ	
17	Port 2 0	17	17				TXD_C		
18	Port 2 1	18	18		SD_B_WP			I/O/IRQ	
19	Port 2 2	19	19	COM3: RTS				I/O/IRQ	
20	Port 2 3	20	20		SD_A_D2			I/O/IRQ	
21	Port 2 4	21	21	COM3: CTS				I/O/IRQ	
22	Port 2 5	22	22		SD_A_D3			I/O/IRQ	
23	Port 2 6	23	23				N/A		
24	Port 2 7	24	24		SD_A_CMD			I/O/IRQ	
25	Port 3 0	25	25				PWM1	I/O/IRQ	32
26	Port 3 1	26	26				V33		
27	Port 3 2	27	27				GND		
28	Port 3 3	28	28		SD_A_CLK			I/O/IRQ	
29	Port 3 4	29	29	CID1: TX				I/O/IRQ	
30	Port 3 5	30	30				GND		
31	Port 3 6	31	31	CID1: RX				I/O/IRQ	
32	Port 3 7	0	32		SD_A_D0			I/O/IRQ	
33	Port 4 0	1	33				GND		
34	Port 4 1	2	34		SD_A_D1			I/O/IRQ	
35	Port 4 2	3	35	CID2: TX				I/O/IRQ	55
36	Port 4 3	4	36				N/C		
37	Port 4 4	5	37	CID2: RX				I/O/IRQ	56

UseAsIO / DataDir / DataInit / IRQCfg

38		6	6	38				N/C		
39		7	7	39				GND		
40	Port 5	0	8	40				N/C		
41		1	9	41				PCIE_TXP		
42		2	10	42				N/C		
43		3	11	43				PCIE_TXN		
44		4	12	44				N/C		
45		5	13	45				GND		
46		6	14	46				GND		
47		7	15	47				PCIE_RXP		
48	Port 6	0	16	48				BOOTSELn		
49		1	17	49				PCIE_RXN		
50		2	18	50	SPI_B_MISO				I/O/IRQ	23
51		3	19	51				GND		
52		4	20	52	SPI_B_MOSI				I/O/IRQ	24
53		5	21	53				PCIE_CLKP		
54		6	22	54	SPI_B_SCLK				I/O/IRQ	25
55		7	23	55				PCIE_CLKN		
56	Port 7	0	24	56	SPI_B_CS1				I/O/IRQ	26
57		1	25	57				GND		
58		2	26	58	SPI_B_CS2				I/O/IRQ	27
59		3	27	59				MPCIE_PERST	I/O/IRQ	
60		4	28	60	SPI_B_IRQ1				I/O/IRQ	28
61		5	29	61				MPCIE_WAKE	I/O/IRQ	
62		6	30	62	SPI_B_IRQ2				I/O/IRQ	29
63		7	31	63				GND		
64	Port 8	0	0	64				GND		
65		1	1	65		SD_B_D2			I/O/IRQ	
66		2	2	66	SPI_A_MISO				I/O/IRQ	33
67		3	3	67		SD_B_D3			I/O/IRQ	
68		4	4	68	SPI_A_MOSI				I/O/IRQ	34
69		5	5	69		SD_B_CMD			I/O/IRQ	
70		6	6	70	SPI_A_SCLK				I/O/IRQ	35
71		7	7	71				V33		
72	Port 9	0	8	72	SPI_A_CS1				I/O/IRQ	36
73		1	9	73		SD_B_CLK			I/O/IRQ	
74		2	10	74	SPI_A_CS2				I/O/IRQ	37
75		3	11	75				GND		
76		4	12	76	SPI_A_IRQ1				I/O/IRQ	38
77		5	13	77		SD_B_D0			I/O/IRQ	
78		6	14	78	SPI_A_IRQ2				I/O/IRQ	39
79		7	15	79		SD_B_D1			I/O/IRQ	
80	Port 10	0	16	80				GND		
81		1	17	81		SD_B_WP			I/O/IRQ	
82		2	18	82	I2C2: DAT				I/O/IRQ	45
83		3	19	83		SD_B_CD			I/O/IRQ	
84		4	20	84	I2C2: CLK				I/O/IRQ	46
85		5	21	85				GND		
86		6	22	86	I2C2: IRQ				I/O/IRQ	48
87		7	23	87				BKLT_PWM	I/O/IRQ	

88	0	Port 11	24	88	I2C2: RST				I/O/IRQ	47
89	1		25	89			VCFL_ON		I/O/IRQ	
90	2		26	90				GND		
91	3		27	91				GND		
92	4		28	92	COM1: RXD				I/O/IRQ	
93	5		29	93			LCD_CLK		I/O/IRQ	
94	6		30	94	COM1: TXD				I/O/IRQ	
95	7		31	95				GND		
96	0	Port 12	0	96	COM4: RXD				I/O/IRQ	14
97	1		1	97			LCD_HSYNC		I/O/IRQ	
98	2		2	98	COM4: TXD				I/O/IRQ	16
99	3		3	99			LCD_VSYNC		I/O/IRQ	
100	4		4	100				GND		
101	5		5	101				GND		
102	6		6	102	COM2: RXD				I/O/IRQ	
103	7		7	103			LCD_R0		I/O/IRQ	
104	0	Port 13	8	104	COM2: TXD				I/O/IRQ	
105	1		9	105			LCD_R1		I/O/IRQ	
106	2		10	106	COM2: RTS				I/O/IRQ	
107	3		11	107			LCD_R2		I/O/IRQ	
108	4		12	108	COM2: CTS				I/O/IRQ	
109	5		13	109			LCD_R3		I/O/IRQ	
110	6		14	110				GND		
111	7		15	111			LCD_R4		I/O/IRQ	
112	0	Port 14	16	112				I2S_MCLK	I/O/IRQ	
113	1		17	113			LCD_R5		I/O/IRQ	
114	2		18	114				GND		
115	3		19	115				GND		
116	4		20	116				I2S_LRCLK	I/O/IRQ	
117	5		21	117			LCD_G0		I/O/IRQ	
118	6		22	118				GND		
119	7		23	119			LCD_G1		I/O/IRQ	
120	0	Port 15	24	120				I2S_SCLK	I/O/IRQ	
121	1		25	121			LCD_G2		I/O/IRQ	
122	2		26	122				GND		
123	3		27	123			LCD_G3		I/O/IRQ	
124	4		28	124				I2S_DOUT	I/O/IRQ	
125	5		29	125			LCD_G4		I/O/IRQ	
126	6		30	126				I2S_DIN	I/O/IRQ	
127	7		31	127			LCD_G5		I/O/IRQ	
128	0	Port 16	0	128				GND		
129	1		1	129				GND		

130			2	130	I2C3: DAT					
131			3	131			LCD_B0		I/O/IRQ	
132			4	132	I2C3: CLK					
133			5	133			LCD_B1		I/O/IRQ	
134			6	134				V33		
135			7	135			LCD_B2		I/O/IRQ	
136		Port 17	8	136				GND		
137			9	137			LCD_B3		I/O/IRQ	
138			10	138				HDMI_DATA2_P		
139			11	139			LCD_B4		I/O/IRQ	
140			12	140				HDMI_DATA2_N		
141			13	141			LCD_B5		I/O/IRQ	
142			14	142				HDMI_DATA1_P		
143			15	143				GND		
144		Port 18	16	144				HDMI_DATA1_N		
145			17	145			LCD_DE		I/O/IRQ	
146			18	146				HDMI_DATA0_P		
147			19	147				GND		
148			20	148				HDMI_DATA0_N		
149			21	149			VLCD_ON		I/O/IRQ	
150			22	150				HDMI_CLK_P		
151			23	151	I2C1: DAT				I/O/IRQ	41
152		Port 19	24	152				HDMI_CLK_N		
153			25	153	I2C1: IRQ				I/O/IRQ	44
154			26	154				GND		
155			27	155	I2C1: CLK				I/O/IRQ	42
156			28	156				HDMI_DDCCEC	I/O/IRQ	
157			29	157	I2C1: RST				I/O/IRQ	43
158			30	158				HDMI_HPD		
159			31	159				GND		
160		Port 20	0	160				GND		
161			1	161				CAMINT_YD9/D0_N	I/O/IRQ	
162			2	162				N/C		
163			3	163				CAMINT_YD8/D0_P	I/O/IRQ	
164			4	164				N/C		
165			5	165				CAMINT_YD2/D1_N	I/O/IRQ	
166			6	166				N/C		
167			7	167				CAMINT_YD1/D1_P	I/O/IRQ	
168		Port 21	8	168				N/C		
169			9	169				CAMINT_YD3/D2_N	I/O/IRQ	

170	2		10	170				N/C		
171	3		11	171				CAMINT_YD0/D2_P	I/O/IRQ	
172	4		12	172				GND		
173	5		13	173				CAMINT_YD/D3_N	I/O/IRQ	
174	6		14	174				N/C		
175	7		15	175				CAMINT_PCLK/D3_P	I/O/IRQ	
176	0	Port 22	16	176				N/C		
177	1		17	177				CAMINT_YD5/CLK_N	I/O/IRQ	
178	2		18	178				N/C		
179	3		19	179				CAMINT_YD6/CLK_P	I/O/IRQ	
180	4		20	180				N/C		
181	5		21	181				GND		
182	6		22	182				N/C		
183	7		23	183				CAMINT_MCLK	I/O/IRQ	
184	0	Port 23	24	184				GND		
185	1		25	185				GND		
186	2		26	186				N/C		
187	3		27	187				CAMINT_YD7	I/O/IRQ	
188	4		28	188				ETH_A_D4-		
189	5		29	189				VCAM		
190	6		30	190				ETH_A_D4+		
191	7		31	191				CAMINT_HREF	I/O/IRQ	
192	0	Port 24	0	192				ETH_A_LED_LINK		
193	1		1	193				CAMINT_PWDN	I/O/IRQ	
194	2		2	194				ETH_A_D3-		
195	3		3	195				CAMINT_VSYNC	I/O/IRQ	
196	4		4	196				ETH_A_D3+		
197	5		5	197				I2C_C_CAMRST	I/O/IRQ	
198	6		6	198				V33		
199	7		7	199				GND		
200	0	Port 25	8	200				ETH_A_D2-		
201	1		9	201				SATA_RX_P		
202	2		10	202				ETH_A_D2+		
203	3		11	203				SATA_RX_N		
204	4		12	204				ETH_A_LED_ACT		
205	5		13	205				SATA_TX_N		
206	6		14	206				ETH_A_D1-		
207	7		15	207				SATA_TX_P		
208	0	Port 26	16	208				ETH_A_D1+		
209	1		17	209				GND		

210	2		18	210				GND		
211	3		19	211				N/C		
212	4		20	212				USB_PWRON	I/O/IRQ	
213	5		21	213				N/C		
214	6		22	214				USB_A_N		
215	7		23	215				GND		
216	0	Port 27	24	216				USB_A_P		
217	1		25	217				USB_DEV_VBUS		
218	2		26	218				GND		
219	3		27	219				USB_DEV_PWR_ON	I/O/IRQ	
220	4		28	220				N/C		
221	5		29	221				USB_DEV_OC	I/O/IRQ	
222	6		30	222				N/C		
223	7		31	223				USB_DEV_ID	I/O/IRQ	
224	0	Port 28	0	224				GND		
225	1		1	225				USB_DEV_N		
226	2		2	226				N/C		
227	3		3	227				USB_DEV_P		
228	4		4	228				N/C		
229	5		5	229				GND		
230	6		6	230				GND		
231	7		7	231				GND		
232	0	Port 29	8	232				GND		
233	1		9							
234	2		10							
235	3		11							
236	4		12							
237	5		13							
238	6		14							
239	7		15							

to old rev. exchanged pins

N/A - not available

N/C - not connected

Table 8: Digital I/O pins- efus

4.3 Port description efusA7UL

The port numbering of efusA7UL is equal to pin number of connector “goldfinger connector”. That means if you want to use pin 1 as I/O, port number is 1.
The efusA7UL connector has a total of 232 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.

				efusTMA7UL									
Digital-IO				efus-Pin	GPIO	PIO-Pin	Function				capabilities		SKIT connector J22
IO-Pin	Port	Registry Settings	UseAsIO / DataDir / DataInit / IRQCfg				COM / I2C / SPI / CAN	SD / MMC	LCD	other			
0	Port 0	0		0						N/A			
1		1		1						5.0V		1	
2		2		2						5.0V		2	
3		3		3						5.0V		3	
4		4		4						5.0V		4	
5		5		5						5.0V		5	
6		6		6						5.0V		6	
7		7		7						GND		7	
8	Port 1	8		8						GND		8	
9		9		9						VBAT		9	
10		10		10						V33		10	
11		11		11						ACOK		11	
12		12		12						RESET_INn		12	
13		13		13						N/C		13	
14		14		14						RESET_OUTn		14	
15		15		15	GPIO4_IO22	CSI_DATA1	RXD_C				I/O/IRQ	15	
16	Port 2	16		16	GPIO1_IO18	UART1_CTS		SD_A_WP			I/O/IRQ	16	
17		17		17	GPIO4_IO21	CSI_DATA0	TXD_C				I/O/IRQ	17	
18		18		18	GPIO1_IO19	UART1_RTS		SD_A_CD			I/O/IRQ	18	
19		19		19	GPIO4_IO23	CSI_DATA02	RTS_C				I/O/IRQ	19	
20		20		20	GPIO2_IO20	SD1_DATA2		SD_A_D2			I/O/IRQ	20	
21		21		21	GPIO4_IO24	CSI_DATA03	CTS_C				I/O/IRQ	21	
22		22		22	GPIO2_IO21	SD1_DATA3		SD_A_D3			I/O/IRQ	22	
23		23		23						N/A		23	
24	Port 3	24		24	GPIO2_IO16	SD1_CMD		SD_A_CMD			I/O/IRQ	24	
25		25		25	GPIO1_IO05	GPIO1_IO05				PWM1	I/O/IRQ	25	32
26		26		26						V33		26	
27		27		27						GND		27	
28		28		28	GPIO2_IO17	SD1_CLK		SD_A_CLK			I/O/IRQ	28	
29		29		29	GPIO1_IO26	UART3_CTS	CAN_A_TX				I/O/IRQ	29	
30		30		30						GND		30	
31		31		31	GPIO1_IO27	UART2_RTS	CAN_A_RX				I/O/IRQ	31	
32	Port 4	0		32	GPIO2_IO18	SD1_DATA0		SD_A_D0			I/O/IRQ	32	
33		1		33						GND		33	
34		2		34	GPIO2_IO19	SD1_DATA1		SD_A_D1			I/O/IRQ	34	
35		3		35	GPIO1_IO22	UART2_CTS	CAN_B_TX				I/O/IRQ	35	55
36		4		36						N/C		36	
37		5		37	GPIO1_IO23	UART2_RTS	CAN_B_RX				I/O/IRQ	37	56
38		6		38						N/C		38	
39		7		39						GND		39	
40	Port 5	8		40						N/C		40	

41		1	9	41						N/C		41		
42		2	10	42						N/C		42		
43		3	11	43						N/C		43		
44		4	12	44						N/C		44		
45		5	13	45						GND		45		
46		6	14	46						GND		46		
47		7	15	47						N/C		47		
48		0	16	48						BOOTSELn		48		
49		1	17	49						N/C		49		
50		2	18	50	**GPIO1_IO31**	UART5_RXD	SPI_B_MISO				I/O/IRQ	50	23	
51		3	19	51						GND		51		
52	Port 6	4	20	52	**GPIO1_IO30**	UART5_TXD	SPI_B_MOSI				I/O/IRQ	52	24	
53		5	21	53						N/C		53		
54		6	22	54	**GPIO1_IO28**	UART4_TXD	SPI_B_SCLK				I/O/IRQ	54	25	
55		7	23	55						N/C		55		
56		0	24	56	**GPIO1_IO29**	UART4_RXD	SPI_B_CS1				I/O/IRQ	56	26	
57		1	25	57						GND		57		
58		2	26	58						N/C		58	27	
59		3	27	59						N/C		59		
60	Port 7	4	28	60	**GPIO5_IO07**	SNVS_TMPR7	SPI_B_IRQ1					60	28	
61		5	29	61						N/C		61		
62		6	30	62						N/C		62	29	
63		7	31	63						GND		63		
64		0	0	64						GND		64		
65		1	1	65	**GPIO3_IO27**	LCD_DATA22		SD_B_D2			I/O/IRQ	65		
66		2	2	66	**GPIO4_IO28**	CSI_DATA07	SPI_A_MISO				I/O/IRQ	66	33	
67		3	3	67	**GPIO3_IO28**	LCD_DATA23		SD_B_D3			I/O/IRQ	67		
68	Port 8	4	4	68	**GPIO4_IO27**	CSI_DATA06	SPI_A_MOSI				I/O/IRQ	68	34	
69		5	5	69	**GPIO3_IO23**	LCD_DATA18		SD_B_CMD			I/O/IRQ	69		
70		6	6	70	**GPIO4_IO25**	CSI_DATA04	SPI_A_SCLK				I/O/IRQ	70	35	
71		7	7	71						V33		71		
72		0	8	72	**GPIO4_IO26**	CSI_DATA05	SPI_A_CS1				I/O/IRQ	72	36	
73		1	9	73	**GPIO3_IO24**	LCD_DATA19		SD_B_CLK			I/O/IRQ	73		
74		2	10	74						N/C		74	37	
75		3	11	75						GND		75		
76	Port 9	4	12	76	**GPIO5_IO06**	SNVS_TMPR6	SPI_A_IRQ1				I/O/IRQ	76	38	
77		5	13	77	**GPIO3_IO25**	LCD_DATA20		SD_B_D0			I/O/IRQ	77		
78		6	14	78								78	39	
79		7	15	79	**GPIO1_IO26**	LCD_DATA21		SD_B_D1			I/O/IRQ	79		
80		0	16	80						GND		80		
81		1	17	81						N/C		81		
82		2	18	82	**GPIO4_IO19**	CSI_VSYNC	I2C_B_DAT				I/O/IRQ	82	45	
83		3	19	83						N/C		83		
84	Port 10	4	20	84	**GPIO4_IO20**	CSI_HSYNC	I2C_B_CLK				I/O/IRQ	84	46	
85		5	21	85						GND		85		
86		6	22	86	**GPIO5_IO1**	SNVS_TMPR1	I2C_B_IRQ				I/O/IRQ	86	48	
87		7	23	87	**GPIO1_IO08**	GPIO1_IO08				BKLT_PWM	I/O/IRQ	87		
88		0	24	88	**GPIO5_IO3**	SNVS_TMPR3	I2C_B_RST				I/O/IRQ	88	47	
89		1	25	89	**GPIO5_IO05**	SNVS_TMPR5			VCFL_ON		I/O/IRQ	89		
90		2	26	90						GND		90		
91		3	27	91						GND		91		
92	Port 11	4	28	92	**GPIO1_IO17**	UART1_RXD	RXD_A				I/O/IRQ	92		
93		5	29	93	**GPIO3_IO00**	LCD_CLK			LCD_CLK		I/O/IRQ	93		
94		6	30	94	**GPIO1_IO16**	UART1_TXD	TXD_A				I/O/IRQ	94		
95		7	31	95						GND		95		
96	Port 12	0	0	96	**GPIO4_IO18**	CSI_PIXCLK	RXD_D				I/O/IRQ	96	14	

97	1		1	97	GPIO3_IO2	LCD_HSYNC				LCD_HSYNC		I/O/IRQ	97	
98	2		2	98	GPIO4_IO17	CSI_MCLK	TXD_D					I/O/IRQ	98	16
99	3		3	99	GPIO3_IO3	LCD_VSYNC				LCD_VSYNC		I/O/IRQ	99	
100	4		4	100							GND		100	
101	5		5	101							GND		101	
102	6		6	102	GPIO1_IO21	UART2_RXD	RXD_B					I/O/IRQ	102	
103	7		7	103	GPIO3_IO05	LCD_DATA00				LCD_R0		I/O/IRQ	103	
104	0	Port 13	8	104	GPIO1_IO20	UART2_TXD	TXD_B					I/O/IRQ	104	
105	1		9	105	GPIO3_IO06	LCD_DATA01				LCD_R1		I/O/IRQ	105	
106	2		10	106	GPIO1_IO25	UART3_RXD	RTS_B					I/O/IRQ	106	
107	3		11	107	GPIO3_IO07	LCD_DATA02				LCD_R2		I/O/IRQ	107	
108	4		12	108	GPIO1_IO24	UART3_TXD	CTS_B					I/O/IRQ	108	
109	5		13	109	GPIO3_IO08	LCD_DATA03				LCD_R3		I/O/IRQ	109	
110	6		14	110							GND		110	
111	7		15	111	GPIO3_IO09	LCD_DATA04				LCD_R4		I/O/IRQ	111	
112	0	Port 14	16	112	GPIO1_IO11	JTAG_TMS					I2S_MCLK	I/O/IRQ	112	
113	1		17	113	GPIO3_IO10	LCD_DATA05				LCD_R5		I/O/IRQ	113	
114	2		18	114							GND		114	
115	3		19	115							GND		115	
116	4		20	116	GPIO1_IO12	JTAG_TDO					I2S_LRCLK	I/O/IRQ	116	
117	5		21	117	GPIO3_IO11	LCD_DATA06				LCD_G0		I/O/IRQ	117	
118	6		22	118							GND		118	
119	7		23	119	GPIO3_IO12	LCD_DATA7				LCD_G1		I/O/IRQ	119	
120	0	Port 15	24	120	GPIO1_IO13	JTAG_TDI					I2S_SCLK	I/O/IRQ	120	
121	1		25	121	GPIO3_IO13	LCD_DATA8				LCD_G2		I/O/IRQ	121	
122	2		26	122							GND		122	
123	3		27	123	GPIO3_IO14	LCD_DATA9				LCD_G3		I/O/IRQ	123	
124	4		28	124	GPIO1_IO15	JTAG_TRST_B					I2S_DOUT	I/O/IRQ	124	
125	5		29	125	GPIO3_IO15	LCD_DATA10				LCD_G4		I/O/IRQ	125	
126	6		30	126	GPIO3_IO14	JTAG_TCK					I2S_DIN	I/O/IRQ	126	
127	7		31	127	GPIO3_IO16	LCD_DATA11				LCD_G5		I/O/IRQ	127	
128	0	Port 16	0	128							GND		128	
129	1		1	129							GND		129	
130	2		2	130	GPIO5_IO08	SNVS_TMPR8	I2C_C_DAT					I/O/IRQ	130	
131	3		3	131	GPIO3_IO17	LCD_DATA12				LCD_B0		I/O/IRQ	131	
132	4		4	132	GPIO5_IO09	SNVS_TMPR9	I2C_C_CLK					I/O/IRQ	132	
133	5		5	133	GPIO3_IO18	LCD_DATA13				LCD_B1		I/O/IRQ	133	
134	6		6	134							V33		134	
135	7		7	135	GPIO3_IO19	LCD_DATA14				LCD_B2		I/O/IRQ	135	
136	0	Port 17	8	136							GND		136	
137	1		9	137	GPIO3_IO20	LCD_DATA15				LCD_B3		I/O/IRQ	137	
138	2		10	138							N/C		138	
139	3		11	139	GPIO3_IO21	LCD_DATA16				LCD_B4		I/O/IRQ	139	
140	4		12	140							N/C		140	
141	5		13	141	GPIO3_IO22	LCD_DATA17				LCD_B5		I/O/IRQ	141	
142	6		14	142							N/C		142	
143	7		15	143							GND		143	
144	0	Port 18	16	144							N/C		144	
145	1		17	145	GPIO3_IO01	LCD_ENABLE				LCD_DE		I/O/IRQ	145	
146	2		18	146							N/C		146	
147	3		19	147							GND		147	
148	4		20	148							N/C		148	
149	5		21	149	GPIO5_IO4	SNVS_TMPR4				VLCD_ON		I/O/IRQ	149	
150	6		22	150							N/C		150	
151	7		23	151	GPIO1_IO03	GPIO1_IO03	I2C_A_DAT					I/O/IRQ	151	41

152	Port 19 0	24	152						N/C		152	
153	1	25	153	**GPIO5_IO00**	SNVS_TMPR0	I2C_A_IRQ				I/O/IRQ	153	44
154	2	26	154						GND		154	
155	3	27	155	**GPIO1_IO02**	GPIO1_IO02	I2C_A_CLK				I/O/IRQ	155	42
156	4	28	156						N/C		156	
157	5	29	157	**GPIO5_IO02**	SNVS_TMPR2	I2C_A_RST				I/O/IRQ	157	43
158	6	30	158						N/C		158	
159	7	31	159						GND		159	
160	Port 20 0	0	160						GND		160	
161	1	1	161						N/C		161	
162	2	2	162						N/C		162	
163	3	3	163						N/C		163	
164	4	4	164						N/C		164	
165	5	5	165						N/C		165	
166	6	6	166						ETH_B_LED_LINK		166	
167	7	7	167						N/C		167	
168	Port 21 0	8	168						N/C		168	
169	1	9	169						N/C		169	
170	2	10	170						N/C		170	
171	3	11	171						N/C		171	
172	4	12	172						N/C		172	
173	5	13	173						N/C		173	
174	6	14	174						ETH_B_RX-		174	
175	7	15	175						N/C		175	
176	Port 22 0	16	176						ETH_B_RX+		176	
177	1	17	177						N/C		177	
178	2	18	178						N/C		178	
179	3	19	179						N/C		179	
180	4	20	180						ETH_B_TX-		180	
181	5	21	181						GND		181	
182	6	22	182						ETH_B_TX+		182	
183	7	23	183						N/C		183	
184	Port 23 0	24	184						GND		184	
185	1	25	185						GND		185	
186	2	26	186						N/C		186	
187	3	27	187						N/C		187	
188	4	28	188						N/C		188	
189	5	29	189						N/C		189	
190	6	30	190						N/C		190	
191	7	31	191						N/C		191	
192	Port 24 0	0	192						ETH_A_LED_LINK		192	
193	1	1	193						N/C		193	
194	2	2	194						N/C		194	
195	3	3	195						N/C		195	
196	4	4	196						N/C		196	
197	5	5	197						N/C		197	
198	6	6	198						V33		198	
199	7	7	199						GND		199	
200	Port 25 0	8	200						ETH_A_RX-		200	
201	1	9	201						N/C		201	
202	2	10	202						ETH_A_RX+		202	
203	3	11	203						N/C		203	
204	4	12	204						ETH_A_LED_ACT		204	
205	5	13	205						N/C		205	
206	6	14	206						ETH_A_TX-		206	

207	7	15	207						N/C		207	
208	0	16	208						ETH_A_TX+		208	
209	1	17	209						GND		209	
210	2	18	210						GND		210	
211	3	19	211						N/C		211	
212	4	20	212						USB_PWRON		212	
213	5	21	213						N/C		213	
214	6	22	214						USB_A_N		214	
215	7	23	215						GND		215	
216	0	24	216						USB_A_P		216	
217	1	25	217						USB_DEV_VBUS		217	
218	2	26	218						GND		218	
219	3	27	219	**GPIO1_IO04**	GPIO1_IO04				USB_DEV_PWR_ON	I/O/IRQ	219	
220	4	28	220						N/C		220	
221	5	29	221	**GPIO1_IO01**	GPIO1_IO01				USB_DEV_OC	I/O/IRQ	221	
222	6	30	222						N/C		222	
223	7	31	223	**GPIO1_IO00**	GPIO1_IO00				USB_DEV_ID	I/O/IRQ	223	
224	0	0	224						GND		224	
225	1	1	225						USB_DEV_P		225	
226	2	2	226						N/C		226	
227	3	3	227						USB_DEV_N		227	
228	4	4	228						N/C		228	
229	5	5	229						GND		229	
230	6	6	230						GND		230	
231	7	7	231						GND		231	
232	0	8	232						GND		232	
233	1	9										
234	2	10										
235	3	11										
236	4	12										
237	5	13										
238	6	14										
239	7	15										

Port 26 | Port 27 | Port 28 | Port 29

N/A - not available
N/C - not connected

Table 9: Digital I/O pins- efus™A7UL

4.4 Port Description PicoCOMA9X

PicoCOMA9X														
Digital-IO			PCA9X-Pin	PIO-Pin	Function							capabilities	SKIT-Pin (J10)
IO-Pin	Port	Registry settings			COM	I2C	SPI1 +CAN	USB	SD/MMC	LCD	other		
0	Port 0	UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1	13	GPIO4_IO30	TXD2							I/O/IRQ	
1			14	GPIO4_IO29	RXD2							I/O/IRQ	
2			15	GPIO4_IO25	RTS2/TXD3 [3)]							I/O/IRQ	
3			16	GPIO4_IO24	CTS2/RXD3 [3)]							I/O/IRQ	
4			17	GPIO7_IO01	TXD1							I/O/IRQ	
5			18	GPIO7_IO05	RXD1							I/O/IRQ	
6			23					OTGVBUS			~~EINT4~~		
7			24	GPIO1_IO12				PWR			EINT8	I/O/IRQ	
8	Port 1		26	GPIO2_IO11			MISO0					I/O/IRQ	3
9			27	GPIO2_IO16			PCS0					I/O/IRQ	4
10			28	GPIO2_IO10			SPCK0					I/O/IRQ	5
11			29	GPIO2_IO15			MOSI0				EINT1	I/O/IRQ	6
12			32	GPIO7_IO08		SDA						I/O/IRQ	9
13			33	GPIO7_IO06		SCL						I/O/IRQ	10
14			34	GPIO6_IO08					DAT0			I/O/IRQ	
15			35	GPIO6_IO09					DAT1			I/O/IRQ	
16	Port 2		36	GPIO6_IO10					DAT2			I/O/IRQ	
17			37	GPIO6_IO11					DAT3			I/O/IRQ	
18			38	GPIO6_IO06					CLK			I/O/IRQ	
19			39	GPIO6_IO07					CMD			I/O/IRQ	
20			40	GPIO1_IO01							EINT2	I/O/IRQ	11
21			41	GPIO1_IO13							PWM1	I/O/IRQ	12
22			43	GPIO3_IO14						R1 (R3)		I/O/IRQ	
23			44	GPIO3_IO15						R2 (R4)		I/O/IRQ	
24	Port 3		45	GPIO3_IO16						R3 (R5)		I/O/IRQ	
25			46	GPIO3_IO17						R4 (R6)		I/O/IRQ	
26			47	GPIO3_IO18						R5 (R7)		I/O/IRQ	
27			48	GPIO3_IO07						G0 (G2)		I/O/IRQ	
28			49	GPIO3_IO08	~~TXD3~~					G1 (G3)		I/O/IRQ	
29			50	GPIO3_IO09	~~RXD3~~					G2 (G4)		I/O/IRQ	
30			51	GPIO3_IO10			~~PCS1~~			G3 (G5)		I/O/IRQ	
31			52	GPIO3_IO11			~~PCS2~~			G4 (G6)		I/O/IRQ	

32			**53**	GPIO3_IO12	~~CTS1~~						G5 (G7)		I/O/IRQ	7
33			**54**	GPIO3_IO02							B1 (B3)	~~AIN0~~	I/O/IRQ	8
34			**55**	GPIO3_IO03							B2 (B4)	~~AIN1~~	I/O/IRQ	-
35		**Port 4**	**56**	GPIO3_IO04							B3 (B5)	~~AIN2~~	I/O/IRQ	-
36			**57**	GPIO3_IO05							B4 (B6)		I/O/IRQ	
37			**58**	GPIO3_IO06							B5 (B7)		I/O/IRQ	
38			**59**	GPIO3_IO00							B6		I/O/IRQ	
39			**60**	GPIO3_IO25							B7		I/O/IRQ	
40			**63**	GPIO3_IO26							**HSYNC** / B0 (B2)		I/O/IRQ	
41			**64**	GPIO3_IO28							**VSYNC** / R0 (R2)		I/O/IRQ	
42			**65**	GPIO3_IO22							VEEK		I/O/IRQ	
43		**Port 5**	**66**	GPIO3_IO27							VLCD-ON		I/O/IRQ	
44			**67**	GPIO3_IO24							VCFL-ON		I/O/IRQ	
45			**68**	GPIO3_IO23							VCD-DEN		I/O/IRQ	
46			**69**	GPIO7_IO04	RTS1								I/O/IRQ	13
47			**11**	GPIO7_IO00	CTS1							EINT 1	I/O/IRQ	1
48			**12**	GPIO1_IO06						CD			I/O/IRQ	2
49			**30**	GPIO1_IO01		SDA	**CANTX**						I/O/IRQ	
50		**Port 6**	**31**	GPIO1_IO00		SCL	**CANRX**						I/O/IRQ	
51														
52														
53														
46														
47														

4.5 Port description PicoCOM1.2

				PicoCOM1.2							
Digital-IO			PC12-Pin	PIO-Pin	CPU-Pad Name	Function			capabilities	SKIT-Pin (J10)
IO-Pin	Port	Registry settings				*COM / I2C / SPI / CAN*	*SD/ MMC*	*other*		
0	Port 0 0	0	13	GPIO1_IO20	UART2_TX_DATA	COM2 TXD			I/O/IRQ	
1	Port 0 1	1	14	GPIO1_IO21	UART2_RX_DATA	COM2 RXD			I/O/IRQ	
2	Port 0 2	2	15	GPIO1_IO22	UART2_CTS_B	COM2 RTS			I/O/IRQ	
3	Port 0 3	3	16	GPIO1_IO23	UART2_RTS_B	COM2 CTS			I/O/IRQ	
4	Port 0 4	4	17	GPIO1_IO24	UART3_TX_DATA	COM1 TXD			I/O/IRQ	
5	Port 0 5	5	18	GPIO1_IO25	UART3_RX_DATA	COM1 RXD			I/O/IRQ	
6	Port 0 9	6	23					USB CNX		
7	Port 0 7	7	24					USB PWR		
8	Port 1 0	8	32	GPIO1_IO31	UART5_RX_DATA	I2C1 SDA			I/O/IRQ	21
9	Port 1 1	9	33	GPIO1_IO30	UART5_TX_DATA	I2C1 SCL			I/O/IRQ	22
10	Port 1 2	10	34	GPIO3_IO25	LCD1_DATA20		SD_DAT0		I/O/IRQ	
11	Port 1 3	11	35	GPIO3_IO26	LCD1_DATA21		SD_DAT1		I/O/IRQ	
12	Port 1 4	12	36	GPIO3_IO27	LCD1_DATA22		SD_DAT2		I/O/IRQ	
13	Port 1 5	13	37	GPIO3_IO28	LCD1_DATA23		SD_DAT3		I/O/IRQ	
14	Port 1 9	14	38	GPIO3_IO24	LCD1_DATA19		SD_CLK		I/O/IRQ	
15	Port 1 7	15	39	GPIO3_IO23	LCD1_DATA18		SD_CMD		I/O/IRQ	
16	Port 2 0	16	40	GPIO1_IO19	UART1_RTS_B			IRQ1	I/O/IRQ	6
17	Port 2 1	17	41	GPIO1_IO18	UART1_CTS_B			GPIO5	I/O/IRQ	
18	Port 2 2	18	43	GPIO1_IO16	UART1_TX_DATA	COM3 TXD			I/O/IRQ	
19	Port 2 3	19	44	GPIO1_IO17	UART1_RX_DATA	COM3 RXD			I/O/IRQ	
20	Port 2 4	20	45	GPIO3_IO00	LCD1_CLK	COM4 TXD			I/O/IRQ	
21	Port 2 5	21	46	GPIO3_IO01	LCD1_ENABLE	COM4 RXD			I/O/IRQ	4
22	Port 2 9	22	47	GPIO3_IO02	LCD1_HSYNC	COM4 RTS			I/O/IRQ	
23	Port 2 7	23	48	GPIO3_IO03	LCD1_VSYNC	COM4 CTS			I/O/IRQ	
24	Port 3 0	24	49	GPIO1_IO26	UART3_CTS_B	COM3 RTS			I/O/IRQ	
25	Port 3 1	25	50	GPIO1_IO27	UART3_RTS_B	COM3 CTS			I/O/IRQ	5
26	Port 3 2	26	55	GPIO4_IO28	CSI_DATA07	SPI1 MISO			I/O/IRQ	15
27	Port 3 3	27	56	GPIO4_IO27	CSI_DATA06	SPI1 MOSI			I/O/IRQ	16
28	Port 3 4	28	57	GPIO4_IO24	CSI_DATA03	SPI1 SCLK			I/O/IRQ	17
29	Port 3 5	29	58	GPIO4_IO26	CSI_DATA05	SPI1 PCS0			I/O/IRQ	18
30	Port 3 9	30	59	GPIO3_IO10	LCD1_DATA05	SPI1 PCS1			I/O/IRQ	19
31	Port 3 7	31	60	GPIO3_IO11	LCD1_DATA06	SPI1 PCS2			I/O/IRQ	20
32	Port 4 0	0	63	GPIO4_IO21	CSI_DATA00			GPIO6	I/O/IRQ	7
33	Port 4 1	1	64	GPIO4_IO22	CSI_DATA01			GPIO7	I/O/IRQ	8
34	Port 4 2	2	65	GPIO4_IO24	CSI_DATA03			GPIO8	I/O/IRQ	9
35	Port 4 3	3	66	GPIO4_IO23	CSI_DATA02			GPIO9	I/O/IRQ	10
36	Port 4 4	4	67	GPIO3_IO21	LCD1_DATA16			GPIO10	I/O/IRQ	11
37	Port 4 5	5	68	GPIO3_IO22	LCD1_DATA17			GPIO11	I/O/IRQ	12
38	Port 4 9	6	69	GPIO3_IO12	LCD1_DATA07			GPIO12	I/O/IRQ	13
39	Port 4 7	7	70	GPIO1_IO04	GPIO1_IO04			GPIO13	I/O/IRQ	14
40	Port 5 0	8	74	GPIO1_IO08	GPIO1_IO08			ADC1	I/O/IRQ	1
41	Port 5 1	9	75	GPIO1_IO05	GPIO1_IO05			ADC2	I/O/IRQ	2
42	Port 5 2	10	76	GPIO1_IO09	GPIO1_IO09			ADC3	I/O/IRQ	3
43	Port 5 3	11	11	GPIO1_IO28	UART4_TX_DATA				I/O/IRQ	23
44	Port 5 4	12	12	GPIO1_IO29	UART4_RX_DATA				I/O/IRQ	24
45	Port 5 5	13								
46	Port 5 9	14								
47	Port 5 7	15								

UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1

Table 10: Digital I/O pins- PicoCOM1.2

4.6 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

4.7 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	--		-						
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		-						

UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1

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		72		72	X21	GPH3_1						WDOUT, IRQ

73	**1**	**Port 9**	**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**		-						

4.8 Port description QBlissA9r2

Digital-IO			QBliss-Pin	Pico-ITX	GPIO-Pin	PAD-Name	Function				capabilities
IO-Pin	Port	Registry settings					*COM / I2C / SPI / CAN*	*SD/ MMC*	*LCD*	*other*	
0	Port 0 0	0	0								
1	Port 0 1	1	1								
2	Port 0 2	2	2								
3	Port 0 3	3	3								
4	Port 0 4	4	4								
5	Port 0 5	5	5								
6	Port 0 9	6	6								
7	Port 0 7	7	7								
8	Port 1 0	8	8								
9	Port 1 1	9	9								
10	Port 1 2	10	10								
11	Port 1 3	11	11								
12	Port 1 4	12	12								
13	Port 1 5	13	13								
14	Port 1 9	14	14								
15	Port 1 7	15	15								
16	Port 2 0	16	16		GPIO3_IO01	PAD_EIM_DA1				SUS_S5#	
17	Port 2 1	17	17	X21	GPIO3_IO09	PAD_EIM_DA9				GPI (WAKE#)	I/O/IRQ
18	Port 2 2	18	18		GPIO3_IO00	PAD_EIM_DA0				SUS_S3#	
19	Port 2 3	19	19		GPIO3_IO02	PAD_EIM_DA2				SUS_STAT#	
20	Port 2 4	20	20	X21	GPIO3_IO03	PAD_EIM_DA3				PWRBTN#	I/O/IRQ
21	Port 2 5	21	21	X21	GPIO3_IO05	PAD_EIM_DA5				SLP_BTN#	I/O/IRQ
22	Port 2 9	22	22	X21	GPIO3_IO04	PAD_EIM_DA4				LID_BTN#	I/O/IRQ
23	Port 2 7	23	23							GND	
24	Port 3 0	24	24							GND	
25	Port 3 1	25	25							GND	
26	Port 3 2	26	26							PWGIN (N/A)	
27	Port 3 3	27	27		GPIO3_IO06	PAD_EIM_DA6				GPI (BATLOW#)	
28	Port 3 4	28	28							RSTBTN#	
29	Port 3 5	29	29							SATA0_TX+	
30	Port 3 9	30	30							SATA1_TX+ (N/A)	
31	Port 3 7	31	31							SATA0_TX-	
32	Port 4 0	0	32							SATA1_TX- (N/A)	
33	Port 4 1	1	33		GPIO3_IO13	PAD_EIM_DA13				SATA_ACT#	
34	Port 4 2	2	34							GND	
35	Port 4 3	3	35							SATA0_RX+	
36	Port 4 4	4	36							SATA1_RX+ (N/A)	
37	Port 4 5	5	37							SATA0_RX-	
38	Port 4 9	6	38							SATA1_RX- (N/A)	
39	Port 4 7	7	39							GND	
40	Port 5 0	8	40							GND	
41	Port 5 1	9	41							BOOTSEL#	
42	Port 5 2	10	42	X17	GPIO1_IO10	PAD_SD2_CLK		SDIO_CLK		SDIO_CLK#	I/O/IRQ
43	Port 5 3	11	43	X17	GPIO1_IO04	PAD_GPIO_4		SDIO_CD		SDIO_CD#	I/O/IRQ
44	Port 5 4	12	44							SDIO_LED (N/A)	
45	Port 5 5	13	45	X17	GPIO1_IO11	PAD_SD2_CMD		SDIO_CMD		SDIO_CMD	I/O/IRQ

UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1

46		14		46	X17	GPIO1_IO02	PAD_GPIO_2		SDIO_WP		SDIO_WP	I/O/IRQ
47	7	15		47	X17	GPIO3_IO14	PAD_EIM_DA14		SDIO_PWR		SDIO_PWR#	I/O/IRQ
48	0	16		48	X17	GPIO1_IO14	PAD_SD2_DAT1		SDIO_DAT1		SDIO_DAT1	I/O/IRQ
49	1	17		49	X17	GPIO1_IO15	PAD_SD2_DAT0		SDIO_DAT0		SDIO_DAT0	I/O/IRQ
50	2	18		50	X17	GPIO1_IO12	PAD_SD2_DAT3		SDIO_DAT3		SDIO_DAT3	I/O/IRQ
51	3	19		51	X17	GPIO1_IO13	PAD_SD2_DAT2		SDIO_DAT2		SDIO_DAT2	I/O/IRQ
52	4	20		52					SDIO_DAT5 (N/A)			
53	5	21		53					SDIO_DAT4 (N/A)			
54	9	22		54					SDIO_DAT7 (N/A)			
55	7	23		55					SDIO_DAT6 (N/A)			
56	0	24		56		GPIO3_IO22	PAD_EIM_D22				USB_OTG_PEN	I/O/IRQ
57	1	25		57							GND	
58	2	26		58							GND	
59	3	27		59		GPIO5_IO16	PAD_DISP0_DAT22				I2S_WS	I/O/IRQ
60	4	28		60		GPIO4_IO12	PAD_KEY_COL3				GP1_I2C_CLK	I/O/IRQ
61	5	29		61		GPIO4_IO13	PAD_KEY_ROW3				I2S_RST#	I/O/IRQ
62	9	30		62		GPIO5_IO12	PAD_DISP0_DAT18				GP1_I2C_DAT	I/O/IRQ
63	7	31		63		GPIO5_IO14	PAD_DISP0_DAT20				I2S_CLK	I/O/IRQ
64	0	0		64	X21	GPIO1_IO30	PAD_ENET_TXD0				SMB_ALERT#	I/O/IRQ
65	1	1		65		GPIO5_IO17	PAD_DISP0_DAT23				I2S_SDI	I/O/IRQ
66	2	2		66	X19	GPIO1_IO3	PAD_GPIO_3				GP0_I2C_CLK	I/O/IRQ
67	3	3		67		GPIO5_IO15	PAD_DISP0_DAT21				I2S_SDO	I/O/IRQ
68	4	4		68	X19	GPIO1_IO11	PAD_GPIO_16				GP0_I2C_DAT	I/O/IRQ
69	5	5		69	TP13	GPIO3_IO07	PAD_EIM_DA7				THRM#	I/O/IRQ
70	9	6		70	X21	GPIO3_IO11	PAD_EIM_DA11				WDTRIG#	I/O/IRQ
71	7	7		71							THRMTRIP#	
72	0	8		72	X21	GPIO3_IO10	PAD_EIM_DA10				WDOUT	I/O/IRQ
73	1	9		73							GND	
74	2	10		74							GND	
75	3	11		75							USB_P7- (NA)	
76	4	12		76							USB_P6- (NA)	
77	5	13		77							USB_P7+ (NA)	
78	9	14		78							USB_P6+ (NA)	
79	7	15		79							USB_6_7_OC# (N/A)	
80	0	16		80							USB_4_5_OC#	
81	1	17		81							USB_P5- (N/A)	
82	2	18		82							USB_P4-	
83	3	19		83							USB_P5+ (N/A)	
84	4	20		84							USB_P4+	
85	5	21		85							USB_2_3_OC#	
86	9	22		86							USB_0_1_OC#	
87	7	23		87							USB_P3-	
88	0	24		88							USB_P2-	
89	1	25		89							USB_P3+	
90	2	26		90							USB_P2+	
91	3	27		91							USB_VBUS	
92	4	28		92		GPIO1_IO24	PAD_ENET_RX_ER				USB_ID	
93	5	29		93							USB_P1-	

94		30	94							USB_P0-		
95	9	31	95							USB_P1+		
96	7	0	96							USB_P0+		
97	0	1	97							GND		
98	1	2	98							GND		
99	2	3	99							LVDS_A0+		
100	3	4	100							LVDS_B0+		
101	Port 12	5	101							LVDS_A0-		
102	4	6	102							LVDS_B0-		
103	5	7	103							LVDS_A1+		
104	9	8	104							LVDS_B1+		
105	7	9	105							LVDS_A1-		
106	0	10	106							LVDS_B1-		
107	1	11	107							LVDS_A2+		
108	2	12	108							LVDS_B2+		
109	Port 13	13	109							LVDS_A2-		
110	3	14	110							LVDS_B2-		
111	4	15	111		GPIO2_IO11	PAD_SD4_DAT3				LVDS_PPEN	I/O/IRQ	
112	5	16	112		GPIO2_IO08	PAD_SD4_DAT0				LVDS_BLEN	I/O/IRQ	
113	9	17	113							LVDS_A3+		
114	7	18	114							LVDS_B3+		
115	0	19	115							LVDS_A3-		
116	1	20	116							LVDS_B2-		
117	Port 14	21	117							GND		
118	2	22	118							GND		
119	3	23	119							LVDS_A_CLK+		
120	4	24	120							LVDS_B_CLK+		
121	5	25	121							LVDS_A_CLK-		
122	9	26	122							LVDS_B_CLK-		
123	7	27	123		GPIO2_IO09	PAD_SD4_DAT1				LVDS_BLT_CTRL		
124	0	28	124		GPIO5_IO02	PAD_EIM_A25				GP_1-Wire_Bus	I/O/IRQ	
125	1	29	125		GPIO2_IO15	PAD_SD4_DAT7				GP2_I2C_DAT	I/O/IRQ	
126	2	30	126		GPIO5_IO08	PAD_DISP0_DAT14				LVDS_BLC_DAT	I/O/IRQ	
127	Port 15	31	127		GPIO2_IO14	PAD_SD4_DAT6				GP2_I2C_CLK	I/O/IRQ	
128	3	0	128		GPIO5_IO09	PAD_DISP0_DAT15				LVDS_BLC_CLK	I/O/IRQ	
129	4	1	129		GPIO4_IO10	PAD_KEY_COL2	CAN0_TX			CAN0_TX		
130	5	2	130		GPIO4_IO11	PAD_KEY_ROW2	CAN0_RX			CAN0_RX		
131	9	3	131							TMDS_CLK+		
132	7	4	132							USB_SSTX1+ (N/A)		
133	0	5	133							TMDS_CLK-		
134	1	6	134							USB_SSTX1- (N/A)		
135	Port 16	7	135							GND		
136	2	8	136							GND		
137	3	9	137							TMDS_LAN1+		
138	4	10	138							DP_AUX+ (N/A)		
139	5	11	139							TMDS_LAN1-		
140	9	12	140							DP_AUX- (N/A)		
141	7	13	141							GND		
142	Port 17	14	142							GND		
143	0	15	143							TMDS_LANE0+		
144	1	16	144							USB_SSRX1- (N/A)		
145	2	17	145							TMDS_LAN0-		
146	Port 18	18	146							USB_SSRX1+ (N/A)		

147		3	19	147								GND	
148		4	20	148								GND	
149		5	21	149								TMDS_LANE2+	
150		9	22	150		GPIO5_IO10	PAD_DISP0_DAT16					HDMI_CTRL_DAT	I/O/IRQ
151		7	23	151								TMDS_LANE2-	
152		0	24	152		GPIO5_IO11	PAD_DISP0_DAT17					HDMI_CTRL_CLK	I/O/IRQ
153		1	25	153								DP_HDP# (N/A)	
154		2	26	154								DP_HDMI_HPD#	
155	Port 19	3	27	155								PCIE_CLK_REF+	
156		4	28	156		GPIO6_IO16	PAD_NANDF_CS3					PCIE_WAKE#	
157		5	29	157								PCIE_CLK_REF-	
158		9	30	158		GPIO6_IO14	PAD_NANDF_CS1					PCIE_RST#	
159		7	31	159								GND	
160		0	0	160								GND	
161		1	1	161	X19	GPIO3_IO25	PAD_EIM_D25					PCIE3_TX+ (N/A)	I/O/IRQ
162		2	2	162	X19	GPIO3_IO31	PAD_EIM_D31					PCIE3_RX+ (N/A)	I/O/IRQ
163		3	3	163	X19	GPIO3_IO24	PAD_EIM_D24					PCIE3_TX- (N/A)	I/O/IRQ
164	Port 20	4	4	164	X19	GPIO3_IO30	PAD_EIM_D30					PCIE3_RX- (N/A)	I/O/IRQ
165		5	5	165								GND	
166		9	6	166								GND	
167		7	7	167								PCIE2_TX+ (N/A)	
168		0	8	168								PCIE2_RX+ (N/A)	
169		1	9	169								PCIE2_TX- (N/A)	
170		2	10	170								PCIE2_RX- (N/A)	
171		3	11	171		GPIO3_IO26	PAD_EIM_D26					UART0_TX	
172	Port 21	4	12	172		GPIO3_IO28	PAD_EIM_D28					UART0_RTS#	
173		5	13	173								PCIE1_TX+ (N/A)	
174		9	14	174								PCIE1_RX+ (N/A)	
175		7	15	175								PCIE1_TX- (N/A)	
176		0	16	176								PCIE1_RX- (N/A)	
177		1	17	177		GPIO3_IO27	PAD_EIM_D27					UART0_RX	
178		2	18	178		GPIO3_IO29	PAD_EIM_D29					UART0_CTS#	
179		3	19	179								PCIE0_TX+	
180	Port 22	4	20	180								PCIE0_RX+	
181		5	21	181								PCIE0_TX-	
182		9	22	182								PCIE0_RX-	
183		7	23	183								GND	
184		0	24	184								GND	
185		1	25	185	X5	GPIO4_IO09	PAD_KEY_ROW1					ESPI_IO_0	I/O/IRQ
186		2	26	186	X5	GPIO4_IO08	PAD_KEY_COL1					ESPI_IO_1	I/O/IRQ
187		3	27	187	X5	GPIO4_IO14	PAD_KEY_COL4					ESPI_IO_2	I/O/IRQ
188	Port 23	4	28	188	X5	GPIO4_IO15	PAD_KEY_ROW4					ESPI_IO_3	I/O/IRQ
189		5	29	189								ESPI_CK (N/A)	
190		9	30	190								ESPI_CS0# (N/A)	
191		7	31	191								ESPI_CS0# (N/A)	
192		0	0	192								LPC_LDRQ# (N/A)	
193		1	1	193								VCC_RTC	
194		2	2	194		GPIO4_IO29	PAD_DISP0_DAT8					GP_PWM_OUT2	I/O/IRQ
195	Port 24	3	3	195	X21	GPIO3_IO12	PAD_EIM_DA12					FAN_TACHOIN	I/O/IRQ
196		4	4	196	X21	GPIO2_IO10	PAD_SD4_DAT2					GP_PWM_OUT1	I/O/IRQ
197		5	5	197								GND	
198		9	6	198								GND	

199	Port 25 7	7	199	X19	GPIO2_IO24	PAD_EIM_CS1					SPI_MOSI	I/O/IRQ
200	Port 25 0	8	200		GPIO2_IO26	PAD_EIM_RW					SPI_CS0#	I/O/IRQ
201	Port 25 1	9	201	X19	GPIO2_IO25	PAD_EIM_OE					SPI_MISO	I/O/IRQ
202	Port 25 2	10	202		GPIO2_IO27	PAD_EIM_LBA					SPI_CS1#	I/O/IRQ
203	Port 25 3	11	203	X19	GPIO2_IO23	PAD_EIM_CS0					SPI_SCK	I/O/IRQ
204	Port 25 4	12	204								MFG_NC4 (N/A)	
205	Port 25 5	13	205								VCC_5V_SB	
206	Port 25 9	14	206								VCC_5V_SB	
207	Port 25 7	15	207								MFG_NC0 (N/A)	
208	Port 26 0	16	208		GPIO4_IO07	PAD_KEY_ROW0	UART4_RXD				MFG_NC2	
209	Port 26 1	17	209		GPIO4_IO06	PAD_KEY_COL0	UART4_TXD				MFG_NC1	
210	Port 26 2	18	210								MFG_NC3 (N/A)	
211	Port 26 3	19	211								NC1 (N/A)	
212	Port 26 4	20	212								NC2 (N/A)	
213	Port 26 5	21	213								NC3 (N/A)	
214	Port 26 9	22	214								NC4 (N/A)	
215	Port 26 7	23	215								NC5 (N/A)	
216	Port 27 0	24	216								NC6 (N/A)	
217	Port 27 1	25	217								NC7 (N/A)	
218	Port 27 2	26	218								NC8 (N/A)	
219	Port 27 3	27	219								VCC9	
220	Port 27 4	28	220								VCC10	
221	Port 27 5	29	221								VCC11	
222	Port 27 9	30	222								VCC12	
223	Port 27 7	31	223								VCC13	
224	Port 28 0	0	224								VCC14	
225	Port 28 1	1	225								VCC15	
226	Port 28 2	2	226								VCC16	
227	Port 28 3	3	227								VCC17	
228	Port 28 4	4	228								VCC18	
229	Port 28 5	5	229								VCC19	
230	Port 28 9	6	230								VCC20	
231	Port 28 7	7	231									

4.9 Port description NetDCUA9

Digital-IO					NetDCUA9									
IO-Pin	Port		Registry settings		J5-Pin	PIO-Pin	Function							capabilities
							COM	I2C	SPI1 +CAN	USB	SD/ MMC	LCD	other	
0	Port 0 0	Port 0	0	UseAsIO / DataDir / DataInit / IRQCfg0 / IRQCfg1	9	EIM_A16							ROW0	I/O/IRQ
1	Port 0 1		1		8	EIM_A17							ROW1	I/O/IRQ
2	Port 0 2		2		7	EIM_A18							ROW2	I/O/IRQ
3	Port 0 3		3		6	EIM_A19							ROW3	I/O/IRQ
4	Port 0 4		4		5	EIM_A20							ROW4	I/O/IRQ
5	Port 0 5		5		4	EIM_A21							ROW5	I/O/IRQ
6	Port 0 6		6		3	EIM_A22							ROW6	I/O/IRQ
7	Port 0 7		7		2	EIM_A23							ROW7	I/O/IRQ
8	Port 1 0	Port 1	8		15	CSI0_DAT4			CLK					I/O/IRQ
9	Port 1 1		9		13	CSI0_DAT7			CS					I/O/IRQ
10	Port 1 2		10		11	CSI0_DAT5		SCL	MOSI					I/O/IRQ

11	3		11		**10**	CSI0_DAT6		SDA	MISO					I/O/IRQ
12	4		12											
13	5		13											
14	6		14											
15	7		15											
16	0	Port 2	16		**24**	EIM_A24							COL0	I/O/IRQ
17	1		17		**23**	EIM_A25							COL1	I/O/IRQ
18	2		18		**22**	EIM_EB2							COL2	I/O/IRQ
19	3		19		**21**	EIM_EB3							COL3	I/O/IRQ
20	4		20		**20**	EIM_BCLK							COL4	I/O/IRQ
21	5		21		**19**	EIM_WAIT							COL5	I/O/IRQ
22	6		22		**18**	EIM_EB0							COL6	I/O/IRQ
23	7		23		**17**	EIM_EB1							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**									

4.10 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

4.11 Programming example

Headerfile:

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

Listing 7: 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 8: Digital I/O: Open a port

G. 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 9: Digital I/O: write data to port

H. 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 10: Digital I/O: changing the port

I. 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 11: Digital I/O: Access individual pin

J. 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 12: Digital I/O: Using Interrupts

K. Closing port

```
CloseHandle(hDIO);
```

Listing 13: Digital I/O: Closing port

5 Driver for Serial I/O (UART)

Implemented on: ASA9, PMA9, QBA9, EFA9

I.MX6 boards have a maximum of five serial ports (UART). Following communication settings are supported by the driver (data bits, parity, stop bit(s)):
(7,N,1), (7,N,2), (8,N,1), (8,N,2).
Installation of the driver is done by setting some registry values under the following registry key:

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

Settings:

Entry	Type	Description
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~~

Table 13: 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.~~

~~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 armStoneA9

armStone										
	COM1 (Debug)				COM2					
Signal	*RXA/Xu_RXD2*	*TXA/Xu_TXD2*	*CTSA/CTS2*	*RTSA/RTS2*	*Xu_RXD1*	*Xu_TXD1*	~~*INTG0Xu_CTS1*~~	~~*INTG1/Xu_RTS1*~~	RXC/Xu_RXD3	TXC/Xu_TXD3
Connector	X15-55	X15-57	X15-58	X15-56	X15-15	X15-13	~~X19-13~~	~~X19-15~~	X15-36	X15-38
armStoneA9	UART2				UART1				UART3	

Table 14: UART – Overview armStoneA9

5.2 UART Overview efusA7UL

efus™A7UL												
	COM2				COM3 (RS485)				COM1 (Debug)		COM4	
SKIT-Signal	*RXD_B*	*TXD_B*	*CTS_B*	*RTS_B*	*RXD_C*	*TXD_C*	*CTS_C*	*RTS_C*	*RXD_A*	*TXD_A*	*RXD_D*	*TXD_D*
SKIT-Connector	J15-3	J15-5	J15-6	J15-4	RS485+ J16-5	RS485- J16-6			J14-3	J14-5	J22-14	J22-16
Connector-Pin	102	104	108	106	15	17	21	19	92	94	96	98
efusA7UL	UART2				UART5				UART1		UART6	

Table 15: UART – Overview efusA7UL

5.3 UART Overview NetDCUA9

NetDCUA9										
	COM1 (Debug)		COM2				COM3		COM3 (RS485)	
SKIT-Signal	*RXD0*	*TXD0*	*RXD1*	*TXD1*	*CTS1*	*RTS1*	*RXD2*	*TXD2*	*RS485+*	*RS485-*
SKIT-Connector	J1E-B2	J1E-B3	J1E-A2	J1E-A3	J1E-A8	J1E-A7	J5A-9	J5A-10	NC	NC
Connector-Pin	J5-12	J5-14	J1-4	J1-6	J1-5	J1-3	J5-9	J5-10	J1-20	J1-24
NetDCUA9	UART4		UART2				UART5 (optional)		UART5 (optional)	

Table 16: UART – Overview NetDCUA9

5.4 UART Overview PicoMODA9

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	

Table 17: UART – Overview PicoMODA9

5.5 UART Overview PicoCOM1.2

PicoCOM1.2										
	COM1				COM2(RS485)				COM3 (Debug)	
SKIT-Signal	*RXD0*	*TXD0*	*CTS0*	*RTS0*	*RXD1*	*TXD1*	*CTS1*	*RTS1*	*RXD2*	*TXD2*
SKIT-Connector	J6-3	J6-5	J6-6	J6-4	RS485+ J8-3	RS485- J8-5			J7-3	J7-5
Connector-Pin	14	13	16	15	18	17	48	49	44	43
PicoCOM1.2	UART3				UART2				UART1	

Table 18: UART – Overview PicoCOM1.2

5.6 UART Overview QBlissA9

QBliss								
	COM1 (Debug)				COM2			
SKIT-Signal	*XuRXD0*	*XuTXD0*	*XuCTS0*	*XuRTS0*	*Xu_RXD1*	*Xu_TXD1*	*INTG0Xu_CTS1*	*INTG1/Xu_RTS1*
SKIT-Connector	X5-3	X5-5	X5-6	X5-4	X19-11	X19-9	X19-13	X19-15
Connector-Pin	186 (LPC_AD0)	185 (LPC_AD1)	187 (LPC_AD2)	188 (LPC_AD3)	161 (PCIE3_TX+)	163 (PCIE3_TX-)	162 (PCIE3_RX+)	164 (PCIE3_RX-)
					177 (EXCD0_CPPE#)	171 (EXCD0_PERST#)	178 (EXCD1_CPPE#)	172 (EXCD1_PERST#)
QBlissA9	UART2				UART3 (optional)			

Table 19: UART – Overview QBlissA9

5.7 UART Overview QBlissA9r2

QBlissA9r2								
	COM1 (Debug)				COM2			
SKIT-Signal	*XuRXD0*	*XuTXD0*	*XuCTS0*	*XuRTS0*	*Xu_RXD1*	*Xu_TXD1*	*INTG0Xu_CTS1*	*INTG1/Xu_RTS1*
SKIT-Connector	X5-3	X5-5	X5-6	X5-4	X19-11	X19-9	X19-13	X19-15
Connector-Pin	186 (LPC_AD0)	185 (LPC_AD1)	187 (LPC_AD2)	188 (LPC_AD3)	161 (PCIE3_TX+)	163 (PCIE3_TX-)	162 (PCIE3_RX+)	164 (PCIE3_RX-)
					177 (EXCD0_CPPE#)	171 (EXCD0_PERST#)	178 (EXCD1_CPPE#)	172 (EXCD1_PERST#)
QBlissA9	**UART5**				**UART3 (optional)**			

Table 20: UART – Overview QBlissA9r2

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]
```

Entry	Type	Description
Type	dword:1	See Table 22: 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 21: 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 22: Martix Keyboard: Type registry value

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

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

Entry	Type	Description
IOCol0	Dword	Number of IO-Pin Pin (see Chapter 4 Digital I/O) you want use for column 0. See Table 30: Matrix Keyboard: Connector J1
…		
IOCol*n*	Dword	Number of IO you want use for last column. See Table 30: Matrix Keyboard: Connector J1

Table 23: 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]
```

Entry	Type	Description
IORow0	Dword	Number of IO-Pin (see Chapter 4 Digital I/O) you want use for row 0. See Table 30: Matrix Keyboard: Connector J1
…		
IORow*n*	Dword	Number of IO you want use for last row. See Table 30: Matrix Keyboard: Connector J1

Table 24: 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]`

Entry	Type	Description
IOStaticKey0	Dword:	Number of IO you want use for static key 0. See Table 30: Matrix Keyboard: Connector J1
StaticKey0	Dword:	PS2 code for static key 0. See Table 27: Matrix Keyboard: PS2 Scan Codes
…		
IOStaticKey*n*	Dword:	Number of IO you want use for last static key. See Table 30: Matrix Keyboard: Connector J1
StaticKey*n*	Dword:	PS2 code for last static key. See Table 27: Matrix Keyboard: PS2 Scan Codes

Table 25: 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 26: 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
'5'	// Scan Code 0x6
'6'	// Scan Code 0x7
'7'	// Scan Code 0x8

V-KEY	PS2-Scan-Code
'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
VK_LMENU	// Scan Code 0x38
VK_SPACE	// Scan Code 0x39
VK_CAPITAL	// Scan Code 0x3A
VK_F1	// Scan Code 0x3B

V-KEY	PS2-Scan-Code
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
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

V-KEY	PS2-Scan-Code
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 27: 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 28: 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 29: 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
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

Pin	IO	Default Interface	Starter-Kit Interface
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 30: 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 14: 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 15: Matrix Keyboard: Example 2

7 Touch Panel Driver

Implemented on: all

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\<Board Type>\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]`

Entry	Type	Default Value	Description
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 31: Touch screen proxy driver settings

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

`[HKEY_LOCAL_MACHINE\SYSTEM\GWE\TOUCHPROXY]`

Entry	Type	Default Value	Description
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 32: 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 MXT224 Touch Driver

The MXT224 is a 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\<Board Type>\TOUCH_MXT224]`

Entry	Value Type	Default Value	Description
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	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.
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 33: Capacitive touch driver registry settings

Note:

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

7.2 EDT Touch Driver

The FT5406 is a 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\<Board Type>\TOUCH_EDT]` or
`[HKEY_LOCAL_MACHINE\DRIVERS\BUILTIN\<Board Type>\TOUCH_EDT_FT5606]`

Entry	Value Type	Default Value	Description
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 34: 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.

7.3 SX865x Touch Driver

The SX8654 is haptic 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\<Board Type>\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.
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

Key	Value Type	Default Value	Comment
			coordinate. SeeTable 36 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 36 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 35: Resistive touch driver registry settings

Amount of time
5
11
22
44
89
178
710
1420
2480
5680
11400
22700
45500
90900
181900

Table 36: 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.

7.4 SiS92XX Touch Driver

Implemented on: kernel V2.50 and later

The SiS9255 is a 32-bit RISC touch screen panel processor with 12-bits ADC for supporting up to 10.1” projected capacitive touch sensor. This processor was developed from SiS and provide in I2C interface for communication with host systems.

This driver can be activated by setting the registry value Flags and support at time 5 finger touch.

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\<Board Type>\TOUCH_SiS92XX]
```

Entry	Value Type	Default Value	Description
ChangeIO	DWORD	99	Touches interrupt IO-Pin number.
ResetIO	DWORD	98	IO-Pin used to trigger controller reset during initialization. A value of -1 disables this functionality.
I2CDevAddr	DWORD	0xB8=(0x5C<<1)	I2C Device address of the touch controller.
I2CDevice	SZ	“I2C2:”	I2C Device name.
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 37: Capacitive touch driver registry settings

If the registry value `SWCalibration` is enabled 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:

- *A touch calibration is not required as the touch controller automatically scales the touch sample to the screen size. Other touch drivers have to be deactivated.*
- *This driver use I2C communication interface with the host. In some cases it is helpful to change clock frequency of I2C bus or IOMUX configuration (drive strength, pull up) for SDA and SCL lines. For more details see description of I2C Driver.*

7.5 eGalax EXC3000 Touch Driver

Implemented on: kernel V2.50 and later

The EXC3000 is a master chip which provides a high performance of projected capacitive touch solution. This touch controller was developed from EETI to support up to 42” projected capacitive touch screen.

This driver can be activated by setting the registry value Flags and support at time 5 finger touch.

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\<Board Type>\TOUCH_EXC3000]
```

Entry	Value Type	Default Value	Description
ChangeIO	DWORD	86	Touches interrupt IO-Pin number.
ResetIO	DWORD	88	IO-Pin used to trigger controller reset during initialization. A value of -1 disables this functionality.
I2CDevAddr	DWORD	0x54=(0x2A<<1)	I2C Device address of the touch controller.
I2CDevice	SZ	“I2C2:”	I2C Device name.
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 38: Capacitive touch driver registry settings

If the registry value `SWCalibration` is enabled 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:

- *A touch calibration is not required as the touch controller automatically scales the touch sample to the screen size. Other touch drivers have to be deactivated.*
- *This driver use I2C communication interface with the host. In some cases it is helpful to change clock frequency of I2C bus or IOMUX configuration (drive strength, pull up) for SDA and SCL lines. For more details see description of* I2C Driver*.*
- *ETTI recommends I2C clock frequency of 400kHz.*

7.6 WM9715 Touch Driver

Implemented on: Kernel V2.50 and later

The WM9715L is a highly integrated input / output device designed for mobile computing and communications. The device can connect directly to a 4-wire touch panel.

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\<Board Type>\TOUCH_WM9715]
```

Entry	Value Type	Default Value	Description
ChangeIO	DWORD	86	Touches interrupt IO-Pin number.
ResetIO	DWORD	-1	IO-Pin used to trigger controller reset during initialization. A value of -1 disables this functionality.
TouchSamples	DWORD	3	Number of samples per point
SettleTime	DWORD	4	Controls touch panel settling time
PenDetectDivider	DWORD	15	Pen detect 68 kOhm Pull-Up is divided by this value.
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.
Flags	DWORD	4	4: Driver is not loaded. 8: Driver will be loaded.

Table 39: Capacitive touch driver registry settings

SettleTime	DELAY (TIME)
0	20.8µs
1	41.7 µs
2	83.3 µs
3	167 µs
4	333 µs
5	667 µs
6	1 ms
7	1.33 ms
8	2 ms
9	2.67 ms
10	3.33 ms
11	4 ms
12	4.67 ms
13	5.33 ms
14	6 ms
15	No delay, switch matrix always on

Table 40: possible settle time settings

8 USB Host Driver

Implemented on: all

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\HCD_HSH1]
```

The registry key for the 2nd , optional USB host is:
```
[HKLM\Drivers\Builtin\HCD_HSH2]
```

Use the following parameters to configure the driver:

Entry	Type/Value	Description
Dll	Hcd_hsh1.dll	Name of the DLL with the driver.
Prefix	HCD	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
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.
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
Flags	Dword:	4: Disable driver from loading. Default for USB0.

Table 41: 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:

Entry	Type	Description
DoNotPromptUser	Dword	Allows disabling the USB driver dialog. Default: 0

Table 42: 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: all

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:

Entry	Type/Value	Description
Prefix	UFN	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Dll	"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 2.0) Default: 0
Priority256	Dword:	Default: 100

Table 43: 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:

Entry	Value	Description
DefaultClientDriver	“USBSER_Class “ “Mass_Storage_Class” “RNDIS”	Select function class of USB device interface.

Table 44: USB Device: Registry settings

10 LCD Driver for FSiMX6

Implemented on: ASA9, PMA9, QBA9, EFA9, QBA9R2, NDCUA9

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 (analogue 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:

Entry	Value	Description
Mode	dword	Number of the predefined configuration or new user configuration.
UseBootMem	dword	Use memory provided by boot loader for frame buffer.
VidMemBase	dword	Base address of video memory. Don't modify this value or add this value if it doesn't exist.
VidMemLen	dword	Size of video memory in MB.
VidMemCache	dword	Use cached video memory for display frame buffer. Default: 0
Verbose	dword	Enables additional output at serial debug port.
AccelLevel	dword	Enables/disables hardware/software acceleration, see 10.4 2D Acceleration Type
OutputDevice	dword	Select output device, see Table 48: LCD – Output Device
HDMI_PHY_CKSYMTXCTRL	dword	HDMI: Clock Symbol and Transmitter Control (see chapter 34.7.10 of i.MX6 Reference Manual)
HDMI_PHY_VLEVCTRL	dword	HDMI: Voltage Level Control (see chapter 34.7.15 of i.MX6 Reference Manual)
HDMI_PHY_CPCE_CTRL	dword	HDMI: Edge Rate Control (see chapter 34.7.7 of i.MX6 Reference Manual)

Table 45: 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
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 ASA9, efusA9	800x480	16	On	On	33MHz

Table 46: LCD - Modes

Mode	Name	XxY	BPP	VCLK
25	HDMI 640x480 Panel	640x480	16	25,2MHz
26	HDMI 720x480 Panel	720x480	16	25MHz
27	HDMI 720x576 Panel	720x576	16	27MHz
28	HDMI XGA Panel	1024x768	16	56MHz
29	HDMI 1280x720 Panel	1280x720	16	74,25MHz
30	HDMI 1080p60 Panel	1920x1080	16	148,5MHz
31	HDMI 1680x1050 Panel	1680x10500	16	108MHz

Table 47: HDMI - 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. For detailed information on how to perform these settings and a series of display driver adjustments please refer to the documentation section NetDCUA9 – Display.

The output device (RGB, LVDS, HDMI) is pre-configured by the selected mode. If you want to override this value (i.e. to use the values of mode 19 for a display connected to LVDS interface) you can set registry value OutputDevice to one of the following values:

Output Device	Description
0x02000	HDMI
0x04000	Single channel LVDS
0x08000	Digital RGB
0x10000	Dual channel LVDS

Table 48: LCD – Output Device

10.1 Default Display Mode

	Digital RGB	LVDS
armStoneA9	19 = 800x480 (ET070080)	
armStoneA9R2	--	18 = 800x480 (CHIMEI G070Y)
efusA9	19 = 800x480 (ET070080)	
PicoMODA9	6 = 800x480 (ET070080)	
QBlissA9	--	18 = 800x480 (CHIMEI G070Y)

Table 49: 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
armStoneA9	LCD_CONFIG_OUT18BIT	LCD_CONFIG_OUT18BIT
armStoneA9R2	--	LCD_CONFIG_OUT18BIT
PicoModA9	LCD_CONFIG_OUT16BIT	LCD_CONFIG_OUT18BIT
EFUSA9	LCD_CONFIG_OUT18BIT	LCD_CONFIG_OUT18BIT
QBlissA9	--	LCD_CONFIG_OUT18BIT

Table 50: LCD - Default LCD Output Width

The configuration can be changed with registry parameter CONFIG.

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]
```

Entry	Type	Description
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.

Entry	Type	Description
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.

Table 51: LCD – Display Mode Registry Settings

10.3.1 Registry Value Type

Value	Meaning
0x00000	Default
0x00002	TFT-Display
0x00004	Colour-Display
0x00100	Enable contrast voltage VEE
0x00200	Output more information to serial debug line
0x02000	Use HDMI interface as output device
0x04000	Use single channel LVDS interface as output device
0x08000	Use digital RGB interface as output device
0x10000	Use dual channel LVDS interface as output device

Table 52: LCD - Display Driver Registry Value Type

10.3.2 Registry Value Config

Symb. Name	Value	Description
LCD_BACKLIGHT	0x00001000	Backlight enable polarity: active low
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 49: 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 53: 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)	240 Ohm
2	120 Ohm
3	80 Ohm
4	60 Ohm
5	48 Ohm
6	40 Ohm
7	34 Ohm

Table 54: LCD - Port Drive Strength

10.4 2D Acceleration Type

The drawing of GDI based display drivers can be done with different optimization types.
By default, all drawing is done by software using GDI functions. With value registry value AccelLevel you can modify this.

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

AccelLevel	Description
Bit 0	Enable acceleration based on software using NEON engine. If NEON can't speed-up drawing, use GDI.
Bit 1	Enable iMX6 hardware acceleration. If hardware can't speed-up drawing try using NEON.

Table 55: LCD - Registry value AccelLevel

10.5 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]
```

Entry	Type	Description
BtnOldSkin	DWORD	Set this value to 1 to enable old button drawing. This also speeds up drawing at all. Default: 0

Table 56: UI Skin / XP Mode

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:

Entry	Type/Value	Description
Dll	SOFTKB.DLL	name of the driver file
Prefix	SIP	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:0	This value specifies the device index, a value from 0 through 9.
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.

Table 57: Softkeybd: Registry Settings

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

All F&S boards support two types of I2C drivers:

- Soft I2C, means bit banging driver using GPIO pins
- Native I2C, means using the internal hardware machine of the CPU.

Following you can see the list of I2C interfaces and the corresponding driver name for each board.

armStoneA9:
At armStoneA9 we have two NI2C drivers and one I2C drivers.
The usage is as follows:

Connector	Pin	Function	Driver
Feature connector	16	SCL	I2C1: (Soft I2C driver)
	17	SDA	
Feature connector	18	SCL	I2C5: (Native I2C driver) (not available on armStoneA5 or armStoenA9R2)
	26	SDA	
Touch connector	3	SCL	I2C3: (Native I2C driver)
	2	SDA	
RGB connector	34	SCL	I2C3: (Native I2C driver)
	32	SDA	
JILI30 LVDS connector	27	SCL	I2C3: (Native I2C driver)
	25	SDA	
HDMI_CTRL_DAT, HDMI_CLK			I2C9: (I2C driver)

Table 58: armStoneA9 I2C driver usage

armStoneA9R2:
At armStoneA9R2 we have two NI2C drivers and one I2C drivers.
The usage is as follows:

Connector	Pin	Function	Driver
Feature connector	16	SCL	I2C1: (Soft I2C driver)
	17	SDA	
~~Feature connector~~	~~18~~	~~SCL~~	~~I2C5: (Native I2C driver)~~
	~~26~~	~~SDA~~	
Touch connector	3	SCL	I2C3: (Native I2C driver)
	2	SDA	
JILI30 LVDS connector	27	SCL	I2C3: (Native I2C driver)
	25	SDA	
HDMI_CTRL_DAT, HDMI_CLK			I2C9: (I2C driver)

Table 59: armStoneA9R2 I2C driver usage

PicoMODA9:
At PicoMODA9 we have two NI2C drivers and one I2C drivers.
The usage is as follows:

Connector	driver
Main connector	I2C3: (native I2C driver)
Touch	I2C2: (native I2C driver), on-board
Audio	I2C1: (native I2C driver)
HDMI_CTRL_DAT, HDMI_CLK	I2C9: (I2C driver), on-board

Table 60: PicoMODA9 I2C driver usage

QBlissA9:
At QBlissA9 we have two NI2C drivers and one I2C driver.
The usage is as follows:

Connector	driver
Main connector	I2C3: (native I2C driver)
HDMI_CTRL_DAT, HDMI_CLK	I2C9: (I2C driver), on-board

Table 61: QBlissA9 I2C driver usage

efusA9:
At efusA9 we have two NI2C drivers and one I2C driver.
The usage is as follows:

Connector	driver
Connector J22	I2C3: (native I2C driver)
Touch	I2C2: (native I2C Driver)
Audio	I2C1: (native I2C Driver)
HDMI_CTRL_DAT, HDMI_CLK	I2C9: (I2C driver), on-board

Table 62: efusA9 I2C driver usage

efusA7UL:
At efusA7UL we have two NI2C drivers and one I2C driver.
The usage is as follows:

SKIT- Connector	driver
Connector J22 and J2X	I2C1: (native I2C driver)
Connector J22	I2C2: (native I2C Driver)
Audio and RTC	I2C3: (I2C driver), on-board

Table 63: efusA7UL I2C driver usage

PicoCOM1.2:
At PicoCOM1.2 we have one NI2C drivers and one I2C driver.
The usage is as follows:

Connector	driver
Connector J3 (Pin 32/33)	I2C1: (native I2C driver)
Audio and RTC	I2C3: (I2C driver), on-board

Table 64: PicoCOM1.2 I2C driver usage

13.1 Soft I2C Driver

Implemented on: all

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:

Entry	Type/Value	Description
Dll	fs_i2c.dll	Name of the DLL with the driver
Prefix	I2C	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
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.
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.
Flags	Dword:	4: Disabled from loading

Table 65: 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 (see Figure 3: F&S I2C Bus test tool). This program lists the available I2C ports and scans the port for devices.

13.2 Native I2C Driver

Implemented on: ASA9, PMA9, QBA9, EFA9, EFA7UL, PC12, QBA9R2, NDCUA9

Board supports native I2C driver.

The registry key for the driver is:

```
[HKLM\Drivers\Builtin\<board>\I2C<n>]
[HKLM\Drivers\Builtin\I2C2]
[HKLM\Drivers\Builtin\I2C3]
```

Use the following parameters to configure the driver:

Entry	Type/Value	Description
Dll	fs_ni2c.dll	Name of the DLL with the driver
Prefix	I2C	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
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.
ClockFreq	Dword:0x30d40	200000 kBit/s (default)
DrvStrength	Dword:	Set drive strength control for SDA/SCL.
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

Table 66: 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 (see Figure 3: F&S I2C Bus test tool). This program lists the available I2C ports and scans the port for devices.

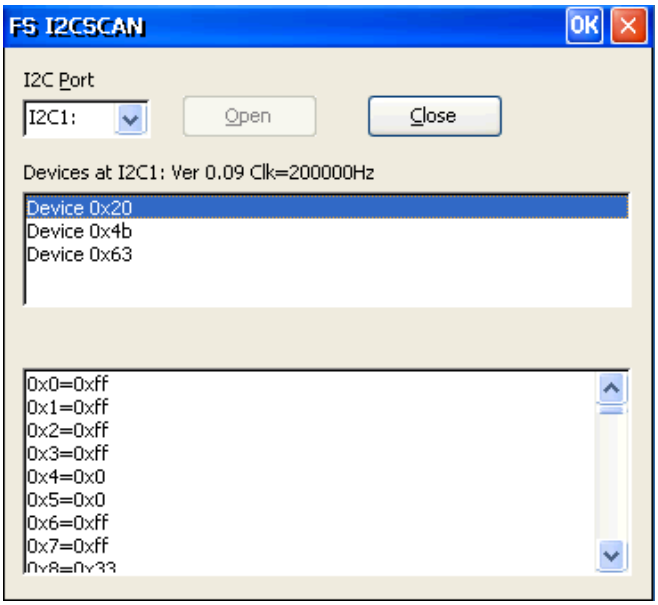

Figure 3: F&S I2C Bus test tool

14 PWM Driver

Implemented on: all

armStoneA9 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.

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

```
[HKLM\Drivers\BuiltIn\<Board type>\PWM<id>]
```

Required settings:

Entry	Type/Value	Description
Dll	FS_PWM.DLL	Name of the DLL with the driver
Prefix	PWM	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
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.
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
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 67: 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 armStoneA9:

Channel	Description
1	TOUT3 (Feature connector pin 32)
2	TOUT1 (Feature connector pin 28)
3	Do not use! Backlight control. Use contrast control of display driver. (Display connector pin 25)
4	TOUT2 (Feature connector pin 30)

Table 68: PWM – armStoneA9 Channel

Table Channel QBlissA9:

Channel	Description
1	(Connector 196)
3	Do not use! Backlight control. Use contrast control of display driver. (Display connector pin 25)
4	(Connector 194)

Table 69: PWM – QBlissA9 Channel

Table Channel efusA9:

Channel	Description
3	Do not use! Backlight control. Use contrast control of display driver. (Display connector pin 25)
4	(Connector pin 25)

Table 70: PWM – efusA9 Channel

Table Channel efusA9X:

Channel	Description
6	Do not use! Backlight control. Use contrast control of display driver. (Connector pin 87)
2	(Connector pin 25)
5	(Connector pin 23)

Table 71: PWM – efusA9X Channel

15 SD/MMC Driver

Implemented on: all

Platform supports SD/MMC driver. There will be a driver for external SD slot and one for internal (only ASA9/PMA9/QBlissA9) SD slot.

The registry key for the on-board slot (armStoneA9/armStoneA9R2) is:

```
[HKLM\Drivers\Builtin\armStoneA9\USDHC3]
[HKLM\Drivers\Builtin\armStoneA9R2\USDHC2]
```

The registry key for the external slot (efusA9) is:

```
[HKLM\Drivers\Builtin\efusA9\USDHC3]
```

Use the following parameters to configure the driver:

Entry	Type/Value	Description
Dll	fs_usdhc.dll	Name of the DLL with the driver
Prefix	SHC	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
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~~
WPIO[1]	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: not set
WPIOPolarity[2]	Dword:	Polarity of WPIO to signal driver write protection. Default: 1
~~WriteProtect~~	~~Dword:<0|1>~~	~~Enable disable write protection. This value will be ored with the hardware WPIO pin.~~
UseCardAvailable	Dword:<0|1>	Use registry value CardAvailable to detect if card is inserted.
CardAvailable	Dword:<0|1>	Card is inserted or not.
MaximumClockFrequency	Dword:	Maximum clock frequency in Hz
BusWidth8Bit	Dword:<0|1>	Enable 8Bit bus width.
Debug	Dword:0|4	Set to 4 to get list of registry settings at serial debug port. Default: 0

Table 72: SD/MMC Driver Registry Settings

[1] Needs driver version 1.3 or higher
[2] Needs driver version 1.3 or higher

16 Native SPI Driver

Implemented on: all

Board supports native SPI driver.

The registry key for the driver is:

```
[HKLM\Drivers\Builtin\<Board type>\SPI<n>]
```

Use the following parameters to configure the driver:

Entry	Type/Value	Description
Dll	fs_nspi.dll	Name of the DLL with the driver
Prefix	SPI	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:0…9	This value specifies the device index, a value from 0 through 9.
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.
DriverMethod	Dword:0…3	0: IRQ 1: Polling 2+3: DMA Up to now, only Polling is implemented.
IntPullUp	Dword:0…4	Enable Internal pull-up or pull-down for MISO, MOSI, SCL and CS0, CS1. 0: Disable internal pull-up (default) 1: 100k pull-down 2: 47k pull-up 3: 100k pull-up 4: 22k pull-up Default: 0
DrvStrength	Dword:0…7	Set drive strength control for MISO, MOSI, SCL and CS0, CS1. 0: Disabled 1: 240 Ohm 2: 120 Ohm 3: 80 Ohm 4: 60 Ohm 5: 48 Ohm 6: 40 Ohm 7: 34 Ohm Default: 7
Flags	Dword:	4: Disabled from loading

Table 73: Native SPI: Registry settings

16.1 efusA9 Port relation

Hardware	Software (Registry)
SPIA	SPI2:
SPIB	SPI1:

Table 74: Native SPI: efusA9 Port relation

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

17 Ethernet Driver

Implemented on: all

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:

Entry	Value	Description
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 75: Ethernet Driver: Registry settings

18 Screen Saver Driver

Implemented on: all

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:

Entry	Value	Description
Dll	FSPMScreenSaver.dll	Name of the DLL with the driver
Prefix	PSS	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
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.
DxOn	Dword:	0
DxOff	Dword:	4
Flags	Dword:	0x10: User mode driver

Table 76: PSS: Registry settings

19 Broadcast Driver

This driver is loaded during system start and sends a broadcast to the network including some device information. The broadcast message could 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:

Entry	Value	Description
Dll	fs_bcsend.dll	Name of the DLL with the driver
Prefix	BCS	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
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 16: Broadcast Driver: broadcast message

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 17: Broadcast Driver: Example BCIoctl

20 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:

Entry	Type	Description
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 18: 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.

21 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:

Entry	Value	Description
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.

22 FSStartup

Implemented on: all

F&S Startup driver creates the start-up functionality for Microsoft® Compact 2013 like Microsoft® Compact 7.

The registry key for the driver is:

```
[HKLM\System\FSStartup]
```

Use the following parameters to configure the driver:

Entry	Value	Description
Dll	FSStartup.dll	Name of the DLL with the driver
Folder	String	Defines the Startup folder Default: \ffsdisk\Startup
IsEnabled	Dword:0|1	This activate(1) or deactivate(0) the driver Default: 1

Table 77: FSStartup registry settings

23 FSMinShell

Implemented on: all

F&S MinShell application replaces the Microsoft® Compact 2013 MinShell application. It fixes some bugs and adds font initialisation and start-up functionality for Microsoft® Compact 2013 like Microsoft® Compact 7.
FSMinShell is in the same way as MinShell by using [HKLM\Init\Launch80].

Features of FSMinShell:

- Call touch calibration if there is no calibration data stored in registry
- Register fonts which are in directory CSIDL_FONTS or CSIDL_WINDOWS
- Start applications (*.exe, *.bat, *.cmd) from directory CSIDL_STARTUP

You can overcome startup functionality if you press the shift key of a connected keyboard during boot.

Shell folders are initialized as follows:

```
[HKEY_LOCAL_MACHINE\SYSTEM\Explorer\Shell Folders]
     "Fonts"="FFSDISK\\FONTS"
     "StartUp"="\\FFSDISK\\StartUp"
```

The above folders are not available by default. You have to create it.

24 CEDDK Functions

The CEDDK functions support some information about the board.

Required includes:

```
#include "ceddk_core.h"
```

24.1 Board Type

Implemented on: all

Get the board type i.e. efusA9.

Signature:

```
int CEDDK_GetBoardType(void);
```

Return:

0	efusA9
1	armStoneA9
2	PicoMODA9
3	QBlissA9
4	armStoneA9r2
6	QBlissA9r2
7	NetDCUA9
8	efusA9X
9	PicoCOMA9X
16	efusA7UL
18	PicoCOM1.2

24.2 Board Name

Implemented on: kernel V1.60 and later

Get the name of the board as string.

Signature:

```
int CEDDK_GetBoardName(WCHAR* szName, int nStrLen);
```

Parameters:

`szName`	Name of the board. Must be a pointer to an array of previously allocated WCHARs. You can set this value to NULL to get the length of the string.
`nStrLen`	Length of the array szName in WCHARs.

Return:

Length of board name in WCHARs

24.3 CPU Core Temperature

Implemented on: Kernel V1.50 and later

Retrieve CPU core temperature.

Signature:
```
DWORD CEDDK_GetCPUCoreTemp(double *pResult);
```

Parameters:

`pResult`	Temperature in C°

Return:

0	Error, see `GetLastError()` for details

Example:
```
#include "ceddk_core.h"

DWORD dwError;
FLOAT fTemperatur;

If(!CEDDK_GetCPUCoreTemp(fTemperatur))
{
        // Temperature value is invalid
        dwError = GetLastError();
        …
}
```

Listing 19: Read temperature

24.4 CPU Core Speed

Implemented on: Kernel V2.40 and later

Retrieve CPU core speed.

Signature:
```
DWORD GetCPUCoreSpeed(void);
```

Parameters:

Return:

0	CPU clock in Hz

Example:
```
#include "ceddk_core.h"

DWORD dwClock;

dwClock = GetCPUCoreSpeed();
```

Listing 20: Read CPU clock

24.5 Processor Information

Implemented on: kernel V1.50 and later

Get processor information.

Signature:

```
DWORD CEDDK_GetSOCInformation(PCEDDK_SOC_INFO pSOCInfo);
```

Parameters:

`pSOCInfo` Pointer to a `CEDDK_SOC_INFO` structure, where the information will be stored

CEDDK_SOC_INFO:

```
typedef struct tag_SOC_INFO
{
    DWORD dwSocType;
    DWORD dwNumCores;
    DWORD dwTemperature;      /* 0=Commercial, 1=Extended Commercial,
                                 2=Industrial, 3=Automotive */
    DWORD dwSpeed;            /* Speed in MHz */
    DWORD dwRevision;
}CEDDK_SOC_INFO, *PCEDDK_SOC_INFO;
```

Listing 21: Struct CEDDK_SOC_INFO

`dwSocType`	The SOC Type see `CEDDK_SOCTYPE`
`dwNumCores`	Total count of cores
`dwTemperature`	Supported temperature range: 0 - Commercial (0 to 95C) 1 – Extended Commercial (-20 to 105C) 2 - Industrial (-40 to 105C) 3 - Automotive (-40 to 125C)
`dwSpeed`	Core Speed in MHz
`dwRevision`	Silicon revision of the SOC

CEDDK_SOCTYPE:

```
typedef enum
{
       CEDDK_TYPE_MX6UL=0x064,
       CEDDK_TYPE_MX6S=0x161,
       CEDDK_TYPE_MX6DL=0x261,
       CEDDK_TYPE_MX6SX=0x062,
       CEDDK_TYPE_MX6D=0x263,
       CEDDK_TYPE_MX6Q=0x463,
       CEDDK_TYPE_UNKNOWN,
} CEDDK_SOCTYPE;
```

Listing 22: Struct CEDDK_SOCTYPE

25 High Resolution Timer

Implemented on: kernel V1.90 and later

F&S High Resolution Timer driver provides a timer which has resolution from 1µs, the lowest timer interval is 100µs and the precision is approximately 25µs (interrupt latency). At the end of each time interval an event is send to the calling application.
In contrast the Windows Multimedia Timer may have a higher resolution (ticks/µs~66), but it can be used only to measure time intervals and not to send events in regular time intervals.

The registry key for the driver is:

```
[HKLM\System\Timer]
```

Use the following parameters to configure the driver:

Entry	Type/Value	Description
Dll	fs_imx6_timer.dll	Name of the DLL with the driver
Prefix	TIM	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:1	This value specifies the device index, a value from 0 through 9.
Order	Dword:0x20	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.
Flags	Dword:n	Device load flags (0x4: disabled from loading)
Priority256	Dword:110	Priority for the timer driver thread.
TogglePin	Dword:0xFFFFFFFF	Number of IO pin to toggle (0XFFFFFFFF = toggle no pin)
Equalize	Dword:0(default)…255	After a timer event the current - and the awaited timer counter values are compared. If the current value exceeds the awaited one the next timer cycle is shorted. The maximal reduction is defined by "Equalize".

Table 78: FS High Resolution Timer registry settings

25.1 Programming example

Headerfile:

```
#include <timerio.h>
```

Listing 23: FS High Resolution Timer: Headerfile

A. Opening a timer instance

```
HANDLE hTIM;
hTIM = CreateFile( _T("TIM1:"), GENERIC_READ|GENERIC_WRITE, 0, NULL, OPEN_EXISTING,
                   FILE ATTRIBUTE NORMAL, NULL );

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

Listing 24: FS High Resolution Timer: Open an instance

B. Create a new timer

```
TIMER_SETTINGS stTS = {0};
WCHAR tEventName[64] = {0};  // Timer event to wait for within the application
stTS.dwWaitUs = 1000         // Timer period in micro seconds, minimum 100µs
stTS.uNumEvents = 0xFFFFFFFF // Number of events carried out after timer is started
                             // 0xFFFFFFFF means infinite events carried out

BOOL bRes = DeviceIoControl( hTIM, IOCTL TIMER CREATE, &stTS, sizeof(TIMER SETTING),
                                   tEventname, sizeof(tEventname), NULL, NULL );

if(!bRes)
{
    ERRORMSG(1,(L" IOCTL TIMER CREATE: Error %d\r\n"), GetLastError());
    return(FALSE);
}

HANDLE hEvent = CreateEvent(NULL, FALSE, FALSE, tEventname);
if(!hEvent)
{
    ERRORMSG(1,(L"Create Event: Error %d\r\n"), GetLastError());
    return(FALSE);
}
```

Listing 25: FS High Resolution Timer: Create a timer

C. Start the timer

```
// Note, before start the timer you may create a particular thread to wait for timer events
// else some events may lost.

DWORD TimerThread(LPVOID lpParameter /*fit in your needed timer parameters*/)
{
    DWORD dwWait;
    DWORD dwTimout
    while(TRUE)
    {
        dwWait = WaitForSingleObjet(lpParameter->hEvent, dwTimeout)
        if(dwWait == WAIT OBJECT 0)
        {
            ResetEvent(lpParameter->hEvent);
            // Do desired action and/or toggle pin via registry
            // …
        }
        // else if (dwWait == … error handling
    }
    return 0;
}
```

Listing 26: FS High Resolution Timer: Before start the timer

```
// Start the timer

if(!bRes)
{
    ERRORMSG(1,(L" IOCTL TIMER START: Error %d\r\n"), GetLastError());
    return(FALSE);
}
```

Listing 27: FS High Resolution Timer: Start the timer

D. Stop the timer and free all resources

```
// Stop the timer
bRes = DeviceIoControl( hTIM, IOCTL_TIMER_STOP, NULL, 0, NULL, 0, NULL, NULL );

if(!bRes)
{
    ERRORMSG(1,(L" IOCTL TIMER STOP: Error %d\r\n"), GetLastError());
    return(FALSE);
}

// You may continue with IOCTL_TIMER_START …, else
// don’t forget to terminate the “event thread” and wait for it, if any

// Free resources
bRes = DeviceIoControl( hTIM, IOCTL_TIMER_DESTROY, NULL, 0, NULL, 0, NULL, NULL );
//
if(!bRes)
{
    ERRORMSG(1,(L" IOCTL TIMER DESTROY: Error %d\r\n"), GetLastError());
    return(FALSE);
}

CloseHandle(hEvent);
CloseHandle(hTIM);
```

Listing 28: FS High Resolution Timer: Stop the timer and free resources

E. Request and control the timer

```
// Request the number of events occurred. Note this IO Control may be used to check if all
// events have been successfully send to the application.
// Applying it to a running timer may retrieve former values.

DWORD dwEvents;
bRes = DeviceIoControl( hTIM, IOCTL TIMER NUM EVENTS, &dwEvents, sizeof(DWORD),
                                                      NULL, 0, NULL, NULL );
//
if(!bRes)
{
    ERRORMSG(1,(L" IOCTL TIMER NUM EVENTS: Error %d\r\n"), GetLastError());
    return(FALSE);
}

// Request the number of missed events. Means events which are not confirmed by the
// application by call ResetEvent()

DWORD dwEventsMissed;
bRes = DeviceIoControl( hTIM, IOCTL TIMER NUM MISSED EVENTS, &dwEventsMissed, sizeof(DWORD),
                                                      NULL, 0, NULL, NULL );
//
if(!bRes)
{
    ERRORMSG(1,(L" IOCTL TIMER NUM MISSED EVENTS: Error %d\r\n"), GetLastError());
    return(FALSE);
}

// Reset the timer. This IOControl does stop the timer immediately and resets the event-/
// missed counter finally the timer is started again

bRes = DeviceIoControl( hTIM, IOCTL_TIMER_RESET, NULL, 0, NULL, 0, NULL, NULL );
//
if(!bRes)
{
    ERRORMSG(1,(L" IOCTL TIMER RESET: Error %d\r\n"), GetLastError());
    return(FALSE);
}

// Adjust new timer settings.
// IOCTL TIMER ADJUST stops the timer immediately without waiting for the current period
// ends. Use IOCTL TIMER ADJUST FROM NEXT EVENT for wait the current periode finshed
// before the new settings are valid.

TIMER_SETTINGS stNewTS = {0};
stNewTS.dwWaitUs = 100
stNewTS.uNumEvents = 2000

BOOL bRes = DeviceIoControl( hTIM, IOCTL TIMER ADJUST, &stNewTS, sizeof(TIMER SETTING),
                                   NULL, 0 NULL, NULL);

if(!bRes)
{
    ERRORMSG(1,(L" IOCTL_TIMER_ADJUST: Error %d\r\n"), GetLastError());
    return(FALSE);
}
```

Listing 29: FS High Resolution Timer: Request and control the timer

26 Flash Correct and Refresh

Implemented on: kernel V2.30 and later

Flash correct and refresh (FCR) is a device driver to scan stored data for possible data refresh before flash accumulates more errors that can be corrected by ECC. This driver runs in background and scans in default configuration one sector per minute.

The registry key for the driver is:

```
[HKLM\Drivers\Builtin\FCR]
```

Use the following parameters to configure the driver:

Entry	Type/Value	Description
Dll	fs_fcr.dll	Name of the DLL with the driver
Prefix	FCR	This required value specifies the driver's device file name prefix. It is a three-character identifier, such as COM.
Index	Dword:1	This value specifies the device index, a value from 0 through 9. At time it can be 1 only.
Order	Dword:0x90	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.
Flags	Dword:n	Device load flags (0x4: disabled from loading)
Priority	Dword:256	Priority for the refresh thread.
Timeout	Dword:1	Waiting time in minutes to read next page
ErrorTolerance	Dword:0	This value would be subtracting from max ECC of half page. (E.g. 1024 bytes) Means: (MAXECC – ErrorTolerance) Default MAXECC – 1 Min: 0 Max: 8 Currently is MAXECC=8
UseDataPartition	Dword:0	Normally the driver scans bootloader and kernel partitions. With this value can be allowed to check data partition. Each bit for one data partition after kernel: Bit0=1 partition 1 has to be checked Bit1=2 partition 2 has to be checked … … … Bit31=32 means that partition 32 has to be checked. Currently we are supporting only 2 data partitions.
LastCheckedPage	Dword: -1	Number of last scanned page. This page would be saved periodical and used after board reset to continue data scan.
SavePageRuns	Dword: -1	Number of scanned pages before value LastCheckedPage can be saved. Default: Number pages per block

Table 79: Flash correct and refresh registry settings

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 one (1) year 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