netX

Program Reference Guide

next Generation of Communication Controller

Preliminary
Changed and extended without notice
Only for general information

Edition: 4
Language: English (EN)

Hilscher Gesellschaft für Systemautomation mbH

Rheinstrasse 15
D-65795 Hattersheim
Germany

Tel. +49 (0) 6190 / 9907 - 0
Fax: +49 (0) 6190 / 9907 - 50

Sales national: +49 (0) 6190 / 9907 - 90
Sales international: +49 (0) 6190 / 9907 - 91
netX-Support: +49 (0) 6190 / 9907 - 97

E-Mail Sales national: vertrieb@hilscher.com
E-Mail Sales international: sales@hilscher.com
E-Mail netX Support: netxsupport@hilscher.com

Web: www.hilscher.com

1 Naming Conventions

Generally for the various functions and parts of a microcontroller a number of acronyms comes into play. For ease of use, in this reference guide a consistent naming scheme is introduced to all the NetX register names which will be used throughout. A full list of register names can be found in appendix A.

The naming of the registers is carried out as follows: The first component determines the register group, e.g. GPIO or DPM, etc., while the subsequent parts, separated by underscores "_", specify the function of the particular register more detailed. Note that the word "register" will never occur in the name as it does not yield any additional information.

Sometimes the notation [m-n] will be used to indicate a set of registers ranging from m to n, e.g. IRQ_XP[0-3] stands for the (sequence of) registers IRQ_XP0, IRQ_XP1, IRQ_XP2 and IRQ_XP3.

2 System Functions

In this chapter, we will discuss some system functions about boot option, IO configuration, watch dog, system reset and status, etc.

The following table is a summary of the registers about these functions.

ARM Address	Register Name	Short Description
0x00100000	BOO_STAT	Bond Out Option Status Register
0x00100004	IO_CFG	IO Configuration Register
0x00100008	IO_CFG_MSK	IO Configuration Mask Register
0x0010000c	RESET_CTRL	Reset Control Register
0x00100034	NETX_REV	netX Revision Register
0x00100070	IO_CFG_ACCESS_KEY	IO Configuration Access Key Register
0x00100200	WDG_TRIG	Watchdog Trigger Register
0x00100204	WDG_CNTR	Watchdog Counter
0x00100208	WDG_IRQ_TIMEOUT	Watchdog Interrupt Timeout
0x0010020c	WDG_RESET_TIMEOUT	Watchdog Reset Timeout
0x001034d8	SYS_STAT	System Status

2.1 BOO – Bond Out Option

BOO_STAT – Bond Out Option Status Register 0x00100000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																										Internal			BOND_OPTION		

Bits	Name	Description	R/W	Default
31:6	reserved	-	R	0x00
5:3	Internal	Reserved for test mode	R	0x00
2:0	BOND_OPTION	Bits reflect the status of the (internally) hardwired pins.	R	0x00

2.2 IO_CFG – IO Configuration

The following two registers (IO_CFG_ACCESS_KEY and IO_CFG) are needed for selecting particular functions of the shared netX pins. By setting the bits of IO_CFG register, the desired functions can be activated. Any bit of the IO_CFG register can only be set, if the corresponding mask bit in the IO_CFG_MSK register is set either.

IO_CFG_ACCESS_KEY – IO Configuration Access Key Register **0x00100070**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																IO_CFG_ACCESS_KEY															

Bits	Name	Description	R/W	Default
31:16	Reserved	-	R	0x00
15:0	IO_CFG_ACCESS_KEY	Access key for next write access.	R/W	0x00

Changing the ASIC Control registers is only possible by the following sequence:
1: read out access key
2: write back access key
3: write desired value to the desired register

The access key will become invalid after each access to any register in the ASIC_CTRL address area (Arm address:0x00100000 – 0x00100034) and has to be read and written again for subsequent accesses.

IO_CFG – IO Configuration Register 0x00100004

Functions can only be selected, if the corresponding bit in IO_CFG_MSK is set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF_SELECT_N	reserved		SEL_E_PWM2_ECLK	SEL_F3_PWM3_ECLK	SEL_F1_PWM3_ECLK	SEL_F0_PWM3_ECLK	SEL_WDG	SEL_ETM	SEL_LED_MII3	SEL_LED_MII2	SEL_MP	SEL_ENC1	SEL_ENC0	SEL_E_RPWM2	SEL_E_FAILURE2	SEL_E_PWM2	SEL_F3_PWM3	SEL_F2_RPWM3	SEL_F2_FAILURE3	SEL_F1_RPWM3	SEL_F1_PWM3	SEL_F0_FAILURE3	SEL_F0_PWM3	SEL_FO1	SEL_FO0	SEL_MII3PWM	SEL_MII23	SEL_MII3	SEL_MII2	SEL_LCD_COL	SEL_LCD_BW

Bits	Name	Description	R/W	Default
31	IF_SELECT_N	1: Host interface modes disabled 0: DPM Mode / Extension Bus Mode / PIO Mode	R/W	0
30:29	reserved	-	R	0x00
28	SEL_E_PWM2_ECLK	select outputs for PWM0 signals at PIO(7-0), with XMAC2 clocked by xmac2_eclk_in (used for xMAC3 fiber optic mode with low clock jitter)	R/W	0
27	reserved	-	R/W	0
26	SEL_F1_PWM3_ECLK	select outputs for PWM1 signals (Vn, W, Wn) at XMAC1 pads, with XMAC3 clocked by xmac3_eclk_in (used for xMAC3 fiber optic mode with low clock jitter)	R/W	0
25	SEL_F0_PWM3_ECLK	select outputs for PWM1 signals (U, Un, V) at XMAC0 pads, with XMAC3 clocked by xmac3_eclk_in (used for xMAC3 fiber optic mode with low clock jitter)	R/W	0
24	SEL_WDG	select pin for clk watchdog / system watchdog at PIO15	R/W	0
23	SEL_ETM	select pins for ETM9 of ARM926 at PIO(30-8)	R/W	0
22	SEL_LED_MII3	select inputs for LEDs of MII3 at PIO(7-4)	R/W	0
21	SEL_LED_MII2	select inputs for LEDs of MII2 at PIO(3-0)	R/W	0
20	SEL_MP	select inputs for encoder signals mp at PIO(7-6)	R/W	0
19	SEL_ENC1	select inputs for encoder signals enc1 at PIO(5-3)	R/W	0
18	SEL_ENC0	select outputs for encoder signals enc0 at PIO(2-0)	R/W	0
17	SEL_E_RPWM2	select output for PWM0 rpwm signal at PIO7	R/W	0
16	SEL_E_FAILURE2	select output for PWM0 failure signal at PIO6	R/W	0
15	SEL_E_PWM2	select outputs for PWM0 signals at PIO (5-0)	R/W	0
14	SEL_F3_PWM3	select outputs for PWM1 signals at XMAC3 pads and XM1_ECLK	R/W	0
13	SEL_F2_RPWM3	select output for PWM1 rpwm signal at XM0_ECLK	R/W	0
12	SEL_F2_FAILURE3	select output for PWM1 failure signal at XM2_ECLK	R/W	0
11	SEL_F1_RPWM3	select output for PWM1 rpwm signal at XM1_IO1	R/W	0
10	SEL_F1_PWM3	select outputs for PWM1 signals (Vn, W, Wn) at xMAC1 pads	R/W	0
9	SEL_F0_FAILURE3	select output for PWM1 failure signal at XM0_IO1	R/W	0
8	SEL_F0_PWM3	select outputs for PWM1 signals (U, Un, V) at xMAC0 pads	R/W	0
7	SEL_FO1	select outputs for Fiber Optic signals of Phy1 at xMAC1 pads	R/W	0

6	SEL_FO0	select outputs for Fiber Optic signals of Phy0 at xMAC0	R/W	0
5	SEL_MII3PWM	select outputs for MII interface of xMAC3 at PIO(25-18) *use together with Bit 3 (SEL_MII3)*	R/W	0
4	SEL_MII23	select outputs for MDIO signals at PIO(17-16)	R/W	0
3	SEL_MII3	select outputs for MII interface of xMAC3 at all xMAC3 pads and XMAC0_ECLK, XMAC1_ECLK, XMAC2_ECLK *use together with Bit 5 (SEL_MII3PWM)*	R/W	0
2	SEL_MII2	select outputs for MII interface of xMAC2 at xMAC2 pads and PIO(14-8) and PIO(30-26)	R/W	0
1	SEL_LCD_COL	select outputs for LCD signals (data lines 8 – 17) at PIO(25-16)	R/W	0
0	SEL_LCD_BW	select outputs for LCD signals (control signals and data 0 – 7) at PIO(30-26) and PIO(15-8)	R/W	0

Hint:

Simple displays (usually black and white STN displays) may require only the control lines and data lines (7:0) of the netX Display Interface. When connecting such displays, only Bit 0 of the IO_CFG register needs to be set, allowing the remaining pins of the display interface to be used as PIOs (PIO [25:16]).

IO_CFG_MSK – IO Configuration Mask Register 0x00100008

This Mask Register is set by the netX internal ROM, according to the capabilities of the particular chip and the licenses found in the external security memory and is hence **not user writeable**.

The structure of this register is identical to the IO_CFG register, while each set bit in the Mask Register represents a selection that can be made in the IO_CFG register, or in other words, any bit in the IO_CFG register can only be set, if the corresponding bit in the Mask Register is also set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF_SELECT_N	reserved		SEL_E_PWM2_ECLK	SEL_F3_PWM3_ECLK	SEL_F1_PWM3_ECLK	SEL_F0_PWM3_ECLK	SEL_WDG	SEL_ETM	SEL_LED_MII3	SEL_LED_MII2	SEL_MP	SEL_ENC1	SEL_ENC0	SEL_E_RPWM2	SEL_E_FAILURE2	SEL_E_PWM2	SEL_F3_PWM3	SEL_F2_RPWM3	SEL_F2_FAILURE3	SEL_F1_PWM3	SEL_F0_FAILURE3	SEL_F0_PWM3	SEL_FO1	SEL_FO0	SEL_MII3PWM	SEL_MII23	SEL_MII3	SEL_MII2	SEL_LCD_COL	SEL_LCD_BW	SEL_F1_PWM3

Bits	Name	Description	R/W	Default
31	IF_SELECT_N	Mask bit for IO_CFG bit IF_SELECT_N	R	-
30:29	reserved	-	R	0x00
28	SEL_E_PWM2_ECLK	Mask bit for IO_CFG bit SEL_E_PWM2_ECLK	R	-
27	reserved	-	R	-
26	SEL_F1_PWM3_ECLK	Mask bit for IO_CFG bit SEL_F1_PWM3_ECLK	R	-
25	SEL_F0_PWM3_ECLK	Mask bit for IO_CFG bit SEL_F0_PWM3_ECLK	R	-
24	SEL_WDG	Mask bit for IO_CFG bit SEL_WDG	R	-
23	SEL_ETM	Mask bit for IO_CFG bit SEL_ETM	R	-
22	SEL_LED_MII3	Mask bit for IO_CFG bit SEL_LED_MII3	R	-
21	SEL_LED_MII2	Mask bit for IO_CFG bit SEL_LED_MII2	R	-
20	SEL_MP	Mask bit for IO_CFG bit SEL_MP	R	-
19	SEL_ENC1	Mask bit for IO_CFG bit SEL_ENC1	R	-
18	SEL_ENC0	Mask bit for IO_CFG bit SEL_ENC0	R	-
17	SEL_E_RPWM2	Mask bit for IO_CFG bit SEL_E_RPWM2	R	-
16	SEL_E_FAILURE2	Mask bit for IO_CFG bit SEL_E_FAILURE2	R	-
15	SEL_E_PWM2	Mask bit for IO_CFG bit SEL_E_PWM2	R	-
14	SEL_F3_PWM3	Mask bit for IO_CFG bit SEL_F3_PWM3	R	-
13	SEL_F2_RPWM3	Mask bit for IO_CFG bit SEL_F2_RPWM3	R	-
12	SEL_F2_FAILURE3	Mask bit for IO_CFG bit SEL_F2_FAILURE3	R	-
11	SEL_F1_RPWM3	Mask bit for IO_CFG bit SEL_F1_RPWM3	R	-
10	SEL_F1_PWM3	Mask bit for IO_CFG bit SEL_F1_PWM3	R	-
9	SEL_F0_FAILURE3	Mask bit for IO_CFG bit SEL_F0_FAILURE3	R	-
8	SEL_F0_PWM3	Mask bit for IO_CFG bit SEL_F0_PWM3	R	-
7	SEL_FO1	Mask bit for IO_CFG bit SEL_FO1	R	-

6	SEL_FO0	Mask bit for IO_CFG bit SEL_FO0	R	-
5	SEL_MII3PWM	Mask bit for IO_CFG bit SEL_MII3PWM	R	-
4	SEL_MII23	Mask bit for IO_CFG bit SEL_MII23	R	-
3	SEL_MII3	Mask bit for IO_CFG bit SEL_MII3	R	-
2	SEL_MII2	Mask bit for IO_CFG bit SEL_MII2	R	-
1	SEL_LCD_COL	Mask bit for IO_CFG bit SEL_LCD_CO	R	-
0	SEL_LCD_BW	Mask bit for IO_CFG bit SEL_LCD_BW	R	-

2.3 RESET – Reset Controller

This register controls the reset functions of the netX chip and indicates the reset state. The reset state shows which resets have occurred, allowing the firmware to detect which resets were active. In order to determine the source of the last reset, the firmware should evaluate and reset these bits during its start sequence. After a power on reset, the RESET_CTRL register is cleared completely.

Changing this register is only possible by the following sequence:
1. read access key
2. write back access key
3. write register

RESET_CTRL – Reset Control Register **0x0010000c**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved					EN_RSTOUTn	RSTOUTn	FIRMW_RES	FIRMW_FLG3	FIRMW_FLG2	FIRMW_FLG1	FIRMW_FLG0	DIS_XPEC3_RES	DIS_XPEC2_RES	DIS_XPEC1_RES	DIS_XPEC0_RES	Reserved								XPEC3_RES	XPEC2_RES	XPEC1_RES	XPEC0_RES	FIRMW_RES	HOST_RES	WDG_RES	RSTINn

Bits	Name	Description	R/W	Default
31:27	reserved	-	R	0x00
26	EN_RSTOUTn	This bit enables the output driver of the reset out pin. When the driver is enabled the physical level can be set by bit 25. When the bit is cleared the reset out pin is in high impendance state	R/W	0
25	RSTOUTn	Programmable reset: This bit controls the signal of the RSTOUT pin of the netX (when enabled by Bit 26). When set, the RSTOUT pin is low, when cleared the pin has high level.	R/W	0
24	FIRMW_RES	Setting this bit activates a firmware reset.	W	0
23	FIRMW_FLG3	Bits 23:20 can be used to save a 4-Bit value which will not be cleared by any reset, except a power on reset.	R/W	0
22	FIRMW_FLG2	Bits 23:20 can be used to save a 4-Bit value which will not be cleared by any reset, except a power on reset.	R/W	0
21	FIRMW_FLG1	Bits 23:20 can be used to save a 4-Bit value which will not be cleared by any reset, except a power on reset.	R/W	0
20	FIRMW_FLG0	Bits 23:20 can be used to save a 4-Bit value which will not be cleared by any reset, except a power on reset.	R/W	0
19	DIS_XPEC3_RES	The xpec3 module can set this bit to avoid being affected by a system reset. For ARM software, this bit is read only .	R	0
18	DIS_XPEC2_RES	The xpec2 module can set this bit to avoid being affected by a system reset. For ARM software, this bit is read only .	R	0
17	DIS_XPEC1_RES	The xpec1 module can set this bit to avoid being affected by a system reset. For ARM software, this bit is read only .	R	0
16	DIS_XPEC0_RES	The xpec0 module can set this bit to avoid being affected by a system reset. For ARM software, this bit is read only .	R	0
15:8	reserved	-	R	0x00

7	XPEC3_RES	reset from XPEC3: This reset is generated by xpec3 reset.	R/W	0
6	XPEC2_RES	reset from XPEC2: This reset is generated by xpec2 reset.	R/W	0
5	XPEC1_RES	reset from XPEC1: This reset is generated by xpec1 reset.	R/W	0
4	XPEC0_RES	reset from XPEC0: This reset is generated by xpec0 reset.	R/W	0
3	FIRMW_RES	reset from FIRMWARE: This reset is activated by arm software. The firmware reset, resets the netX chip completely (system reset) and sets this bit. It can be cleared by trying to set it through a write access.	R/W	0
2	HOST_RES	reset from Host interface: If the internal Host Interface module generates a reset, the netX chip will be completely reset (system reset) and this bit is set. It can be cleared by trying to set it through a write access. There is a 1ms wait state from setting the bit till execution of the reset request.	R/W	0
1	WDG_RES	reset from System WDG: If the netX watchdog is activated and a watchdog timeout occurs the, system will be reset and this bit is set. It can be cleared by trying to set it through a write access.	R/W	0
0	RSTINn	reset from external pin: The netX chip has an external reset input If this reset is activated the netX chip will be completely reset (system reset) and this bit is set. It can be cleared by trying to set it through a write access.	R/W	0

The firmware can disable the internal system reset signals to the XPEC modules, allowing these modules to continue to run even while the chip is performing a reset, however a power on reset will always reset the complete chip and hence also the XPECs.

Code Examples:

Firmware Reset:

```
/* Defines the POKE Macro */
#define POKE(addr, val) (*(volatile unsigned int *)(addr) = (unsigned int)(val))

/* Defines the RESET_CTRL - Register */
#define RESET_CTRL 0x0010000c

int main (void)
  {
    POKE (RESET_CTRL, 0x1000008);      /* Activates a FW-Reset */
  }
```

2.4 NETX_REV – netX Revision

NETX_REV – netX Revision Register **0x00100034**

This register reflects the silicon revision of the netX.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VERSION																															

Bits	Name	Description	R/W	Default
31:0	VERSION	0x41 = 'A' : first version of the netX 0x42 = 'B' : second version of the netX … The boot loader initializes this value.	R	0x00

2.5 WDG – Watchdog

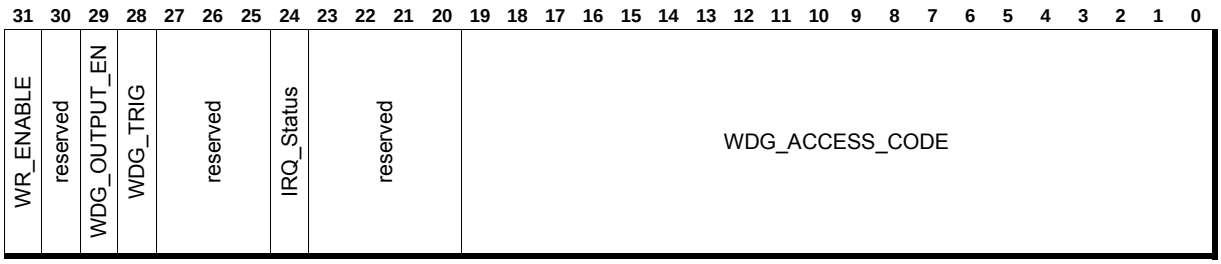
The netX system watchdog is used for supervision of the netX status. After a power on reset, the watchdog timer is disabled. The firmware has to load the watchdog timer with two timeout values in order to arm the watchdog. During operation the timout counter has to be retriggered continuously. There are two timeout values for programming. One is for setting a timeout to generate an interrupt and the other value is for setting a timeout when a system reset occure. The timer has a fixed time base of 100µs. The timeout can be configured in a wide range. The following formula is used to calculate the desired values:

T_{IRQ} = WDG_IRQ_TIMEOUT x 100 µs
T_{RESET} = (WDG_IRQ_TIMEOUT + WDG_RESET_TIMEOUT) x 100 µs

The watchdog works in two stages: When the IRQ timeout counter has reached zero, an Interrupt is generated, to indicate that the watchdog will soon perform a reset and needs attention. When the reset timeout counter reaches zero as well, the netX will be reset by the watchdog.

The timeout register values are always loaded into the watchdog timer when the timer is being retriggered.

WDG_TRIG – Watchdog Trigger Register **0x00100200**



Bits	Name	Description	R/W	Default
31:	WR_ENABLE	Write enable bit for timeout register. As long as this bit is not set all write accesses to the timeout register are ignored.	R/W	0
30	Reserved	-	R	0
29	WDG_ACT_EN	Watchdog Active Enable. If this bit is set, the WDGACT output signal(PIN D16) is enabled.	R/W	0
28	WDG_TRIG	Watchdog trigger bit. Bit must be set to trigger the watchdog counter. When read, this bit is always ‘0’.	R/W	0
27:25	reserved	-	R	0x00
24	IRQ_Status	Interrupt request bit. Set by watchdog when an IRQ timeout has occurred. This bit can be cleared by trying to set it through a write access.	R/W	0
23:20	reserved	-	R	0x00
19:0	WDG_ACCESS_CODE	Watchdog access code for retriggering the watchdog or modifying Bits 31:24. A read access provides the 20 bit code, to be written back with the next write access.	R/W	0x00

Attention: writing bits [31:24] is only possible when writing the correct access code at the same time. Hence only 32 bit accesses to this register are possible. To get the next valid access code it is necessary to read the WDG_TRIG register. The lowest 20 bits provide the new access code, which is generated by a pseudo random counter.

WDG_CNTR – Watchdog Counter 0x00100204

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																WDG_COUNTER															

Bits	Name	Description	R/W	Default
31:17	Reserved	-	R	0x00
16:0	WDG_COUNTER	Current watchdog counter value.	R	0x00

WDG_IRQ_TIMEOUT – Watchdog Interrupt Timeout 0x00100208

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WDG_IRQ_TIMEOUT															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	WDG_IRQ_TIMEOUT	Watchdog interrupt request timeout value.	R/W[1]	0x00

WDG_RESET_TIMEOUT – Watchdog Reset Timeout 0x0010020c

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																WDG_RESET_TIMEOUT															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	WDG_RESET_TIMEOUT	Watchdog reset request timeout value.	R/W[1]	0x00

[1] Write access to the register is only possible when the WR_ENABLE bit in the Watchdog Trigger Register WDG_TRIG is set.

2.6 SYS_STAT – System Status

The general status of a netX based system is usually indicated by the System LED, which can either consist of a dual LED or two single LEDs (see netX Product Brief for further information).

SYS_STA – System Status **0x001034d8**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved						RUN_DRV	RDY_DRV	reserved				RUN_POL	RDY_POL	RUN_IN	RDY_IN	NETX_STA_CODE								HOST_STATE[3:0]				NETX_STATE[3:2]		RUN	RDY

Bits	Name	Description	R/W	Default
31:26	reserved	-	R	0x00
25	RUN_DRV	Driver enable for RUN LED. Enables output driver when set.	R/W	0
24	RDY_DRV	Driver enable for RDY LED. Enables output driver when set.	R/W	0
23:20	reserved	-	R	0x00
19	RUN_POL	Output polarity RUN LED; outsig = RUN exor RUN_POL	R/W	0
18	RDY_POL	Output polarity RUN LED; outsig = RDY exor RDY_POL	R/W	0
17	RUN_IN	Physical input signal level at RUN pin	R	0
16	RDY_IN	Physical input signal level at RDY pin	R	0
15:8	NETX_STA_CODE	netX status code. The netX status codes are software defined. The predefined code values are: F0h: Status after power on reset	R/W	0xf0
7:4	HOST_STATE[3:0]	User defined status signals	R	0x00
3:2	NETX_STATE[3:2]	User defined status signals	R/W	0
1	RUN	Signal Level of the RUN LED output	R/W	0
0	RDY	Signal level of the RDY LED output	R/W	0

Hint: The lower 16 bits can also be accessed from an external host processor through the dual-port memory interface, while Bits (7:4) can be written and all other Bits can be read only.

Code Example:

```
/* Defines the POKE Macro */
#define POKE(addr, val) (*(volatile unsigned int *)(addr) = (unsigned int)(val))

/* Defines the System-Status - Register */
#define SYS_STA 0x001034d8

int main (void)
  {
    /* Turn ON the SYS-LED to Green */
    POKE (SYS_STA, 0x030c0002);

    /* Turn ON the SYS-LED to Red */
    POKE (SYS_STA, 0x030c0001);

    /* Turn OFF the SYS-LED */
    POKE (SYS_STA, 0x030c0000);
  }
```

3 Memory Controller

The Memory Controller can drive SRAM, FLASH and SDRAM without any additional glue logic. The bus width and wait state parameters can be set individually for each memory area, allowing bus widths of 8, 16 or 32 Bit and wait states of up to 63 clock cycles.

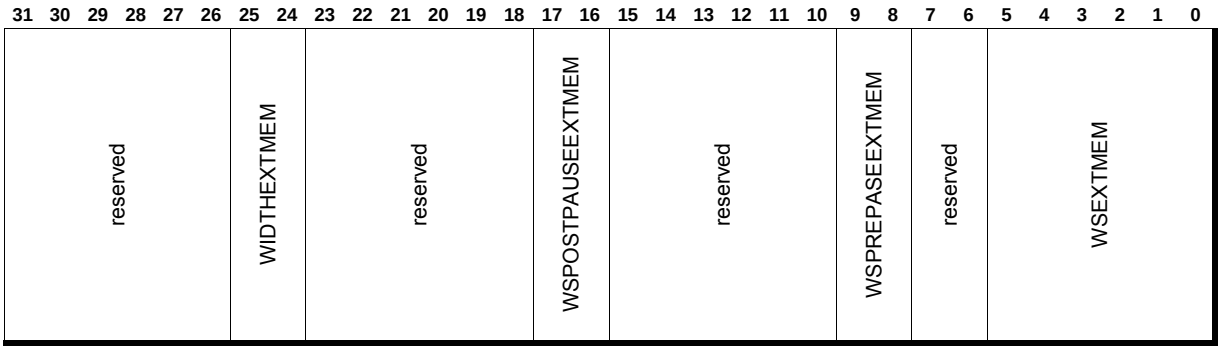
With the 24 Bit address bus and the three chip select signals up to 3 x 16 MByte SRAM or FLASH can be accessed. The address space can be enlarged for 16 Bit and 32 Bit devices to 3 x 32 and 3 x 64 MByte by using the addresses as word or dword addresses. In that case the selection for a single byte is done with the signals MEM_DQM0-3.

The followings is a summary of memory controller registers.

ARM Address	Register Name	Short Description
0x00100100	MEM_SRAM0_CTRL	Memory SRAM Control Register for Chip Select Area 0
0x00100104	MEM_SRAM1_CTRL	Memory SRAM Control Register for Chip Select Area 1
0x00100108	MEM_SRAM2_CTRL	Memory SRAM Control Register for Chip Select Area 2
0x00100140	MEM_SDRAM_CFG_CTRL	Memory SDRAM Configuration Control Register
0x00100144	MEM_SDRAM_TIMING_CTRL	Memory SDRAM Timing Control Register
0x00100148	MEM_SDRAM_MODE	Memory SDRAM Mode Register
0x0010014c	MEM_SDRAM_EXT_MODE	Memory SDRAM Extended Mode Register
0x00100180	MEM_PRIO_TIMESLOT_CTRL	Memory Priority Timeslot Control Register
0x00100184	MEM_PRIO_ACCESS_CTRL	Memory Priority Access Control Register

3.1 MEM_SRAM – Memory Controller for SRAM and FLASH

MEM_SRAM0_CTRL – Memory SRAM Control Register for Chip Select Area 0 0x00100100
MEM_SRAM1_CTRL – Memory SRAM Control Register for Chip Select Area 1 0x00100104
MEM_SRAM2_CTRL – Memory SRAM Control Register for Chip Select Area 2 0x00100108



Bits	Name	Description	R/W	Default
31:26	reserved	-	R	0x00
25:24	WIDTHEXTMEM	Data path width of ExtMem0 area. The external address bus will be shifted according to the selected memory width. So the external maximum address range varies depending on the databus width. 00 : 8 bit A_{ext}[23:0] reflect the byte address 01 : 16 bit A_{ext}[23:0] reflect the word address 10 : 32 bit A_{ext}[23:0] reflect the dword address 11 : reserved	R/W	00
23:18	reserved	-	R	0x00
17:16	WSPOSTPAUSEEXTMEM	additional wait states after access (0 – 3 cycles)	R/W	11
15:10	reserved	-	R	0x00
9:8	WSPREPASEEXTMEM	additional wait states for setup time nCS, Aext to nOE, nWE (0 – 3 cycles)	R/W	11
7:6	reserved	-	R	00
5:0	WSEXTMEM	Wait states (0 – 63 cycles)	R/W	11111

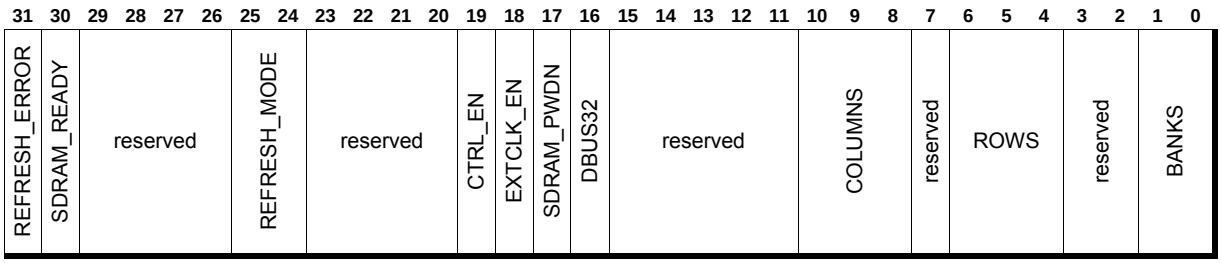
3.2 MEM_SDRAM – Memory Controller for SDRAM

The SDRAM controller can drive all SDRAM Single Data Rate Types from 16 MBit to 512 MBit, providing a 1 GByte address space from 0x80000000 to 0xBFFFFFFF.
The following parameters can be set:

- Number of banks 2, 4, 8
- Number of rows 2k, 4k, 8k, 16k, 32k, 64k
- Number of columns 256, 512, 1k, 2k, 4k, 8k, 16k
- Data width 16 Bit, 32 Bit
- Refresh-mode priority adjustable in 4 steps
- Power save mode SDRAM-Self-refresh-Mode with disabled clock
  switch on / off SDRAM Controller

For further information on the Memory Controller, please refer to the netX_Product_Brief.

MEM_SDRAM_CFG_CTRL – Memory SDRAM Configuration Control Register 0x00100140



Bits	Name	Description	R/W	Default
31	REFRESH_ERROR	When set by the SDRAM controller, this bit indicates that a Refresh could not be performed within the interval determined by the REFRESH_MODE (25:24) and T_REFI (17:16, MEM_SDRAM_TIMING_CTRL) bits, due to high system load. This bit can be cleared by a write access.	R/W	0
30	SDRAM_READY	This bit is set when SDRAM is ready for access. If Bit CTRL_EN is cleared or SDRAM_PWDN is set, SDRAM_READY will automatically be cleared. It will be set after SDRAM has been initialized or after power down wake up.	R/W	0
29:26	reserved	-	R	0x00
25:24	REFRESH_MODE	Refresh priortity mode 00 : fixed interval (T_REFI us, refresh has highest priority) 01 : collect up to 8 refreshes (data access has higher priority than refresh, default) 10 : collect up to 16 refreshes (data access has higher priority than refresh) 11 : collect up to 2047 refreshes (for SDRAM only, data access has higher priority than refresh)	R/W	0x01
23:20	reserved	-	R	0x00
19	CTRL_EN	SDRAM controller enable The MEM_SDRAM_TIMING_CTRL and the MEM_SDRAM_MODE registers can only be written while this bit is cleared. Upon setting this bit, the SDRAM controller will be enabled, running the following SDRAM initialization procedure (100 MHz, t_clk = 10 ns). NOP (200 us = 20,000 t_clk, running sd_clk (if extclk_en), n_cs low, cke high)	R/W	0

		PRECHARGE ALL NOP (160 ns = 16 t_clk) 7x (AUTO REFRESH, NOP (310 ns = 31 t_clk)) AUTO REFRESH NOP (220 ns = 22 t_clk) LOAD MODE REGISTER (with settings done by these configuration registers) NOP (40 ns = 4 t_clk) ACTIVATE (for first access, if requested, sdram_ready will be set to 1 here) Accesses requested before sdram_ready is 1 will be blocked (no ready). The external SDRAM-clk will not run if the controller is disabled.		
18	EXTCLK_EN	external SDRAM clock enable	R/W	0
17	SDRAM_PWDN	SDRAM Power Down If this bit is set, the Controller will move SDRAM to power down self refresh mode (no data loss) and stop the external SDRAM clock.	R/W	0
16	DBUS32	SDRAM data bus width 0 : SDRAM bus is 16 bit wide. (default) 1 : SDRAM bus is 32 bit wide.	R/W	0
15:11	reserved	-	R	0x00
10:8	COLUMNS	column address coding. 000 : 256 (A0..A7) (default) 001 : 512 (A0..A8) 010 : 1k (A0..A9) 011 : 2k (A0..A9,A11) 100 : 4k (A0..A9,A11,A12) 101 : 8k (A0..A9,A11..A13) (for future devices) 110 : 16k (A0..A9,A11..A14) (for future devices) 111 : reserved	R/W	
7	reserved	-	R	0
6:4	ROWS	row address coding. 000 : 2k (A0..A10) (default) 001 : 4k (A0..A11) 010 : 8k (A0..A12) 011 : 16k (A0..A13) 100 : 32k (A0..A14) (for future devices) 101 : 64k (A0..A15) (for future devices) 11x : reserved	R/W	0x00
3:2	reserved	-	R	0x00
1:0	BANKS	bank address coding; mapped to A18 (=BA2), A17 (=BA1) and A16(BA0). 00 : 2 01 : 4 (default) 10 : 8 11 : reserved	R/W	0x01

MEM_SDRAM_TIMING_CTRL – Memory SDRAM Timing Control Register 0x00100144

This register can only be modified, while the SDRAM-Controller is disabled (Bit CTRL_EN of MEM_SDRAM_CFG_CTRL is cleared).

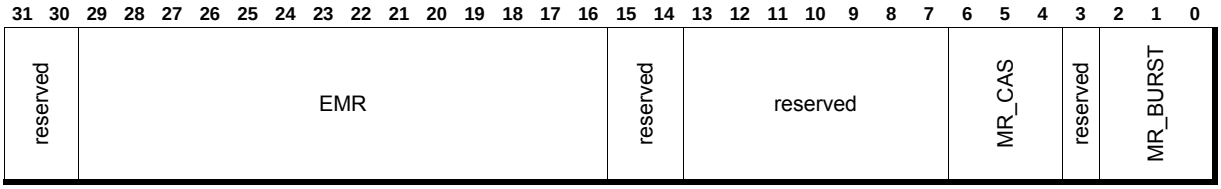
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved			BYPASS_NEG_DELAY	reserved	DATA_SAMPLE_PHASE			MEM_SDCLK_SSNEG	MEM_SDCLK_PHASE			reserved		T_REFI		T_RFC				reserved	T_RAS			T_RP		T_WR		reserved		T_RCD	

Bits	Name	Description	R/W	Default
31:29	reserved	-	R	00
28	BYPASS_NEG_DELAY	0 : use phase shifted (negative delayed) SDRAM loopback clock for data sampling. 1 : bypass phase shift logic for SDRAM data sampling use SDRAM loopback clock for data sampling (for system clock frequency ≤ 80 MHz)	R/W	0
27	reserved	-	R	0
26:24	DATA_SAMPLE_PHASE	0..5: adjustable phase-shift for data sampling SDRAM loopback clock (clk_sdloopback) depending external capacitive load and SDRAM access time (t_AC). The phase can be shifted in 1.25 ns steps. clk_sdloopback will internally rise (sample SDRAM read data) at the data_sample_phase+4th clk400 edge after rise of external mem_sdclk (including external capacitive load). For correct settings, the delays depending on external capacitive have to be considered. Data sampling has to be done at least 8 ns after internal changes of SDRAM ctrl-signals (mem_sd*-signals, driven by clk_memsig) .	R/W	011
23	MEM_SDCLK_SSNEG	Bit must be set	R/W	1
22:20	MEM_SDCLK_PHASE	0..5: adjustable phase-shift for external SDRAM clock depending on external capacitive load on mem_sdclk-signal to match SDRAM ctrl-signal setup times. The phase can be shifted in 1.25 ns steps. mem_sdlk will internally rise at the mem_sdclk_phase+2nd clk400 edge after internal changes of SDRAM ctrl-signals (mem_sd*-signals, driven by clk_memsig), where the 1st egde is defined by the mem_sdclk_ssneg-bit. For correct settings, delays depending on external capacitive have to be considered.	R/W	000
19:18	reserved	-	R	00
17:16	T_REFI	Average Periodic refresh interval (time = 3.90 us * 2^T_REFI) 00 : 3.90 us 01 : 7.80 us (default) 10 : 15.60 us 11 : 31.20 us	R/W	01

15:12	T_RFC	REFRESH to Command time Minimum time interval between AUTOREFRESH command and next command (next AUTOREFRESH or ACTIVE). In some SDRAM datasheets also called auto refresh period. 0x00 : 4 clks 0x01 : 5 clks : 0x0f : 19 clks (default)	R/W	1111
11	reserved	-	R	0
10:8	T_RAS	ACTIVE to PRECHARGE command time Minimum time interval between ACTIVE command (row adress and RAS asserted) and PRECHARGE command (deactivation of row). In some SDRAM datasheets also called row active time. 000 : 3 clks 001 : 4 clks : 111 : 10 clks (default)	R/W	111
7:6	T_RP	PRECHARGE command period time (Precharge to command) Minimum time interval between PRECHARGE (deactivation of row) and next command. 00 : 1 clk 01 : 2 clks 10 : 3 clks (default) 11 : reserved	R/W	11
5:4	T_WR	WRITE recovery time Minimum time interval between last write data (data in) and Precharge command (deactivation of row). 00 : 1 clk 01 : 2 clks 10 : 3 clks (default) 11 : reserved	R/W	11
3:2	reserved	-	R	00
1:0	T_RCD	ACTIVE to READ or WRITE time Minimum time interval between ACTIVE command (row adress and RAS asserted) and READ or WRITE command (column address and CAS asserted). In some SDRAM datasheets also called RAS-to-CAS time. This parameter will also be used to configure t_RRD (minimum time between consecutive ACTIVE commands to different banks) 00 : 1 clk 01 : 2 clks 10 : 3 clks (default) 11 : reserved	R/W	11

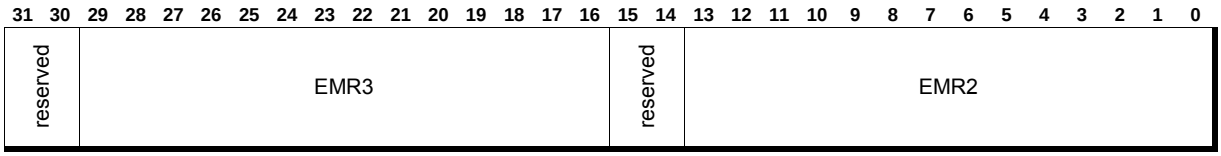
MEM_SDRAM_MODE – Memory SDRAM Mode Register 0x00100148

The SDRAM Controller will initialize the Mode Register of the connected SDRAM component(s) with the value, stored in this register, after enabling the SDRAM Controller (200 µs SDRAM memory initialization procedure).



Bits	Name	Description	R/W	Default
31:30	reserved	-	R	00
29:16	EMR	Extended SDRAM Mode Register. These bits are reserved and must not be modified!	R/W	0x00
15:14	reserved	-	R	00
13:7	reserved	SDRAM Mode Register. Reflect bits [13:7] of mode register of SDRAM component. These bits are reserved and must not be modified!	R/W	00 0000 0
6:4	MR_CAS	SDRAM Mode Register. Reflect CAS latency bits of mode register of SDRAM component (Bits [6:4]). The controller supports two settings: 010: CL2 011: CL3 (default)	R/W	011
3	reserved	SDRAM Mode Register. Reflects bit [3] of mode register of SDRAM component. This bit is reserved and must not be modified!	R/W	0
2:0	MR_BURST	SDRAM Mode Register. Reflect Burst Length Bits of mode register of SDRAM component (Bits [2:0]). The controller, which is fixed to 4 DWORD Bursts, only supports two settings, which are **automatically** done by the DBUS32 Bit in the MEM_SDRAM_CFG_CTRL register: 010: Burst Length 4 (data width 32 Bit) 011: Burst Length 8 (data width 16 Bit, default)	R	011

MEM_SDRAM_EXT_MODE – Memory SDRAM Extended Mode Register 0x0010014c



Bits	Name	Description	R/W	Default
31:30	reserved	-	R	00
29:16	EMR3	Extended Mode Register 3 for SDRAM devices These bits are reserved and must not be modified!	R	0x00
15:14	reserved	-	R	00
13:0	EMR2	Extended Mode Register 2 for SDRAM devices These bits are reserved and must not be modified!		0x00

3.3 MEM_PRIO – Memory Priority Controller

MEM_PRIO_TIMESLOT_CTRL – Memory Priority Timeslot Control Register 0x00100180

Memory interface master timeslot priority control register.

Note:
Any master can access in one timeslot ((ts_accessrate_mX*ts_length_mX)/64) + 1 times (i.e. at maximum (ts_accessrate_mX)/64 bandwidth on external memory bus, ts_accessrate_mX is programmed by MEM_PRIO_ACCESS_CTRL register).

Priority control will watch data accesses on external memory data bus (SDRAM and non SDRAM), including pauses on non SDRAM-accesses, not including control commands to SDRAM. Any master requesting more accesses will be forced to wait for the remaining timeslot.

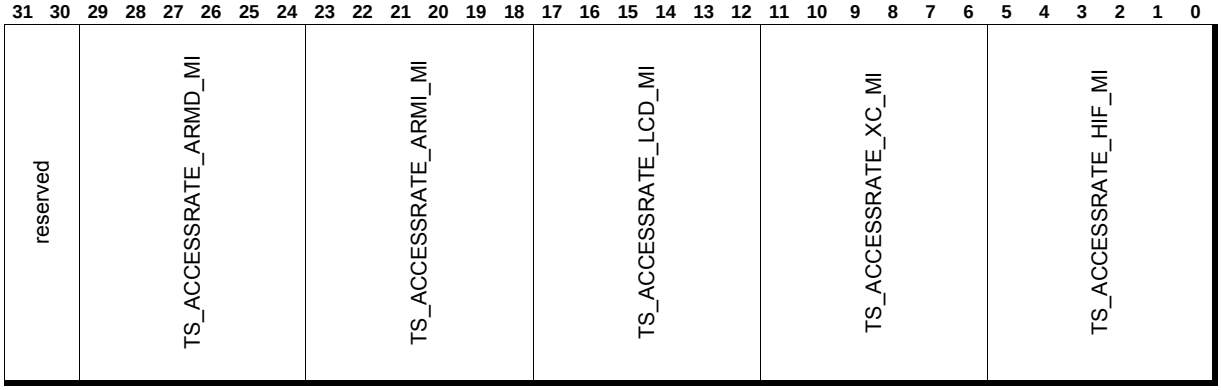
master channel m0: Host Bus Interface (highest priority)
master channel m1: XC
master channel m2: LCD-Controller
master channel m3: ARM instruction channel
master channel m4: ARM data channel (lowest priority)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved													TS_LENGTH_ARMD_MI			reserved		TS_LENGTH_ARMI_MI		reserved		TS_LENGTH_LCD_MI		reserved		TS_LENGTH_XC_MI		reserved		TS_LENGTH_HIF_MI	

Bits	Name	Description	R/W	Default
31:19	reserved	-	R	0x00
18:16	TS_LENGTH_ARMD_MI	0..7: the timeslot of master m4 (ARM data fetch) is on external memory interface 64*2^ts_length_ARMD_mi system clock cycles		111
15	reserved	-	R	0
14:12	TS_LENGTH_ARMI_MI	0..7: the timeslot of master m3 (ARM instruction fetch) is on external memory interface 64*2^ts_length_ARMI_mi system clock cycles		111
11	reserved	-	R	0
10:8	TS_LENGTH_LCD_MI	0..7: the timeslot of master m2 is on external memory interface 64*2^ts_length_LCD_mi system clock cycles		111
7	reserved	-	R	0
6:4	TS_LENGTH_XC_MI	0..7: the timeslot of master m1 is on external memory interface 64*2^ts_length_XC_mi system clock cycles		111
3	reserved	-	R	0
2:0	TS_LENGTH_HIF_MI	0..7: the timeslot of master m0 is on external memory interface 64*2^ts_length_HIF_mi system clock cycles		111

MEM_PRIO_ACCESS_CTRL – Memory Priority Access Control Register 0x00100184

Control Register for master channel accesses per timeslot on external memory interface.
This register may be partially locked by the LOCK_MEM_PRIO_CTRL register in ASIC_CTRL address area.
For detailed priority controlling read note at MEM_PRIO_TIMESLOT_CTRL register description.



Bits	Name	Description	R/W	Default
31:30	reserved	-	R	00
29:24	TS_ACCESSRATE_ARMD_MI	0..63: master m4 (ARM data fetch) is allowed to request ((ts_accessrate_ARMD_mi*ts_length_ARMD_mi)/64) + 1 accesses on external memory	R/W	111111
23:18	TS_ACCESSRATE_ARMI_MI	0..63: master m3 (ARM instruction fetch) is allowed to request ((ts_accessrate_ARMI_mi*ts_length_ARMI_mi)/64) + 1 accesses on external memory	R/W	111111
17:12	TS_ACCESSRATE_LCD_MI	0..63: master m2 is alowed to request ((ts_accessrate_LCD_mi*ts_length_LCD_mi)/64) + 1 accesses on external memory	R/W	111111
11:6	TS_ACCESSRATE_XC_MI	0..63: master m1 is allowed to request ((ts_accessrate_XC_mi*ts_length_XC_mi)/64) + 1 accesses on external memory	R/W	111111
5:0	TS_ACCESSRATE_HIF_MI	0..63: master m0 is allowed to request ((ts_accessrate_HIF_mi*ts_length_HIF_mi)/64) + 1 accesses on external memory	R/W	111111

4 Extension Bus

In order to use the Extension Bus, the netX host interface must be configured for 'Extension Bus Mode' in register DPM_ARM_IF_CFG0. Further, it is also important to configure each required signal line of the Extension Bus for host interface mode in registers DPM_ARM_IO_MODE0 and DPM_ARM_IO_MODE1(please refer to section 5.2:DPM_ARM – Dual-Port Memory ARM Side). Unused signal lines of the Extension Bus may be left configured as I/O mode (e.g. unused upper address lines or data lines 15-8 when using an 8 Bit device only).

Please note, that the configuration for Extension Bus is done separately for each chip select, resulting in a set of four identical registers. For details on the Extension bus timing parameters, please refer to the appropriate chapter of the “netX Product Brief” and the netX “Technical Reference Guide” documents.

EXT_CFG_CS0 – Extension Bus Configuration Chip Select 0 **0x00103610**
EXT_CFG_CS1 – Extension Bus Configuration Chip Select 1 **0x00103614**
EXT_CFG_CS2 – Extension Bus Configuration Chip Select 2 **0x00103618**
EXT_CFG_CS3 – Extension Bus Configuration Chip Select 3 **0x0010361c**

31 30 29	28 27 26	25 24 23	22 21 20	19 18 17	16 15 14 13 12	11 10 9 8 7	6	5	4	3	2	1	0
Talewidth	Tadrhold	Tcson	Trdon	Twron	Trdwroff	Trdwrcyc	WAIT_POLARITY	WAIT_EN	nRD_MODE	DS_MODE	nWR_MODE	8/16BIT	CS_EN

Bits	Name	Description	R/W	Default
31:29	Talewidth	Delay time from start of cycle until ALE inactive	R/W	0x00
28:26	Tadrhold	Delay time from start of cycle until invalid address at the data bus.	R/W	0x00
25:23	Tcson	Delay time from start of cycle until Chip Select low	R/W	0x00
22:20	Trdon	Delay time from start of cycle until RD low in system clocks	R/W	0x00
19:17	Twron	Delay time from start of cycle until WR low in system clocks	R/W	0x00
16:12	Trdwroff	Delay time from start of cycle until RD and WR inactive for accesses in system clocks	R/W	0x00
11:7	Trdwrcyc	Set the end of an access cycle in system clocks The value 0x00 and 0x01 are interpreted as 0x20	R/W	0x00
6	WAIT_POLARITY	WAIT input polarity sampled before rising edge of RD/WR end to lengthen a cycle.	R/W	0
5	WAIT_EN	External Wait Enable	R/W	0
4	nRD_MODE	nRD mode. When this bit is zero the interface pin function as normal low active read signal. If the bit is set the pin signals the data direction for each transfer, RD/#WR = DIR	R/W	0
3	DS_MODE	Data Strobe Mode 0 nWR/nWRL/nWRH 1 nDS/nDSL/nDSH	R/W	0
2	nWR_MODE	nWR mode. If this bit is set the nWR signal is only active for low byte write accesses (nWRL).	R/W	
1	8/16BIT	Interface width selection: 0 = 8 Bit, 1 = 16 Bit	R/W	0
0	CS_EN	Chip Select Enable	R/W	0

5 Dual-Port Memory

In order to use the netX Dual-port memory interface, the netX host interface must be configured to 'µP Bus Mode' in the DPM_ARM_IF_CFG0 register. Further, it is also important to configure each required signal line of the DPM to host interface mode in the DPM_ARM_IO_MODE0 and DPM_ARM_IO_MODE1 registers. Unused signal lines of the DPM may be left configured as I/O mode (e.g. data lines 15-8 when operating in 8 Bit mode only).

For detailed information about DPM, please refer to the appropriate chapter of the netX_Product_Brief and the netX_Technical_Reference_Guide documents.

5.1 DPM_HOST – Dual-Port Memory Host Side

The following two tables show the Dual-port memory structure as it is seen from the host side. The first table shows the global configuration area with control and status registers, always located between 0xfe00 and 0xffff.

Host Address	Arm Address	Register Name	Short Description
0xfffc	0x001031fc	reserved	-
0xfff8	0x001031f8	reserved	-
0xfff4	0x001031f4	reserved	-
0xfff0	0x001031f0	DPM_HOST_INT_EN0	Interrupt Enable 0
0xffec	0x001031ec	reserved	-
0xffe8	0x001031e8	reserved	-
0xffe4	0x001031e4	reserved	-
0xffe0	0x001031e0	DPM_HOST_INT_STAT0	Interrupt Status 0
0xffdc	0x001031dc	DPM_HOST_RESET_REQ	Reset Request
0xffd8	0x001031d8	DPM_HOST_SYS_STAT	System Status
0xffd4	0x001031d4	DPM_HOST_TMR_START_VAL	Timer Start Value
0xffd0	0x001031d0	DPM_HOST_TMR_CTRL	Timer Control
0xffcc	0x001031cc	reserved	-
0xffc8	0x001031c8	DPM_HOST_WDG_ARM_TIMEOUT	Watchdog ARM Timeout
0xffc4	0x001031c4	DPM_HOST_WDG_HOST_TRIG	Watchdog Host Trigger
0xffc0	0x001031c0	DPM_HOST_WDG_HOST_TIMEOUT	Watchdog Host Timeout, read only
0xffbc - 0xfe00	0x001031bc-0x00103100	reserved	-

Address space from 0x0000 to 0xfdff is data memory area. *The following table is only an example, as the actual DPM memory layout is user programmable and hence completely firmware dependant.*

Host Address	Short Description		
0xfdff - 0xa000	Unused area with nearly 24k bytes		
0x9fff - 0x9ffc	Host→netX Handshake 16 bit	netX→Host Handshake 16 bit	
0x9ffb – 0x3800	Data Memory Block 8 with nearly 10k bytes		
0x37ff – 0x3000	Data Memory Block 7 with 2k bytes		
0x6fff - 0x6000	Data Memory Block 6 with 4k bytes		
0x5fff - 0x5000	Data Memory Block 5 with 4k bytes		
0x4fff - 0x4ffc	Host→netX Handshake	netX→Host Handshake	Handshake Data Memory 2 Byte

0x4ffb - 0x2800	Data Memory Block 4 with nearly 10k bytes
0x27ff - 0x2000	Data Memory Block 3 with 2k bytes
0x1fff - 0x1000	Data Memory Block 2 with 4k bytes
0x0fff - 0x0000	Data Memory Block 1 with 4k bytes

The DPM host side registers are described following:

DPM_HOST_INT_EN0 – DPM Host Side Interrupt Enable 0 0xfff0

31	30	29	28 27	26	25	24 23 22 21 20 19 18 13 16	15 14 13 12 11 10 9 8 3 6 5 4 3 2 1 0
INT_EN	MEM_LCK	WDG_NETX	reserved	SYS_STA	TMR	reserved	HS_EVENT[15:0]

Bits	Name	Description	R/W	Default
31	INT_EN	Global interrupt enable	R/W	0
30	MEM_LCK	Memory lock interrupt enable	R/W	0
29	WDG_NETX	Watchdog timeout of netX supervision interrupt enable	R/W	0
28:27	reserved	-	R	0x00
26	SYS_STA	System status change interrupt enable	R/W	0
25	TMR	Timer interrupt enable	R/W	0
24:16	reserved	-	R	0x00
15	HS_EVENT15	Handshake event 15 interrupt enable	R	0
14	HS_EVENT14	Handshake event 14 interrupt enable	R	0
13	HS_EVENT13	Handshake event 13 interrupt enable	R	0
12	HS_EVENT12	Handshake event 12 interrupt enable	R	0
11	HS_EVENT11	Handshake event 11 interrupt enable	R	0
10	HS_EVENT10	Handshake event 10 interrupt enable	R	0
9	HS_EVENT9	Handshake event 9 interrupt enable	R	0
8	HS_EVENT8	Handshake event 8 interrupt enable	R	0
7	HS_EVENT7	Handshake event 7 interrupt enable	R	0
6	HS_EVENT6	Handshake event 6 interrupt enable	R	0
5	HS_EVENT5	Handshake event 5 interrupt enable	R	0
4	HS_EVENT4	Handshake event 4 interrupt enable	R	0
3	HS_EVENT3	Handshake event 3 interrupt enable	R	0
2	HS_EVENT2	Handshake event 2 interrupt enable	R	0
1	HS_EVENT1	Handshake event 1 interrupt enable	R	0
0	HS_EVENT0	Handshake event 0 interrupt enable	R	0

This register controls the host side interrupt behaviour of the chip. The interrupt status flags will be always set upon the corresponding event, but the physical interrupt line will only be active when the enable bit is set. All interrupt requests can be enabled or disabled independently and are wired together to the global interrupt request for the host interface, however no interrupt will be generated unless the global interrupt enable bit (bit 31) is also set.

DPM_HOST_INT_STAT0 – DPM Host Side Interrupt Status 0 0xffe0

31	30	29	28 27	26	25	24	23 22 21 20 19 18 13 16	15 14 13 12 11 10 9 8 3 6 5 4 3 2 1 0
INT_REQ	MEM_LCK	WDG_NETX	reserved	SYS_STA	TMR	reserved	IRQ_VECTOR[7:0]	HS_EVENT[15:0]

Bits	Name	Description	R/W	Default
31	INT_REQ	Interrupt request	R	0
30	MEM_LCK	Flag of Memory lock interrupt	R/W	0
29	WDG_NETX	Flag of Watchdog timeout netX supervision interrupt	R/W	0
28:27	reserved	-	R	0x00
26	SYS_STA	Flag of System status change interrupt	R/W	0
25	TMR	Flag of Timer interrupt	R/W	0
24	reserved	-	R	0x00
23:16	IRQ_VECTOR[7:0]	Interrupt Vector according to the status flags 0x00 : no interrupt 0x10 : handshake event interrupt 0 0x11 : handshake event interrupt 1 0x12 : handshake event interrupt 2 0x13 : handshake event interrupt 3 0x14 : handshake event interrupt 4 0x15 : handshake event interrupt 5 0x16 : handshake event interrupt 6 0x13 : handshake event interrupt 3 0x18 : handshake event interrupt 8 0x19 : handshake event interrupt 9 0x1a : handshake event interrupt 10 0x1b : handshake event interrupt 11 0x1c : handshake event interrupt 12 0x1d : handshake event interrupt 13 0x1e : handshake event interrupt 14 0x1f : handshake event interrupt 15 0x20 - 0x5f : reserved 0x60 : Memory lock interrupt 0x61 : Watchdog timeout ARM supervision 0x62 - 0x6f reserved 0x70 : System status change interrupt 0x71 : Timer interrupt 0x72 - 0xff : Reserved	R	0x00
15:0	HS_EVENT[15:0]	Flag of Handshake event [15:0] interrupt	R/W	0x00

This register does not only show the interrupt status flag of each interrupt event, but also provides the interrupt vector. The flags are always set when the corresponding interrupt events occur, regardless of the interrupt enable flag, but the physical interrupt which is reflected by the interrupt vector, will only be generated when the corresponding enable flag is set. An interrupt status flag can be cleared by trying to set the flag through a write access. When a handshake register cell is read, the corresponding interrupt status flag will be cleared automatically. The interrupt flag will be not cleared when the interrupt event reoccurs at the same time.

DPM_HOST_RESET_REQ – DPM Host Side Reset Request 0xffdc

The host system can cause a chip reset by writing a special sequence to this register. After writing the last word of the sequence, the netX will perform a reset after a delay of 1 ms.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																DEL_CNT[16:8]								DEL_CNT[7:0] CONTROL[7:0]							

Bits	Name	Description	R/W	Default
31:17	reserved	-	R	0x00
16:8	DEL_CNT[16:8]	Delay counter value (not linear counting)	R	0x00
7:0	DEL_CNT[7:0] CONTROL[7:0]	If the written value is not equal to the delay counter then the delay counter is reset to 0000h. Writing the same value as read (only low byte) will switch one cycle of delay counter step. When the delay counter switches eight times and reaches the 80h value the delay counter will start to count up automatically and could not be stopped. After 1 ms the netX performs a hardware reset.	R/W	0x00

DPM_HOST_SYS_STAT – DPM Host Side System Status 0xffd8

The netX provides an ARM side system status register, called 'SYS_STAT', which shows the status of the netX chip. Some bits of this register some can be controlled from netX side and some bits can be controlled from host side. The DPM_HOST_SYS_STAT register is an alias mapping to the SYS_STAT register, but only the lower 16 bits can be accessed, and only bits(7:4) can be written from the host side. It is possible to enable a system status interrupt for the host system, then any write access from netX side to the data will generate an interrupt to the host.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																NETX_STA_CODE[7:0]								HOST_STATE [3:0]				NETX_STATE[3:2]		RUN	RDY

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:8	NETX_STA_CODE	netX status code. The netX status codes are software defined. The predefined code values are: F0h: Status during and after power on reset	R	0xf0
7:4	HOST_STATE[3:0]	User defined host status signals	R/W	0x00
3:2	NETX_STATE[3:2]	User defined netX status signals	R	0x00
1	RUN	Signal level of the RUN LED output	R	0
0	RDY	Signal level of the RDY LED output	R	0

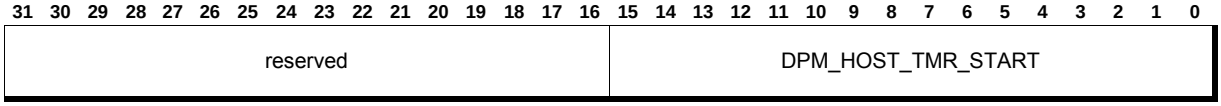
See also chapter 'SYS_STAT – System Status' on page 16.

DPM_HOST_TMR_START_VAL – DPM Host Side Timer Start Value 0xffd4

The host has the possibility to use this timer for cyclic interrupt generation or as a count down timer. The timer is a 16 bit count down timer and can generate an interrupt at the host system side. There are two different timer modes. In one mode the timer will stop after counting down to zero and set the interrupt event only one time. The other mode will set the interrupt too but the timer will automatically be reloaded with the configured timeout value, allowing the generation of a cyclic interrupt.

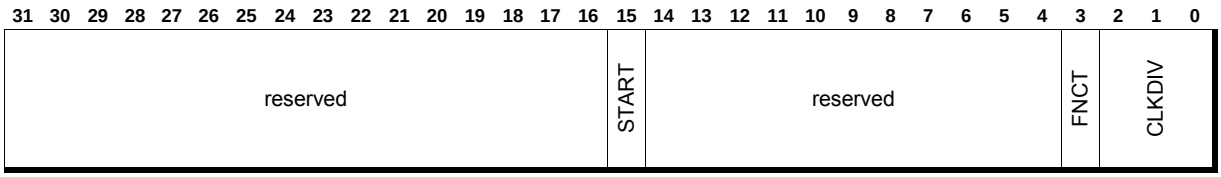
The first step in programming the timer is to set the timer start count value DPM_HOST_TMR_START in the 'DPM_HOST_TMR_START_VAL' register. The timer operation is controlled by the 'DPM_HOST_TMR_CTRL' register. When setting the start flag, the counter will be loaded with the timer start count value. This can also be done during timer operation to reload the timer.

The clock divider field provides the possibility to select different clock time bases. The divider for the time base generation will be reset to its default value, when the counter start flag is being set or the divider value is being changed.



Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	DPM_HOST_TMR_START	Timer start value for count down or cyclic reload. The timeout time can calculated by the following formula: $T_{TIMEOUT}$ = TMR_START x Time base	R/W	0x00

DPM_HOST_TMR_CTRL – DPM Host Side Timer Control 0xffd0



Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15	START	Start Timer Count. Writing a one bit will load the timer with the timer start value and then the timer counts down to zero. A zero bit stops the timer from counting.	R/W	0
14:4	reserved	-	R	0x00
3	FNCT	Timer mode function 0 : Timer stops after count down to zero 1 : Timer start again with start value (reload mode)	R/W	0
2:0	CLKDIV	Timer clock divider 000 : 100 µs 001 : 10 µs 010 : 1 µs 011 : 100 ns 1xx : Reserved (100µs)	R/W	0x00

DPM_HOST_WDG_ARM_TIMEOUT – DPM Host Side Watchdog ARM 0xffc8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																TIMEOUT_VAL															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	TIMEOUT_VAL	Timeout value $T_{TIMEOUT}$ = TIMEOUT_VAL x 100 µs	R/W	0x00

DPM_HOST_WDG_HOST_TRIG – DPM Host Side Watchdog Host Trigger 0xffc4

This 16 bit count down watchdog timer allows supervision of the host system. The netX chip can set the timeout value, which can read by the host system. It is possible for the netX to change the timeout value at any time. The watchdog timer can be disabled by setting the timeout value to zero, which is the default reset value.

When the host watchdog is enabled, the host system has to trigger the timer during the watchdog timeout period to avoid setting the host timeout event which cause a host timeout interrupt at netX side. When triggering the watchdog timer by the host system the timeout value will be always loaded into the timer register. Setting another timeout value from netX side will not change the actual count value of the watchdog timer. Only when the host system triggers the watchdog timer, the timeout data value will be loaded into the watchdog timer. So after configuration by the netX, the host system has to trigger always one time the watchdog timer to start the supervision.

The time base of the watchdog timer is fixed to 100 µs. The timeout range can selected between 100µs and 6.50 sec. The following formular gives the timeout value according to the register setting.

TIMEOUT_HOST = TIMEOUT_VALUE x 100 µs

For triggering the watchdog timer there is a register which can be accessed from the host system. Triggering the watchdog timer is only possible with a special access code. This access code is generated by a pseudo random generator. This ensures save operation. The following sequence must be done to trigger the watchdog timer.

1.) reading the DPM_HOST_WDG_HOST_TRIG register to get the next WDG_TRIGGER_CODE
2.) writing back the new watchdog access code to the DPM_HOST_WDG_HOST_TRIG register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								WDG_TRIGGER_CODE							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	WDG_TRIGGER_CODE	Watchdog trigger code (read-write back-value)	R/W	0x00

DPM_HOST_WDG_HOST_TIMEOUT – DPM Host Side Watchdog Host Timeout 0xffc0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																TIMEOUT_VAL															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	TIMEOUT_VAL	Timeout value	R	0x00

Handshake Register

For the synchronisation of the data transfer between the host system with user application and the netX chip, 16 handshake register pairs with interrupt capability are available. It is possible to select 8 or 16 bit width handshake register mode. The Dual-port memory offset address of each register pair is set by the netX. If the handshake register pair is located at the upper end of a data memory block then the structure is equal to standard Dual-port memories which are often used to couple two microprocessor systems.

The base address of the 8 or 16 bit width handshake register pairs in the Dual-port memory must be a DWORD aligned address. These address can be set anywhere in the Dual-port memory address range. Only the control block at top area of 512 bytes can not be used. The absolute address in the Dual-port memory of the handshake register depends on the register width. The following example shows the address generation.

8 bit register width ➔ Host _ netX handshake register = BASE_ADDRESS[15:2] + 03h
netX _ Host handshake register = BASE_ADDRESS[15:2] + 02h
16 bit register width ➔ Host _ netX handshake register = BASE_ADDRESS[15:2] + 00h
netX _ Host handshake register = BASE_ADDRESS[15:2] + 02h

Any access of the host to the handshake registers are not mapped into the netX data memory area. These special accesses are directly mapped into the handshake registers. If the 8 bit handshake register mode is selected any access from host side to memory location at byte 0 and byte 1 of the DWORD aligned handshake pair base address will be also mapped into the handshake registers.

After a netX reset the handshake register function is generally disabled. Any write access from the host interface to the upper register of each handshake pair sets an interrupt status flag at the netX side. A write access from netX into the lower register of each handshake pair sets the interrupt status at the host system side. With the interrupt mask registers all interrupt requests as the result of active interrupt status flags can be individually enabled or disabled. The interrupt flag will be cleared when the corresponding interface side reads the handshake cell. Another possibility for releasing the interrupts is to write a one bit to the interrupt status flag register.

The netX chip can access all handshake registers by different memory addresses. The internal hardware logic recognized the accesses to the handshake register only by the physical memory access address. When the netX makes a read access to the netX _ host handshake registers at the memory address location accessed by the host system, the host interrupt will also be cleared like reading from host interface. This function is only for debugging. Tue to programmable Dual-port memory structure it might be possible that there are register or data memory block overlapping. A hardware priority decoder of all data memory blocks and register will solve such situation. The following table show the priority encoding of the memory blocks:

Priority Level	Memory Block Name
Highest	Global Control Block
	Handshake Register Pair 0
	.
	.
	Handshake Register Pair 15
lowest	Data Memory Block [7:0]

Dual-port memory data priority level 1

The detailed description of the DPM is shown in the manual netX DPM Interface.

DPM Host Side Handshake Register Pair (Host→netX and netX→Host)

The location of each handshake register pair is programmed from ARM side and could be located at any dword address in the area 0x0000 until 0xfdfc. The maximum handshake register pairs are limited to 16.

16 Bit Handshake Register (Host→netX / netX→Host) Area: 0x0000 - 0xfdfc

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Host→NETX_DATA [15:0]																NETX→Host_DATA [15:0]															

Bits	Name	Description	R/W	Default
31:16	HOST→NETX_DATA[15:0]	Handshake Data Flags host to netX [15:0]	R/W	0x00
15:0	NETX→HOST_DATA[15:0]	Handshake Data Flags netX to host [15:0]	R	0x00

8 Bit Handshake Register (Host→netX / netX→Host) Area: 0x0000 - 0xfdfc

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HOST→NETX_DATA [7:0]								NETX→HOST_DATA [7:0]								DATA_MEMORY[15:8]								DATA_MEMORY[7:0]							

Bits	Name	Description	R/W	Default
31:24	HOST→NETX_DATA[7:0]	Handshake Data Flags host to netX [7:0]	R/W	0x00
23:16	NETX→HOST_DATA[7:0]	Handshake Data Flags netX to host [7:0]	R	0x00
15:8	DATA_MEMORY[15:8]	Data Memory [15:8]	R/W	0x00
7:0	DATA_MEMORY[7:0]	Data Memory [7:0]	R/W	0x00

5.2 DPM_ARM – Dual-Port Memory ARM Side

The following table is a summary of the Dual-Port Memory registers.

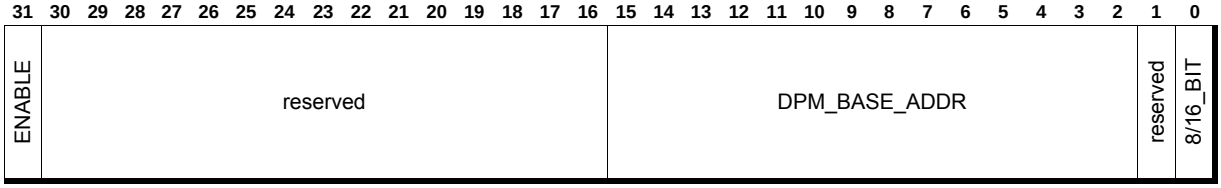
ARM Address	Register Name	Short Description
0x00103ffc - 0x00103800	reserved	-
0x001037fc - 0x00103700	reserved	-
0x001036fc - 0x001036c0	Reserved	-
0x001036bc	DPM_ARM_HS_CTRL15	Handshake Control Register 15
0x001036b8	DPM_ARM_HS_CTRL14	Handshake Control Register 14
0x001036b4	DPM_ARM_HS_CTRL13	Handshake Control Register 13
0x001036b0	DPM_ARM_HS_CTRL12	Handshake Control Register 12
0x001036ac	DPM_ARM_HS_CTRL11	Handshake Control Register 11
0x001036a8	DPM_ARM_HS_CTRL10	Handshake Control Register 10
0x001036a4	DPM_ARM_HS_CTRL9	Handshake Control Register 9
0x001036a0	DPM_ARM_HS_CTRL8	Handshake Control Register 8
0x0010369c	DPM_ARM_HS_CTRL3	Handshake Control Register 3
0x00103698	DPM_ARM_HS_CTRL6	Handshake Control Register 6
0x00103694	DPM_ARM_HS_CTRL5	Handshake Control Register 5
0x00103690	DPM_ARM_HS_CTRL4	Handshake Control Register 4
0x0010368c	DPM_ARM_HS_CTRL3	Handshake Control Register 3
0x00103688	DPM_ARM_HS_CTRL2	Handshake Control Register 2
0x00103684	DPM_ARM_HS_CTRL1	Handshake Control Register 1
0x00103680	DPM_ARM_HS_CTRL0	Handshake Control Register 0
0x0010367c	DPM_ARM_DB_MAP7	Data Block Mapping Address Register 7
0x00103678	DPM_ARM_DB_END7	Data Block 7 End Address
0x00103674	DPM_ARM_DB_MAP6	Data Block Mapping Address Register 6
0x00103670	DPM_ARM_DB_END6	Data Block 6 End Address
0x0010366c	DPM_ARM_DB_MAP5	Data Block Mapping Address Register 5
0x00103668	DPM_ARM_DB_END5	Data Block 5 End Address
0x00103664	DPM_ARM_DB_MAP4	Data Block Mapping Address Register 4
0x00103660	DPM_ARM_DB_END4	Data Block 4 End Address
0x0010365c	DPM_ARM_DB_MAP3	Data Block Mapping Address Register 3
0x00103658	DPM_ARM_DB_END3	Data Block 3 End Address
0x00103654	DPM_ARM_DB_MAP2	Data Block Mapping Address Register 2
0x00103650	DPM_ARM_DB_END2	Data Block 2 End Address
0x0010364c	DPM_ARM_DB_MAP1	Data Block Mapping Address Register 1
0x00103648	DPM_ARM_DB_END1	Data Block 1 End Address
0x00103644	DPM_ARM_DB_MAP0	Data Block Mapping Address Register 0
0x00103640	DPM_ARM_DB_END0	Data Block 0 End Address
0x0010363c	reserved	-
0x00103638	DPM_ARM_IO_DATA1	Input / Output Data Register 1
0x00103634	DPM_ARM_IO_DRV_EN1	Input / Output Driver Enable Register 1
0x00103630	DPM_ARM_IO_MODE1	Input / Ouput Mode Register 1

0x0010362c	Reserved	-
0x00103628	DPM_ARM_IO_DATA0	Input / Output Data Register 0
0x00103624	DPM_ARM_IO_DRV_EN0	Input / Ouput Driver Enable Register 0
0x00103620	DPM_ARM_IO_MODE0	Input / Ouput Mode Register 0
0x0010361c	EXT_CFG_CS3	Extension Bus Configuration Chip Select 3
0x00103618	EXT_CFG_CS2	Extension Bus Configuration Chip Select 2
0x00103614	EXT_CFG_CS1	Extension Bus Configuration Chip Select 1
0x00103610	EXT_CFG_CS0	Extension Bus Configuration Chip Select 0
0x0010360c	DPM_ARM_IF_CFG1	Interface Configuration Register 1
0x00103608	DPM_ARM_IF_CFG0	Interface Configuration Register 0
0x00103604	DPM_ARM_CLKOUT_CFG	Clockout Configuration Register
0x00103600	Reserved	-
0x001035fh - 0x00103540	Reserved	-
0x0010353c	DPM_ARM_HS_DATA15	Handshake Data Register 15
0x00103538	DPM_ARM_HS_DATA14	Handshake Data Register 14
0x00103534	DPM_ARM_HS_DATA13	Handshake Data Register 13
0x00103530	DPM_ARM_HS_DATA12	Handshake Data Register 12
0x0010352c	DPM_ARM_HS_DATA11	Handshake Data Register 11
0x00103528	DPM_ARM_HS_DATA10	Handshake Data Register 10
0x00103524	DPM_ARM_HS_DATA9	Handshake Data Register 9
0x00103520	DPM_ARM_HS_DATA8	Handshake Data Register 8
0x0010351c	DPM_ARM_HS_DATA7	Handshake Data Register 7
0x00103518	DPM_ARM_HS_DATA6	Handshake Data Register 6
0x00103514	DPM_ARM_HS_DATA5	Handshake Data Register 5
0x00103510	DPM_ARM_HS_DATA4	Handshake Data Register 4
0x0010350c	DPM_ARM_HS_DATA3	Handshake Data Register 3
0x00103508	DPM_ARM_HS_DATA2	Handshake Data Register 2
0x00103504	DPM_ARM_HS_DATA1	Handshake Data Register 1
0x00103500	DPM_ARM_HS_DATA0	Handshake Data Register 0
0x001034fc	Reserved	-
0x001034f8	Reserved	-
0x001034f4	Reserved	-
0x001034f0	DPM_ARM_INT_EN0	Interrupt Enable 0 Register
0x001034ec	Reserved	-
0x001034e8	Reserved	-
0x001034e4	Reserved	-
0x001034e0	DPM_ARM_INT_STAT0	Interrupt Status 0 Register
0x001034dc	Reserved	-
0x001034d8	DPM_ARM_SYS_STAT	System Status Register
0x001034d4	Reserved	-
0x001034d0	Reserved	-
0x001034cc	DPM_ARM_WDG_ARM_TRIG	Watchdog Trigger ARM
0x001034c8	DPM_ARM_WDG_ARM_TIMEOUT	Watchdog Timeout ARM, read only

0x001034c4	Reserved	-
0x001034c0	DPM_ARM_WDG_HOST_TIMEOUT	Watchdog Timeout Host
0x001034bc	DPM_ARM_CIS_MAP	Card Information Structure Mapping Address
0x001034b8 - 0x00103400	Reserved	-
0x001033fc - 0x00103300	Reserved	-
0x001032fc - 0x00103240	Reserved	-
0x0010323c	DPM_HOST_HS_DATA15	Handshake Data Register 15
0x00103238	DPM_HOST_HS_DATA14	Handshake Data Register 14
0x00103234	DPM_HOST_HS_DATA13	Handshake Data Register 13
0x00103230	DPM_HOST_HS_DATA12	Handshake Data Register 12
0x0010322c	DPM_HOST_HS_DATA11	Handshake Data Register 11
0x00103228	DPM_HOST_HS_DATA10	Handshake Data Register 10
0x00103224	DPM_HOST_HS_DATA9	Handshake Data Register 9
0x00103220	DPM_HOST_HS_DATA8	Handshake Data Register 8
0x0010321c	DPM_HOST_HS_DATA3	Handshake Data Register 3
0x00103218	DPM_HOST_HS_DATA6	Handshake Data Register 6
0x00103214	DPM_HOST_HS_DATA5	Handshake Data Register 5
0x00103210	DPM_HOST_HS_DATA4	Handshake Data Register 4
0x0010320c	DPM_HOST_HS_DATA3	Handshake Data Register 3
0x00103208	DPM_HOST_HS_DATA2	Handshake Data Register 2
0x00103204	DPM_HOST_HS_DATA1	Handshake Data Register 1
0x00103200	DPM_HOST_HS_DATA0	Handshake Data Register 0
0x001031fc	Reserved	-
0x001031f8	Reserved	-
0x001031f4	Reserved	-
0x001031f0	DPM_HOST_INT_EN0	Interrupt Enable 0
0x001031ec	Reserved	-
0x001031e8	Reserved	-
0x001031e4	Reserved	-
0x001031e0	DPM_HOST_INT_STAT0	Interrupt Status 0
0x001031dc	DPM_HOST_RES_REQ	Reset Request
0x001031d8	DPM_HOST_SYS_STAT	System Status
0x001031d4	DPM_HOST_TMR_START_VAL	Timer Start Value
0x001031d0	DPM_HOST_TMR_CTRL	Timer Control
0x001031cc	Reserved	-
0x001031c8	DPM_HOST_WDG_ARM_TIMEOUT	Watchdog ARM Timeout
0x001031c4	DPM_HOST_WDG_HOST_TRIG	Watchdog Host Trigger
0x001031c0	DPM_HOST_WDG_HOST_TIMEOUT	Watchdog Host Timeout, read only
0x001031bc - 0x00103100	reserved	-
0x001030fc - 0x00103000	reserved	-

Hint!
Register with grey high lighted names will be accessed from host system. They are mapped into the netX memory range. Access from ARM side is only for debugging.

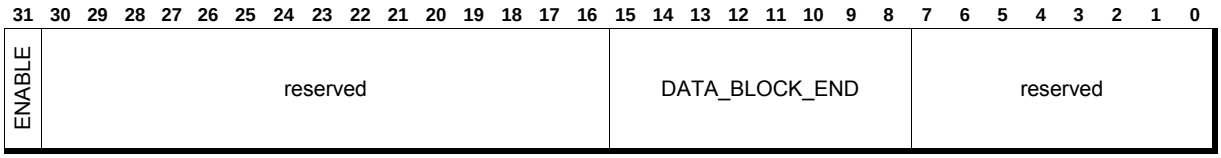
DPM_ARM_HS_CTRL0 – DPM ARM Side Handshake Control Register 0 0x00103680
DPM_ARM_HS_CTRL1 – DPM ARM Side Handshake Control Register 1 0x00103684
DPM_ARM_HS_CTRL2 – DPM ARM Side Handshake Control Register 2 0x00103688
DPM_ARM_HS_CTRL3 – DPM ARM Side Handshake Control Register 3 0x0010368c
DPM_ARM_HS_CTRL4 – DPM ARM Side Handshake Control Register 4 0x00103690
DPM_ARM_HS_CTRL5 – DPM ARM Side Handshake Control Register 5 0x00103694
DPM_ARM_HS_CTRL6 – DPM ARM Side Handshake Control Register 6 0x00103698
DPM_ARM_HS_CTRL7 – DPM ARM Side Handshake Control Register 7 0x0010369c
DPM_ARM_HS_CTRL8 – DPM ARM Side Handshake Control Register 8 0x001036a0
DPM_ARM_HS_CTRL9 – DPM ARM Side Handshake Control Register 9 0x001036a4
DPM_ARM_HS_CTRL10 – DPM ARM Side Handshake Control Register 10 0x001036a8
DPM_ARM_HS_CTRL11 – DPM ARM Side Handshake Control Register 11 0x001036ac
DPM_ARM_HS_CTRL12 – DPM ARM Side Handshake Control Register 12 0x001036b0
DPM_ARM_HS_CTRL13 – DPM ARM Side Handshake Control Register 13 0x001036b4
DPM_ARM_HS_CTRL14 – DPM ARM Side Handshake Control Register 14 0x001036b8
DPM_ARM_HS_CTRL15 – DPM ARM Side Handshake Control Register 15 0x001036bc



Bits	Name	Description	R/W	Default
31	ENABLE	When set the handshake register pair function is enabled.	R/W	0
30:16	reserved	-	R	0x00
15:2	DPM_BASE_ADDR	Base address of handshake register pair in Dual-Port memory.	R/W	0x00
1	reserved	-	R	0
0	8/16_BIT	Select the handshake register pair width 0 : 8 bit cell 1 : 16 bit cell	R/W	0

When more than one register pair is enabled the internal priorization logic selects the base address in ascending order from handshake control register 0 to 15.

DPM_ARM_DB_END0 – DPM ARM Side Data Block 0 End Address 0x00103640
DPM_ARM_DB_END1 – DPM ARM Side Data Block 1 End Address 0x00103648
DPM_ARM_DB_END2 – DPM ARM Side Data Block 2 End Address 0x00103650
DPM_ARM_DB_END3 – DPM ARM Side Data Block 3 End Address 0x00103658
DPM_ARM_DB_END4 – DPM ARM Side Data Block 4 End Address 0x00103660
DPM_ARM_DB_END5 – DPM ARM Side Data Block 5 End Address 0x00103668
DPM_ARM_DB_END6 – DPM ARM Side Data Block 6 End Address 0x00103670
DPM_ARM_DB_END7 – DPM ARM Side Data Block 7 End Address 0x00103678



Bits	Name	Description	R/W	Default
31	ENABLE	Data Block Enable. When set the block mapping function is enabled.	R/W	0
30:16	reserved	-	R	0x00
15:8	DATA_BLOCK_END	Data Block End Address	R/W	0x00
7:0	reserved	-	R	0x00

These registers set the end address of each data block in the Dual-Port memory. It is possible to set each end address anywhere in the 64 kbyte linear address range. The first memory block will always start at address 0x0000.

Data Block Number	Priority	Start Base Address	:	End Base Address
DATA_BLOCK0	highest	0x0000	:	DATA_BLOCK_END0 - 0x0001
DATA_BLOCK1		DATA_BLOCK_END0	:	DATA_BLOCK_END1 - 0x0001
DATA_BLOCK2		DATA_BLOCK_END1	:	DATA_BLOCK_END2 - 0x0001
DATA_BLOCK3	:	DATA_BLOCK_END2	:	DATA_BLOCK_END3 - 0x0001
DATA_BLOCK4	:	DATA_BLOCK_END3	:	DATA_BLOCK_END4 - 0x0001
DATA_BLOCK5		DATA_BLOCK_END4	:	DATA_BLOCK_END5 - 0x0001
DATA_BLOCK6		DATA_BLOCK_END5	:	DATA_BLOCK_END6 - 0x0001
DATA_BLOCK7	lowest	DATA_BLOCK_END6	:	DATA_BLOCK_END7 - 0x0001

Attention!
The end addresses must be ascending from memory block end register one to eight. There is a priority encoder with a sort order from register one to eight.
The handshake register has higher priority if the programmed memory block address range overlaps with the handshake register pair. It is the same with the global status register block.

DPM_ARM_DB_MAP0 – DPM ARM Side Data Block 0 Address Mapping 0x00103644
DPM_ARM_DB_MAP1 – DPM ARM Side Data Block 1 Address Mapping 0x0010364c
DPM_ARM_DB_MAP2 – DPM ARM Side Data Block 2 Address Mapping 0x00103654
DPM_ARM_DB_MAP3 – DPM ARM Side Data Block 3 Address Mapping 0x0010365c
DPM_ARM_DB_MAP4 – DPM ARM Side Data Block 4 Address Mapping 0x00103664
DPM_ARM_DB_MAP5 – DPM ARM Side Data Block 5 Address Mapping 0x0010366c
DPM_ARM_DB_MAP6 – DPM ARM Side Data Block 6 Address Mapping 0x00103674
DPM_ARM_DB_MAP7 – DPM ARM Side Data Block 7 Address Mapping 0x0010367c

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NETX_MAP_ADDR																								reserved							

Bits	Name	Description	R/W	Default
31:8	NETX_MAP_ADDR	Data block base address in netX memory range	R/W	0x00
7:0	reserved	-	R	0x00

These registers set the physical base addresses in the netX address range of each data block enabled in the Dual-Port memory.

Data Block	netX Physical Mapping Address
DATA_BLOCK0	Physical_netX_Address0 = DATA_BLOCK_0_NETX_MAP_ADDR + 0x00000000
DATA_BLOCK1	Physical_netX_Address1 = DATA_BLOCK_1_NETX_MAP_ADDR + DATA_BLOCK_END0
DATA_BLOCK2	Physical_netX_Address2 = DATA_BLOCK_2_NETX_MAP_ADDR + DATA_BLOCK_END1
DATA_BLOCK3	Physical_netX_Address3 = DATA_BLOCK_3_NETX_MAP_ADDR + DATA_BLOCK_END2
DATA_BLOCK4	Physical_netX_Address4 = DATA_BLOCK_4_NETX_MAP_ADDR + DATA_BLOCK_END3
DATA_BLOCK5	Physical_netX_Address5 = DATA_BLOCK_5_NETX_MAP_ADDR + DATA_BLOCK_END4
DATA_BLOCK6	Physical_netX_Address6 = DATA_BLOCK_6_NETX_MAP_ADDR + DATA_BLOCK_END5
DATA_BLOCK7	Physical_netX_Address7 = DATA_BLOCK_7_NETX_MAP_ADDR + DATA_BLOCK_END6

Data Memory Area / Data Memory Blocks

The complete data memory area can be divided up to eight data memory blocks for data transfer between the netX and the host system. They are located at the address range 0x0000 until 0xFDFF. The start address in the Dual-port memory of each data block is programmable by the netX. Accesses to these data memory blocks are physically mapped to accesses in the netX memory range. This could be an internal memory area, an internal register or an external memory. The internal netX memory base mapping addresses are also programmable by the netX.

Due to the programmable memory block structure there is no special address base for accessing the data memory blocks at netX side.

When the host system makes an access to a memory address of the Dual-port memory which is not mapped, the access is not performed and an interrupt event occures on the host side (Memory Lock Interrupt). The write access will be ignored and a read access will contain always the data '0bad_0bad'.

The 'DPM_ARM_DB_END0-7' registers, controlled by the netX, set the end address of each data memory block in the Dual-Port memory. It is possible to set eight end addresses anywhere in the 64 kByte address range. The area of each block is defined from the previous end pointer to the next end pointer minus one. The first memory block will always start at address 0. All start addresses must be a multiple of 256 bytes. The end addresses must be ascending from memory block end register one to eight. There is a priority encoder with a sort order from register one to eight. The handshake register has higher priority if the programmed data memory block address range overlaps with the handshake register pair. It is the same with the global control block.

The 'DPM_ARM_DB_MAP0-7' registers, controlled by the netX, set the physical memory mapping address of each data memory block. The physical address can be set by a multiple of 256 Bytes. The physical base address for data memory block one to eight is calculated by the following formula:

$$DB\_MAPPING_{(0)} = NETX\_ADDRESS_{(0)}$$

$$DB\_MAPPING_{(x)} = NETX\_ADDRESS_{(x)} - DB\_END_{(x-1)}$$

$$x = 1\ldots7$$

The following table shows an example for a programmed Dual-port memory structure with eight data memory blocks. The first four data areas are mapped linear to netX memory address 0x80000000 and following (external memory). The second four data memory blocks are mapped into internal memory address 0x00010000 and following.

Block Number	DPM Address	DB_END	Block Size	netX Address	DB_MAPPING
0	0x0000 - 0x1FFF	0x2000	0x2000	0x80000000	0x80000000
1	0x2000 - 0x3FFF	0x4000	0x2000	0x80020000	0x80000000
2	0x4000 - 0x4FFF	0x5000	0x1000	0x80040000	0x80000000
3	0x5000 - 0x7FFF	0x8000	0x3000	0x80050000	0x80000000
4	0x8000 - 0x9FFF	0xA000	0x2000	0x00010000	0x00008000
5	0xA000 - 0xAFFF	0xB000	0x1000	0x00012000	0x00008000
6	0xB000 - 0xBFFF	0xC000	0x1000	0x00013000	0x00008000
7	0xC000 - 0xFDFF	0xFE00	0x3E00	0x00014000	0x00008000

The register settings for this example are as the followings:

Register	Value	Register	Value
DPM_ARM_DB_END0	0x80002000	DPM_ARM_DB_MAP0	0x80000000
DPM_ARM_DB_END1	0x80004000	DPM_ARM_DB_MAP1	0x80000000
DPM_ARM_DB_END2	0x80005000	DPM_ARM_DB_MAP2	0x80000000
DPM_ARM_DB_END3	0x80008000	DPM_ARM_DB_MAP3	0x80000000
DPM_ARM_DB_END4	0x8000A000	DPM_ARM_DB_MAP4	0x00008000
DPM_ARM_DB_END5	0x8000B000	DPM_ARM_DB_MAP5	0x00008000
DPM_ARM_DB_END6	0x8000C000	DPM_ARM_DB_MAP6	0x00008000
DPM_ARM_DB_END7	0x8000FE00	DPM_ARM_DB_MAP7	0x00008000

DPM_ARM_IO_DATA1 – DPM ARM Side Input / Output Data 1 0x00103638

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved											PIO_DATA[84:64]																				

Bits	Name	Description	R/W	Default
31:21	reserved	-	R	0x00
20:0	PIO_DATA[84:64]	Input or Output Data of each I/O Pin Read: 0 : Physical Input Level is 0 1 : Physical Input Level is 1 Write: 0 : Sets the output pin level to 0 when configured as output 1 : Sets the output pin level to 1 when configured as output.	R/W	0x00

These programmable input / output pins are multiplexed with the host interface. So there are two modes of operation. When the host interface is switched to 'I/O Mode' then all pins work as normal input / output lines. The second mode of operation is when the Extension Bus or Dual-Port memory mode is selected. Then unused interface lines can be used as programmable input / output pins. It is possible to switch each pin between interface operation and programmable input / output pin via the DPM_ARM_IO_MODE register.

DPM_ARM_IO_DRV_EN1 – DPM ARM Side Input / Output Driver Enable 1 0x00103634

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved											PIO_DRV[84:64]																				

Bits	Name	Description	R/W	Default
31:21	reserved	-	R	0x00
20:0	PIO_DRV[84:64]	Driver output enable of each I/O Pin 0 : Output driver disabled 1 : Output driver enabled	R/W	0x00

DPM_ARM_IO_MODE1 – DPM ARM Side Input / Output Mode 1 0x00103630

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IN_CONTROL		reserved									PIO_MODE[84:64]																				

Bits	Name	Description	R/W	Default
31:30	IN_CONTROL	Input data control 00 : Input data latched by nPOR 01 : Input data always sampled with 100 MHz system clock Input 10 : data only sampled when PIO[77] pin is low. 11 : Input data only sampled when PIO[77] pin is high.	R/W	0x0
29:21	reserved	-	R	0x00
20:0	PIO_MODE[84:64]	Pin mode selection 0 : PIO Mode 1 : Host interface Mode	R/W	0x00

DPM_ARM_IO_DATA0 – DPM ARM Side Input / Output Data 0 0x00103628

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIO_DATA[63:32]																															

Bits	Name	Description	R/W	Default
31:0	PIO_DATA[63:32]	Input or Output Data of each I/O Pin Read: 0 : Physical Input Level is 0 1 : Physical Input Level is 1 Write: 0 : Sets the output pin level to 0 when configured as output 1 : Sets the output pin level to 1 when configured as output	R/W	0x00

DPM_ARM_IO_DRV_EN0 – DPM ARM Side Input / Output Driver Enable 0 0x00103624

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIO_DRV[63:32]																															

Bits	Name	Description	R/W	Default
31:0	PIO_DRV[63:32]	Driver output enable of each I/O Pin 0 : Output driver disabled 1 : Output driver enabled	R/W	0x00

DPM_ARM_IO_MODE0 – DPM ARM Side Input / Output Mode 0 0x00103620

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIO_MODE[63:32]																															

Bits	Name	Description	R/W	Default
31:0	PIO_MODE[63:32]	Pin mode selection 0 : PIO Mode 1 : Host interface mode	R/W	0x00

DPM_ARM_IF_CFG1 – DPM ARM Side Interface Configuration Register 1 0x0010360c

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRITE_PROTECT	IRQ_POL_PIO72	IRQ_POL_PIO47	IRQ_POL_PIO40	IRQ_POL_PIO36	IRQ_POL_PIO35	DATAOUT_VALID_TIME		DIS_BUSY_TIMEOUT	reserved			ADR_IN[15:12]				CS_COMP_SRC[19:12]								CS_COMP_VAL[19:12]							

Bits	Name	Description	R/W	Default
31	WRITE_PROTECT	When this bit is set any following write accesses to the register are impossible. The bit will be only cleared by a netX system reset.	R/W	0
30	IRQ_POL_PIO72	External interrupt polarity for PIO[72]	R/W	0
29	IRQ_POL_PIO47	External interrupt polarity for PIO[47]	R/W	0
28	IRQ_POL_PIO40	External interrupt polarity for PIO[40]	R/W	0
27	IRQ_POL_PIO36	External interrupt polarity for PIO[36]	R/W	0
26	IRQ_POL_PIO35	External interrupt polarity for PIO[35]	R/W	0
25:24	DATAOUT_VALID_TIME	This value sets the delay between valid read data available at outputs and de-assertion of the BUSY- / assertion of the READY signal in system clocks (10 ns @ 100 MHz) 00 : 0 system clock 01 : 1 system clock 10 : 2 system clock 11 : 3 system clock	R/W	0x00
23	DIS_BUSY_TIMEOUT	When this flag is not set, any access to the netX host interface will be aborted after 256 system clocks (2.56µs @ 100 MHz) and the BUSY /READY signal will be released. This feature avoids system lockups when a read is not possible (e.g. access to SDRAM while SDRAM controller has not been configured and enabled).	R/W	0
22:20	reserved	-	R	0x00
19:16	ADR_IN[15:12]	Logic level for Dual-Port memory address lines [15:12] when used as input / output pin (pio mode) or when the chip select comparator is enabled.	R/W	0x00
15:8	CS_COMP_SRC[19:12]	Chip select compare source for address [19:12] 0 : internal (compare with CS_COM_VAL) 1 : external (compare with signals at pins SEL_A[19:12])	R/W	0x00
7:0	CS_COMP_VAL[19:12]	Chip Select compare value when internal compare source is used	R/W	0x00

DPM_ARM_IF_CFG0 – DPM ARM Side Host Interface Configuration Register 0 0x00103608

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DISABLE_WR	HIF_MODE			RD_CTRL		WR_CTRL		BE1_MODE			BE0_MODE			CIS_MODE		WAIT_DRV		WAIT_MODE	WAIT_POLARITY	CS_MODE			IRQ_MODE		IRQ_POLARITY	ALE_MODE		ADDR_MODE	OE_MODE		

Bits	Name	Description	R/W	Default
31	DISABLE_WR	When this bit is set any consecutive write accesses to the register are blocked. This bit will only be cleared by a netX system reset.	R/W	0
30:28	HIF_MODE	Host interface mode selection. This selection is only relevant when the 'IF_SELECT_N' bit is not set in the 'IO Configuration Register' IO_CFG. 000 : Disabled outputs (all host interface pins in high imped. mode) 001 : Extension Bus 010 : µP Bus 8 bit (Dual-Port memory) 011 : µP Bus 16 bit (Dual-Port memory) 100 : I/O Mode 101 : Reserved (all host interface pins in high impedance mode) 110 : Reserved (all host interface pins in high impedance mode) 111 : Reserved (all host interface pins in high impedance mode)	R/W	0x00
27:26	RD_CTRL	These bits configure the control lines for Dual-Port memory read accesses to netX 00 : RDn; PIO[52] A read cycle is selected when the RDn line has a low level 01 : RDn, A0, BHEn; PIO[52,73,43] A read cycle is selected when the RDn (RD/WRn = DIR) line has a high level and the A0 (BE0n) or BHEn (BE1n) line (or both of them) have a low level (e.g. Motorola 68000) 10 : Read access disabled 11 : RDn, BHEn; PIO[52,43] A read cycle is selected when the RDn (RD/WRn = DIR) line has a high level and the BHEn (EN) line has a high level (e.g. Motorola 6800).	R/W	0x00
25:24	WR_CTRL	These bits configure the control lines for Dual-Port memory write accesses to netX 00 : WRLn; PIO[45] A write cycle is selected when the WRLn (WRn) line has a low level 01 : RDn, A0, BHEn; PIO[52,73,43] A write cycle is selected when the RDn (RD/WRn = DIR) line has low level and the A0 (BE0n) or BHEn (BE1n) line or both of them have low level (e.g. Motorola 68000).	R/W	0x00

		10 : WRLn, WRHn; PIO[45,44] A write cycle is selected when the WRLn (low byte) or the WRHn (high byte) line (or both of them) have a low level. 11 : RDn, BHEn; PIO[52,43] A write cycle is selected when the RDn (RD/WRn = DIR) line has a low level and the BHEn (EN) line has a high level (e.g. Motorola 6800).		
23:21	BE1_MODE	Byte Enable 1 (internal signal) generation : 000 : BHEn; PIO[43] The high byte is selected when the BHEn (= CE2n) signal has a low level (PCMCIA Mode) 001 : A0; PIO[73] The high byte is selected when the A0 line has a high level (e.g. 8 bit Dual-Port memory interface). 010 : RDn, WRLn; PIO[52,45] The high byte is selected when the RDn or WRLn lines have low level (only one write signal, only 16 Bit accesses) 011 : RDn, WRHn; PIO[52,44] The high byte is selected when the RDn or WRHn lines have low level. 100 : internal A0 line controlled via PIO Mode or ALE Mode (A0==1) 101 : nBHE; PIO[43] The high byte is selected when the BHEn (SBHE) line has a high level (ISA Mode) 101 : high byte access always active 11x : high byte access always active	R/W	0x00
20:18	BE0_MODE	Byte Enable 0 (internal signal) generation : 000 : CS0n, PIO[51] The low byte byte is selected when the CS0n (= CE1n) signal has a low level (PCMCIA Mode) 001 : A0; PIO[73] The low byte is selected when the A0 line has a low level (e.g. 8 bit Dual-Port memory interface). 010 : RDn, WRLn; PIO[52,45] The low byte is selected when the RDn or WRLn lines have a low level. 011 : internal A0 line controlled via PIO Mode or ALE Mode (A0==0) 1xx : low byte access always active	R/W	0x00
17:16	CIS_MODE	The CIS memory array is selected by the following condition. 00 : Never 01 : Always 10 : by low WRHn / PIO[44] signal == REGn (PCMCIA Mode) 11 : by low PIO[40] signal	R/W	0x00
15:14	WAIT_DRV	Wait mode output drive control; RDY / PIO[46] 00 : high impedance output 01 : push / pull output 10 : open drain / open source output 11 : sustained tri state output	R/W	0x00

13	WAIT_MODE	0 : WAIT / BUSY mode function. An active signal indicates that the current access can not yet be finished by the host CPU. 1 : READY mode function. An active signal indicates that the host CPU may finish the current access.	R/W	0
12	WAIT_POLARITY	0 : low active polarity output 1 : high active polarity output	R/W	0
11:9	CS_MODE	These bits configure the logic for internal chip select generation 000 : chip access is disabled 001 : chip select generated by internal address comparator 010 : CS0n, BHEn; PIO[51,43] The netX DPM is selected when CS0n or BHEn lines have a low level (PCMCIA Mode) 011 : CS0n, BHEn; PIO[51,43] The netX DPM is selected when CS0n or BHEn lines have a high level 100 : CS0n, PIO[51] The netX DPM is selected when the CS0n line has a low level. 101 : chip select disabled 110 : Internal Address Comparator and ALE; PIO[35] The netX chip is selected when the internal address comparator detects a hit and the ALE (= AEN) has a low level 111 : Internal Address Comparator and ALE; PIO[35] The netX chip is selected when the internal address comparator detects a hit and the ALE (= AEN) has a high level	R/W	0x00
8:7	IRQ_MODE	Select the interrupt pin output function of IRQ / INT pin (PIO[47]) 00 : high impedance output 01 : fixed output level 10 : push pull output 11 : open drain / open source output	R/W	0x00
6	IRQ_POLARITY	0 : active low polarity output 1 : active high polarity output	R/W	0
5:4	ALE_MODE	Selection of address input mode (PIO[35] == ALE/AEN). This function is only activated when no input / output mode is selected for the PIO[35] pin. 00 : address latching on low signal level of PIO[35] 01 : address latching on high signal level of PIO[35] 10 : address latching on falling PIO[35] edge 11 : address latching on rising PIO[35] edge	R/W	0x00
3	ADDR_MODE	0 : non multiplexed 8/16 bit address mode 1 : multiplexed 8/16 bit address mode	R/W	0
2:0	OE_MODE	Selection bits for output driver control of data lines for read accesses 000 : RDn, BHEn, A0; PIO[52,43,73] Output drivers are enabled by a high RDn (RD/WRn) signal and a low BHEn (high byte) or a low A0 (low byte) signal. (e.g. Motorola 68000)	R/W	0x00

		001 : RDn, BHEn, A0; PIO[52,43,73] Output drivers are enabled by a low RDn (RDn) signal and a low BHEn (high byte enable) or a low A0 (low byte enable) signal (e.g. 80186) 010 : RDn; PIO[52] Output drivers are enabled by a low RDn signal. 011 : RDn, BHEn, CS0n; PIO[52,43,51] The output drivers are enabled by a low RDn signal and a low CS0n (CE0n) or a low BHEn (CE1n) signal (PCMCIA Mode). 100 : RDn, BHEn; PIO[52,43] The output drivers are enabled when the RDn (RD/WRn = DIR) signal is high and the BHEn (EN) signal is high (e.g. Motorola 6800) 101 : reserved (output drivers disabled) 11x : reserved (output drivers disabled)		

The following table shows some configuration examples.

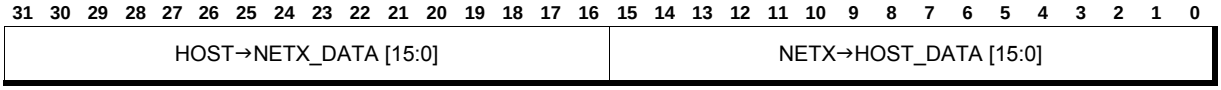
Connection Type	DPM_ARM_IF_CFG0
Intel, 8 Bit non multiplexed, one write signal	x_010_00_00_???_???_??_??_?_?_???_??_?_??_?_???
Intel, 8 Bit multiplexed, one write signal	x_010_00_00_???_???_??_??_?_?_???_??_?_??_?_???
Intel, 16 Bit non-multiplexed, one write signal	x_011_00_00_???_???_??_??_?_?_???_??_?_??_?_???
Intel, 16 Bit non-multiplexed, write low/high signal	x_011_00_00_???_???_??_??_?_?_???_??_?_??_?_???
Motorola, 8 Bit non multiplexed	x_010_00_00_???_???_??_??_?_?_???_??_?_??_?_???
Motorola Coldfire, 16 Bit non-multiplexed	x_011_00_00_???_???_??_??_?_?_???_??_?_??_?_???
Motorola MC68000	x_011_00_00_???_???_??_??_?_?_???_??_?_??_?_???

DPM_ARM_CLKOUT_CFG – DPM ARM Side Clock out configuration 0x00103604

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLKOUT_EN	reserved	CLK_SEL																													

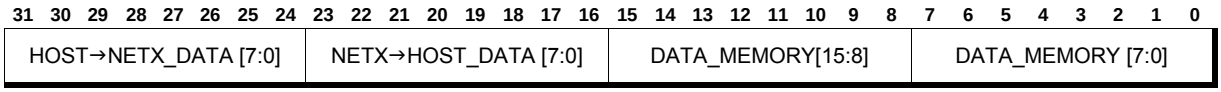
Bits	Name	Description	R/W	Default
31	CLKOUT_EN	CLKOUT driver enable.	R/W	0
30	reserved	-	R/W	0
29:0	CLK_SEL	Clock output frequency selection. F = Value x 200 MHz / 2^{32}	R/W	0x00

16 Bit Handshake Data



Bits	Name	Description	R/W	Default
31:16	HOST→NETX_DATA[15:0]	Handshake Data Flags host to netX [15:0]	R	0x00
15:0	NETX→Host_DATA[15:0]	Handshake Data Flags netX to host [15:0]	R/W	0x00

8 Bit Handshake Data



Bits	Name	Description	R/W	Default
31:24	HOST→NETX_DATA[7:0]	Handshake Data Flags host to netX [7:0]	R	0x00
23:16	NETX→HOST_DATA[7:0]	Handshake Data Flags netX to host [7:0]	R/W	0x00
15:8	DATA_MEMORY [15:8]	Data Memory [15:8]	R/W	0x00
7:0	DATA_MEMORY [7:0]	Data Memory [7:0]	R/W	0x00

DPM_ARM_INT_EN0 – DPM ARM Side Interrupt Enable 0 0x001034f0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GLB_EN	MEM_LCK	WDG	INT_PIO72	INT_PIO47	INT_PIO40	INT_PIO36	INT_PIO35	reserved								HANDSHAKE_EVENT[15:0]																

Bits	Name	Description	R/W	Default
31	GLB_EN	Global Interrupt Enable	R/W	0
30	MEM_LCK	Memory Lock Error (generated by invalid access from host side)	R/W	0
29	WDG	Watchdog timeout host supervision interrupt enable	R/W	0
28	INT_PIO72	External interrupt enable for pin PIO72	R/W	0
27	INT_PIO47	External interrupt enable for pin PIO47	R/W	0
26	INT_PIO40	External interrupt enable for pin PIO40	R/W	0
25	INT_PIO36	External interrupt enable for pin PIO36	R/W	0
24	INT_PIO35	External interrupt enable for pin PIO35	R/W	0
23:16	reserved	-	R	0x00
15:0	HS[15:0]	Handshake event 15:0 interrupt enable	R/W	0x00

This register control the interrupt behaviour of the chip. The interrupt status flags will be always set on the event but the physical interrupt line will only be active when the enable bit is set. All interrupt requests could be enabled or disabled independently and are wired together to the global interrupt request for the host interface.

DPM_ARM_INT_STAT0 – DPM ARM Side Interrupt Status 0 0x001034e0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQ_REQ	MEM_LCK	WDG_IRQ	INT_PIO72	INT_PIO47	INT_PIO40	INT_PIO36	INT_PIO35	IRQ_VECTOR[7:0]								HANDSHAKE_EVENT[15:0]																	

Bits	Name	Description	R/W	Default
31	IRQ_REQ	Global signaling of interrupt request	R	0
30	MEM_LCK	Flag of Memory Lock Error	R/W	0
29	WDG_IRQ	Flag of Watchdog timeout host supervision	R/W	0
28	INT_PIO72	Flag of External interrupt status for pin PIO72	R/W	0
27	INT_PIO47	Flag of External interrupt status for pin PIO47	R/W	0
26	INT_PIO40	Flag of External interrupt status for pin PIO40	R/W	0
25	INT_PIO36	Flag of External interrupt status for pin PIO36	R/W	0
24	INT_PIO35	Flag of External interrupt status for pin PIO35	R/W	0
23:16	IRQ_VECTOR[7:0]	Interrupt Vector generated by the interrupt status flags 0x00 : no interrupt 0x10 : handshake event interrupt 0 0x11 : handshake event interrupt 1 0x12 : handshake event interrupt 2 0x13 : handshake event interrupt 3 0x14 : handshake event interrupt 4 0x15 : handshake event interrupt 5 0x16 : handshake event interrupt 6 0x17 : handshake event interrupt 7 0x18 : handshake event interrupt 8 0x19 : handshake event interrupt 9 0x1a : handshake event interrupt 10 0x1b : handshake event interrupt 11 0x1c : handshake event interrupt 12 0x1d : handshake event interrupt 13 0x1e : handshake event interrupt 14 0x1f : handshake event interrupt 15 0x20 - 0x5f : reserved 0x60 : Memory Lock Error 0x61 : Watchdog timeout host supervision 0x62 : PIO72 interrupt request 0x63 : PIO47 interrupt request 0x64 : PIO40 interrupt request 0x65 : PIO36 interrupt request 0x66 : PIO35 interrupt request 0x67 - 0xff : reserved	R	0x00
15:0	HS[15:0]	Flag of Handshake Event Status [15:0]	R/W	0x00

This register does not only show the interrupt status flags of each interrupt event, but also the interrupt vector. The flags are always set when the interrupt event occurs regardless of the interrupt enable flag, but the physical interrupt which is indicated by the interrupt vector will only be generated when the corresponding enable flag is set. The interrupt status flags can be cleared by writing a one bit to the flag. When the handshake register cell is read, the corresponding interrupt status flags will be cleared too. It makes no difference if an 8 bit or 16 bit access occurs. The interrupt flag will not be cleared when the interrupt event occurs again while the cell is being read.

DPM_ARM_SYS_STAT – System Status 0x001034d8

This register is just the register ‘SYS_STAT’ described in section 2.6, please refer to chapter 'SYS_STAT – System Status' on page 16.

DPM_ARM_WDG_ARM_TRIG – DPM ARM Side Watchdog ARM Supervision Trigger 0x001034cc

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								WDG_ACCESS_CODE							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	WDG_ACCESS_CODE	Watchdog access code for triggering (read-write back-value)	R/W	0x00

DPM_ARM_WDG_ARM_TIMEOUT – DPM ARM Side Watchdog ARM Timeout 0x001034c8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																TIMEOUT_VAL															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	TIMEOUT_VAL	Timeout value $T_{TIMEOUT}$ = TIMEOUT_VAL x 100 µs	R	0x00

DPM_ARM_WDG_HOST_TIMEOUT – DPM ARM Side Watchdog Host Timeout 0x001034c0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																TIMEOUT_VAL															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	TIMEOUT_VAL	Timeout value $T_{TIMEOUT}$ = TIMEOUT_VAL x 100 µs	R/W	0x00

DPM_ARM_CIS_MAP – DPM ARM Side CIS Mapping Address 0x001034bc

In Dual-port memory mode it is possible to allocate a 256 byte data field which contains an image of the card information structure CIS. The netX has to enable this function and has to store the data information via software into the data field. To access the CIS from host side it is possible to program the CIS-Mode in the 'DPM_ARM_IF_CFG0 – DPM ARM Side Host Interface Configuration Register 0'. Different access types are possible. For CIS accesses only the lower address lines DPM_ADR0-7 are used.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CIS_MAPPING																								reserved						WR_ENABLE	CIS_ENABLE

Bits	Name	Description	R/W	Default
31:8	CIS_MAPPING	CIS Mapping Address. This value sets the mapping address for CIS accesses in the netX memory range. The data length is fixed to 256 byte.	R/W	0x00
7:2	reserved	-	R	0x00
1	WR_ENABLE	If this bit is set write accesses to the CIS are enabled. otherwise write accesses are ignored.	R/W	0
0	CIS_ENABLE	Enable CIS MODE	R/W	0

6 Peripheral Functions

6.1 GPIOs – General Purpose IOs and Timers

There are 16 general purpose IOs (GPIOs) in total, 12 of them are shared with the UARTs. And there ane also 5 internal timers all together. As both the GPIO control registers and Timer control registers are in the same address block, and also because the GPIOs are used as input or output signal to work with the internal timers, we disscuss both GPIO and Timers in this section.

The following table is a summary of registers of GPIOs and Timers.

ARM Address	Register Name	Short Description
0x00100800	GPIO_CFG0	GPIO 0 Configuration Register
0x00100804	GPIO_CFG1	GPIO 1 Configuration Register
0x00100808	GPIO_CFG2	GPIO 2 Configuration Register
0x0010080c	GPIO_CFG3	GPIO 3 Configuration Register
0x00100810	GPIO_CFG4	GPIO 4 Configuration Register
0x00100814	GPIO_CFG5	GPIO 5 Configuration Register
0x00100818	GPIO_CFG6	GPIO 6 Configuration Register
0x0010081c	GPIO_CFG7	GPIO 7 Configuration Register
0x00100820	GPIO_CFG8	GPIO 8 Configuration Register
0x00100824	GPIO_CFG9	GPIO 9 Configuration Register
0x00100828	GPIO_CFG10	GPIO 10 Configuration Register
0x0010082c	GPIO_CFG11	GPIO 11 Configuration Register
0x00100830	GPIO_CFG12	GPIO 12 Configuration Register
0x00100834	GPIO_CFG13	GPIO 13 Configuration Register
0x00100838	GPIO_CFG14	GPIO 14 Configuration Register
0x0010083c	GPIO_CFG15	GPIO 15 Configuration Register
0x00100840	GPIO_THRSH_CAPT0	GPIO 0 Threshold or Capture Register
0x00100844	GPIO_THRSH_CAPT1	GPIO 1 Threshold or Capture Register
0x00100848	GPIO_THRSH_CAPT2	GPIO 2 Threshold or Capture Register
0x0010084c	GPIO_THRSH_CAPT3	GPIO 3 Threshold or Capture Register
0x00100850	GPIO_THRSH_CAPT4	GPIO 4 Threshold or Capture Register
0x00100854	GPIO_THRSH_CAPT5	GPIO 5 Threshold or Capture Register
0x00100858	GPIO_THRSH_CAPT6	GPIO 6 Threshold or Capture Register
0x0010085c	GPIO_THRSH_CAPT7	GPIO 7 Threshold or Capture Register
0x00100860	GPIO_THRSH_CAPT8	GPIO 8 Threshold or Capture Register
0x00100864	GPIO_THRSH_CAPT9	GPIO 9 Threshold or Capture Register
0x00100868	GPIO_THRSH_CAPT10	GPIO 10 Threshold or Capture Register
0x0010086c	GPIO_THRSH_CAPT11	GPIO 11 Threshold or Capture Register
0x00100870	GPIO_THRSH_CAPT12	GPIO 12 Threshold or Capture Register
0x00100874	GPIO_THRSH_CAPT13	GPIO 13 Threshold or Capture Register
0x00100878	GPIO_THRSH_CAPT14	GPIO 14 Threshold or Capture Register
0x0010087c	GPIO_THRSH_CAPT15	GPIO 15 Threshold or Capture Register

0x00100880	GPIO_CNTR0_CTRL	GPIO counter 0 control register
0x00100884	GPIO_CNTR1_CTRL	GPIO counter 1 control register
0x00100888	GPIO_CNTR2_CTRL	GPIO counter 2 control register
0x0010088c	GPIO_CNTR3_CTRL	GPIO counter 3 control register
0x00100890	GPIO_CNTR4_CTRL	GPIO counter 4 control register
0x00100894	GPIO_CNTR0_MAX	GPIO counter 0 max values
0x00100898	GPIO_CNTR1_MAX	GPIO counter 1 max values
0x0010089c	GPIO_CNTR2_MAX	GPIO counter 2 max values
0x001008a0	GPIO_CNTR3_MAX	GPIO counter 3 max values
0x001008a4	GPIO_CNTR4_MAX	GPIO counter 4 max values
0x001008a8	GPIO_CNTR0_CNT	GPIO counter 0 current value
0x001008ac	GPIO_CNTR1_CNT	GPIO counter 1 current value
0x001008b0	GPIO_CNTR2_CNT	GPIO counter 2 current value
0x001008b4	GPIO_CNTR3_CNT	GPIO counter 3 current value
0x001008b8	GPIO_CNTR4_CNT	GPIO counter 4 current value
0x001008bc	GPIO_IRQ_MSK_SET	GPIO Interrupt Request Enable Mask
0x001008c0	GPIO_IRQ_MSK_RESET	GPIO Interrupt Request Disable Mask
0x001008c4	GPIO_SYSTIME_NS_CMP	GPIO System Time NS Compare Value
0x001008c8	GPIO_OUT	GPIO Output Register
0x001008cc	GPIO_IN	GPIO Input Register
0x001008d0	GPIO_IRQ	GPIO Interrupt Request

GPIOs have the following features:

- each GPIO can be configured individually as input or output, inverted or non inverted or UART signal.
- Each GPIO can be assigned to one of the Timers or the System Time in order to be used as capture input or PWM output.
- Each GPIO can generate an interrupt, when it is configured as one of the capture modes.

Each GPIO has its own configuration register GPIO_CFGi, which can be used to read or write the corresponding IO configuration individually. All inputs can be read in the GPIO_IN register, respectively can be written in the GPIO_OUT register.

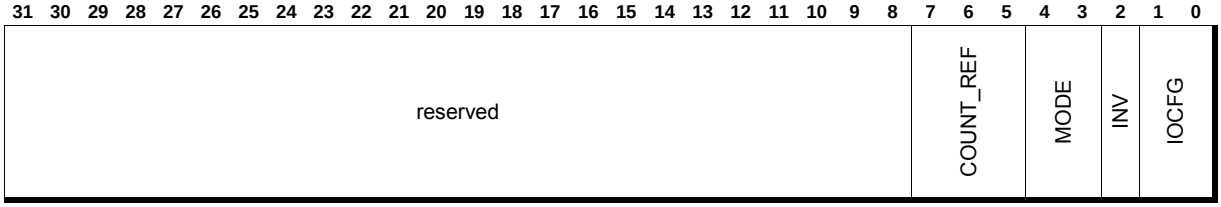
If a GPIO wants to generate an Interrupt then the corresponding interrupt request bit must be set in the GPIO_IRQ_MSK_SET register. On the contrary, each interrupt can be disabled by setting the register GPIO_IRQ_MSK_RESET. When a GPIO generates an interrupt, then the corresponding bit in register GPIO_IRQ will be automatically set.

The internal timers are five 32-Bit Counters, all of them are configurable:

- to count from zero to a maximum value and backward (symmetric Mode)
- to count from zero to a maximum value and set back to zero (asymmetric Mode)
- single shot or count continuously
- to generate an interrupt if it reaches zero
- to count external events
- to set back to zero by an external event
- to capture the timer value by an external events
- to generate a PWM signal by comparing the timer value with a threshold

Any GPIO can be assigned as external events. This can be a rising or falling edge, or a high or low level at the GPIO by setting the inverting bit at the GPIO configuration register. The counter value can be read and overwritten at any time.

GPIO_CFG0 – GPIO 0 Configuration Register **0x00100800**
GPIO_CFG1 – GPIO 1 Configuration Register **0x00100804**
GPIO_CFG2 – GPIO 2 Configuration Register **0x00100808**
GPIO_CFG3 – GPIO 3 Configuration Register **0x0010080c**
GPIO_CFG4 – GPIO 4 Configuration Register **0x00100810**
GPIO_CFG5 – GPIO 5 Configuration Register **0x00100814**
GPIO_CFG6 – GPIO 6 Configuration Register **0x00100818**
GPIO_CFG7 – GPIO 7 Configuration Register **0x0010081c**
GPIO_CFG8 – GPIO 8 Configuration Register **0x00100820**
GPIO_CFG9 – GPIO 9 Configuration Register **0x00100824**
GPIO_CFG10 – GPIO 10 Configuration Register **0x00100828**
GPIO_CFG11 – GPIO 11 Configuration Register **0x0010082c**
GPIO_CFG12 – GPIO 12 Configuration Register **0x00100830**
GPIO_CFG13 – GPIO 13 Configuration Register **0x00100834**
GPIO_CFG14 – GPIO 14 Configuration Register **0x00100838**
GPIO_CFG15 – GPIO 15 Configuration Register **0x0010083c**



Bits	Name	Description	R/W	Default
31:8	reserved		R	0x00
7:5	COUNT_REF	counter reference 000 : counter 0 001 : counter 1 010 : counter 2 011 : counter 3 100 : counter 4 111 : system time	R/W	0x00
4:3	MODE	defines the GPIO mode - depends on 'IOCFG' mode : Input mode 00 : read mode 01 : capture mode (continued) at rising edge 10 : capture mode (once) at rising edge 11 : capture mode (high level) Output mode 00 : set to 0 01 : set to 1 10 : set to gpio_out[i] (i is the corresponding number of GPIO) 11 : pwm mode	R/W	0x00
2	INV	0 : don't invert, 1 : invert input/output value	R/W	0
1:0	IOCFG	defines the input/output configuration 00 : gp input mode 01 : gp output mode 10 : UART mode (signal lines switched to UART modul) 11 : reserved	R/W	0x00

GPIO_THRSH_CAPT0 – GPIO 0 Threshold or Capture Register	**0x00100840**
GPIO_THRSH_CAPT1 – GPIO 1 Threshold or Capture Register	**0x00100844**
GPIO_THRSH_CAPT2 – GPIO 2 Threshold or Capture Register	**0x00100848**
GPIO_THRSH_CAPT3 – GPIO 3 Threshold or Capture Register	**0x0010084c**
GPIO_THRSH_CAPT4 – GPIO 4 Threshold or Capture Register	**0x00100850**
GPIO_THRSH_CAPT5 – GPIO 5 Threshold or Capture Register	**0x00100854**
GPIO_THRSH_CAPT6 – GPIO 6 Threshold or Capture Register	**0x00100858**
GPIO_THRSH_CAPT7 – GPIO 7 Threshold or Capture Register	**0x0010085c**
GPIO_THRSH_CAPT8 – GPIO 8 Threshold or Capture Register	**0x00100860**
GPIO_THRSH_CAPT9 – GPIO 9 Threshold or Capture Register	**0x00100864**
GPIO_THRSH_CAPT10 – GPIO 10 Threshold or Capture Register	**0x00100868**
GPIO_THRSH_CAPT11 – GPIO 11 Threshold or Capture Register	**0x0010086c**
GPIO_THRSH_CAPT12 – GPIO 12 Threshold or Capture Register	**0x00100870**
GPIO_THRSH_CAPT13 – GPIO 13 Threshold or Capture Register	**0x00100874**
GPIO_THRSH_CAPT14 – GPIO 14 Threshold or Capture Register	**0x00100878**
GPIO_THRSH_CAPT15 – GPIO 15 Threshold or Capture Register	**0x0010087c**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															

Bits	Name	Description	R/W	Default
31:0	VAL	holds the counter threshold value for the pwm mode or the captured value in the capture mode	R/W	0x00

GPIO_CNTR0_CTRL – GPIO Counter 0 Control **0x00100880**
GPIO_CNTR1_CTRL – GPIO Counter 1 Control **0x00100884**
GPIO_CNTR2_CTRL – GPIO Counter 2 Control **0x00100888**
GPIO_CNTR3_CTRL – GPIO Counter 3 Control **0x0010088c**
GPIO_CNTR4_CTRL – GPIO Counter 4 Control **0x00100890**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																					GPIO_REF				SEL_EVENT	RST_EN	CNT_EVENT	IRQ_EN	ONCE	SYM_NASYM	RUN

Bits	Name	Description	R/W	Default
31:11	reserved	-	R	0x00
10:7	GPIO_REF	gpio reference (0 - 15)	R/W	0x00
6	SEL_EVENT	select external event 0 : (high) level 1 : (pos.) edge	R/W	0
5	RST_EN	1 : enable automatic reset 0 : disable automatic reset	R/W	0
4	CNT_EVENT	1 : count external event (edge, level) 0 : count every clock cycle	R/W	0
3	IRQ_EN	1 : enable interrupt request 0 : disable interrupt request	R/W	0
2	ONCE	1 : count once 0 : count continue	R/W	0
1	SYM_NASYM	1 : symmetric mode (triangle) 0 : asymmetric mode (sawtooth)	R/W	0
0	RUN	1 : start counter 0 : stop counter	R/W	0

GPIO_CNTR0_MAX – GPIO Counter 0 Maximum Value **0x00100894**
GPIO_CNTR1_MAX – GPIO Counter 1 Maximum Value **0x00100898**
GPIO_CNTR2_MAX – GPIO Counter 2 Maximum Value **0x0010089c**
GPIO_CNTR3_MAX – GPIO Counter 3 Maximum Value **0x001008a0**
GPIO_CNTR4_MAX – GPIO Counter 4 Maximum Value **0x001008a4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															

Bits	Name	Description	R/W	Default
31:0	VAL	max. counter value where the counter switches from upcounting to downcounting (symmetric mode) or jumps to zero (asymmetric mode)	R/W	0x00

GPIO_CNTR0_CNT – GPIO Counter 0 Current Value **0x001008a8**
GPIO_CNTR1_CNT – GPIO Counter 1 Current Value **0x001008ac**
GPIO_CNTR2_CNT – GPIO Counter 2 Current Value **0x001008b0**
GPIO_CNTR3_CNT – GPIO Counter 3 Current Value **0x001008b4**
GPIO_CNTR4_CNT – GPIO Counter 4 Current Value **0x001008b8**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															

Bits	Name	Description	R/W	Default
31:0	VAL	current counter value	R/W	0x00

GPIO_IRQ_MSK_SET – GPIO Interrupt Enable **0x001008bc**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved										SYSTIME	CNT4	CNT3	CNT2	CNT1	CNT0	GPIO[15:0]															

Bits	Name	Description	R/W	Default
31:22	reserved	-	R	0x00
21	SYSTIME	enable interrupt request if sys_time_ns = gpio_systime_ns_cmp	R/W	0
20	CNT4	enable interrupt request for counter4	R/W	0
19	CNT3	enable interrupt request for counter3	R/W	0
18	CNT2	enable interrupt request for counter2	R/W	0
17	CNT1	enable interrupt request for counter1	R/W	0
16	CNT0	enable interrupt request for counter0	R/W	0
15:0	GPIO[15:0]	enable interrupt request for GPIO[15:0]	R/W	0

GPIO_IRQ_MSK_RESET – GPIO Interrupt Disable 0x001008c0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved										SYSTIME	CNT4	CNT3	CNT2	CNT1	CNT0	GPIO[15:0]															

Bits	Name	Description	R/W	Default
31:22	reserved	-	W	0x00
21	SYSTIME	write access with '1' disables interrupt request for sys_time	W	0
20	CNT4	write access with '1' disables interrupt request for counter4	W	0
19	CNT3	write access with '1' disables interrupt request for counter3	W	0
18	CNT2	write access with '1' disables interrupt request for counter2	W	0
17	CNT1	write access with '1' disables interrupt request for counter1	W	0
16	CNT0	write access with '1' disables interrupt request for counter0	W	0
15:0	GPIO[15:0]	write access with '1' disables interrupt request for corresponding GPIO[15:0]	W	0x00

GPIO_SYSTIME_NS_CMP – GPIO System Time NS Compare Value 0x001008c4

Compares this value with systime_ns considering incontinous behaviour of systime_ns

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															

Bits	Name	Description	R/W	Default
31:0	VAL	compare value for systime	R/W	0x00

GPIO_OUT– GPIO Output Register 0x001008c8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																VAL															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	VAL	GPIO[15:0] output values	R/W	0x00

GPIO_IN– GPIO Input Register 0x001008cc

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																VAL															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	VAL	GPIO[15:0] input values	R	0x00

GPIO_IRQ– GPIO Interrupt Register 0x001008d0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										SYS_TIME	CNT4	CNT3	CNT2	CNT1	CNT0	GPIO															

Bits	Name	Description	R/W	Default
31:22	reserved	-	R	0x00
21	SYS_TIME	hold the interrupt bit for sys_time	R/W	0
20	CNT4	hold the interrupt bit for counter4	R/W	0
19	CNT3	hold the interrupt bit for counter3	R/W	0
18	CNT2	hold the interrupt bit for counter2	R/W	0
17	CNT1	hold the interrupt bit for counter1	R/W	0
16	CNT0	hold the interrupt bit for counter0	R/W	0
15:0	GPIO	holds the interrupt bits for every GPIO write access with '1' resets the appropriate irq	R/W	0x00

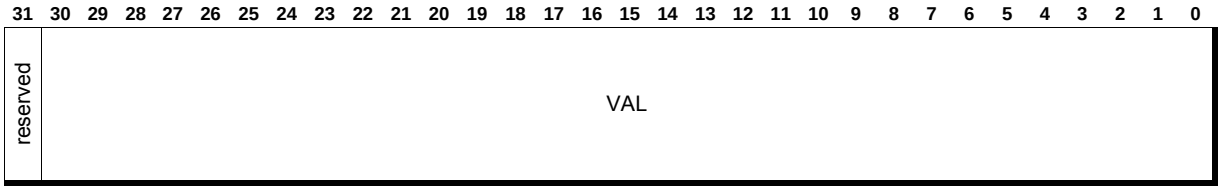
6.2 PIO – Programmable Input Output

The netX chip contains several programmable input / output lines. These lines could be used for the realisation of fieldbus I/O slaves or other digital controlling or monitoring. Each PIO can be used as simple input or output without any additional features. In total there are 84 PIO pins. The first 31 PIO pins (PIO0-PIO30) are shared with Motion Control pins, LCD pins and ETM pins. The 53 other Pins (PIO32-PIO84) are shared with Host Interface Pins. (PIO31 does not exist).

PIO0 - PIO30 pins are configured by the following three registers. The other 53 PIO pins are controlled by the host interface block. Please refer to registers DPM_ARM_IO_DATA1, DPM_ARM_IO_DATA2, DPM_ARM_IO_MODE1, DPM_ARM_IO_MODE2, DPM_ARM_IO_DRV_EN1 and DPM_ARM_IO_DRV_EN2 in section 5.2 DPM_ARM – Dual-Port Memory ARM Side.

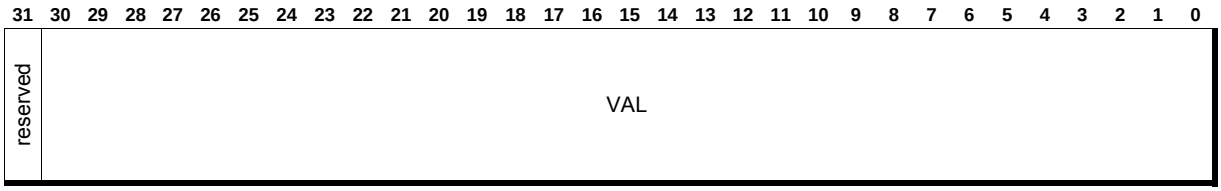
ARM Address	Register Name	Short Description
0x00100900	PIO_IN	PIO Input Register
0x00100904	PIO_OUT	PIO Output Register
0x00100908	PIO_OUT_EN	PIO Output Enable Register

PIO_IN – PIO Input Register 0x00100900



Bits	Name	Description	R/W	Default
31	reserved	-	R	0x00
30:0	VAL	PIO[30:0] input values	R	0x00

PIO_OUT – PIO Output Register 0x00100904



Bits	Name	Description	R/W	Default
31	reserved	-	R	0
30:0	VAL	PIO[30:0] output values	R/W	0x00

PIO_OUT_EN – PIO Output Enable Register 0x00100908

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	VAL																														

Bits	Name	Description	R/W	Default
31	reserved	-	R	0
30:0	VAL	PIO[30:0] output enables	R/W	0x00

6.3 UART – Universal Asynchronous Receiver Transmitter

There are three UART ports, which are 16550-compliant with 16 bytes transmit and receive FIFOs. They can be configured to support speeds up to 3.125 MBaud. The interface supports configurations of:

- five, six, seven, or eight data-bit transfers
- one or two stop bits
- even, odd, or no parity
- IrDA SIR encoding and decoding

The request-to-send (RTS) and clear-to-send (CTS) modem control signals are also available with the interface for hardware flow control. Special features like stick parity and adjustable FIFO trigger level are implemented.

The following table is a summary of registers of UART.

ARM Address	Register Name	Short Description
0x00100a00	UART0_DATA	UART0 Data Register
0x00100a04	UART0_STAT	UART0 Status Register
0x00100a08	UART0_LINE_CTRL	UART0 Line Control Register
0x00100a0c	UART0_BAUD_DIV_MSB	UART0 Baud Rate Divisor MSB
0x00100a10	UART0_BAUD_DIV_LSB	UART0 Baud Rate Divisor LSB
0x00100a14	UART0_CTRL	UART0 Control Register
0x00100a18	UART0_FLAG	UART0 Flag Register
0x00100a1c	UART0_INT_ID	UART0 Interrupt Identification Register
0x00100a20	UART0_IRDA_LO_PWR_CNTR	UART0 IrDA Low Power Counter Register
0x00100a24	UART0_RTS_CTRL	UART0 RTS Control Register
0x00100a28	UART0_RTS_LEAD_CYC	UART0 RTS Leading Cycles
0x00100a2c	UART0_RTS_TRAIL_CYC	UART0 RTS Trailing cycles
0x00100a30	UART0_OUT_DRV_EN	UART0 UART Output Driver Enable Register
0x00100a34	UART0_BAUD_MODE_CTRL	UART0 Baud Rate Mode Control Register
0x00100a38	UART0_RX_FIFO_IRQ_LVL	UART0 Receive FIFO Interrupt Trigger Level
0x00100a3c	UART0_TX_FIFO_IRQ_LVL	UART0 Transmit FIFO Interrupt Trigger Level
0x00100a40	UART1_DATA	UART1 Data Register
0x00100a44	UART1_STAT	UART1 Status Register
0x00100a48	UART1_LINE_CTRL	UART1 Line Control Register
0x00100a4c	UART1_BAUD_DIV_MSB	UART1 Baud Rate Divisor MSB
0x00100a50	UART1_BAUD_DIV_LSB	UART1 Baud Rate Divisor LSB
0x00100a54	UART1_CTRL	UART1 Control Register
0x00100a58	UART1_FLAG	UART1 Flag Register
0x00100a5c	UART1_INT_ID	UART1 Interrupt Identification Register
0x00100a60	UART1_IRDA_LO_PWR_CNTR	UART1 IrDA Low Power Counter Register
0x00100a64	UART1_RTS_CTRL	UART1 RTS Control Register
0x00100a68	UART1_RTS_LEAD_CYC	UART1 RTS Leading Cycles
0x00100a6c	UART1_RTS_TRAIL_CYC	UART1 RTS Trailing cycles
0x00100a70	UART1_OUT_DRV_EN	UART1 UART Output Driver Enable Register
0x00100a74	UART1_BAUD_MODE_CTRL	UART1 Baud Rate Mode Control Register

0x00100a78	UART1_RX_FIFO_IRQ_LVL	UART1 Receive FIFO Interrupt Trigger Level
0x00100a7c	UART1_TX_FIFO_IRQ_LVL	UART1 Transmit FIFO Interrupt Trigger Level
0x00100a80	UART2_DATA	UART2 Data Register
0x00100a84	UART2_STAT	UART2 Status Register
0x00100a88	UART2_LINE_CTRL	UART2 Line Control Register
0x00100a8c	UART2_BAUD_DIV_MSB	UART2 Baud Rate Divisor MSB
0x00100a90	UART2_BAUD_DIV_LSB	UART2 Baud Rate Divisor LSB
0x00100a94	UART2_CTRL	UART2 Control Register
0x00100a98	UART2_FLAG	UART2 Flag Register
0x00100a9c	UART2_INT_ID	UART2 Interrupt Identification Register
0x00100aa0	UART2_IRDA_LO_PWR_CNTR	UART2 IrDA Low Power Counter Register
0x00100aa4	UART2_RTS_CTRL	UART2 RTS Control Register
0x00100aa8	UART2_RTS_LEAD_CYC	UART2 RTS Leading Cycles
0x00100aac	UART2_RTS_TRAIL_CYC	UART2 RTS Trailing cycles
0x00100ab0	UART2_OUT_DRV_EN	UART2 UART Output Driver Enable Register
0x00100ab4	UART2_BAUD_MODE_CTRL	UART2 Baud Rate Mode Control Register
0x00100ab8	UART2_RX_FIFO_IRQ_LVL	UART2 Receive FIFO Interrupt Trigger Level
0x00100abc	UART2_TX_FIFO_IRQ_LVL	UART2 Transmit FIFO Interrupt Trigger Level

The ARM CPU reads and writes data and control/status information via the peripheral bus interface. The UART module can generate four individually-maskable interrupts which are combined to a single interrupt so that the output is asserted if any of the individual interrupts are asserted and unmasked.

If a framing, parity or break error occurs during reception, the appropriate error bit is set, and is stored in the FIFO. If an overrun condition occurs, the overrun register bit is set immediately and FIFO data is prevented from being overwritten.

The following is a brief introduction of different blocks in UART module.

Baud rate generator

The baud rate generator contains free-running counters which generate the internal Baud16 or IrLPBaud16 signal. Baud16 or IrLPBaud16 provide timing information for UART transmit and receive control. Baud16 is a stream of pulses with a width of 10 ns and a frequency of sixteen times the baud rate.

Transmit FIFO

The transmit FIFO is an 8-bit wide, 16-bit depth, first-in, first-out memory buffer. CPU data written across the bus interface is stored in the FIFO until read out by the transmit logic. The transmit FIFO can be disabled to act as a one-byte holding register.

Receive FIFO

The receive FIFO is an 11-bit wide, 16-bit depth, first-in, first-out memory buffer. Received data, and corresponding error bits, are stored in the receive FIFO by the receive logic until read out by the CPU across the bus interface. The FIFO can be disabled to act as a one-byte holding register.

Transmitter

The transmit logic performs parallel-to-serial conversion on the data read from the transmit FIFO. Control logic outputs the serial bit stream begins with a start bit, data bits, least significant bit (LSB) first, followed by parity bit, and then stop bits according to the programmed configuration in control registers.

Receiver
The receive logic performs serial-to-parallel conversion on the received bit stream after a valid start pulse has been detected. Parity, frame error checking and line break detection are also performed, and the data with associated parity, framing and break error bits is written to the receive FIFO.

Interrupt logic
Four individual maskable active HIGH interrupts are generated in the UART module and are combined to one interrupt output. This output is generated as an OR function of the individual interrupt requests. The single combined interrupt is used with the system interrupt controller that provides another level of masking on a per-peripheral basis. This allows use of modular device drivers which will always know where to find the interrupt source control register bits.

IrDA SIR Endec
The Transmitter and Receiver block contain an IrDA SIR protocol Endec. The SIR protocol Endec can be enabled for serial communication via signals nSIROUT and SIRIN to an infrared transducer instead of using the signals TXD and RXD. The SIR protocol Endec can both receive and transmit, but it is half-duplex only, so it cannot receive while transmitting, or vice versa.

The SIR transmit encoder modulates the Non Return-to-Zero (NRZ) transmit bit stream. The IrDA SIR physical layer specifies use of a Return To Zero, Inverted (RZI) modulation scheme which represents logic 0 as an infrared light pulse. The modulated output pulse stream is transmitted to an external output driver and infrared Light Emitting Diode (LED).

In normal mode the transmitted pulse width is specified as three times the period of the internal x16 clock (Baud16), that is, 3 / 16 of a bit period.

Low-power mode of the transmit infrared pulse is set to 3 times the period of the internal generated IrLPBaud16 signal. The frequency of IrLPBaud16 signal is set by writing the appropriate divisor value to UARTILPR.

The active low encoder output is normally LOW for the marking state (no light pulse). The encoder outputs a high pulse to generate an infrared light pulse representing a logic 0 or spacing state.

The SIR receive decoder demodulates the return-to-zero bit stream from the infrared detector and outputs the received NRZ serial bit stream to the internal logic. The decoder input is normally HIGH (marking state) in the idle state (the transmit encoder output has the opposite polarity to the decoder input).
A start bit is detected when the decoder input is LOW. Regardless of being in normal or low-power mode, a start bit is deemed valid if the decoder is still LOW, one period of IrLPBaud16 after the LOW was first detected.

UART communication
Data received or transmitted is stored in two 16-byte FIFOs, though the receive FIFO has an extra three bits per character for status information.

For transmission, data is written into the transmit FIFO. This causes a data frame to start transmitting with the parameters indicated in UARTLCR. Data continues to be transmitted until there is no data left in the transmit FIFO. The BUSY signal goes HIGH as soon as data is written to the transmit FIFO (that is, the FIFO is non-empty) and remains asserted HIGH while data is being transmitted. BUSY is negated only when the transmit FIFO is empty, and the last character has been transmitted from the shift register, including the stop bits. BUSY can be asserted HIGH even though the UART module may no longer be enabled.

When the receiver is idle (RXD continuously 1, in the marking state) and a LOW is detected on the data input (a start bit has been received), the receive counter, with the clock enabled by Baud16, begins running and data is sampled on the eighth cycle of that counter (half way through a bit period). The start

bit is valid if RXD is still LOW on the eighth cycle of Baud16, otherwise a false start bit is detected and it is ignored.

If the start bit was valid, successive data bits are sampled on every 16th cycle of Baud16 (that is, one bit period later) according to the programmed length of the data characters. The parity bit is then checked if parity mode was enabled.
Lastly, a valid stop bit is confirmed if RXD is HIGH, otherwise a framing error has occurred. When a full word has been received, the data is stored in the receive FIFO, with any error bits associated with that word.

Error bits

The three error bits are stored in bits 10:8 of the receive FIFO, and are associated to a particular character. There is an additional error which indicates an overrun error but it is not associated with a particular character in the receive FIFO. The overrun error is set when the FIFO is full and the next character has been completely received in the shift register. The data in the shift register is overwritten but it is not written into the FIFO.
FIFO bits 7:0 : received data
FIFO bit 8 : framing error
FIFO bit 9 : parity error
FIFO bit 10 : break error

Disabling the FIFOs

Additionally, it is possible to disable the FIFOs. In this case, transmit and receive sides of the UART module have 1-byte holding registers (the bottom entry of the FIFOs). The overrun bit is set when a word has been received and the previous one was not yet read.

The UARTs are shared with the General Purpose IOs to save pins. So to use the UART Interface, the 'IOCFG' bits in the 'GPIO Configuration Registers' must be set to UART mode (IOCFG[1:0] = 10). This configuration mode switches the signal paths from the GPIO pins to the UART module. The following table shows the UART pins and the corresponding GPIO signal lines.

GPIO Pin	UART Signal	Corresponding Configuration Register to be programmed (IOCFG[1:0] = 10)
GPIO0	UART0_rxd	GPIO_CFG0: GPIO 0 Configuration Register (0x00100800)
GPIO1	UART0_txd	GPIO_CFG1: GPIO 1 Configuration Register (0x00100804)
GPIO2	UART0_cts	GPIO_CFG2: GPIO 2 Configuration Register (0x00100808)
GPIO3	UART0_rts	GPIO_CFG3: GPIO 3 Configuration Register (0x0010080c)
GPIO4	UART1_rxd	GPIO_CFG4: GPIO 4 Configuration Register (0x00100810)
GPIO5	UART1_txd	GPIO_CFG5: GPIO 5 Configuration Register (0x00100814)
GPIO6	UART1_cts	GPIO_CFG6: GPIO 6 Configuration Register (0x00100818)
GPIO7	UART1_rts	GPIO_CFG7: GPIO 7 Configuration Register (0x0010081c)
GPIO8	UART2_rxd	GPIO_CFG8: GPIO 8 Configuration Register (0x00100820)
GPIO9	UART2_txd	GPIO_CFG9: GPIO 9 Configuration Register (0x00100824)
GPIO10	UART2_cts	GPIO_CFG10: GPIO 10 Configuration Register (0x00100828)
GPIO11	UART2_rts	GPIO_CFG11: GPIO 11 Configuration Register (0x0010082c)

The UART communication is controlled by 16 registers which are described as follows.

UART0_DATA – UART 0 Data Register **0x00100a00**
UART1_DATA – UART 1 Data Register **0x00100a40**
UART2_DATA – UART 2 Data Register **0x00100a80**

These registers are the send and receive register. The data is read form the interface or written to the interface.

For data to be transmitted:
• If the FIFOs are enabled, data written to this location is pushed onto the 16 byte deep transmit FIFO.
• If the FIFOs are not enabled, data is stored into the transmitter holding register.

The write operation initiates transmission from the UART. The data is prefixed with a start bit, appended with the appropriate parity bit (if parity is enabled), and a stop bit. The resultant word is then transmitted. Before you can send any data you have to enable the UARTi_TXD or UARTi_RTS driver (see UART_OUT_DRV_EN register).

For received data:
• If the FIFOs are enabled, the data byte is extracted and a 3-bit status (break, frame and parity) is pushed onto the 11-bit wide receive FIFO.
• If the FIFOs are not enabled, the data byte and status are stored in the receiving holding register (the bottom of the receive FIFO).

The received data must be read first from UART_DATA registers, followed by the status error associated with the data from UART_STAT registers. This read sequence cannot be reversed. The UART must be disabled before any of the control registers are reprogrammed.

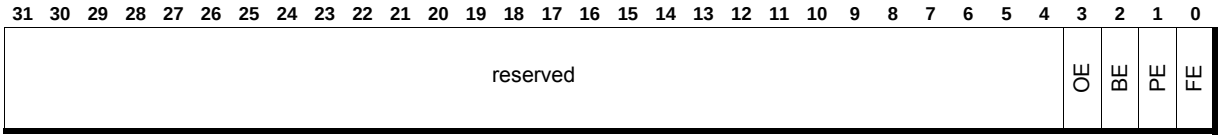
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								DATA							

Bits	Name	Description	R/W	Default
7:0	DATA	Receive data character by reading. Transmit data character by writing.	R/W	0x00

UART0_STAT – UART 0 Status Register **0x00100a04**
UART1_STAT – UART 1 Status Register **0x00100a44**
UART2_STAT – UART 2 Status Register **0x00100a84**

Receive status is read from UART_STAT registers. The status information corresponds to the data character read from UART_DATA registers prior to reading UART_STAT registers. A write to UART_STAT registers clears the framing, parity, break and overrun errors. All the bits are cleared to 0 on reset.

The received data character must be read first from UART_DATA before reading the error status associated with that data character from UART_STAT. This read sequence cannot be reversed, since the status register UART_STAT is updated only when a read occurs from the data register UART_DATA.



Bits	Name	Description	R/W	Default
31:4	reserved	-	R	0x00
3	OE	Overrun Error The FIFO contents remain valid since no further data is written when the FIFO is full, only the contents of the shift register are overwritten. The ARM must now read the data in order to empty the FIFO. 0: after a write to UART_STAT register. 1: if data is received and the FIFO is already full.	R/W	0
2	BE	Break Error In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to 1 (marking state) and the next valid start bit is received. 0: after a write to UART_STAT register. 1: if a break condition was detected, indicating that the received data input was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits).	R/W	0
1	PE	Parity Error In FIFO mode, this error is associated with the character at the top of the FIFO. 0: by a write to UART_STAT register. 1: it indicates that the parity of the received data character does not match the parity selected in UART_LINE_CTRL (bit 2).	R/W	0
0	FE	Framing Error In FIFO mode, this error is associated with the character at the top of the FIFO. 0: by a write to UART_STAT register. 1: it indicates that the received character did not have a valid stop bit (a valid stop bit is 1).	R/W	0

UART0_LINE_CTRL – UART 0 Line Control Register **0x00100a08**
UART1_LINE_CTRL – UART 1 Line Control Register **0x00100a48**
UART2_LINE_CTRL – UART 2 Line Control Register **0x00100a88**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																									WLEN		FEN	STP2	EPS	PEN	BRK

Bits	Name	Description	R/W	Default
31:7	reserved	-	R	0x00
6:5	WLEN	Word length. The bits indicate the number of data bits transmitted or received in a frame as follows: 00 : 5 bits 01 : 6 bits 10 : 7 bits 11 : 8 bits	R/W	00
4	FEN	Enable FIFOs. 0: the FIFOs are disabled (character mode) that is, the FIFOs become 1-byte-deep holding registers. 1: transmit and receive FIFO buffers are enabled (FIFO mode).	R/W	0
3	STP2	Two Stop Bits Select. The receive logic does not check for two stop bits being received. 1: two stop bits are transmitted at the end of the frame.	R/W	0
2	EPS	Even Parity Select. This bit has no effect when parity is disabled by Parity Enable (bit 1) being cleared to 0. 1: even parity generation and checking is performed during transmission and reception, which checks for an even number of 1 in data and parity bits. 0: odd parity is performed which checks for an odd number of 1.	R/W	0
1	PEN	Parity Enable. 0: parity is disabled and no parity bit added to the data frame. 1: parity checking and generation is enabled.	R/W	0
0	BRK	Send Break. 0: for normal use this bit must be cleared to 0. 1: a low level is continually output on the uart_txd output, after completing transmission of the current character. This bit must be asserted for at least one complete frame transmission time in order to generate a break condition. The transmit FIFO contents remain unaffected during a break condition.	R/W	0

UART Baud Rate Generation

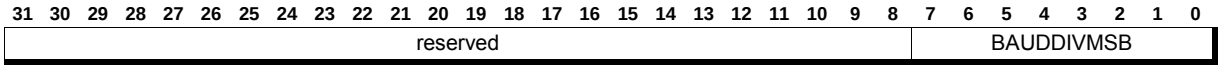
Depending on the UART_BAUD_MODE_CTRL[0] bit the baud rate is calculated by the following formulars:

BAUDDIV = (100 MHz / (16 • BaudRate)) – 1 (UART_BAUD_MODE_CTRL[0] = 0)

BAUDDIV = ((Baud Rate • 16) / 100 MHz) • 2^{16} . (UART_BAUD_MODE_CTRL[0] = 1)

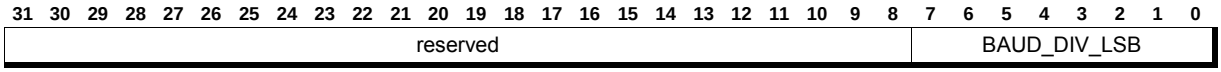
The maximum Baud rate is 3.125 MBaud.

UART0_BAUD_DIV_MSB – UART 0 Baud Rate Divisor MSB 0x00100a0c
UART1_BAUD_DIV_MSB – UART 1 Baud Rate Divisor MSB 0x00100a4c
UART2_BAUD_DIV_MSB – UART 2 Baud Rate Divisor MSB 0x00100a8c



Bits	Name	Description	R/W	Default
31:8	Reserved	-	R	0x00
7:0	BAUDDIV_MSB	Baud Rate Divisor [15:8]. Most significant byte of baud rate divisor (BAUDDIV).	R/W	0x00

UART0_BAUD_DIV_LSB – UART 0 Baud Rate Divisor LSB 0x00100a10
UART1_BAUD_DIV_LSB – UART 1 Baud Rate Divisor LSB 0x00100a50
UART2_BAUD_DIV_LSB – UART 2 Baud Rate Divisor LSB 0x00100a90



Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	BAUDDIV_LSB	Baud Rate Divisor [7:0]. Least significant byte of baud rate divisor (BAUDDIV).	R/W	0x00

Note 1:
The data values of the UART_BAUD_DIV_MSB and UART_BAUD_DIV_LSB registers are first stored into a shadow register. Only after a write to the UART_LINE_CTRL register the programmed baudrate is used by the UART baudrate generator.

Note 2:
A divisor value of zero is illegal, and so no transmission or reception will occur.

UART0_CTRL – UART 0 Control Register **0x00100a14**
UART1_CTRL – UART 1 Control Register **0x00100a54**
UART2_CTRL – UART 2 Control Register **0x00100a94**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								LBE	RTIE	TIE	RIE	MSIE	SIRLP	SIREN	UARTEN

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7	LBE	Loop back enable. This bit is cleared to 0 on reset, which disables the loop back mode. Loop Back is only in IrDA mode possible.	R/W	0
6	RTIE	Receive timeout interrupt enable. 1: the receive timeout interrupt is enabled.	R/W	0
5	TIE	Transmit interrupt enable. 1: the transmit interrupt is enabled.	R/W	0
4	RIE	Receive interrupt enable. 1: the receive interrupt is enabled.	R/W	0
3	MSIE	Modem status interrupt enable. 1: the modem status interrupt is enabled.	R/W	0
2	SIRLP	IrDA SIR low power mode. This bit selects the IrDA encoding mode. 0: low level bits are transmitted as an active high pulse with a width of 3 / 16 the of the bit period. 1: low level bits are transmitted with a pulse width which is 3 times the period of the IrLPBaud16 input signal, regardless of the selected bit rate. Setting this bit uses less power, but may reduce transmission distance.	R/W	0
1	SIREN	SIR enable. 1: the IrDA SIR Endec is enabled. This bit has no effect if the UART is not enabled by bit 0 being set to 1. When the IrDA SIR Endec is enabled, data is transmitted and received on nSIROUT and SIRIN. Uart_txd remains in the marking state (set to 1). Signal transitions or modem status inputs will have no effect. When the IrDA SIR Endec is disabled, nSIROUT remains cleared to 0 (no light pulse generated), and signal transitions on SIRIN will have no effect.	R/W	0
0	UARTEN	UART enable. If this bit is set to 1, the UART is enabled. Data transmission and reception occurs for either UART signals or SIR signals according to the setting of SIR Enable (bit 1).	R/W	0

UART0_FLAG – UART 0 Flag Register **0x00100a18**
UART1_FLAG – UART 1 Flag Register **0x00100a58**
UART2_FLAG – UART 2 Flag Register **0x00100a98**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								TXFE	RXFF	TXFF	RXFE	BUSY	DCD	DSR	CTS

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7	TXFE	Transmit FIFO empty. The meaning of this bit depends on the state of the FEN bit in the UART_LINE_CTRL register. If the FIFO is disabled, this bit is set when the transmit holding register is empty. If the FIFO is enabled, the TXFE bit is set when the transmit FIFO is empty.	R	0
6	RXFF	Receive FIFO full. The meaning of this bit depends on the state of the FEN bit in the UART_LINE_CTRL register. If the FIFO is disabled, this bit is set when the receive holding register is full. If the FIFO is enabled, the RXFF bit is set when the receive FIFO is full.	R	0
5	TXFF	Transmit FIFO full. The meaning of this bit depends on the state of the FEN bit in the UART_LINE_CTRL register. If the FIFO is disabled, this bit is set when the transmit holding register is full. If the FIFO is enabled, the TXFF bit is set when the transmit FIFO is full.	R	0
4	RXFE	Receive FIFO empty. The meaning of this bit depends on the state of the FEN bit in the UART_LINE_CTRL register. If the FIFO is disabled, this bit is set when the receive holding register is empty. If the FIFO is enabled, the RXFE bit is set when the receive FIFO is empty.	R	0
3	BUSY	UART busy. If this bit is set to 1, the UART is busy transmitting data. This bit remains set until the complete byte, including all the stop bits, has been sent from the shift register. This bit is set as soon as the transmit FIFO becomes non-empty (regardless of whether the UART is enabled or not).	R	0
2	DCD	Data carrier detect. This bit is actually not supported. Always read as 0.	R	0
1	DSR	Data set ready. This bit is actually not supported. Always read as 0.	R	0
0	CTS	Clear to send. This bit is the complement of the modem status input pin UARTi_CTS. That is (when CTS_POL of UART_RTS_CTRL register is not set) the bit is 1 when the modem status input is 0. When CTS_POL of UART_RTS_CTRL register is set to 1 this bit is inverted.	R	0

UART0_INT_ID – UART 0 Interrupt Identification Register **0x00100a1c**
UART1_INT_ID – UART 1 Interrupt Identification Register **0x00100a5c**
UART2_INT_ID – UART 2 Interrupt Identification Register **0x00100a9c**

UART_INT_ID is the interrupt identification register/interrupt clear register. The memory location has different functions. Interrupt status is read from UART_INT_ID. A write to this register clears the modem status interrupt.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																												RTIS	TIS	RIS	MIS

Bits	Name	Description	R/W	Default
31:4	reserved	-	R	0x00
3	RTIS	Receive timeout interrupt status. 1: if the receive timeout interrupt is asserted.	R/W	0
2	TIS	Transmit interrupt status. 1: if the transmit interrupt is asserted.	R/W	0
1	RIS	Receive interrupt status. 1: if the receive interrupt is asserted.	R/W	0
0	MIS	Modem Interrupt Status. 1: if the modem status interrupt is asserted.	R/W	0

Interrupts
There are four different interrupts generated by the UART. They are combined into a single interrupt request which is an OR function of the individual masked sources. This output is connected to the system interrupt controller VIC to provide another level of masking on an individual peripheral basis. The combined UART interrupt is asserted if any of the four individual interrupts above are asserted and enabled.

The transmit and receive dataflow interrupts have been separated from the status interrupts. This allows to be used in a DMA controller, so that data can be read or written in response to just the FIFO trigger levels. The status of the individual interrupt sources can be read from UART_INT_ID.

The modem status interrupt
is asserted if the modem status line UARTi_CTS change. It is cleared by writing to the UART_INT_ID register.

The receive interrupt
changes state when one of the following events occurs:
• If the FIFOs are enabled and the receive FIFO is half or more full (it contains eight or more words), then the receive interrupt is asserted HIGH. The receive interrupt is cleared by reading data from the receive FIFO until it becomes less than half full. By changing the value of UART_RX_FIFO_IRQ_LVL register you can change the interrupt trigger level.
• If the FIFOs are disabled (have a depth of one location) and data is received thereby filling the location, the receive interrupt is asserted HIGH. The receive interrupt is cleared by performing a single read of the receive FIFO.

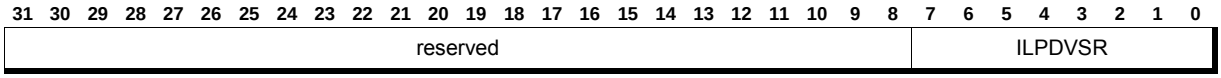
The transmit interrupt
changes state when one of the following events occurs:

• If the FIFOs are enabled and the transmit FIFO is at least half empty (it has space for eight or more words), then the transmit interrupt is asserted HIGH. It is cleared by filling the transmit FIFO to more than half full. By changing the value of UART_TX_FIFO_IRQ_LVL register you can change the interrupt trigger level.
• If the FIFOs are disabled (have a depth of one location) and there is no data present in the transmitters single location, the transmit FIFO is asserted HIGH. It is cleared by performing a single write to the transmitter FIFO.
The transmit interrupt is not qualified with the UART Enable signal, which allows operation in one of two ways. Data can be written to the transmit FIFO prior to enabling the UART and the interrupts. Alternatively, the UART and interrupts can be enabled so that data can be written to the transmit FIFO by an interrupt service routine.

The receive timeout interrupt
is asserted when the receive FIFO is not empty and no further data is received over a 32-bit period. The receive timeout interrupt is cleared when the FIFO becomes empty through reading all the data (or by reading the holding register).

UART0_IRDA_LO_PWR_CNTR – UART 0 IrDA Low Power Counter Register 0x00100a20
UART1_IRDA_LO_PWR_CNTR – UART 1 IrDA Low Power Counter Register 0x00100a60
UART2_IRDA_LO_PWR_CNTR – UART 2 IrDA Low Power Counter Register 0x00100aa0

UART_IRDA_LO_PWR_CNTR is the IrDA low-power counter register. This is an 8-bit read/write register that stores the low-power counter divisor value used to generate the internal IrLPBaud16 (16*Baudrate=IrLPBaud16) signal.



Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	ILPDVSR	IrDA Low Power Divisor [7:0]. 8-bit low-power divisor value.	R/W	0x00

The low power divisor value is calculated as follows:

ILPDVSR = (100MHz / 16*Baudrate) – 1

Note:
Zero is an illegal value. Programming a zero value will result in no IrLPBaud16 pulses being generated.

UART0_RTS_CTRL – UART 0 RTS Control Register **0x00100a24**
UART1_RTS_CTRL – UART 1 RTS Control Register **0x00100a64**
UART2_RTS_CTRL – UART 2 RTS Control Register **0x00100aa4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								STICK	CTS_POL	CTS_CTR	RTS_POL	MOD2	COUNT	RTS	AUTO

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7	STICK	Parity bit works as stick bit. 0 : the parity bit not stick and normally calculated. 1 : parity bit is the inverted bit EPS of UART_LINE_CTRL register.	R/W	0
6	CTS_POL	CTS polarity. 0 : CTS input is active low. 1 : CTS input is active high.	R/W	0
5	CTS_CTR	CTS control. 0 : the UART starts transmitting regardless CTS input pin. 1 : the UART starts transmitting only if the CTS input signal is active. After each character which has been send the UART checks if the CTS input is still active. If it is active it continues transmitting otherwise it will wait for the CTS input.	R/W	0
4	RTS_POL	RTS polarity. 0 : RTS output is active low. 1 : RTS output is active high.	R/W	0
3	MOD2	There are two modes you can choose when AUT0 is set to 1. 0 : After every character which has been send the internal state machine goes into the trail state which means that the bit stream is stopped for a while (see UART_RTS_TRAIL_CYC register). 1 :The internal state machine goes only into the trail state when the transmit FIFO is empty.	R/W	0
2	COUNT	RTS counter time base. 0 : the internal counter bases on baud times. 1 : the forerun and trail cycles of RTS Signal are counted is system clock cycles (100 MHz).	R/W	0
1	RTS	If AUTO=0 then the RTS output is set by this bit.	R/W	0
0	AUTO	0 : RTS output is controlled directly by bit 1 of UART_RTS_CTRL register. 1 : RTS output is automatically assigned by the internal state machine. See also bit 2-6 of UART_RTS_CTRL register.	R/W	0

UART0_RTS_LEAD_CYC – UART 0 RTS Leading Cycles 0x00100a28
UART1_RTS_LEAD_CYC – UART 1 RTS Leading Cycles 0x00100a68
UART2_RTS_LEAD_CYC – UART 2 RTS Leading Cycles 0x00100aa8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								LEAD_CYC							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	LEAD_CYC	Number of leading cycles in system clocks or baud rate cycles.	R/W	0x00

UART0_RTS_TRAIL_CYC – UART 0 RTS Trailing cycles 0x00100a2c
UART1_RTS_TRAIL_CYC – UART 1 RTS Trailing cycles 0x00100a6c
UART2_RTS_TRAIL_CYC – UART 2 RTS Trailing cycles 0x00100aac

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								TRAIL_CYC							

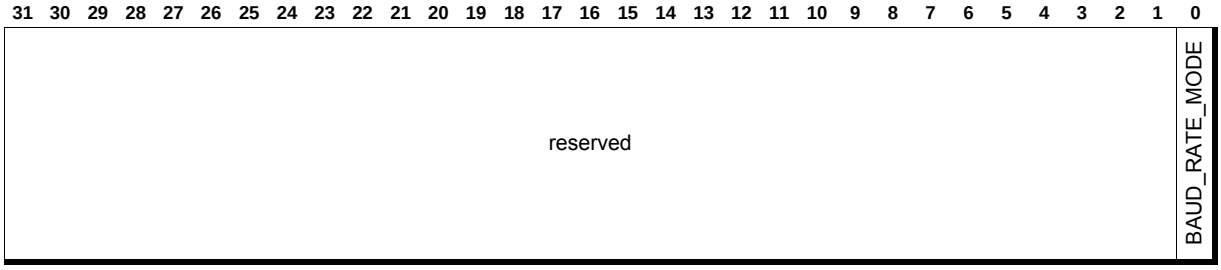
Bits	Name	Description	R/W	Default
31:8	Reserved	-	R	0x00
7:0	TRAIL_CYC	Number of trail cycles in system clocks or baud rate cycles.	R/W	0x00

UART0_OUT_DRV_EN – UART 0 Output Driver Enable Register 0x00100a30
UART1_OUT_DRV_EN – UART 1 Output Driver Enable Register 0x00100a70
UART2_OUT_DRV_EN – UART 2 Output Driver Enable Register 0x00100ab0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																														DRVRTS	DRVTX

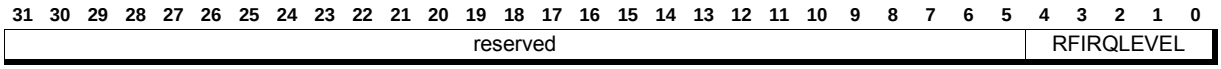
Bits	Name	Description	R/W	Default
31:2	reserved	-	R	0x00
1	DRVRTS	This bit enables the driver for UARTi_RTS output pin.	R/W	0
0	DRVTX	This bit enables the driver for UARTi_TXD output pin.	R/W	0

UART0_BAUD_MODE_CTRL – UART 0 Baud Rate Mode Control Register 0x00100a34
UART1_BAUD_MODE_CTRL – UART 1 Baud Rate Mode Control Register 0x00100a74
UART2_BAUD_MODE_CTRL – UART 2 Baud Rate Mode Control Register 0x00100ab4



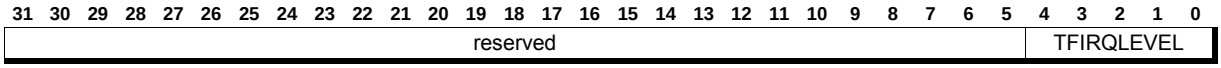
Bits	Name	Description	R/W	Default
31:1	reserved	-	R	0x00
0	BAUD_RATE_MODE	Sets the generation method of baudrate. See the 'BAUDDIV' of UART_BAUD_DIV registers for calculating the baudrate.		0

UART0_RX_FIFO_IRQ_LVL – UART 0 Receive FIFO Interrupt Trigger Level 0x00100a38
UART1_RX_FIFO_IRQ_LVL – UART 1 Receive FIFO Interrupt Trigger Level 0x00100a78
UART2_RX_FIFO_IRQ_LVL – UART 2 Receive FIFO Interrupt Trigger Level 0x00100ab8



Bits	Name	Description	R/W	Default
31:5	reserved	-	R	0x00
4:0	RFIRQLEVEL	IRQ trigger level of the receive FIFO. Choose a number between 1 and 16. The UART receive interrupt will be set if the number of received bytes in the receive FIFO are greater than or equal RFIRQLEVEL.	R/W	0x08

UART0_TX_FIFO_IRQ_LVL – UART 0 Transmit FIFO Interrupt Trigger Level 0x00100a3c
UART1_TX_FIFO_IRQ_LVL – UART 1 Transmit FIFO Interrupt Trigger Level 0x00100a7c
UART2_TX_FIFO_IRQ_LVL – UART 2 Transmit FIFO Interrupt Trigger Level 0x00100abc



Bits	Name	Description	R/W	Default
31:5	reserved	-	R	0x00
4:0	TFIRQLEVEL	IRQ trigger level of the transmit FIFO. Choose a number between 1 and 16. The UART transmit interrupt will be set if the number of transmitted bytes in the transmit FIFO are less than TFIRQLEVEL.	R/W	0x08

6.4 SPI – Serial SPI-Interface

Beside the I2C, the SPI is the most common serial interface for peripherals and memory components. It can be also used to connect Smart Media Cards.

The SPI unit supports all four transfer modes with 16 different speeds and can be configured as master or slave. As master the maximum speed is 25 MHz. The interface is completely interrupt driven and provides separate 16x16 bit FIFOs for incoming and outgoing data. The fuel level of the FIFOs can be read at any time. The IRQ output signal is the disjunction of 7 internal IRQ signals. Each of them are maskable by software and can be read and reset by software either.

There are three select signals available to connect up to three different SPI devices without any additional glue logic. These chip select signals can be controlled by either the internal SPI communication state machine or directly by software.

The burst length can be programmed and is from 1 to 128. Also the burst delay is programmable (0-7 SPI clock cycles).

The following table is a summary of registers of SPI.

ARM Address	Register Name	Short Description
0x00100c00	SPI_DATA	SPI Data Register
0x00100c04	SPI_STAT	SPI Status Register
0x00100c08	SPI_CTRL	SPI Control Register
0x00100c0c	SPI_INT_CTRL	SPI Interrupt Control Register

SPI_DATA – SPI Data Register 0x00100c00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved														DR_VALID1	DR_VALID0	DATA_BYTE_1								DATA_BYTE_0							

Bits	Name	Description	R/W	Default
31:18	reserved	-	R	0x00
17	DR_VALID1	valid bit for data_byte_1 This bit shows if DATA_BYTE_1 Is valid.	R/W	0
16	DR_VALID0	valid bit for data_byte_0 This bit shows if DATA_BYTE_0 Is valid.	R/W	0
15:8	DATA_BYTE_1	data byte 1	R/W	0x00
7:0	DATA_BYTE_0	data byte 0	R/W	0x00

SPI_STAT – SPI Status Register 0x00100c04

Shows the actual status of the SPI interface. Bits 24..18 show occurred interrupts, writing ones into these bits deletes the interrupts. Writing into other bits has no effect

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved						SR_SELECTED	SR_OUT_FULL	SR_OUT_EMPTY	SR_OUT_FW	SR_OUT_FUEL	SR_IN_FULL	SR_IN_RECDATA	SR_IN_FUEL	SR_OUT_FUEL_VAL									SR_IN_FUEL_VAL									

Bits	Name	Description	R/W	Default
31:26	reserved	-	R	0x00
25	SR_SELECTED	external master has access to spi-interface	R	0
24	SR_OUT_FULL	output FIFO is full	R/W	0
23	SR_OUT_EMPTY	output FIFO is empty and interface sending data (occures only in slave mode)	R/W	0
22	SR_OUT_FW	ARM is writing data too fast into output FIFO	R/W	0
21	SR_OUT_FUEL	adjustable fuel value of output FIFO reached	R/W	0
20	SR_IN_FULL	input FIFO is full	R/W	0
19	SR_IN_RECDATA	valid data bytes in input FIFO	R/W	0
18	SR_IN_FILL_LEVEL	adjustable fill level of input FIFO reached	R/W	0
17:9	SR_OUT_FILL_VAL	output FIFO fill value (number of bytes)	R	0x00
8:0	SR_IN_FILL_VAL	input FIFO fill value (number of bytes)	R	0x00

SPI_CTRL – SPI Control Register 0x00100c08

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CR_EN	CR_MS	CR_CPOL	CR_NCPHA	CR_BURST			CR_BURSTDELAY			CR_CLR_OUTFIFO	CR_CLR_INFIFO	reserved								CS_MODE	CR_SS			CR_WRITE	CR_READ	reserved	CR_SPEED				CR_SOFTRESET

Bits	Name	Description	R/W	Default
31	CR_EN	0 : disable SPI interface 1 : enable SPI interface	R/W	0x00
30	CR_MS	0 : slave mode 1 : master mode	R/W	0x00
29	CR_CPOL	0 : rising edge of spi_sck is primary 1 : falling edge of spi_sck is primary	R/W	0x00
28	CR_NCPHA	relative to CR_cpol 1 : change data to primary spi_sck edge data are activ to secondary spi_sck edge 1 : change data to secondary spi_sck edge data are active to primary spi_sck edge	R/W	0x00
27:25	CR_BURST	burst length = 2^CR_burst	R/W	0x00
24:22	CR_BURSTDELAY	delay between transmission of 2 data bytes (0 to 7 SCK cycles)	R/W	0x00
21	CR_CLR_OUTFIFO	clear output FIFO	R/W	0x00
20	CR_CLR_INFIFO	clear input FIFO	R/W	0x00
19:12	reserved	-	R	0x00
11	CS_MODE	0 : chip select is directly controlled by software. (see bits CR_SS). 1 : chip select is generated automatically by the internal state machine.	R/W	0
10:8	CR_SS	external slave select SPI_CSn[2:0] 0 : slave not selected (physical output level is high) 1 : slave selected (physical output level is low)	R/W	0x00
7	CR_WRITE	1 : spi interface write data enable	R/W	0x00
6	CR_READ	1 : spi interface read data enable	R/W	0x00
5	reserved	-	R	0
4:1	CR_SPEED	SPI Clock Speed 0001 : 0,05 MHz 0010 : 0,1 MHz 0011 : 0,2 MHz 0100 : 0,5 MHz 0101 : 1 MHz 0110 : 1,25 MHz 0111 : 2 MHz 1000 : 2,5 MHz 1001 : 3,3333 MHz 1010 : 5 MHz 1011 :10 MHz 1100 :12,5 MHz 1101 :16,6666 MHz 1110 :25 MHz 1111 :reserved	R/W	0x00
0	CR_SOFTRESET	not implemented	R/W	0

SPI_INT_CTRL – SPI Interrupt Control Register 0x00100c0c

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved							IR_OUT_FULL_EN	IR_OUT_EMPTY_EN	IR_OUT_FW_EN	IR_OUT_FUEL_EN	IR_IN_FULL_EN	IR_IN_RECDATA_EN	IR_IN_FUEL_EN	IR_OUT_FUEL										IR_IN_FUEL								

Bits	Name	Description	R/W	Default
31:25	reserved	-	R	0x00
24	IR_OUT_FULL_EN	IRQ enable for output FIFO is full	R/W	0
23	IR_OUT_EMPTY_EN	IRQ enable for output FIFO is empty and interface sending data (occures only in slave mode)	R/W	0
22	IR_OUT_FW_EN	IRQ enable for ARM is writing data to fast into output FIFO	R/W	0
21	IR_OUT_FUEL_EN	IRQ enable for adjustable fuel value of output FIFO reached	R/W	0
20	IR_IN_FULL_EN	IRQ enable for input FIFO is full	R/W	0
19	IR_IN_RECDATA_EN	IRQ enable for valid data bytes in input FIFO	R/W	0
18	IR_IN_FUEL_EN	IRQ enable for adjustable fill level of input FIFO reached	R/W	0
17:9	IR_OUT_FUEL	adjustable full level for output FIFO	R/W	0x00
8:0	IR_IN_FUEL	adjustable full level for input FIFO	R/W	0x00

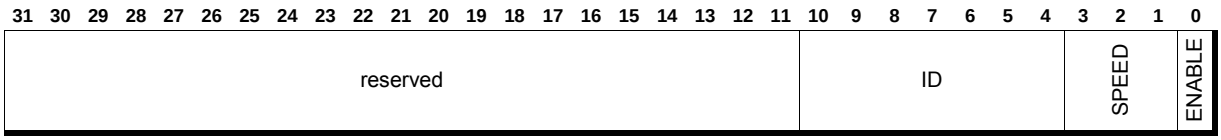
6.5 I2C – Serial I2C-Interface

The I2C Bus Interface Unit provides a general purpose 2-pin serial communication port. This is a standard interface for a wide range of peripheries and memory components. The netX chip is always I2C master. There are 8 different speeds for the I2C_SCL signal (clock). The maximum speed is 1MHz.

There is only two registers about setting I2C.

ARM Address	Register Name	Short Description
0x00100d00	I2C_CTRL	I2C Control Register
0x00100d04	I2C_DATA	I2C Data Register

I2C_CTRL – I2C Control Register 0x00100d00



Bits	Name	Description	R/W	Default
31:11	reserved	-	R	0x00
10:4	ID	Slave device ID	R/W	0x00
3:1	SPEED	speed select when 100 MHz system clock: 000 : 25 kHz 001 : 50 kHz 010 : 100 kHz (normal mode) 011 : 200 kHz 100 : 400 kHz (fast mode) 101 : 600 kHz 110 : 800 kHz 111 : 1000 kHz	R/W	0x00
0	ENABLE	0 : interface disable 1 : interface enable	R/W	0

I2C_DATA – I2C Data Register 0x00100d04

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																	ACK		RDF	CMD3	CMD2	CMD1	CMD0	DATA							

Bits	Name	Description	R/W	Default
31:15	reserved	-	R	0x00
14:13	ACK	Number of acknowledge bits send by slave in actual command. Allows software to detect abnormalities of slaves.	R/W	0x00
12	RDF	1 : Read Data Finished: reading data word from slave is finished	R/W	0
11	CMD3	Command Bit 3: execute 0 : no operation, ready, next command can be programmed by CPU 1 : execute command, interface busy	R/W	0
10	CMD2	Command Bit 2: start 0 : nothing 1 : send Start Condition + ID	R/W	0
9	CMD1	Command Bit 1: read -write 0 : write Byte 1 : read Byte	R/W	0
8	CMD0	Command Bit 0: stop 0 : nothing 1 : send Stop Condition	R/W	0
7:0	DATA	transmit and receive data byte	R/W	0x00

6.6 SYS_TIME – System time with IEEE 1588 functionality

The precision System Time is counted with the 100 MHz clock and has a resolution of 10 ns from the view of the application. Due drift, aging or failure of the crystal this time can be differ from a system wide master clock which is very often needed in Real-time Ethernet system.

Therefore the System Time is not realized as an usually counter but is an adder which adds every clock period 10 to the actual time value. If the 100 MHz clock has a deviation to the master clock then the added value will slightly differ of 10 with a resolution of 2^{-28} ns to compensate that failure. This could be calculated based on the protocol of IEEE 1588 or other Real-time Ethernet functions.

The System Time is leaded in two 32-Bit registers. One contains the second value and the other the nanoseconds value from time zero. The application has to read the second value first, because this will freeze the nanoseconds register to get a consistent System Time.

The following table is a summary of registers of system time.

ARM Address	Register Name	Short Description
0x00101100	SYS_TIME_NS	System Time Nanosecond Register
0x00101104	SYS_TIME_S	System Time Second Register
0x00101108	SYS_TIME_NS_BOR	System Time Nanoseconds Border Register
0x0010110c	SYS_TIME_NS_ADD_UP	System Time Nanoseconds Add Up Register
0x00101110	SYS_TIME_S_CMP	System Time Second Compare Register
0x00101114	SYS_TIME_S_CMP_EN	System Time Second Compare Enable Register
0x00101118	SYS_TIME_S_CMP_INT	System Time Second Compare Interrupt Register

SYS_TIME_NS – System Time Nanosecond Register 0x00101100

The SYS_TIME_NS register provides the lower part of the System time in nanoseconds. Used for counting nanoseconds according IEEE 1588.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYS_TIME_NS																															

Bits	Name	Description	R/W	Default
31:0	SYS_TIME_NS	Lower part of the System time in nanoseconds.	R/W	0x00000000

SYS_TIME_S – System Time Second Register 0x00101104

The SYS_TIME_S register provides the upper part of the System time in seconds. Used for counting seconds according IEEE 1588.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYS_TIME_S																															

Bits	Name	Description	R/W	Default
31:0	SYS_TIME_S	Higher part of the system time in seconds. Value is incremented if SYS_TIME_NS reaches SYS_TIME_NS_BOR.	R/W	0x00000000

SYS_TIME_NS_BOR – System Time Nanoseconds Border Register 0x00101108

The SYS_TIME_NS_BOR register configures the border for the lower part of the system time.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYS_TIME_NS_BOR																															

Bits	Name	Description	R/W	Default
31:0	SYS_TIME_NS_BOR	SYS_TIME_NS counts from 0 to this value (including this value). I.e. SYS_TIME_NS counts modulo (SYS_TIME_NS_BOR +1). Default value corresponds to 1 second.	R/W	0x3b9ac9ff

SYS_TIME_NS_ADD_UP – System Time Nanoseconds Add Up Register 0x0010110c

The SYS_TIME_NS_ADD_UP register provides the value which is add up to SYS_TIME_NS register per clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYS_TIME_NS_ADD_UP																															

Bits	Name	Description	R/W	Default
31:0	SYS_TIME_NS_ADD_UP	Each clock cycle (SYS_TIME_NS_ADD_UP >> 28) will be added to System time. I.e. value 0x10000000 can be used for counting in 10ns steps. Default value corresponds to 1 ns per step.	R/W	0xa0000000

SYS_TIME_S_CMP – System Time Second Compare Register 0x00101110

The SYS_TIME_S_CMP register provides the compare value to SYS_TIME_S.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYS_TIME_S_CMP_VAL																															

Bits	Name	Description	R/W	Default
31:0	SYS_TIME_S_CMP_VAL	Compare value with SYS_TIME_S (seconds). Set SYS_TIME_S_CMP_INT register if SYS_TIME_S_CMP_EN is set.	R/W	0x00

SYS_TIME_S_CMP_EN – System Time Second Compare Enable Register 0x00101114

The SYS_TIME_S_CMP_EN register enables or disables the comparison to SYS_TIME_S.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																															ENABLE

Bits	Name	Description	R/W	Default
31:1	reserved	-	R	0x00
0	ENABLE	Enable compare with SYS_TIME_S (seconds). Automatic set with the value in register SYS_TIME_S _CMP, automatic reset after interrupt.	R/W	0

SYS_TIME_S_CMP_INT – System Time Second Compare Interrupt Register 0x00101118

The SYS_TIME_S_CMP_INT register signals an interrupt because SYS_TIME_S is equal SYS_TIME_S_CMP.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																															COMPARE_IRQ

Bits	Name	Description	R/W	Default
31:1	reserved	-	R	0x00
0	COMPARE_IRQ	System time compare interrupt. Set if SYS_TIME_S is equal SYS_TIME_S _CMP. To clear the IRQ you have to write a ‘1’ at this bit position.	R/W	0

6.7 RTC – Real Time Clock

The Real-time Clock is clocked from a 32.768-KHz crystal and can be powered separately for battery back up application. Therefore the value of the Real-time clock is valid after power up.

The RTC provides a constant frequency output with a programmable alarm register. This alarm register can be used to generate an interrupt to the ARM.

When no battery buffered RTC is used the power supply must be directly connected to the netX power supplies.

Moreover the internal backup RAM is powered over the RTC power supply pins. It is only accessable when the power supply is connected and the RTC power good signal has a high level.

The following table is a summary of registers of the Real-time Clock.

ARM Address	Register Name	Short Description
0x00101200	RTC_1HZ_CNTR	RTC 1Hz COUNTER
0x00101204	RTC_32KHZ_CNTR_CUR_VAL	RTC 32KHz Counter Current Value
0x00101208	RTC_32KHZ_CNTR_LAT_VAL	RTC 32KHz Counter Latched Value
0x0010120c	RTC_IRQ2ISOLATE_CYC	Number of Clock_32kHz Cycles
0x00101210	RTC_INT_MSK	Isolated Area Interrupt Mask Register
0x00101214	RTC_INT_STAT	Isolated Area Interrupt Status Register
0x00101218	RTC_ISOLATED	Isolated Area is Currently Isolated

RTC_1HZ_CNTR – RTC 1Hz COUNTER 0x00101200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTC_1HZ_COUNTER																															

Bits	Name	Description	R/W	Default
31:0	RTC_1HZ_COUNTER	1Hz counter value. When reading the RTC_32KHZ_CNTR_CUR_VAL, it is stored into the RTC_32KHZ_CNTR_LAT_VAL register.	R/W	0x00

RTC_32KHZ_CNTR_CUR_VAL – RTC 32KHz Counter Current Value 0x00101204

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTC_32KHZ_CNTR_CUR_VAL																															

Bits	Name	Description	R/W	Default
31:0	RTC_32KHZ_CUR_VAL	32KHz counter value	R	0x00

RTC_32KHz_CNTR_LAT_VAL – RTC 32KHz Counter Latched Value 0x00101208

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTC_32KHZ_CNTR_LAT_VAL																															

Bits	Name	Description	R/W	Default
31:0	RTC_32KHZ_CNTR_LAT_VAL	32KHz counter value latched on RTC_1HZ_CNTR register read	R	0x00

RTC_IRQ2ISOLATE_CYC – RTC Number of Clock_32kHz Cycles 0x0010120c

Numbers of RTC clock cycles between power fail interrupt (power good goes low) and activation of RTC and buffered SRAM isolation. During this time the ARM-software must write all data to buffered SRAM and then has do stop all write accesses to the RTC and buffered SRAM.
Because this value is stored inside the isolated area (with no reset on power up), the default value has to be programmed during startup.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								IRQ2ISO							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	IRQ2ISO	number of 32 kHz clock cycles before isolation the RTC and buffered SRAM module	W	0x00

RTC_INT_MSK – RTC Isolated Area Interrupt Mask Register 0x00101210

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																													EN_CLK32_FAULT2	EN_CLK32_FAULT1	EN_POWER_DOWN

Bits	Name	Description	R/W	Default
31:3	reserved	-	R	0x00
2	EN_CLK32_FAULT2	enable clk32_fault2 interrupt	R/W	0
1	EN_CLK32_FAULT1	enable clk32_fault1 interrupt	R/W	0
0	EN_POWER_DOWN	enable power_good interrupt	R/W	0

RTC_INT_STAT – RTC Isolated Area Interrupt Status Register 0x00101214

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																													CLK32_FAULT2	CLK32_FAULT1	POWER_DOWN

Bits	Name	Description	R/W	Default
31:3	reserved	-	R	0x00
2	CLK32_FAULT2	If clk32 is not running, this bit will be set all 131072 (128k) system clock cycles (100MHz: 1.31ms) After resetting this bit, the next interrupt will be generated not before 128k system clk cycles passed.	R/W	0
1	CLK32_FAULT1	If clk32 is not running, this bit will be set all 4096 system clock cycles (100MHz: 40,96us) After resetting this bit, the next interrupt will be generated not before 4096 system clk cycles passed.	R/W	0
0	POWER_DOWN	power_good signal from external became 0	R/W	0

RTC_ISOLATED – RTC Isolated Area Status 0x00101218

When the isolated area is currently isolated all write accesses to registers of RTC or BACKUP_RAM have no effect, read accesses to registers of RTC will deliver 0, to BACKUP_RAM an abort.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																															NOT_ISOLATED

Bits	Name	Description	R/W	Default
31:1	reserved	-	R	0x00
0	NOT_ISOLATED	0 : Isolated Area is currently isolated 1 : Isolated Area is currently not isolated This bit will be set asynchronous by powergood (powergood must be stable at power on reset). It will be set to zero RTC_IRQ2ISOLATE_CYC 32khz clocks plus 2 (for synchronisation) after powergood pin goes down. If this bit is 0, any access to backup SRAM will generate an abort (HRESP=1). The ARM-CPU will switch to instruction or data abort mode!!! No valid access to backup SRAM and RTC-registers can be done in isolation mode!!!	R	0

6.8 LCD – LCD-Display Controller

When the LCD interface is to be used, Bits 0 and Bit 1 in the 'IO_CFG – IO Configuration Register' have to be set (Bit 0 is always required, Bit 1 may be left cleared, depending on the connected display.

The LCD Controller provides all the necessary control signals to interface directly to a variety of color and monochrome LCD panels:

- Active matrix TFT panels with up to 18-bit bus interface
- Single-panel monochrome STN panels, 4-bit and 8-bit bus interface
- Dual-panel monochrome STN panels, 4-bit and 8-bit bus interface per panel
- Single-panel color STN panels, 8-bit bus interface
- Dual-panel color STN panels, 8-bit bus interface per panel

The LCD Controller can be programmed to support a wide range of panel resolutions such as:

- 320 x 200
- 320 x 240
- 640 x 200
- 640 x 240
- 640 x 480

It supports 15 grey-level mono, 3375 color STN and 32K color TFT with a programmable timing for the different display panels.

The following table shows which data bits are used for the different types of display.

	STN color single panel	**STN color dual panel**	**STN 4-Bit mono single panel**	**STN 4-Bit mono dual panel**	**STN 8-Bit mono dual panel**	**STN 8-Bit mono dual panel**	**TFT 18-Bit color**
LCD_D0	PD[7]	UPD[7]	PD[3]	UPD[3]	PD[7]	UPD[7]	Intensity Bit
LCD_D1	PD[6]	UPD[6]	PD[2]	UPD[2]	PD[6]	UPD[6]	RED[0]
LCD_D2	PD[5]	UPD[5]	PD[1]	UPD[1]	PD[5]	UPD[5]	RED[1]
LCD_D3	PD[4]	UPD[4]	PD[0]	UPD[0]	PD[4]	UPD[4]	RED[2]
LCD_D4	PD[3]	UPD[3]			PD[3]	UPD[3]	RED[3]
LCD_D5	PD[2]	UPD[2]			PD[2]	UPD[2]	RED[4]
LCD_D6	PD[1]	UPD[1]			PD[1]	UPD[1]	Intensity Bit
LCD_D7	PD[0]	UPD[0]			PD[0]	UPD[0]	GREEN[0]
LCD_D8		LPD[7]		LPD[3]		LPD[7]	GREEN[1]
LCD_D9		LPD[6]		LPD[2]		LPD[6]	GREEN[2]
LCD_D10		LPD[5]		LPD[1]		LPD[5]	GREEN[3]
LCD_D11		LPD[4]		LPD[0]		LPD[4]	GREEN[4]
LCD_D12		LPD[3]				LPD[3]	Intensity Bit
LCD_D13		LPD[2]				LPD[2]	BLUE[0]
LCD_D14		LPD[1]				LPD[1]	BLUE[1]
LCD_D15		LPD[0]				LPD[0]	BLUE[2]
LCD_D16							BLUE[3]
LCD_D17							BLUE[4]

PD: Panel data
UPD: Upper panel data
LPD: Lower panel data

The data bit PD[i] corresponds to the pixel position. Means PD[0] is the leftmost pixel on the panel and PD[7] is the rightmost pixel within the 8-Bit data.

Here is also a summary of all the registers about LCD.

ARM Address	Register Name	Short Description
0x00104000	LCD_TIMING0	LCD Horizontal Axis Panel Control Register
0x00104004	LCD_TIMING1	LCD Vertical Axis Panel Control Register
0x00104008	LCD_TIMING2	LCD Clock and Signal Polarity Control Register
0x0010400c	LCD_TIMING3	LCD Line End Control Register
0x00104010	LCD_UP_PAN_BASE	LCD Upper Panel Frame Base Address Registers
0x00104014	LCD_LO_PAN_BASE	LCD Lower Panel Frame Base Address Registers
0x00104018	LCD_INT_MSK_SET_CLR	LCD Interrupt Mask Set/Clear Register
0x0010401c	LCD_CTRL	LCD Control Register
0x00104020	LCD_RAW_INT_STAT	LCD Raw Interrupt Status Register
0x00104024	LCD_MSK_INT_STAT	LCD Masked Interrupt Status Register
0x00104028	LCD_INT_CLR	LCD Interrupt Clear Register
0x0010402c	LCD_UP_PAN_CUR	LCD Upper Panel Current Address Value Registers
0x00104030	LCD_LO_PAN_CUR	LCD Lower Panel Current Address Value Registers
0x00104200	LCD_PALETTE_BASE	LCD Coulour Palette Register Base
0x001043fc	LCD_PALETTE_END	LCD Coulour Palette Registers End
0x00104fe0	CLCD_PERIPH_ID0	LCD Peripheral Identification Register 0
0x00104fe4	CLCD_PERIPH_ID1	LCD Peripheral Identification Register 1
0x00104fe8	CLCD_PERIPH_ID2	LCD Peripheral Identification Register 2
0x00104fec	CLCD_PERIPH_ID3	LCD Peripheral Identification Register 3
0x00104ff0	CLCD_PRIME_CELL_ID0	LCD Prime Cell Identification Register 0
0x00104ff4	CLCD_PRIME_CELL_ID1	LCD Prime Cell Identification Register 1
0x00104ff8	CLCD_PRIME_CELL_ID2	LCD Prime Cell Identification Register 2
0x00104ffc	CLCD_PRIME_CELL_ID3	LCD Prime Cell Identification Register 3

LCD_TIMING0 – LCD Horizontal Axis Panel Control Register 0x00104000

This register controls the Horizontal Sync width, Horizontal Front porch and Horizontal Back porch period. The data path latency forces some restrictions on the usable minimum values for horizontal porch width in STN mode. The minimum values are HSW = 2 and HBP = 2.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HBP								HFP								HSW								PPL						reserved	

Bits	Name	Description	R/W	Default
31:24	HBP	Horizontal back porch, is the number of CLCP periods between the falling edge of CLLP and the start of active data. Program with value minus 1. The 8-bit HBP field specifies the number of pixel clock periods inserted at the beginning of each line or row of pixels. After the line clock for the previous line has been deasserted, the value in HBP counts the number of pixel clocks to wait before starting the next display line. HBP can generate a delay of 1-256 pixel clock cycles.	R/W	0x00
23:16	HFP	Horizontal front porch, is the number of CLCP periods between the end of active data and the rising edge of CLLP. Program with value minus 1. The 8-bit HFP field sets the number of pixel clock intervals at the end of each line or row of pixels, before the LCD line clock is pulsed. When a complete line of pixels is transmitted to the LCD driver, the value in HFP counts the number of pixel clocks to wait before asserting the line clock. HFP can generate a period of 1-256 pixel clock cycles.	R/W	0x00
15:8	HSW	Horizontal synchronization pulse width, is the width of the CLLP signal in CLCP periods. Program with value minus 1. The 8-bit HSW field specifies the pulse width of the line clock in passive mode, or the horizontal synchronization pulse in active mode.	R/W	0x00
7:2	PPL	Pixels-per-line. Actual pixels-per-line = 16 * (PPL + 1). The PPL bit field specifies the number of pixels in each line or row of the screen. PPL is a 6-bit value that represents between 16 and 1024 PPL. PPL controls how much data is read from the DMA input buffers through to the gray scaler.	R/W	0x00
1:0	reserved	-	R	0x00

Note!
There are some horizontal timing restrictions according to the display type. DMA requests new data at the start of a horizontal display line. Some time must be allowed for the DMA transfer and for the data to propagate down the FIFO path in the LCD interface. The data path latency forces some restrictions on the usable minimum values for horizontal porch width. The following table shows the limitations.

Display Mode	HSW	HPB	HFP	Panel Clock Divisor (PCD)
STN	2 … 255	2 … 255		
Single Panel	3 … 255	5 … 255	5 … 255	1 (100 MHZ/3).
Dual Panel	3 … 255 6 … 255	5 … 255 10 … 255	5 … 255 5 … 255	5 (100 MHZ/7) 4 (100 MHZ/6)

LCD_TIMING1 – LCD Vertical Axis Panel Control Register 0x00104004

LCD_TIMING1 – LCD Vertical Axis Panel Control Register 0x00104004

This register controls the Lines per screen, Vertical Sync width, Vertical Front porch and Vertical Back porch period.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBP								VFP								VSW						LPP									

Bits	Name	Description	R/W	Default
31:24	VBP	Vertical back porch is the number of inactive lines at the start of a frame, after vertical synchronization period. Program to 0 on passive displays or reduced contrast results. The 8-bit VBP field specifies the number of line clocks inserted at the beginning of each frame. The VBP count starts just after the vertical synchronization signal for the previous frame has been negated for active mode, or the extra line clocks have been inserted as specified by the VSW bit field in passive mode. After this has occurred, the count value in VBP sets the number of line clock periods inserted before the next frame. VBP generates from 0–255 extra line clock cycles.	R/W	0x00
23:16	VFP	Vertical front porch is the number of inactive lines at the end of frame, before vertical synchronization period. Program to 0 on passive displays or reduced contrast results. The 8-bit VFP field specifies the number of line clocks to insert at the end of each frame. When a complete frame of pixels is transmitted to the LCD display, the value in VFP is used to count the number of line clock periods to wait. After the count has elapsed the vertical synchronization signal, CLFP, is asserted in active mode, or extra line clocks are inserted as specified by the VSW bit-field in passive mode. VFP generates from 0–255 line clock cycles.	R/W	0x00
15:10	VSW	Vertical synchronization pulse width is the number of horizontal synchronization lines. Must be small (for example, program to zero) for passive STN LCDs. Program to the number of lines required minus one. The higher the value the worse the contrast on STN LCDs. The 6-bit VSW field specifies the pulse width of the vertical synchronization pulse. The register is programmed with the number of line clocks in VSync minus one. Number of horizontal synchronization lines. Must be small (for example, program to 0) for passive STN LCDs. Program to the number of lines required minus 1. The higher the value the worse the contrast on STN LCDs.	R/W	0x00
9:0	LPP	Lines per panel is the number of active lines per screen. Program to number of lines required minus 1. The LPP field specifies the total number of lines or rows on the LCD panel being controlled. LPP is a 10-bit value that allows 1-1024 lines. The register is programmed with the number of lines per LCD panel minus 1. For dual panel displays this register is programmed with the number of lines on each of the upper and lower panels.	R/W	0x00

LCD_TIMING2 – LCD Clock and Signal Polarity Control Register 0x00104008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCD_HI					BCD	CPL										reserved	IEO	IPC	IHS	IVS	ACB					CLKSEL	PCD_LO				

Bits	Name	Description	R/W	Default
31:27	PCD_HI	Upper five bits of Panel Clock Divisor. The ten-bit PCD field, comprising PCD_HI and PCD_LO (bits [4:0]), is used to derive the LCD panel clock frequency CLCP from the 100 MHZ frequency: CLCP = 100 MHZ/(PCD+2). For mono STN displays with a four or eight-bit interface, the panel clock is a factor of four and eight down on the actual individual pixel clock rate. For color STN displays, 2 2 /3 pixels are output per CLCP cycle, therefore the panel clock is 0.375 times. For TFT displays the pixel clock divider can be bypassed by setting the LCD_Timing2[26] BCD bit.	R/W	0x00
26	BCD	Bypass pixel clock divider. Setting this to 1 bypasses the pixel clock divider logic. This is mainly used for TFT displays.	R/W	0x00
25:16	CPL	Clocks per line. This field specifies the number of actual CLCP clocks to the LCD panel on each line. This is the number of pixel per line divided by 1 for TFT, 4 or 8 for mono passive, or 2 2 /3 for color passive, minus one. This must be correctly programmed in addition to PPL for the LCD controller to work correctly.	R/W	0x00
15	reserved	-	R	0
14	IEO	Invert output enable: 0 : CLAC output pin is active HIGH in TFT mode 1 : CLAC output pin is active LOW in TFT mode. The Invert Output Enable (IOE) bit is used to select the active polarity of the output enable signal in TFT mode. In this mode, the CLAC pin is used as an enable that indicates to the LCD panel when valid display data is available. In active display mode, data is driven onto the LCD data lines at the programmed edge of CLCP when CLAC is in its active state.	R/W	0x00
13	IPC	Invert panel clock: 0 : Data is driven on the LCDs data lines on the rising-edge of CLCP 1 : Data is driven on the LCDs data lines on the falling-edge of CLCP. The IPC bit is used to select the edge of the panel clock on which pixel data is driven out onto the LCD data lines.	R/W	0x00
12	IHS	Invert horizontal synchronization: 0 : CLLP pin is active HIGH and inactive LOW 1 : CLLP pin is active LOW and inactive HIGH. The Invert HSync (IHS) bit is used to invert the polarity of the CLLP signal.	R/W	0x00
11	IVS	Invert vertical synchronization: 0 : CLFP pin is active HIGH and inactive LOW 1 : CLFP pin is active LOW and inactive HIGH. The Invert VSync (IVS) bit is used to invert the polarity of the CLFP signal.	R/W	0x00

10:6	ACB	AC bias pin frequency. The AC bias pin frequency is only applicable to STN displays, which require the pixel voltage polarity to be periodically reversed to prevent damage due to DC charge accumulation. Program this field with the required value minus 1 to apply the number of line clocks between each toggle of the AC bias pin, CLAC. This field has no effect if the CLCDC is operating in TFT mode when the CLAC pin is used as a data enable signal.	R/W	0x00
5	CLKSEL	This bit drives the CLCD_CLKSEL signal which is used as the select signal for the external LCD clock multiplexer.	R/W	0x00
4:0	PCD_LO	Lower five bits of Panel Clock Divisor. The ten-bit PCD field, comprising PCD_HI (bits [31:27]) and PCD_LO, is used to derive the LCD panel clock frequency CLCP from the 100 MHZ frequency, CLCP = 100 MHZ/(PCD+2). For mono STN displays with a four or eight-bit interface, the panel clock is a factor of four and eight down on the actual individual pixel clock rate. For color STN displays, 2 2 /3 pixels are output per CLCP cycle, so the panel clock is 0.375 times. You can bypass the pixel clock divider for TFT displays by setting the LCD_Timing2[26] BCD bit.	R/W	0x00

Note:
The data path latency forces restrictions on the usable minimum values for the panel clock divider.

Display Mode	**Panel Clock Divisor (PCD)**
Single Panel Color	1 (100 MHZ/3).
Dual Panel Color	4 (100 MHZ/6)
Single Panel Color 4 bit	2 (100 MHZ/4)
Dual Panel Color 4 bit	6 (100 MHZ/8)
Single Panel Color 8 bit	6 (100 MHZ/8)
Dual Panel Color 8 bit	14 (100 MHZ/16)

LCD_TIMING3 – LCD Line End Control Register 0x0010400c

LCD_TIMING3 is a read/write register that controls the enabling of line-end signal CLLE. When enabled, a positive pulse, four 100 MHZ periods wide, is output on CLLE after a programmed delay set by the LED bits. If the line-end signal is disabled then it is held permanently LOW.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved															LEE	reserved									LED						

Bits	Name	Description	R/W	Default
31:17	reserved	-	-	-
16	LEE	LCD Line end enable: 0 = CLLE disabled (held LOW) 1 = CLLE signal active.	R/W	0x00
15:7	reserved	-	R	0x00
6:0	LED	Line-end signal delay from the rising-edge of the last panel clock, CLCP. Program with number of 100 MHZ clock periods minus 1.	R/W	0x00

LCD_UP_PAN_BASE – LCD Upper Panel Frame Base Address Registers 0x00104010

LCD_UP_PAN_BASE and LCD_LO_PAN_BASE are the colour LCD DMA Frame Address Registers. They are read/write registers used to program the base address of the frame buffer. The LCD_LO_PAN_BASE is used for the lower panel of dual panel STN displays. The LCD_UP_PAN_BASE is used for:
• TFT displays
• single panel STN displays
• the upper panel of dual panel STN displays.

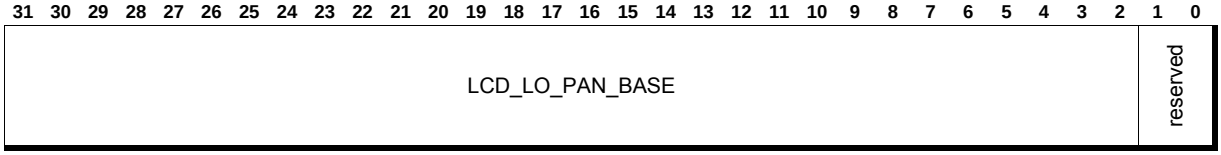
You must initialize LCD_UP_PAN_BASE (and LCD_LO_PAN_BASE for dual panels) before enabling the CLCDC. You can change the value mid-frame to enable double-buffered video displays to be created. These registers are copied to the corresponding current registers at each LCD vertical synchronization. This event causes the LNBU bit and an optional interrupt to be generated. You can use the interrupt to reprogram the base address when generating double-buffered video.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCD_UP_PAN_BASE																														reserved	

Bits	Name	Description	R/W	Default
31:2	LCD_UP_PAN_BASE	LCD upper panel base address. This is the start address of the upper panel frame data in memory and is word aligned.	R/W	0x00
1:0	reserved	-	R	0x00

LCD_LO_PAN_BASE – LCD Lower Panel Frame Base Address Registers 0x00104014

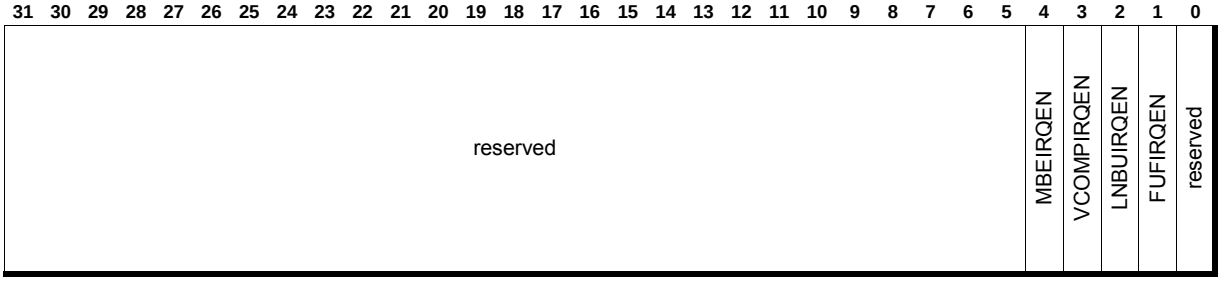
This is a register to program the lower base address of the frame buffer. The base address is word aligned, hence only the bits 31:2 are used. LCD_LO_PAN_BASE is used for the lower panel of dual panel STN displays.



Bits	Name	Description	R/W	Default
31:2	LCD_LO_PAN_BASE	LCD lower panel base address. This is the start address of the lower panel frame data in memory and is word aligned.	R/W	0x00
1:0	reserved	-	R	00

LCD_INT_MSK_SET_CLR – LCD Interrupt Mask Set/Clear Register 0x00104018

LCD_INT_MSK_SET_CLR is the Interrupt Mask Set/Clear Register. Setting bits in this register enables the corresponding raw interrupt LCD_RAW_INT_STAT bit values to be passed to the LCD_MSK_INT_STAT Register.



Bits	Name	Description	R/W	Default
31:5	reserved	-	R	0x00
4	MBEIRQEN	AHB master error interrupt request enable	R/W	0
3	VCOMPIRQEN	Vertical compare interrupt request enable	R/W	0
2	LNBUIRQEN	Next base update interrupt request enable	R/W	0
1	FUFIRQREN	FIFO underflow interrupt request enable	R/W	0
0	reserved	-	R	0

LCD_CTRL – LCD Control Register 0x0010401c

This register controls the mode in which the LCD controller operates.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved															WATERMARK	reserved		LCDVCOMP		LCDPWR	BEPO	BEBO	BGR	LCDDUAL	LCDMONO8	LCDTFT	LCDBW	LCDBPP			LCDEN

Bits	Name	Description	R/W	Default
31:17	reserved	-	R	0x0
16	WATERMARK	LCD DMA FIFO Watermark level: 0 : FIFO Watermark level will be set to 4-words 1 : FIFO Watermark level will be set to 8-words	R/W	0
15:14	reserved	-	R	0x00
13:12	LCDVCOMP	Interrupt generation: 00 : start of vertical synchronization 01 : start of back porch 10 : start of active video 11 : start of front porch.	R/W	0x00
11	LCDPWR	LCD power enable: 0 : power not gated through to LCD panel and CLD[23:0] signals disabled, (held LOW) 1 : power gated through to LCD panel and CLD[23:0] signals enabled, (active)	R/W	0
10	BEPO	Big endian pixel ordering within a byte 0 : Little endian pixel ordering within a byte 1 : Big endian pixel ordering within a byte The BEPO bit selects between little and big-endian pixel packing for 1, 2 and 4 bpp display modes. It has no effect on 8 or 16 bpp pixel formats.	R/W	0
9	BEBO	Big-endian byte order: 0 : little-endian byte order 1 : big-endian byte order.	R/W	0
8	BGR	RGB of BGR format selection: 0 : RGB normal output 1 : BGR red and blue swapped.	R/W	0
7	LCDDUAL	LCD interface is dual panel STN: 0 : single panel LCD is in use 1 : dual panel LCD is in use.	R/W	0
6	LCDMONO8	Monochrome LCD buswidth selection. LCDMONO8 has no meaning in other modes and must be programmed to 0. 0 : mono LCD uses 4-bit interface 1 : mono LCD uses 8-bit interface.	R/W	0
5	LCDTFT	LCD is TFT: 0 : LCD is an STN display, use gray scaler 1 : LCD is TFT, do not use gray scaler.	R/W	0
4	LCDBW	STN LCD is monochrome (black and white): 0 : STN LCD is colour 1 : STN LCD is monochrome. This bit has no meaning in TFT mode.	R/W	0

3:1	LCDBPP	LCD bits per pixel: 000 : 1 bpp 001 : 2 bpp 010 : 4 bpp 011 : 8 bpp 100 : 16 bpp 101 : 24 bpp (TFT panel only) : not supported here 110 : reserved 111 : reserved.	R/W	0x00
0	LCDEN	LCD controller enable: 0 : CLLP, CLCP, CLFP, CLAC, and CLLE disabled (held LOW) 1 : CLLP, CLCP, CLFP, CLAC, and CLLE enabled (active). Refer to LCD powering up and powering down sequence support on page 1-5 for details on LCD power sequencing.	R/W	0

Interrupts

There are four interrupts generated by the LCD controller. The following are individual mask able active HIGH interrupts:
• MBEIRQ AHB master error interrupt
• VCOMPIRQ Vertical compare interrupt
• LNBUIRQ LCD next base address update
• FUFIRQ FIFO underflow

The outputs are also output as a combined single interrupt INTR. Each of the four individual mask able interrupts is enabled or disabled by changing the mask bits in the LCD_INT_MSK_SET_CLR Register. Provision of individual outputs as well as a combined interrupt output enables you to use either:
• a global interrupt service routine
• modular device drivers to handle interrupts.
The status of the individual interrupt sources can be read from the LCD_RAW_INT_STAT Register.

MBEIRQ:
The master bus error interrupt is asserted when an ERROR response is received by the master interface during a transaction with a slave. When such an error is encountered, the master interface enters an error state and remains in this state until clearance of the error has been signalled to it. On completion of the respective interrupt service routine, the master bus error interrupt can be cleared by writing a 1 to the MBECLR bit within the LCD_INT_CLR Register. This action releases the master interface from its ERROR state to the start of FRAME state, enabling a fresh frame of data display to be initiated.

VCOMPIRQ:
The vertical compare interrupt is asserted when one of four vertical display regions, selected using the Control Register, is reached. The interrupt can be made to occur at the start of:
• vertical synchronization
• back porch
• active video
• front porch.
You can clear the interrupt by writing a 1 to the VCOMPCLR bit in the LCD_INT_CLR Register.

LNBUIRQ:
The LCD next base address update interrupt is asserted when either the LCD_UP_PAN_BASE or the LCD_LO_PAN_BASE values are transferred to the LCD_UP_PAN_CUR or LCD_LO_PAN_CUR

incrementers respectively. This signals to the system that it is safe to update the LCD_UP_PAN_BASE or the LCD_LO_PAN_BASE Registers with new frame base addresses if required.
You can clear the interrupt by writing a 1 to the LNBUCLR bit in the LCD_INT_CLR Register.

FUFIRQ:
The FIFO underflow interrupt is asserted when internal data is requested from an empty DMA FIFO. Internally, individual upper and lower panel DMA FIFO underflow interrupt signals are generated and FUFIRQ is the single combined version of these.
You can clear the interrupt by writing a 1 to the FUFCLR bit in the LCD_INT_CLR Register.

LCD_RAW_INT_STAT – LCD Raw Interrupt Status Register 0x00104020

A read of this register returns five bits that may generate interrupts when they are set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																											MBESTAT	VCOMPSTAT	LNBUSTAT	FUFSTAT	reserved

Bits	Name	Description	R/W	Default
31:5	reserved	-	R	0x00
4	MBESTAT	AHB Master bus error status, set when the AHB Master encounters a bus error response from a slave.	R	0
3	VCOMPSTAT	Vertical compare, set when one of the four vertical regions, selected through the LCD Control Register, is reached.	R	0
2	LNBUSTAT	LCD next address base update, mode dependent, set when the Current Base Address Registers have been successfully updated by the next Address Registers. Signifies that a new next address can be loaded if double buffering is in use.	R	0
1	FUFSTAT	FIFO underflow, set when either the upper or lower DMA FIFOs have been read accessed when empty causing an underflow condition to occur.	R	0
0	reserved	-	R	0

LCD_MSK_INT_STAT – LCD Masked Interrupt Status Register 0x00104024

LCD_MSK_INT_STAT is a read-only register. It is a bit-by-bit logical AND of the LCD_RAW_INT_STAT Register and the LCD_INT_MSK_SET_CLR Register. A logical OR of all interrupts is provided to the system interrupt controller.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																											MPEIRQ	VCOMPIRQ	LNBUIRQ	FUFIRQ	reserved

Bits	Name	Description	R/W	Default
31:5	reserved	-	R	0x00
4	MBEIRQ	AHB master error interrupt request status bit	R	0
3	VCOMPIRQ	Vertical compare interrupt request status bit	R	0
2	LNBUIRQ	LCD next base address update interrupt request status bit	R	0
1	FUFIRQ	FIFO underflow interrupt request status bit	R	0
0	reserved	-	R	0

LCD_INT_CLR – LCD Interrupt Clear Register 0x00104028

The LCD_INT_CLR is a write-only register. Writing a logic 1 to the relevant bit clears the corresponding interrupt.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																											MBECLR	VCOMPCLR	LNBUCLR	FUFCLR	reserved

Bits	Name	Description	R/W	Default
31:5	reserved	-	W	0x00
4	MBECLR	Clear AHB Master error interrupt	W	0
3	VCOMPCLR	Clear vertical compare interrupt	W	0
2	LNBUCLR	Clear LCD next base address update interrupt	W	0
1	FUFCLR	Clear FIFO underflow interrupt	W	0
0	reserved	-	W	0

LCD_UP_PAN_CUR – LCD Upper Panel Current Address Value Registers 0x0010402c

LCD_UP_PAN_CUR and LCD_LO_PAN_CUR are read-only registers that contain an approximate value of the upper and lower panel data DMA addresses when read. The registers can change at any time and therefore can only be used as a mechanism for coarse delay.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCD_UP_CUR																															

Bits	Name	Description	R/W	Default
31:0	LCD_UP_CUR	Contains the approximate current upper panel data DMA address	R	0x00

LCD_LO_PAN_CUR – LCD Lower Panel Current Address Value Registers 0x00104030

This register contains the current DMA address for display data read of the lower panel of dual STN displays.
This register's value may change at any moment, and is not normally read from, but can be read to determine the approximate position within the buffer, or for test.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCD_LO_PAN_CUR																															

Bits	Name	Description	R/W	Default
31:0	LCD_LO_PAN_CUR	Contains the approximate current lower panel data DMA address	R	0x00

LCD_PALETTE_BASE-LCD_PALETTE_END – Colour Palette Registers 0x00104200-0x001043fc

The LCD_PALETTE Register contains 256 palette entries organized as 128 locations of two entries per dword. Only TFT displays use all of the palette entry bits.
Each dword location contains two palette entries. This means that 128 dword locations are used for the palette. When configured for little-endian byte ordering, bits [15:0] are the lower numbered palette entry and bits [31:16] are the higher numbered palette entry. When configured to big-endian byte ordering this is reversed because bits [31:16] are the low numbered palette entry and bits [15:0] are the high numbered entry.

Each dword of the 128 dwords has the following structure:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCD_PALETTE_BASE																															

Bits	Name	Description	R/W	Default
31	-	I: Intensity (see bit 15)	R/W	0
30:26	-	B[4:0]: Blue palette data	R/W	0x00
25:21	-	G[4:0]: Green palette data	R/W	0x00
20:16	-	R[4:0]: Red palette data	R/W	0x00
15	-	I: Intensity bit. Can be used as the LSB of the R, G, and B inputs to a 6:6:6 TFT display, doubling the number of colors to 64K, where each color has two different intensities.	R/W	0
14:10	-	B[4:0]: Blue palette data	R/W	0x00
9:5	-	G[4:0]: Green palette data	R/W	0x00
4:0	-	R[4:0]: Red palette data For STN displays, only the four MSBs (bits [4:1]) are used. For monochrome displays only the red palette data is used. All of the Palette Registers have the same bit fields.	R/W	0x00

CLCD_PERIPH_ID0 – LCD Peripheral Identification Register 0 0x00104fe0

The CLCD_PERIPH_ID0 Register is hard-coded and the fields in the register determine the reset value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								PartNumber0							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	PartNumber0	These bits read back as 0x10	R	0x10

CLCD_PERIPH_ID1 – LCD Peripheral Identification Register 1 0x00104fe4

The CLCD_PERIPH_ID1 Register is hard-coded and the fields in the register determine the reset value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								Designer0				PartNumber1			

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:4	Designer0	These bits read back as 0x1	R	0x1
3:0	PartNumber1	These bits read back as 0x1	R	0x1

CLCD_PERIPH_ID2 – LCD Peripheral Identification Register 2 0x00104fe8

The CLCD_PERIPH_ID2 Register is hard-coded and the fields in the register determine the reset value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								Revision				Designer1			

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:4	Revision	These bits read back as 0x0	R	0x0
3:0	Designer1	These bits read back as 0x4	R	0x4

CLCD_PERIPH_ID3 – LCD Peripheral Identification Register 3 0x00104fec

The CLCD_PERIPH_ID3 Register is hard-coded and the fields in the register determine the reset value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								Configuration							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	Configuration	These bits read back as 0x00	R	0x00

CLCD_PRIME_CELL_ID0 – LCD Prime Cell Identification Register 0 0x00104ff0

The CLCD_PRIME_CELL_ID0 Register is hard-coded and the fields in the register determine the reset value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								PCELLIDID0							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	PCELLIDID0	These bits read back as 0x0D	R	0x0D

CLCD_PRIME_CELL_ID1 – LCD Prime Cell Identification Register 1 0x00104ff4

The CLCD_PRIME_CELL_ID1 Register is hard-coded and the fields in the register determine the reset value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								PCELLIDID1							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	PCELLIDID1	These bits read back as 0xF0	R	0xF0

CLCD_PRIME_CELL_ID2 – LCD Prime Cell Identification Register 2 0x00104ff8

The CLCD_PRIME_CELL_ID2 Register is hard-coded and the fields in the register determine the reset value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								PCELLIDID2							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	PCELLIDID2	These bits read back as 0x05	R	0x05

CLCD_PRIME_CELL_ID3 – LCD Prime Cell Identification Register 3 0x00104ffc

The CLCD_PRIME_CELL_ID3 Register is hard-coded and the fields in the register determine the reset value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								PCELLIDID3							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	PCELLIDID3	These bits read back as 0xB1	R	0xB1

6.9 USB – Serial USB-Interface

The integrated USB V 1.1 interface is fully compliant with the USB specification. It supports both full and low speed transfers and can act as a host or a device. The USB unit includes an integrated transceiver and provides eight pipes. Their direction, transfer type and FIFO size can be configured at run-time. All low-level USB operations are realized in hardware. The software only has to manage the enumeration process and data transfer from and to the FIFOs.

The following table is a summary of registers of USB.

ARM Address	Register Name	Short Description
0x00120000	USB_ID	USB ID Register
0x00120004	USB_CTRL	USB Control Register
0x00120008	USB_FRM_TMR	USB Frame Timer Register
0x0012000c	USB_MAIN_EV	USB Main Event Register
0x00120010	USB_MAIN_EV_MSK	USB Main Event Mask Register
0x00120014	USB_PIPE_EV	USB Pipe Event Register
0x00120018	USB_PIPE_EV_MSK	USB Pipe Event Mask Register
0x00120024	USB_PIPE_SEL	USB Pipe Select Register
0x0012002c	USB_PORT_STAT	USB Port Status Register
0x00120030	USB_PORT_CTRL	USB Port Control Register
0x00120034	USB_PORT_STAT_CHG_EV	USB Port Status Change Event Register
0x00120038	USB_PORT_STAT_CHG_EV_MSK	USB Port Status Change Event Mask Register
0x00120040	USB_PIPE_CTRL	USB Pipe Control Register
0x00120044	USB_PIPE_CFG	USB Pipe Configuration Register
0x00120048	USB_PIPE_ADDR	USB Pipe Address Register
0x0012004c	USB_PIPE_STAT	USB Pipe Status Register
0x00120050	USB_PIPE_DATA_PTR	USB Pipe Data Pointer Register
0x00120054	USB_PIPE_DATA_TOT	USB Pipe Total Bytes Register
0x00120058	USB_PIPE_ALT_DATA_PTR	USB Pipe Alternative Data Pointer Register
0x0012005c	USB_PIPE_ALT_DATA_TOT	USB Pipe Alternative Data Total Bytes Register
0x00120060	USB_DBG_CTRL	USB Debug Control Register
0x00120064	USB_DBG_PID	USB Debug PID Register
0x00120068	USB_DBG_STAT	USB Debug Status Register
0x0012006c	USB_TEST	USB Test Register
0x00120080	USB_MAIN_CFG	USB Main Configuration Register
0x00120084	USB_MODE_CFG	USB Mode Configuration Register
0x00120088	USB_CORE_CTRL	USB Core Control and Status Register
0x00130000	USB_FIFO	FIFO for USB-Interface

USB_ID – USB ID Register 0x00120000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																			CORE_ID					REV_ID							

Bits	Name	Description	R/W	Default
31:13	reserved	-	R	0x00
12:8	CORE_ID	Core-ID This bitfield shows the unique ID of the core.	R	0x07
7:0	REV_ID	Revision ID These bits show the revision number of the core.	R	0x00

USB_CTRL – USB Control Register 0x00120004

This register is only valid if the core works in host mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																											PSE	ASE	HRS	XSUSP	CSUSP

Bits	Name	Description	R/W	Default
31:5	reserved	-	R	0x00
4	PSE	Periodic Schedule Enable When set to 1 and the Host Run/Stop Bit is set, the core schedules periodic transactions (interrupt and isochronous). If software sets this bit to zero all pending periodic transactions of the current frame will be completed.	R/W	0
3	ASE	Async Schedule Enable When set to 1 and the Host Run/Stop Bit is set, the core schedules asynchronous transactions (bulk and control). If software sets this bit to zero, all pending asynchronous transactions of the current frame will be completed.	R/W	0
2	HRS	Host Run/Stop When set to 1, the core starts framework and transaction scheduling. If software sets this bit to zero, all pending transactions of the current frame will be completed, and the Host Controller Halted. Event Bit is set at the end of the frame, and no more SOFs will be sent. The bit will be cleared by hardware if a serious error was detected.	R/W	0
1	XSUSP	XCVR Suspend This bit controls the xcvr suspend n output signal. If this bit is set to 1, the low-active xcvr suspendm n output will be set to 0 to enable the low-power mode of the connected Tranceiver. Any change at the ser rx_dm or ser_rx_dp signal will force the deactivation of the xcvr_suspend_n output signal. (Returns to 1)	R/W	0
0	CSUSP	Core Suspend This bit controls the core suspend n output signal. If this bit is set to 1, the low-active core suspend n output will be set to 0 to enable the low-power mode of this core. This signal should be connected to an external clock controller. Any change at the signals ser_rx_dm or ser_rx_dp will force the deactivation of the core suspend n output signal. (Returns to 1)	R/W	0

USB_FRM_TMR – USB Frame Timer Register 0x00120008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																			AGSOF	FTLOCK	FRAME_NR										

Bits	Name	Description	R/W	Default
31:13	reserved	-	R	0x00
12	AGSOF	Artificially Generated SOF This field is only valid, when the core works in peripheral mode. This bit is set, if the Frame Event Bit of the Main Event Register was generated artificially in the case of a lost SOF token.	R	0
11	FTLOCK	Frame Timer Locked This field is only valid, when the core works in peripheral mode. It indicates that the internal frame timer is locked. It will be set if more than two consecutive SOF tokens are received, and will be cleared if at least two consecutive SOF tokens are missed.	R	0
10:0	FRAME_NR	Current Frame Number This field shows the current frame number. If the core works in host mode the frame timer is a free running counter that will be incremented every 1 ms. This register cannot be written unless the Host Controller is in the Halted state. If the core works in peripheral mode the frame timer will synchronize to incoming SOF tokens. This field shows the frame number of the current received SOF token.	R/W	0x00

USB_MAIN_EV – USB Main Event Register 0x0012000c

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																										BWERR_EV	HCHA_EV	GPIPE_EV	GPORT_EV	FRM32_EV	FRM_EV

Bits	Name	Description	R/W	Default
31:6	reserved	-	R	0x00
5	BWERR_EV	Bandwidth Error Event This field is only valid, when the core works in host mode. This bit will be set by hardware if no sufficient bandwidth is available to complete all periodic transfers in a frame. It leads to an interrupt, if the Bandwidth Error Event Mask Bit of the Main Event Mask Register is set. Write a 1 in this position to clear this bit.	R/W	0
4	HCHA_EV	Host Controller Halted Event This field is only valid, when the core works in host mode. The bit is set, if the host controller has stopped as a result of setting the Host Run/Stop Bit to the halted state and all pending transactions of a frame are completed. It leads to an interrupt, if the corresponding HC Halted Mask Bit of the Main Event Mask Register was set. Write a 1 in this position to clear this bit.	R/W	0
3	GPIPE_EV	Global Pipe Transfer Event The bit is set if one of the bits of the Pipe Event Register is set. It leads to an interrupt, if the Global Pipe Event Mask Bit of the Main Event Mask Register is set.	R	0
2	GPORT_EV	Global Port Status Change Event The bit is set if one of the bits of the Port Status Change Event Register is set. It leads to an interrupt, if the Global Port Status Change Event Mask Bit of the Main Event Mask Register is set.	R	0
1	FRM32_EV	Frame 32 Event The bit is set after every 32nd SOF token. It leads to an interrupt, if the corresponding Frame 32 Event Mask Bit of the Main Event Mask Register was set. Can be used by software timers. Write a 1 in this position to clear this bit.	R/W	0
0	FRM_EV	Frame Event The bit is set every 1 ms. It leads to an interrupt, if the Frame Event Mask Bit of the Main Event Mask Register was set. Write a 1 in this position to clear this bit.	R/W	0

USB_MAIN_EV_MSK – USB Main Event Mask Register 0x00120010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																										BWERR_EM	HCHA_EM	GPIPE_EM	GPORT_EM	FRM32_EM	FRM_EM

Bits	Name	Description	R/W	Default
31:6	reserved	-	R	0x00
5	BWERR_EM	Bandwidth Error Event Mask Setting this bit will lead to an interrupt if the Bandwidth Error Event Bit is active.	R/W	0
4	HCHA_EM	HC Halted Mask Setting this bit will lead to an interrupt if the Host Controller Halted Event Bit is active.	R/W	0
3	GPIPE_EM	Global Pipe Event Mask Setting this bit will lead to an interrupt if the Global Pipe Transfer Event Bit is active.	R/W	0
2	GPORT_EM	Global Port Status Change Event Mask Setting this bit will lead to an interrupt if the Global Port Status. Change Event Bit is active.	R/W	0
1	FRM32_EM	Frame 32 Event Mask Setting this bit will lead to an interrupt if the Frame 32 Event Bit is active.	R/W	0
0	FRM_EM	Frame Event Mask Setting this bit will lead to an interrupt if the Frame Event Bit is active.	R/W	0

USB_PIPE_EV – USB Pipe Event Register 0x00120014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								PIPE_EV							

Bits	Name	Description	R/W	Default
31:8	reserved	-	-	-
7:0	PIPE_EV	Pipe Event Flag for each of the eight pipes. If the core works in host mode this event is triggered by the core if one of the following conditions apply: The pipe goes inactive and the Active Status Bit of the Pipe Status Register was cleared by hardware because of normal termination or an error event. A data buffer was completed: the Data Buffer Valid Bit or the Alternative Data Buffer Valid Bit was cleared by hardware. If the core works in peripheral mode this event is triggered by the core if a data buffer was completed: the Data Buffer Valid Bit or the Alter-native Data Buffer Valid Bit was cleared by hardware. Write a 1 in this position to clear this bit.	R/W	0x00

USB_PIPE_EV_MSK – USB Pipe Event Mask Register 0x00120018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								PIPE_EM							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	PIPE_EM	Pipe Event Mask Setting one of these bits enables the pipe event interrupt generation for the appropriated pipe.	R/W	0x00

USB_PIPE_SEL – USB Pipe Select Register 0x00120024

The physical register space for the pipe configuration and status register is paged by the 'USB Pipe Select Register'.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																													PIPE_SEL		

Bits	Name	Description	R/W	Default
31:5	reserved	-	R	0x00
4:0	PIPE_SEL	Pipe Select Depending on the number of supported pipes the core is configured to, one of the pipes may be selected using this register for access of the appropriate status and configuration registers. The width of this bitfield depends on the parameter number of pipes and is equal to log2(NOP). This bitfield is coded as follows: 000 : Pipe 0 : 111 : Pipe 7	R/W	0x00

USB_PORT_STAT – USB Port Status Register 0x0012002c

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																						LINESTATE		OCURC	DLS	PCS	CONN_ID	VB_SESS_END	VB_SESS_VLD	VA_SESS_VLD	VBUS_VLD

Bits	Name	Description	R/W	Default
31:10	reserved	-	R	0x00
9:8	LINESTATE	USB Line State This bit shows the current line state: LINESTATE[1:0] = [ser_rx_dp, ser_rx_dm]	R	0x00
7	OCURC	Over Current Condition This bit will be set to one if the core has detected an overcurrent condition, other-wise it is zero. An overcurrent condition is defined as vb_vld is deasserted but vb on is set.	R	0
6	DLS	Device Low Speed This field is only valid if the core works in host mode. This bit indicates, if the connected device is a full or low speed device. The bit is only valid if the Port Connect Status Bit indicates a connected device. 0 : Full speed device connected 1 : Low speed device connected	R	0
5	PCS	Port Connect Status This field is only valid if the core works in host mode. This bit indicates, if a device is connected. 0 : No device attached [label = ”PCS DISCONN”] 1 : Device attached to root port [label = ”PCS CONN”]	R	0
4	CONN_ID	USB Connector ID Value This bit shows the ID value of the USB connector. If set to one, it indicates that the core works as B-Device, with initial peripheral mode. If set to zero, the core will work as A-Device, with initial host mode. The core can change the host/peripheral roles using the Host Negotiation Protocol (HNP). Therefore, the real mode of the core depends on this bit in conjunction with the selected termination. (refer the Termination Select Bit). The core will update this bit, when a falling edge is detected at the ID-Pullup Enable Bit! 0 : A-Device (initial host) [label = ”CONN ID A”] 1 : B-Device (initial peripheral) [label = ”CONN ID B”]	R	0
3	VB_SESS_END	VB Session End Hardware will set this bit to one if the vb_sess_end input is set, indicating VBUS is above VB-SESS-END.	R	0
2	VB_SESS_VLD	VB Session Valid Hardware will set this bit to one if the vb_sess_vld input is set, indicating VBUS is above VB-SESS-VLD.	R	0
1	VA_SESS_VLD	VA Session Valid Hardware will set this bit to one if the va_sess_vld input is set, indicating VBUS is above VA-SESS-VLD.	R	0
0	VBUS_VLD	Vbus Valid Hardware will set this bit to one if the vb_vld input is set, indicating VBUS is above 4.4 V. For OTG Devices this bit is set to one, when Vbus is above VA-VBUS-VLD. When Vbus goes below thisthreshold an overcurrent condition did occur.	R	0

USB_PORT_CTRL – USB Port Control Register 0x00120030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved								P_LEN								reserved			ID_PU	reserved	VBS_ON	DCHRG	TERM_ENA	TERM_SEL	VB_ON	PSUSP	PENA	FPRESU	URESET	PTESTC	

Bits	Name	Description	R/W	Default
31:24	reserved	-	R	0x00
23:16	P_LEN	Pulse Length It defines the pulse length in milliseconds if the core wants to request a session using the data line pulsing or VBUS pulsing method. Setting this bitfield to zero allows software controlled pulse length. If set to a non-zero value, this field will be updated every millisecond by hardware. 0x00 : Software controlled pulse length. Software must reset the VBUS Session Request Control Bit or the Termination Enable Bit manually. 0x01 : 1 ms pulse length. Hardware will reset the VBUS Session Request Control Bit and the Termination Enable Bit automatically after 1 ms. : 0xFF : 255 ms pulse length. Hardware will reset the VBUS Session Request Control Bit and the Termination Enable Bit automatically after 255 ms. If the this bitfield counts down from one to zero, the Pulse End Event Bit of the Main Event Register will be set.	R/W	0x00
15:13	reserved	-	R	0x00
12	ID_PU	ID-Pullup Enable Software can control the ID-pullup output signal (id pullup) using this bit. The core will update the status of the USB Connector ID Value Bit only if this bit becomes inactive.	R/W	0
11	reserved	-	R	0x00
10	VBS_ON	VBUS Session Request Control Software can control the power circuitry (chrg_vbus) using this bit. If set to 1, the vbs_on output will be set.	R/W	0
9	DCHRG	Enable Discharge Circuitry Software can control the discharge circuitry using this bit to accelerate the discharge of the host bus power capacitors. Setting this bit to 1 will lead to setting the dischrg_vbus output.	R/W	0
8	TERM_ENA	Termination Enable Software can enable the data line termination at the OTG port using this bit. If set to 0, the rpu_ena and rpd_ena outputs are tied to zero. Software can set this bit after it has detected the following conditions after power on reset: • The ID value indicates the core must work as B-Device and is initially in peripheral mode and valid bus power was detected • The ID value indicates the core must work as A-Device Software can set this bit to switch the host/peripheral role following the Host Negotiation Protocol. The pull-up resistor can also be switched on if software wants to request a session using data line pulsing (according to Session Request Protocol). This requires that the VBUS Control Bit and the VBUS Session Request Control Bit must indicate the powered off state, and will also lead to setting the dlp_active output.	R/W	0

7	TERM_SEL	Termination Select Software can select the data line termination at the port using this bit: 0 : Host termination. If the Termination Enable Bit is set, this will lead to setting rpd_ena = 1 and rpu_ena = 0. [label = ”TERM SEL HOST”] 1 : Device termination. If the Termination Enable Bit is set, this will lead to setting rpu_ena = 1 and rpd_ena = 0. [label = ”TERM SEL DEV”] Software can set this bit after it has detected the following conditions after power on reset: • The ID value indicates the core must work as B-Device and is initially in peripheral mode • Valid bus power was detected Software can set this bit to switch the host/peripheral role following the Host Negotiation Protocol.	R/W	0
6	VB_ON	VBUS Control Software can control the power circuitry (vb_on output) using this bit. If set to 1, the vb_on output will be set. Note: Software must set this bit to be able to detect a connected device!	R/W	0
5	PSUSP	Port Suspend This field is only valid if the core works in host mode. The bit can be set by software if the device driver wants to set the connected device into suspended state. Setting the bit will block data propagation on the root port. If the bit is set, the core is sensitive to resume detection. The bit must also be set to prepare the core for changing the host/peripheral role. (Host Negotiation Protocol) Setting the Force Resume Bit will automatically clear this bit.	R/W	0
4	PENA	Port Enable This field is only valid if the core works in host mode. The bit will be set automatically by hardware after the software has finished the USB reset. The core will start sending SOF tokens if the connected device is working at full speed, or low speed keepalive strobes if the connected device is working at low speed.	R/W	0
3	FPRESU	Force Resume If the core works in host mode: The bit can be set by software if, the connected device has to finish its suspend mode. The bit will be set by hardware, if a Remote Wakeup condition was detected. (Resume Event Bit is set). In that case, software must finish the resume sequence by setting this bit to zero. The bit will remain set until the resume sequence was completed by hardware by a low speed EOP. (indicated by Resume Complete Event Bit) If the core works in peripheral mode: The bit can be set by software if it wants to wake up the host/hub it is connected (Remote Wakeup). Software is responsible for the timing of the remote wakeup resume, and must set the bit for at least 1 ms, but no more than 15 ms.	R/W	0
2	URESET	USB Reset This field is only valid if the core works in host mode. When software writes a one to this bit, an USB reset sequence (SE0 state) will be started by the core. Writing a zero to this bit will terminate the reset sequence. There may be a delay before the bit status changes to zero, because hardware will clear the bit if it has completed the reset sequence. Software must keep the bit at one long enough as defined in the USB Specification 2.0, chapter 7.3.2. (10..20 ms) 0 : No USB reset driven. [label = ”URESET INACTIVE”] 1 : Core drives USB reset. [label = ”URESET ACTIVE”]	R/W	0

1:0	PTESTC	Port Test Control This bitfield allows to set the port into specific test modes. When the field is zero, the port is not operating in test mode. The encoding of test mode is: 00 : Test mode disabled [label = ”PTESTC DIS”] 01 : Test J State [label = ”PTESTC JST”] 10 : Test K State [label = ”PTESTC KST”] 11 : Test SE0 [label = ”PTESTC SE0”]	R/W	0x00

USB_PORT_STAT_CHG_EV – USB Port Status Change Event Register 0x00120034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																							P_END_EV	PWRSC_EV	CDC_EV	URES_EV	SUSP_EV	RSUC_EV	RSU_EV	BERR_EV	OCU_EV

Bits	Name	Description	R/W	Default
31:9	-	-	-	-
8	P_END_EV	Pulse End Event This field is only valid, when the core works in peripheral mode. The bit is set if the Pulse Length Field changes from one to zero. Write a 1 in this position to clear this bit.	R/W	0
7	PWRSC_EV	Power Status Change Event The bit is set if one of the following bits has changed: • Vbus Valid Bit • VA Session Valid Bit • VB Session Valid Bit • VB Session End Bit Write a 1 in this position to clear this bit.	R/W	0
6	CDC_EV	Connect/Disconnect Event This field is only valid if the core works in host mode. If set to one, this bit indicates that the Port Connect Status Bit value has changed. Write a 1 in this position to clear this bit.	R/W	0
5	URES_EV	USB Reset Event If the core works in host mode: After software has completed the USB reset by clearing the USB Reset Bit bit, this event bit will be set after the reset was completed by hardware. (The core continues driving the reset after URESET is deasserted until the next Start Of Frame) If the core works in peripheral mode. This bit is set if an USB reset condition was detected. (data lines more than 2,5 µs SE0). Write a 1 in this position to clear this bit.	R/W	0
4	SUSP_EV	Suspend Event This bit is only valid if the core works in peripheral mode. This bit indicates a lack of bus activity for more than 3 ms. Software can use this event to activate a low power mode. Note: If the low power mode causes that the clock of the USB Core is removed, the bounding hardware is responsible to turn on the clock controller if a state other than IDLE state at the USB data lines is asserted. This requires some additional, asynchronous circuitry. If HNP was not enabled previously, software shall set the Port Suspend Bit to be sensitive for resume signaling, and enter the low power state. If HNP was enabled, software can now switch to host mode according to the HNP rules. Write a 1 in this position to clear this bit.	R/W	0

3	RSUC_EV	Resume Complete Event If the core works in host mode: This bit will be set after the core has completed the resume sequence with a low speed EOP. If the core works in peripheral mode: This bit is set if the resume sequence was completed by the host with a low speed EOP. Write a 1 in this position to clear this bit.	R/W	0
2	RSU_EV	Resume Event If the core works in host mode: This bit is set if the host driver has set the port in suspended state, and a resume condition driven by the connected device was detected. (remote wakeup) If the core works in peripheral mode: This bit is set if resume condition was detected (more than 2.5 µs K-State). Write a 1 in this position to clear this bit.	R/W	0
1	BERR_EV	Babble Error Event This field is only valid, when the core works in host mode. this bit gets only to a one when the root port is disabled due to the appropriate conditions existing at the EOF2 point (/USB-Spec20/ chapter 11.2.5). Write a 1 in this position to clear this bit.	R/W	0
0	OCU_EV	Over Current Event This bit will be set to one if the core has detected an overcurrent condition. (refer the Over Current Condition Bit) Write a 1 in this position to clear this bit.	R/W	0

USB_PORT_STAT_CHG_EV_MSK – USB Port Status Change Event Mask Register 0x00120038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																							P_END_EM	PWRSC_EM	CDC_EM	URES_EM	SUSP_EM	RSUC_EM	RSU_EM	BERR_EM	OCU_EM

Bits	Name	Description	R/W	Default
31:9	reserved	-	R	0x00
8	P_END_EM	Pulse End Event Mask Setting this bit will lead to an interrupt if the Pulse End Event Bit is active.	R/W	0
7	PWRSC_EM	Power Status Change Event Mask Setting this bit will lead to an interrupt if the Power Status Change Event Bit is active.	R/W	0
6	CDC_EM	Connect/Disconnect Event Mask Setting this bit will lead to an interrupt if the Connect/Disconnect Event Bit is active.	R/W	0
5	URES_EM	USB Reset Event Mask Setting this bit will lead to an interrupt if the USB Reset Event Bit is active.	R/W	0
4	SUSP_EM	Suspend Event Mask Setting this bit will lead to an interrupt if the Suspend Event Bit is active.	R/W	0
3	RSUC_EM	Resume Complete Event Mask Setting this bit will lead to an interrupt if the Resume Complete Event Bit is active.	R/W	0
2	RSU_EM	Resume Event Mask Setting this bit will lead to an interrupt if the Resume Event Bit is active.	R/W	0
1	BERR_EM	Babble Error Event Mask Setting this bit will lead to an interrupt if the Babble Error Event Bit is active.	R/W	0
0	OCU_EM	Over Current Event Mask Setting this bit will lead to an interrupt if the Over Current Event Bit is active.	R/W	0

Pipe Register Group

This section defines the interface data structures, which will be used to communicate control/status information and data between the software and the hardware.
If the core works in peripheral mode:
Pipe[0] is reserved to handle control transfers. No other pipe can be used for control transfer. Because the control pipe is defined as bidirectional pipe, the core is sensitive to tokens for both directions. A SETUP transaction will always be accepted with an ACK handshake as required by the USB Specification. If no data buffer is available, the default response to OUT or IN tokens is NAK.
All pipe registers will be set to their reset values after the core has detected an USB reset.

USB_PIPE_CTRL – USB Pipe Control Register 0x00120040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																													ACT	TPID	

Bits	Name	Description	R/W	Default
31:3	reserved	-	R	0x00
2	ACT	Activate Pipe If the core works in host mode: Setting this bit will initiate one or more transactions with the given configuration for pipe. The bit will be cleared automatically if the transfer is finished or an error condition has occurred. Data transfer is only initiated, if valid buffer space is provided by software. If the core works in peripheral mode: Setting this bit will enable the response to an endpoint with the given pipe parameters. If no valid buffer space is provided, the endpoint will respond with a NAK handshake. The bit will be set automatically for pipe[0] after an USB reset was detected. It will be automatically cleared by hardware for all other pipes after the detection of an USB reset.	R/W	0
1:0	TPID	Token PID/Direction If the core works in host mode: This field indicates the PID (and direction) used for the token to be sent. If the core works in peripheral mode: This field indicates the direction of data flow for this endpoint. For receive direction, the bitfield must be set to TPID IN. For transmit direction, the bitfield must be set to TPID OUT. The bitfield will be automatically set to TPID SETUP for pipe[0] after a SETUP token was received. 00 : OUT transfer [label = ”TPID_OUT”] 01 : IN transfer [label = ”TPID_IN”] 10 : SETUP transfer [label = ”TPID_SETUP”] 11 : Reserved.	R/W	0

USB_PIPE_CFG – USB Pipe Configuration Register 0x00120044

Access Rules:
If the core works in host mode:
The contents of this register must not be changed when the Activate Pipe Bit or the Active Status Bit is one.
If the core works in peripheral mode:
The contents of this register (except the Stall Pipe Bit) must not be changed when the Activate Pipe Bit is one.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	IOT	HIDBE	SKIPISO	PI				POFF								STALL	ACID	EPS	STRM	ET		MPS									

Bits	Name	Description	R/W	Default
31	reserved	-	R	0
30	IOT	Interrupt on Transaction. This field is only valid if the core works in host mode. If this bit is set, the core will assert an interrupt after every performed transaction on this pipe, independently on the transaction status. That means, also for NAKed transactions or after an error condition (timeout, CRC-error) an interrupt will be generated. Setting this bit is only useful if the transaction scheduling must be performed by software.	R/W	0
29	HIDBE	Halt on ISO Data Buffer Error This field is only valid if the core works in host mode. If the Skip ISO Token Bit is not set, and the core sends an IN token for isochronous IN pipes, and there is no buffer available, the core will halt this pipe, when the HIDBE bit is activated. If HIDBE is deactivated, the core will not change the pipe state.	R/W	0
28	SKIPISO	Skip ISO Token This field is only valid if the core works in host mode. When this bit is set, the HC will send no IN/OUT token for isochronous pipes, when no data buffer space is available. In the case, the IN token is send (the Skip ISO Token Bit is 0) and there is no bufferspace the HC will not write any data to the RAM. The received data is lost. For OUT pipes the HC will send an OUT-DATA0 sequence, when no buffer space is available.	R/W	0
27:24	PI	Polling Interval This field is only valid if the core works in host mode. This value defines the polling interval for this pipe as 2 PI , if the endpoint is configured as an interrupt or isochronous endpoint. The polling intervals for full/low speed interrupt endpoints is encoded as follows: 0000 : 1 frame 0001 : 2 frames 0010 : 4 frames : 1000 : 256 frames PI must be less/equal 8. Setting PI to a value greater 8 will lead to undefined results.	R/W	0x00
23:16	POFF	Polling Offset This field is only valid if the core works in host mode and for interrupt endpoints. Defines the polling interval frame offset the pipe is scheduled for. The pipe is polled if PI OFF[PI-1:0] matches the current frame number slice FRAME NR[PI-1:0]. (8 >= PI >= 1)	R/W	0x00

15	STALL	Stall Pipe This bit is only valid if the core works in peripheral mode. Setting this bit will lead to a STALL response to an endpoint with the given pipe parameters. The bit will be automatically cleared for pipe[0] after an USB reset was detected or a SETUP token was received.	R/W	0
14	ACID	Accept corrupted ISO Data When this bit is set, the core accepts also corrupted data, transfered by isochronous receive pipes. Packets with CRC error are accepted completely. Packets with bitstuff error are accepted until the bitstuff error occurs.	R/W	0
13	EPS	Endpoint Speed This field is only valid if the core works in host mode. Indicates the speed of the function with this endpoint. 0 : Full speed [label = ”EPS FULL”] 1 : Low speed [label = ”EPS LOW”] If the device connected to the root port is a full speed, and the bit is set to low speed, a PRE token will be generated automatically.	R/W	0
12	STRM	Streaming Mode This bit is only valid for receive pipes/endpoints. It should be used for high bandwidth periodic receive pipes/endpoints. If this bit is activated, the streaming mode is used for this pipe: • The data received by several bus transactions can be stored in one buffer. • The inter buffer byte alignment is done, so that different buffers can be concatenated without shifting the entire data by 1 or more bytes. If this bit is deactivated for this pipe: • The data of one transaction for this pipe is stored in a separate buffer • No inter buffer byte alignment is done.	R/W	0
11:10	ET	Endpoint Type Indicates the transfer type for the transactions of this pipe/endpoint: 00 : Control Transfer [label = ”ET CTRL”] 01 : Isochronous Transfer [label = ”ET ISO”] 10 : Bulk Transfer [label = ”ET BULK”] 11 : Interrupt Transfer [label = ”ET INT”]	R/W	0x00
9:0	MPS	Maximum Packet Size This field defines the maximum number of bytes per data packet that can be sent to or received from the endpoint of this pipe. If the core works in host mode: If more bytes are received, and the core is configured for host mode, a babble error will be generated. If the core works in peripheral mode: The bitfield will be ignored if the received transaction was a SETUP transaction. (SETUP transactions must always be accepted)	R/W	0

USB_PIPE_ADDR – USB Pipe Address Register 0x00120048

Access Rules:
If the core works in host mode:
The contents of this register must not be changed when the Activate Pipe Bit or the Active Status Bit is one.
If the core works in peripheral mode:
The contents of this register (except the Stall Pipe Bit) must not be changed when the Activate Pipe Bit is one.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																					EPADDR							ERNR			

Bits	Name	Description	R/W	Default
31:11	reserved	-	R	0x00
10:4	EPADDR	Endpoint Address This field defines the USB device address of this pipe. It must be set for both host and peripheral mode. If working in peripheral mode, this field defines the device address. Software must set this field only for pipe[0] after the successful completion of a SetAddress-request. The value of this field of pipe[0] is used for all other pipes. It will be automatically reset to 0x00 if an USB reset was detected.	R/W	0x00
3:0	ERNR	Endpoint Number It specifies the endpoint number of this pipe/endpoint. This field must be set for both host and peripheral mode. If the core works in peripheral mode: The bitfield will be automatically set to 0x0 for pipe[0].	R/W	0x00

USB_PIPE_STAT – USB Pipe Status Register 0x0012004c

This register contains the complete status information of the selected pipe and can be changed by hardware.
Access Rules:
If the core works in host mode:
The contents of this register must not be changed when the Activate Pipe Bit or the Active Status Bit is one.
If the core works in peripheral mode:
The contents of this register (except the Stall Pipe Bit) must not be changed when the Activate Pipe Bit is one.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																						CERR		DBERR	ACTS	HALT	BBL	DBSEL	DT	DBOFF	

Bits	Name	Description	R/W	Default
31:10	reserved	-	R	0x00
9:8	CERR	Error Counter This field is only valid if the core works in host mode. Shows the status of a 2-bit down counter that, keeps track of the number of consecutive errors detected. The core decrements the counter if the transaction fails. If the counter counts from one to zero, the pipe will be halted and sets the Pipe Halted Bit. When the Activate Pipe Bit of the Pipe Control Register goes from 0x0 to 0x1, the core resets this bitfield.	R	0x3
7	DBERR	Data Buffer Error This field is only valid if the core works in host mode. Set to a 1 during the status update to indicate that the core is unable to keep up with the reception of incoming data (overrun). This bit is only valid for isochronous IN pipes. For all other kinds of transfers the core ensures, that there is enough buffer space for at least 1 packet with maximum payload size. When the Activate Pipe Bit of the Pipe Control Register goes from 0x0 to 0x1, the core resets this bit.	R	0
6	ACTS	Active Status This field is only valid if the core works in host mode. This bit shows the current status of the pipe. It will be set, after the Activate Pipe Bit was set to 1. It will be cleared, after the Activate Pipe Bit was set to 0, or the pipe was halted due to several transfer events (pipe halted, babble, short packet for control/bulk pipes etc.).	R	0
5	HALT	Pipe Halted This field is only valid if the core works in host mode. Set to a 1 by the core during status update if the core works in host mode to indicate that a serious error has occurred at the device/endpoint. This can be caused by babble errors, the error counter counting down to zero, or reception of the STALL handshake from the device. Any time that a transaction results in the this bit being set to a one, the Activate Pipe Bit is also set to 0. When the Activate Pipe Bit of the Pipe Control Register goes from 0x0 to 0x1, the core resets this bit.	R	0
4	BBL	Babble detected This field is only valid if the core works in host mode. Set to a 1 by the Host Controller during status update when a babble is detected during the transaction. In addition to setting this bit, the hardware also sets the Pipe Halted Bit to a 1 (only for IN endpoints). When the Activate Pipe Bit of the Pipe Control Register goes from 0x0 to 0x1, the core resets this bit.	R	0

3	DBSEL	Selected Data Buffer This bit is only valid if the pipe is configured to have an alternative data buffer. Other-wise the bit will stuck at zero. The bit indicates, which buffer is currently processed by hardware: 0 : (Normal) Data Buffer processed. [label = ”DBSEL NORMAL”] 1 : Alternative Data Buffer processed. [label = ”DBSEL ALTERNATE”] The bit will be updated by hardware if the data controller switches between the data buffers. The bit can also be changed by software.	R/W	0
2	DT	Data Toggle This bit shows the current data toggle. It is valid for all bulk, control and interrupt transfers. This bit will be changed by hardware according to the current data toggle state. The bit can also be changed by software. 0 : DATA0 data toggle value [label = ”DT DATA0”] 1 : DATA1 data toggle value [label = ”DT DATA1”] If the core works in peripheral mode: When SETUP token was received, the bit will be automatically set to DT DATA1 for pipe[0].	R/W	0
1:0	DBOFF	Data Byte Offset For transmit transfers, this field indicates the current byte of data to be sent within a data word. For receive transfers, this field points to the first byte within the data word where the first received byte must be stored. If the core parameter DW (Data Width) is set to 16, only the lowest bit of this bitfield is valid. The bitfield can be changed by software. • For transmit pipes: this bitfield traces the current byte offset for this pipe, it is set by hardware to 0 after an buffer becomes invalid. • For streaming receive pipes: this bitfield traces the current byte offset for this pipe. If the pipe works in streaming mode (refer Streaming Mode Bit), this byte offset is also used, to eliminate byte shifting operations by software. Software has only to initialize the byte offset at the beginning of the transfer. • For non-streaming receive pipes: this bitfield traces the current byte offset for this pipe, it is set by hardware to 0 after an buffer becomes invalid. If the core works in peripheral mode: The bitfield will be automatically set to 0x0 for pipe[0] if a SETUP transaction is received.	R/W	0

USB_PIPE_DATA_PTR – USB Pipe Data Pointer Register 0x00120050

The contents of this register can be changed by hardware.
Access Rules: the contents of this register must not be changed when the access to the Pipe Configuration Register is granted and the Data Buffer Valid Bit is active.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																						DPTR									

Bits	Name	Description	R/W	Default
31: 10	reserved	-	R	0x00
9:0	DPTR	Data Pointer For transmit transfers, this field points to the first byte of data to be sent. For receive transfers, this field points to the address where the first received byte must be stored. The width of this field depends on the address width of the used RAM. It is important to note, that this bitfield represents a real DATAWORD POINTER. That means the complete address is calculated by appending the valid DBOFF bits to this bitfield. If the core works in peripheral mode: The bitfield will be automatically set to 0x0000 for pipe[0] if a SETUP transaction is received.	R/W	0x00

USB_PIPE_DATA_TOT – USB Pipe Total Bytes Register 0x00120054

The contents of this register can be changed by hardware.
Access Rules: the contents of this register must not be changed when the access to the Pipe Configuration Register is granted and the Data Buffer Valid Bit is active.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBV	30:TBW reserved																		TBW-1:0 TBYTES												

Bits	Name	Description	R/W	Default
31	DBV	Data Buffer Valid Software must set this bit to indicate that the data described by the Pipe Data Pointer Register and Pipe Total Bytes Register is valid, respectively the buffer is valid for incoming data. The bit will be cleared by hardware after the buffer was finished (e.g. all data was transmitted or received).	R/W	0
30:TBW	reserved	-	R	0x00
TBW-1:0	TBYTES	Total Bytes To Transfer This bitfield contains the total number of bytes to be transmitted or received. This field will be decremented according to the number of bytes received/transmitted with every transaction. Software must ensure that the field is only written if the data buffer is not valid. Writing to this register if the data is validated will lead to undefined results. For receive pipes, software must set this field always to a multiple of Maximum Packet Size Field. Transfers will only be initiated if TBYTES is greater/equal MPS. If the core works in peripheral mode: The bitfield will be ignored by hardware if the received transaction was a SETUP transaction. (SETUP transactions must always be accepted)	R/W	0x00

TBW: Total Bytes Width

USB_PIPE_ALT_DATA_PTR – USB Pipe Alternative Data Pointer Register 0x00120058

Access Rules: the contents of this register must not be changed when the access to the Pipe Configuration Register is granted and the Alternative Data Buffer Valid Bit is active

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																						ALT_DATA_PTR									

Bits	Name	Description	R/W	Default
31:10	reserved	-	R	0x00
9:0	ALT_DATA_PTR	Alternative Data Pointer This register does only exist, if alternate buffer support is enabled for this pipe. Otherwise this bitfield will stuck at zero. For transmit transfers, this field points to the first byte of the alternative data buffer to be sent. For receive transfers, this field points to the address where the first received byte must be stored. The width of this field depends on the address width of the used RAM..	R/W	0x00

USB_PIPE_ALT_DATA_TOT – USB Pipe Alternative Data Total Bytes Register 0x0012005c

Access Rules: the contents of this register must not be changed when the access to the Pipe Configuration Register is granted and the Alternative Data Buffer Valid Bit is active.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADBV	30:TBW reserved																				TBW-1:0 ATBYTES										

Bits	Name	Description	R/W	Default
31	ADBV	Alternative Data Buffer Valid This bit is only valid if the pipe is configured to have an alternative data buffer . Otherwise the bit will stuck at zero. Software must set this bit to indicate that the data described by the Pipe Alternative Data Pointer Register and Pipe Alternative Data Total Bytes Register is valid, respectively the buffer is valid for incoming data. The bit will be cleared by hardware: • After an receive buffer was finished, this condition is checked during the status update. The Buffer will be finished when the Total Bytes To Transfer Field is less Maximum Packet Size Field and Selected Data Buffer Bit is 1. • After an transmit buffer was finished, this condition is checked during the status update. The Buffer will be finished when the Total Bytes To Transfer Field is 0x0 and Selected Data Buffer Bit is 1. • The pipe state (Active Status Bit) has changed from active to inactive and the core works in host mode.	R/W	0
30:TBW	reserved	-	R	0x00
TBW-1:0	ATBYTES	Alternative Total Bytes To Transfer It contains the total number of bytes of the alternative data buffer to be transmitted or received. This field will be decremented according to the number of bytes received/transmitted with every transaction. Software must ensure that the field is only written if the alternative data buffer is not valid. Writing to this register if the data is validated will lead to undefined results!	R/W	0x00

TBW: Total Bytes Width

Debug Register Group

These registers provide various debug capabilities. They will not be used for normal operation. Debug mode is enabled if one of the Debug Control Register bits are set to one.
Debug mode is only valid for pipe 0. The basic configuration of this pipe will be used for a transaction and changes according to the debug configuration.
The core can be configured to support debug or extended debug capability. Therefore, some registers or bits can be not valid (stuck at zero).

USB_DBG_CTRL – USB Debug Control Register 0x00120060

Debug mode is enabled if one of the control bits is set to one.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																						UDTPID	UDHSPID	UDDPID	FRXCRC16G	FRXCRC5G	FRXCRCE	FTXCRC16E	FTXCRC5E	DBSTX	DBSERRDET

Bits	Name	Description	R/W	Default
31:10	reserved	-	R	0x00
9	UDTPID	Use Debug Token PID If the bit is set, the host controller will use the Debug Token PID Field for the next transaction. The bit will only exist, if extended debug capability support was implemented. Otherwise the bit will stuck at zero.	R/W	0
8	UDHSPID	Use Debug Handshake PID If the bit is set, the host controller will use the Debug Handshake PID Field for the next transaction. The bit will only exist, if extended debug capability support was implemented. Otherwise the bit will stuck at zero.	R/W	0
7	UDDPID	Use Debug Data PID If the bit is set, the host controller will use the Debug Data PID Field for the next transaction. The bit will only exist, if extended debug capability support was implemented. Otherwise the bit will stuck at zero.	R/W	0
6	FRXCRC16G	Force Receive Good CRC16 If this bit is set, the core will treat every received data CRC16 as a good CRC. The bit will only exist, if debug capability support was implemented. Otherwise the bit will stuck at zero.	R/W	0
5	FRXCRC5G	Force Receive Good CRC5 If this bit is set, the core will treat every received CRC5 of an incoming token or SOF as a good CRC. The bit will only exist, if debug capability support was implemented. Otherwise the bit will stuck at zero.	R/W	0
4	FRXCRCE	Force Receive CRC Error If this bit is set, the core will treat every received CRC as a wrong CRC. The bit will only exist, if debug capability support was implemented. Otherwise the bit will stuck at zero.	R/W	0
3	FTXCRC16E	Force Transmit CRC16 Error If this bit is set, the core will generate a wrong data CRC16 (Ones complement). The bit will only exist, if debug capability support was implemented. Otherwise the bit will stuck at zero.	R/W	0

2	FTXCRC5E	Force Transmit CRC5 Error If this bit is set, the core will generate a wrong CRC5 for every token/SOF (Ones complement). The bit will only exist, if debug capability support was implemented. Otherwise the bit will stuck at zero.	R/W	0
1	DBSTX	Disable Bitstuffing Transmit If this bit is set, the core will not perform bitstuffing at the output stream. The bit will only exist, if debug capability support was implemented. Otherwise the bit will stuck at zero.	R/W	0
0	DBSERRDET	Disable Bitstuff Error Detection If this bit is set, the core will not report bitstuff errors. Because every bitstuff error will lead to a CRC error, the completion code will show only CRC errors instead of bitstuff errors. The bit will only exist, if debug capability support was implemented. Otherwise the bit will stuck at zero.	R/W	0

USB_DBG_PID – USB Debug PID Register 0x00120064

This register will only exist, if extended debug capability support was implemented. Otherwise the bit will stuck at zero.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved								DHSPID								DTPID								DDPID							

Bits	Name	Description	R/W	Default
31:24	reserved	-	R	0x00
23:16	DHSPID	Debug Handshake PID This bitfield will be sent as handshake PID if the device has returned data.	R/W	0x00
15:8	DTPID	Debug Token PID This bitfield will be sent as token PID if debug mode is enabled instead of an IN/OUT/SETUP token PID.	R/W	0x00
7:0	DDPID	Debug Data PID This bitfield will be sent as data PID if debug mode is enabled instead of an DATA0, DATA1, DATA2 or MDATA PID.	R/W	0x00

USB_DBG_STAT – USB Debug Status Register 0x00120068

This register will only exist, if extended debug capability support was implemented. Otherwise the bit will stuck at zero.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								DRXPIP							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	DRXPIP	Debug Receive PID This bitfield contains the received PID of the last transaction, independently of the trans-action status.	R	0x00

USB_TEST – USB Test Register 0x0012006c

This register exists for system integration tests only.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RW32																															

Bits	Name	Description	R/W	Default
31:0	RW32	Read Write 32 This test field can be read and written in any order and with any contents. Every write access stores the 1's complement of the written value.	R/W	0x00

Core Configurations Register Group

The following registers show the parameters the core is configured with All bitfields mirror the core's configuration, it is not possible to change the configuration by writing these registers!

USB_MAIN_CFG – USB Main Configuration Register 0x00120080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved				RAW_CFG				reserved												DW_CFG						NOP_CFG					

Bits	Name	Description	R/W	Default
31:30	reserved	-	R	0x00
29:24	RAW_CFG	RAM Address Width Configuration This field shows the address width of the Dual Port RAM the core is configured to.	R	0x0a
23:12	reserved	-	R	0x00
11:6	DW_CFG	Data Width Configuration This bitfield shows the data width (DW) of the core.	R	0x20
5:0	NOP_CFG	Number of Pipes Configuration This field shows the number of pipes (NOP) the core is configured with.	R	0x08

USB_MODE_CFG – USB Mode Configuration Register 0x00120084

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved														XDBG_CFG	DBG_CFG	reserved								ABUFF_CFG							

Bits	Name	Description	R/W	Default
31:18	reserved	-	R	0x00
17	XDBG_CFG	Extended Debug Configuration This bit shows whether this core is configured for extended debug support.	R	0
16	DBG_CFG	Debug Configuration This bit shows whether this core is configured for debug support.	R	0
15:8	reserved	-	R	0x00
7:0	ABUFF_CFG	Alternative Buffer Configuration This bitfield shows which pipe is configured for alternative Buffer usage.	R	0x00

USB_CORE_CTRL – USB Core Control and Status Register 0x00120088

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved							USB_IRQ	UCIF_RDY	DISCHRG_VBUS	VB_ON	DLP_ACTIVE	CHRG_VBUS	VBUS_VLD	VB_SESS_VLD	VB_SESS_END	VA_SESS_VLD	OVER_CURRENT	XCVR_SUSPEND_	CORE_SUSPEND_	ALT_BUFFER_SUPPORT								SOFT_ID_DIG	XTD_DBG_SUPPORT	DBG_SUPPORT	RESET

Bits	Name	Description	R/W	Default
31:25	reserved	-	R	0x00
24	USB_IRQ	reflects usb_irq	R	0
23	UCIF_RDY	reflects ucif_rdy	R	0
22	DISCHRG_VBUS	reflects dischrg_vbus	R	0
21	VB_ON	reflects vb_on	R	0
20	DLP_ACTIVE	reflects dlp_active	R	0
19	CHRG_VBUS	reflects chrg_vbus	R	0
18	VBUS_VLD	reflects vbus_vld	R	0
17	VB_SESS_VLD	reflects vb_sess_vld	R	0
16	VB_SESS_END	reflects vb_sess_end	R	0
15	VA_SESS_VLD	reflects va_sess_vld	R	0
14	OVER_CURRENT	reflects over_current	R	0
13	XCVR_SUSPEND_N	reflects xcvr_suspend_n	R	0
12	CORE_SUSPEND_N	reflects core_suspend_n	R	0
11:4	ALT_BUFF_SUPPORT	alt buffer support	R/W	0x00
3	SOFT_ID_DIG	to set id_dig via software	R/W	0
2	XTD_DBG_SUPPORT	enables extended debug support	R/W	0
1	DBG_SUPPORT	enables debug support	R/W	0
0	RESET	software reset	R/W	0

USB_FIFO – FIFO for USB-Interface from 0x00130000 to 0x0013ffff

The function of the USB_FIFO is sending and receiving frames.

6.10 VIC – Vectored Interrupt Controller

The Vectored Interrupt Controller (VIC) supports 32 interrupt sources, where 16 can be vectored. The interrupt priority and the type of interrupt(IRQ or FIQ) are configurable. All Interrupts can be masked. Some of the Interrupts represent the result of the or-function of up to 32 single interrupts of a function block. Inside the Interrupt service routine these interrupts have to be checked and reset.

The following table shows the different interrupt sources:

Bit Position	Name	Description
31	-	reserved
30	TIMER4	Timer 4
29	TIMER3	Timer 3
28	-	reserved
27	-	reserved
26	ISO_AREA	Power is going down (see RTC for details)
25	INT_PHY	Interrupt from internal PHYs
24	MSYNC3	Motion synchronization channel 3 (= logic OR of xpec3_irq[15:12])
23	MSYNC2	Motion synchronization channel 2 (=logic OR of xpec2_irq[15:12])
22	MSYNC1	Motion synchronization channel 1 (=logic OR of xpec1_irq[15:12])
21	MSYNC0	Motion synchronization channel 0 (=logic OR of xpec0_irq[15:12])
20	COM3	Communication channel 3 (=logic OR of xpec3_irq[11:0])
19	COM2	Communication channel 2 (=logic OR of xpec2_irq[11:0])
18	COM1	Communication channel 1 (=logic OR of xpec1_irq[11:0])
17	COM0	Communication channel 0 (=logic OR of xpec0_irq[11:0])
16	GPIO	Other external Interrupts from GPIO 0-14
15	HIF	HIF interrupt
14	LCD	LCD-Controller interrupt
13	I2C	Reserved
12	SPI	SPI interrupt
11	USB	USB interrupt
10	UART2	UART 2
9	UART1	UART 1
8	UART0	UART 0
7	WATCHDOG	reserved
6	GPIO15	External interrupt at GPIO 15
5	SYSTIME_S	System time IRQ from SYSTIME module
4	SYSTIME_NS	System time ns compare interrupt from GPIO module
3	TIMER2	Timer 2 / Counter 2 from GPIO Module
2	TIMER1	Timer 1 / Counter 1 from GPIO Module
1	TIMER0	Timer 0 / Counter 0 from GPIO Module
0	SW	Reserved for Software Interrupt, ARM standard configuration

6.10.1 Summary of VIC Registers

ARM Address	Register Name	Short Description
0x001ff000	VIC_IRQ_STAT	IRQ Status Register
0x001ff004	VIC_FIQ_STAT	FIQ Status Register
0x001ff008	VIC_RAW_INT_STAT	Raw Interrupt Status Register
0x001ff00c	VIC_INT_SEL	Interrupt Select Register
0x001ff010	VIC_INT_EN	Interrupt Enable Register
0x001ff014	VIC_INT_EN_CLR	Interrupt Enable Clear Register
0x001ff018	VIC_SWI	Software Interrupt Register
0x001ff01c	VIC_SWI_CLR	Software Interrupt Clear Register
0x001ff020	VIC_PROT_EN	Protection Enable Register
0x001ff030	VIC_VECT_ADDR	Vector Address Register
0x001ff034	VIC_DFLT_VECT_ADDR	Default Vector Address Register
0x001ff100	VIC_VECT_ADDR0	Vector Address Register 0
0x001ff104	VIC_VECT_ADDR1	Vector Address Register 1
0x001ff108	VIC_VECT_ADDR2	Vector Address Register 2
0x001ff10c	VIC_VECT_ADDR3	Vector Address Register 3
0x001ff110	VIC_VECT_ADDR4	Vector Address Register 4
0x001ff114	VIC_VECT_ADDR5	Vector Address Register 5
0x001ff118	VIC_VECT_ADDR6	Vector Address Register 6
0x001ff11c	VIC_VECT_ADDR7	Vector Address Register 7
0x001ff120	VIC_VECT_ADDR8	Vector Address Register 8
0x001ff124	VIC_VECT_ADDR9	Vector Address Register 9
0x001ff128	VIC_VECT_ADDR10	Vector Address Register 10
0x001ff12c	VIC_VECT_ADDR11	Vector Address Register 11
0x001ff130	VIC_VECT_ADDR12	Vector Address Register 12
0x001ff134	VIC_VECT_ADDR13	Vector Address Register 13
0x001ff138	VIC_VECT_ADDR14	Vector Address Register 14
0x001ff13c	VIC_VECT_ADDR15	Vector Address Register 15
0x001ff200	VIC_VECT_CTRL0	Vector Control Register 0
0x001ff204	VIC_VECT_CTRL1	Vector Control Register 1
0x001ff208	VIC_VECT_CTRL2	Vector Control Register 2
0x001ff20c	VIC_VECT_CTRL3	Vector Control Register 3
0x001ff210	VIC_VECT_CTRL4	Vector Control Register 4
0x001ff214	VIC_VECT_CTRL5	Vector Control Register 5
0x001ff218	VIC_VECT_CTRL6	Vector Control Register 6
0x001ff21c	VIC_VECT_CTRL7	Vector Control Register 7
0x001ff220	VIC_VECT_CTRL8	Vector Control Register 8
0x001ff224	VIC_VECT_CTRL9	Vector Control Register 9
0x001ff228	VIC_VECT_CTRL10	Vector Control Register 10
0x001ff22c	VIC_VECT_CTRL11	Vector Control Register 11
0x001ff230	VIC_VECT_CTRL12	Vector Control Register 12
0x001ff234	VIC_VECT_CTRL13	Vector Control Register 13
0x001ff238	VIC_VECT_CTRL14	Vector Control Register 14
0x001ff23c	VIC_VECT_CTRL15	Vector Control Register 15

6.10.2 VIC Overview

The Vectored Interrupt Controller (VIC) provides a software interface to the interrupt system. In a system with an interrupt controller, software must determine the source that is requesting service and where its services routine is loaded. A VIC does both of these in hardware. It supplies the starting address(or vector address) of the service routine corresponding to the highest priority requesting interrupt source.

In an ARM processor based system, two levels of interrupt are available:

- Fast Interrupt Request (FIQ) for fast, low latency interrupt handling.
- Interrupt Request (IRQ) for more general interrupts.

Only a single FIQ source at a time is generally used in a system, to provide a true low-latency interrupt. This has the following benefits:

- You can execute the interrupt service routine directly without determining the source of the interrupt.
- Interrupt latency is reduced. You can use the banked registers available for FIQ interrupts more efficiently, because a context save is not required.

There are 32 interrupt lines. The VIC uses a bit position for each different interrupt source. The software can control each request line to generate software interrupts.

There are 16 vectored interrupts available, which can also be used as nonvectored interrupts if the system does not support interrupt vector addresses. These interrupts can only generate an IRQ interrupt. The vectored and none vectored IRQ interrupts provide an address for an Interrupt Service Routine (ISR). Reading from the vector interrupt address register, VIC_VECT_ADDR, provides the address of the ISR, and updates the interrupt priority hardware that masks out the current and any lower priority interrupt requests. Writing to the VIC_VECT_ADDR register indicates to the interrupt priority hardware that the current interrupt is serviced, allowing lower priority interrupts to go active.

The FIQ interrupt has the highest priority, followed by interrupt vector0-15, and nonvectored IRQ interrupts have the lowest priority. Among vectored IRQ interrupts, the priority level is fixed by hardware, vector0 has the highest priority level and vector15 has the lowest.

A programmed interrupt request allows you to generate an interrupt under software control. This register VIC_INT_SEL is typically used to downgrade an FIQ interrupt to an IRQ interrupt. This is done by clearing the FIQ and setting up a software IRQ instead.

Note:
The priority of the FIQ over IRQ is set by the ARM processor. The VIC can raise both an FIQ and an IRQ at the same time.

6.10.3 Interrupt Generation

Interrupt request generation

In the following the generation of FIQ and IRQ is described:
The interrupt request logic receives the interrupt requests from the peripheral and combines them with the software interrupt requests. It then masks out the interrupt request which are not enabled, and routes the enabled interrupt requests to either VIC_FIQ_STAT[31:0] or VIC_IRQ_STAT[31:0].

Vectored interrupt generation

There are 16 vectored interrupt blocks which generate 16 vectored interrupt signals. A vectored interrupt is only generated if the following are true:

- the selected interrupt is active
- the selected interrupt is currently the highest priority requesting interrupt
- the selected interrupt is enabled in the vector control register (VIC_VECT_CTRLI[0-15]).

Software interrupts

The software can control the source interrupt lines to generate software interrupts. These interrupts are generated before interrupt masking, in the same way as external source interrupts. Software interrupts are cleared by writing to the software interrupt clear register, VIC_SWI_CLR. This is normally done at the end of the interrupt service routine.

6.10.4 Interrupt priority logic

The interrupt priority block prioritizes the following requests:

- none vectored interrupt requests
- vectored interrupt requests.

The highest-priority request generates an IRQ interrupt if the interrupt is not currently being serviced. The FIQ interrupt has the highest priority (outside the Interrupt priority logic), followed by interrupt vector 0 to interrupt vector 15. none vectored IRQ interrupts have the lowest priority.

6.10.5 Interrupt Flow Sequence

Vectored interrupt flow sequence:

The following procedure shows the sequence for the vectored interrupt flow:

- An interrupt occurs.
- The ARM processor branches to the IRQ interrupt vector.
- Read the VIC_VECT_ADDR register and branch to the interrupt service routine. This can be done using an LDR PC instruction. Reading the VIC_VECT_ADDR register updates the hardware priority register of the interrupt controller.
- Stack the workspace so that IRQ interrupts can be re-enabled.
- Enable the IRQ interrupts so that a higher priority can be serviced.
- Execute the Interrupt Service Routine (ISR).
- Clear the requesting interrupt in the peripheral, or write to the VIC_SWI_CLR register if the request was generated by a software interrupt.
- Disable the interrupts and restore the workspace.
- Write to the VIC_VECT_ADDR register. This clears the respective interrupt in the internal interrupt priority hardware.
- Return from the interrupt. This re-enables the interrupts.

Simple interrupt flow sequence:

The following procedure shows how you can use the interrupt controller without using vectored interrupts or the interrupt priority hardware.

- An interrupt occurs.
- Branch to IRQ or FIQ interrupt vector.
- Branch to the interrupt handler.
- Interrogate the VIC_IRQ_STAT register to determine which source generated the interrupt, and prioritize the interrupts if there are multiple active interrupt sources. This takes a number of instructions to compute.
- Branch to the correct ISR.
- Execute the ISR.
- Clear the requesting interrupt in the peripheral, or write to the VIC_SWI_CLR if the request was generated by a software interrupt.
- Check the VIC_IRQ_STAT register to ensure that no other interrupt is active. If there is an active request, go to Step 4 (Interrogate …).
- Return from the interrupt. This re-enables the interrupts.

Note:
If the above flow is used, you must not read or write to the VIC_VECT_ADDR register.

6.10.6 VIC Registers Descriptions

VIC_IRQ_STAT – VIC IRQ Status Register **0x001ff000**

The VIC_IRQ_STAT register provides the status of the interrupts after registers VIC_INT_EN and VIC_INT_SEL are configured. When an IRQ interrupt occurs, the corresponding bit of the interrupt source will be set to 1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	TIMER4	TIMER3	reserved		ISO_AREA	INT_PHY	MSYNC3	MSYNC2	MSYNC1	MSYNC0	COM3	COM2	COM1	COM0	GPIO	HIF	LCD	I2C	SPI	USB	UART2	UART1	UART0	WATCHDOG	GPIO15	SYSTIME_S	SYSTIME_NS	TIMER2	TIMER1	TIMER0	SW

Bits	Name	Description	R/W	Default
31:0	VIC_IRQ_STAT	Interrupt Status.	R	0x00000000

VIC_FIQ_STAT – VIC FIQ Status Register **0x001ff004**

The VIC_ FIQ _STAT register provides the status of a FIQ interrupt if the register VIC_INT_EN is enabled and the register VIC_INT_SEL select a FIQ interrupt. When a FIQ interrupt occurs, the corresponding bit of the interrupt source will be set to 1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	TIMER4	TIMER3	reserved		ISO_AREA	INT_PHY	MSYNC3	MSYNC2	MSYNC1	MSYNC0	COM3	COM2	COM1	COM0	GPIO	HIF	LCD	I2C	SPI	USB	UART2	UART1	UART0	WATCHDOG	GPIO15	SYSTIME_S	SYSTIME_NS	TIMER2	TIMER1	TIMER0	SW

Bits	Name	Description	R/W	Default
31:0	VIC_FIQ_STAT	Fast Interrupt Status	R	0x00000000

VIC_RAW_INT_STAT – VIC Raw Interrupt Status Register 0x001ff008

The VIC_RAW_INT_STAT register provides the status of the source interrupts (and software interrupts) to the interrupt controller.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	TIMER4	TIMER3	reserved		ISO_AREA	INT_PHY	MSYNC3	MSYNC2	MSYNC1	MSYNC0	COM3	COM2	COM1	COM0	GPIO	HIF	LCD	I2C	SPI	USB	UART2	UART1	UART0	WATCHDOG	GPIO15	SYSTIME_S	SYSTIME_NS	TIMER2	TIMER1	TIMER0	SW

Bits	Name	Description	R/W	Default
31:0	VIC_RAW_INT_STAT	Raw Interrupt Status	R	0x00000000

VIC_INT_SEL – VIC Interrupt Select Register 0x001ff00c

The VIC_INT_SEL register selects whether the corresponding interrupt source generates an FIQ or an IRQ interrupt.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	TIMER4	TIMER3	reserved		ISO_AREA	INT_PHY	MSYNC3	MSYNC2	MSYNC1	MSYNC0	COM3	COM2	COM1	COM0	GPIO	HIF	LCD	I2C	SPI	USB	UART2	UART1	UART0	WATCHDOG	GPIO15	SYSTIME_S	SYSTIME_NS	TIMER2	TIMER1	TIMER0	SW

Bits	Name	Description	R/W	Default
31:0	VIC_INT_SEL	Selects type of interrupt for interrupt request: 1: FIQ interrupt 0: IRQ interrupt	R/W	0x00000000

VIC_INT_EN – VIC Interrupt Enable Register 0x001ff010

The VIC_INT_EN register enables the interrupt request lines, by unmasking the interrupt sources for the IRQ interrupt. Clearing these bits are only possible by writing to the 'VIC_INT_EN_CLR - VIC Interrupt Enable Clear Register'

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	TIMER4	TIMER3	reserved		ISO_AREA	INT_PHY	MSYNC3	MSYNC2	MSYNC1	MSYNC0	COM3	COM2	COM1	COM0	GPIO	HIF	LCD	I2C	SPI	USB	UART2	UART1	UART0	WATCHDOG	GPIO15	SYSTIME_S	SYSTIME_NS	TIMER2	TIMER1	TIMER0	SW

Bits	Name	Description	R/W	Default
31:0	VIC_INT_EN	Enable the interrupt request lines. Read: 0: Interrupt disabled. 1: Interrupt enabled. Allows interrupt request to processor. Write: 0: no effect. 1: set the bit	R/W	0x00000000

VIC_INT_EN_CLR – VIC Interrupt Enable Clear Register 0x001ff014

The VIC_INT_EN_CLR register is for clearing bits in the VIC_INT_EN register. Writing a one bit will result in clearing the corresponding bit in the 'VIC_INT_EN - VIC Interrupt Enable Register'

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	TIMER4	TIMER3	reserved		ISO_AREA	INT_PHY	MSYNC3	MSYNC2	MSYNC1	MSYNC0	COM3	COM2	COM1	COM0	GPIO	HIF	LCD	I2C	SPI	USB	UART2	UART1	UART0	WATCHDOG	GPIO15	SYSTIME_S	SYSTIME_NS	TIMER2	TIMER1	TIMER0	SW

Bits	Name	Description	R/W	Default
31:0	VIC_INT_EN_CLR	Clear bits in the VIC_INT_EN register. 0: has no effect. 1: clears the corresponding bit in the VIC_INT_EN register.	W	0x00000000

VIC_SWI – VIC Software Interrupt Register 0x001ff018

The VIC_SWI register is used to generate software interrupts.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	TIMER4	TIMER3	reserved		ISO_AREA	INT_PHY	MSYNC3	MSYNC2	MSYNC1	MSYNC0	COM3	COM2	COM1	COM0	GPIO	HIF	LCD	I2C	SPI	USB	UART2	UART1	UART0	WATCHDOG	GPIO15	SYSTIME_S	SYSTIME_NS	TIMER2	TIMER1	TIMER0	SW

Bits	Name	Description	R/W	Default
31:0	VIC_SWI	Setting a bit generates a software interrupt for the specific source before interrupt masking. Read: 0: Software Interrupt inactive(reset). 1: Software Interrupt active. Write: 0: has no effect. 1: software interrupt enabled.	R/W	0x00000000

VIC_SWI_CLR – VIC Software Interrupt Clear Register 0x001ff01c

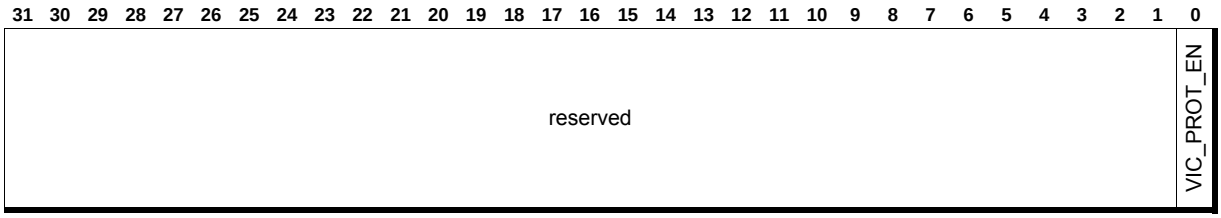
The VIC_SWI_CLR register clears bits in the VIC_SWI register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	TIMER4	TIMER3	reserved		ISO_AREA	INT_PHY	MSYNC3	MSYNC2	MSYNC1	MSYNC0	COM3	COM2	COM1	COM0	GPIO	HIF	LCD	I2C	SPI	USB	UART2	UART1	UART0	WATCHDOG	GPIO15	SYSTIME_S	SYSTIME_NS	TIMER2	TIMER1	TIMER0	SW

Bits	Name	Description	R/W	Default
31:0	VIC_SWI_CLR	Clear corresponding bits in the VIC_SWI register. 0: has no effect. 1: software interrupt disabled in the VIC_SWI register.	W	0x00000000

VIC_PROT_EN – VIC Protection Enable Register 0x001ff020

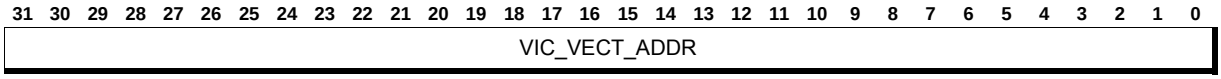
The VIC_PROT_EN register enables or disables protected register access, stopping register accesses when the processor is in User mode.



Bits	Name	Description	R/W	Default
31:1	reserved	-	R	0x00
0	VIC_PROT_EN	0: VIC registers are accessable 1: VIC registers are not accessable	R/W	0

VIC_VECT_ADDR – VIC Vector Address Register 0x001ff030

The VIC_VECT_ADDR register contains the Interrupt Service Routine (ISR) address of the currently active interrupt.

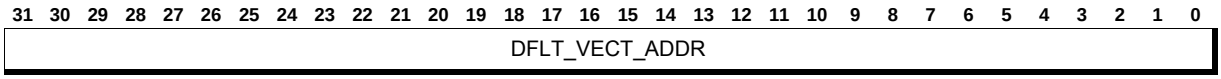


Bits	Name	Description	R/W	Default
31:0	VIC_VECT_ADDR	Contains the address of the currently active ISR. Any writes to this register clear the current interrupt.	R/W	0x00000000

Note:
Reading from this register provides the address of the ISR, and indicates to the priority hardware that the interrupt is being serviced. Writing to this register indicates to the priority hardware that the interrupt has been serviced. The register should be used as follows:
• The ISR reads the VIC_VECT_ADDR register when an IRQ interrupt is generated
• At the end of the ISR, the VIC_VECT_ADDR register is written to, to update the priority hardware.
Reading or writing to the register at other times can cause incorrect operation.

VIC_DFLT_VECT_ADDR – VIC Default Vector Address Register 0x001ff034

The VIC_DFLT_VECT_ADDR register contains the default ISR address.



Bits	Name	Description	R/W	Default
31:0	VIC_DFLT_VECT_ADDR	Contains the address of the default ISR handler.	R/W	0x00000000

VIC_VECT_ADDR0 – VIC Vector Address Registers 0	**0x001ff100**
VIC_VECT_ADDR1 – VIC Vector Address Registers 1	**0x001ff104**
VIC_VECT_ADDR2 – VIC Vector Address Registers 2	**0x001ff108**
VIC_VECT_ADDR3 – VIC Vector Address Registers 3	**0x001ff10c**
VIC_VECT_ADDR4 – VIC Vector Address Registers 4	**0x001ff110**
VIC_VECT_ADDR5 – VIC Vector Address Registers 5	**0x001ff114**
VIC_VECT_ADDR6 – VIC Vector Address Registers 6	**0x001ff118**
VIC_VECT_ADDR7 – VIC Vector Address Registers 7	**0x001ff11c**
VIC_VECT_ADDR8 – VIC Vector Address Registers 8	**0x001ff120**
VIC_VECT_ADDR9 – VIC Vector Address Registers 9	**0x001ff124**
VIC_VECT_ADDR10 – VIC Vector Address Registers 10	**0x001ff128**
VIC_VECT_ADDR11 – VIC Vector Address Registers 11	**0x001ff12c**
VIC_VECT_ADDR12 – VIC Vector Address Registers 12	**0x001ff130**
VIC_VECT_ADDR13 – VIC Vector Address Registers 13	**0x001ff134**
VIC_VECT_ADDR14 – VIC Vector Address Registers 14	**0x001ff138**
VIC_VECT_ADDR15 – VIC Vector Address Registers 15	**0x001ff13c**

The VIC_VECT_ADDR0-15 registers contain the ISR vector addresses. These registers must only be updated when the relevant interrupts are disabled. Receiving an interrupt while the vector address is being written to can result in unpredictable behavior.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VECT_ADDR0-15																															

Bits	Name	Description	R/W	Default
31:0	VIC_VECT_ADDR0-15	Contains ISR vector address.	R/W	0x00000000

VIC_VECT_CTRL0 – VIC Vector Control Registers 0 **0x001ff200**
VIC_VECT_CTRL1 – VIC Vector Control Registers 1 **0x001ff204**
VIC_VECT_CTRL2 – VIC Vector Control Registers 2 **0x001ff208**
VIC_VECT_CTRL3 – VIC Vector Control Registers 3 **0x001ff20c**
VIC_VECT_CTRL4 – VIC Vector Control Registers 4 **0x001ff210**
VIC_VECT_CTRL5 – VIC Vector Control Registers 5 **0x001ff214**
VIC_VECT_CTRL6 – VIC Vector Control Registers 6 **0x001ff218**
VIC_VECT_CTRL7 – VIC Vector Control Registers 7 **0x001ff21c**
VIC_VECT_CTRL8 – VIC Vector Control Registers 8 **0x001ff220**
VIC_VECT_CTRL9 – VIC Vector Control Registers 9 **0x001ff224**
VIC_VECT_CTRL10 – VIC Vector Control Registers 10 **0x001ff228**
VIC_VECT_CTRL11 – VIC Vector Control Registers 11 **0x001ff22c**
VIC_VECT_CTRL12 – VIC Vector Control Registers 12 **0x001ff230**
VIC_VECT_CTRL13 – VIC Vector Control Registers 13 **0x001ff234**
VIC_VECT_CTRL14 – VIC Vector Control Registers 14 **0x001ff238**
VIC_VECT_CTRL15 – VIC Vector Control Registers 15 **0x001ff23c**

VIC_VECT_CTRL0-15 registers select interrupt source and set if it is enabled.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																										ENABLE	INT_SOURCE				

Bits	Name	Description	R/W	Default
31:6	reserved	-	R	0x00
5	ENABLE	Enables vector interrupt. This bit is cleared on reset.	R/W	0
4:0	INT_SOURCE	Select interrupt source. You can select any of the 32 interrupt sources.	R/W	0x00

7 Motion Control Functions

7.1 PWM – Pulse-width modulation for Motion Control

The two identical PWM modules of the netX are directly connected to the communication function blocks and can be accessed by xMAC2 (PWM0 block) and xMAC3 (PWM1 block). When accessing the PWM blocks from the ARM CPU, the corresponding xMAC must be switched off!

The parameter TD, configuring the dead time of the complementary PWM signal pairs, is common to all PWM signals of one block.

The following table lists all registers to PWM functions:

ARM Address	Register Name	Short Description
0x0016295c	PWM0_CFG	PWM 0 Configuration Register
0x0016395c	PWM1_CFG	PWM 1 Configuration Register
0x00162960	PWM0_STAT	PWM 0 Status Register
0x00163960	PWM1_STAT	PWM 1 Status Register
0x00162964	PWM0_TP	PWM 0 Period
0x00163964	PWM1_TP	PWM 1 Period
0x00162968	PWM0_TU	PWM 0 Channel U Low Phase Width
0x00163968	PWM1_TU	PWM 1 Channel U Low Phase Width
0x0016296c	PWM0_TV	PWM 0 Channel V Low Phase Width
0x0016396c	PWM1_TV	PWM 1 Channel V Low Phase Width
0x00162970	PWM0_TW	PWM 0 Channel W Low Phase Width
0x00163970	PWM1_TW	PWM 1 Channel W Low Phase Width
0x00162974	PWM0_TD	PWM 0 Dead Time Counter Preload
0x00163974	PWM1_TD	PWM 1 Dead Time Counter Preload
0x00162980	PWM0_CNT	Actual Counter Motor PWM 0 Period
0x00163980	PWM1_CNT	Actual Counter Motor PWM 1 Period
0x00162988	PWM0_STRTIME	Captured System Time at Start Point of Motor PWM 0 Period
0x00163988	PWM1_STRTIME	Captured System Time at Start Point of Motor PWM 1 Period
0x00162978	RPWM0_TP	Resolver PWM 0 Period
0x00163978	RPWM1_TP	Resolver PWM 1 Period
0x0016297c	RPWM0_TR	Resolver PWM 0 Pulse
0x0016397c	RPWM1_TR	Resolver PWM 1 Pulse
0x00162984	RPWM0_CNT	Actual Counter Resolver PWM 0 Period
0x00163984	RPWM1_CNT	Actual Counter Resolver PWM 1 Period
0x0016298c	RPWM0_STRTIME	Captured System time at Start Point of Resolver PWM 0 Period
0x0016398c	RPWM1_STRTIME	Captured System time at Start Point of Resolver PWM 1 Period

PWM0_CFG – PWM 0 Configuration Register **0x0016295c**
PWM1_CFG – PWM 1 Configuration Register **0x0016395c**

This register allows to set and reset outputs manually or via counters.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																RUN_RP	RUN_MP	reserved			AUTO_W	AUTO_V	AUTO_U	PWM_MODE	FO_MODE	VAL_WN	VAL_W	VAL_VN	VAL_V	VAL_UN	VAL_U

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15	RUN_RP	0 : reset/stop Resolver PWM immediately 1 : run Resolver PWM	R/W	0
14	RUN_MP	0 : stop Motor PWM after the actual cycle 1 : run Motor PWM	R/W	0
13:11	reserved	-	R	000
10	AUTO_W	0 : set gpio_pin[4] to VAL_W set gpio_pin[5] to VAL_WN 1 : set gpio_pin[4] to output of channel W set gpio_pin[5] to output of channel WN	R/W	0
9	AUTO_V	0 : set gpio_pin[2] to VAL_V set gpio_pin[3] to VAL_VN 1 : set gpio_pin[2] to output of channel V set gpio_pin[3] to output of channel VN	R/W	0
8	AUTO_U	0 : set gpio_pin[0] to VAL_U set gpio_pin[1] to VAL_UN 1 : set gpio_pin[0] to output of channel U set gpio_pin[1] to output of channel UN	R/W	0
7	PWM_MODE	Motor PWM Mode: 0 : off 1 : on PWM Mode has a higher priority as Fiber Optic Mode	R/W	0
6	FO_MODE	Fiber Optic Mode: 0 : off 1 : on	R/W	0
5	VAL_WN	Motor PWM Mode: see bit AUTO_W Fiber Optic Mode: Value of gpio_pins[5] if serial output date = 1	R/W	0
4	VAL_W	Motor PWM Mode: see bit AUTO_W Fiber Optic Mode: Value of gpio_pins[4] if serial output date = 1	R/W	0
3	VAL_VN	Motor PWM Mode: see bit AUTO_V Fiber Optic Mode: Value of gpio_pins[3] if serial output date = 1	R/W	0
2	VAL_V	Motor PWM Mode: see bit AUTO_V Fiber Optic Mode:	R/W	0

		Value of gpio_pins[2] if serial output date = 1		
1	VAL_UN	Motor PWM Mode: see bit AUTO_U Fiber Optic Mode: Value of gpio_pins[1] if serial output date = 1	R/W	0
0	VAL_U	Motor PWM Mode: see bit AUTO_U Fiber Optic Mode: Value of gpio_pins[0] if serial output date = 1	R/W	0

PWM0_STAT – PWM 0 Status Register **0x00162960**
PWM1_STAT – PWM 1 Status Register **0x00163960**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																							SYSTIME_REQ	PWMERR_REQ	PWMERR	reserved					SYSTC

Bits	Name	Description	R/W	Default
31:9	reserved	-	R	0x00
8	SYSTIME_REQ	System time captured at RPWM = 0 Write a 1 at this position to clear this bit.	R/W	0
7	PWMERR_REQ	Latched input signal FAILURE Write a 1 at this position to clear this bit.	R/W	0
6	PWMERR	Actual input signal FAILURE Write a 1 at this position to clear this bit.	R/W	0
5:1	reserved	-	R	0x00
0	SYSTC	System time captured at beginning of motor PWM Write a 1 at this position to clear this bit.	R/W	0

PWM0_TP - PWM 0 Period **0x00162964**
PWM1_TP - PWM 1 Period **0x00163964**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																TP															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	TP	Length of PWM Period (bit 0 unused due to triangular counting) in clock cycles.	R/W	0

PWM0_TU – PWM 0 Channel U Low Phase Width **0x00162968**
PWM1_TU – PWM 1 Channel U Low Phase Width **0x00163968**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																LEVEL_U															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	LEVEL_U	Width of cannel U low phase in clock cycles.	R/W	0

PWM0_TV – PWM 0 Channel V Low Phase Width **0x0016296c**
PWM1_TV – PWM 1 Channel V Low Phase Width **0x0016396c**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																LEVEL_V															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	LEVEL_V	Width of cannel V low phase in clock cycles.	R/W	0

PWM0_TW – PWM 0 Channel W Low Phase Width **0x00162970**
PWM1_TW – PWM 1 Channel W Low Phase Width **0x00163970**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																LEVEL_W															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	LEVEL_W	Width of cannel W low phase in clock cycles.	R/W	0

PWM0_TD – PWM 0 Dead Time Counter Preload **0x00162974**
PWM1_TD – PWM 1 Dead Time Counter Preload **0x00163974**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																PRELOAD_D															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	PRELOAD_D	counter preload for dead time in clock cycles.	R/W	0

PWM0_CNT – Current Counter Motor PWM 0 Period **0x00162980**
PWM1_CNT – Current Counter Motor PWM 1 Period **0x00163980**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																PWM_CNT															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	PWM_CNT	Current counter Motor PWM Period	R/W	0

PWM0_STRTIME – Captured System Time at Start Point of Motor PWM 0 Period **0x00162988**
PWM1_STRTIME – Captured System Time at Start Point of Motor PWM 1 Period **0x00163988**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																PWM_STRTIME															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	PWM_STRTIME	Captured Systime at start point of Motor PWM Period	R/W	0

RPWM0_TP – Resolver PWM 0 Period **0x00162978**
RPWM1_TP – Resolver PWM 1 Period **0x00163978**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																RES_TP															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	RES_TP	Resolver PWM Period	R/W	0

RPWM0_TR – Resolver PWM 0 Pulse **0x0016297c**
RPWM1_TR – Resolver PWM 1 Pulse **0x0016397c**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																RES_TR															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	RES_TR	Resolver PWM Pulse	R/W	0

RPWM0_CNT – Current Counter Resolver PWM 0 Period **0x00162984**
RPWM1_CNT – Current Counter Resolver PWM 1 Period **0x00163984**

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																RPWM_CNT															

Bits	Name	Description	R/W	Default
31:16	Reserved	-	R	0x00
15:0	RPWM_CNT	Current counter Resolver PWM Period	R/W	0

RPWM0_STRTIME – Captured System time at Start Point of Resolver PWM 0 Period 0x0016298c
RPWM1_STRTIME – Captured System time at Start Point of Resolver PWM 1 Period 0x0016398c

This register has a write pipeline delay of 1 clock cycle.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																RPWM_STRTIME															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	RPWM_STRTIME	Captured System time at start point of Resolver PWM Period	R/W	0

7.2 ENC – Quadrature Encoders

The following registers control the two Quadrature Encoder input blocks of the netX, which can be accessed from xMAC3 (or from the ARM CPU, if xMAX3 is switched off).

The following table lists all Encoder related signals.

ARM Address	Register Name	Short Description
0x0016399c	ENC_STAT	Encoder Position and Capture Status
0x00163990	ENC_CFG	Encoder Configuration Register
0x00163998	ENC_CMD	Encoder Command Register
0x001639a0	ENC0_POS	Actual Position Encoder 0
0x001639a8	ENC1_POS	Actual Position Encoder 1
0x001639a4	ENC0_NULL_POS	Sampled Null Position Encoder 0
0x001639ac	ENC1_NULL_POS	Sampled Null Position Encoder 1
0x001639b0	ENC0_EDGE_TIME	System Time at Last Edge of Encoder 0
0x001639b4	ENC1_EDGE_TIME	System Time at Last Edge of Encoder 1
0x00163994	ENC_CFG_CAPT	Encoder Capture Configuration Register
0x001639b8	ENC_CAPT0	Encoder Capture Register 0
0x001639bc	ENC_CAPT1	Encoder Capture Register 1
0x001639c0	ENC_CAPT2	Encoder Capture Register 2
0x001639c4	ENC_CAPT3	Encoder Capture Register 3

ENC_STAT – Encoder Position and Capture Status 0x0016399c

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																MP1	MP0	ENC1_SIGN	ENC0_SIGN	CAP3	CAP2	CAP1	CAP0	ENC1_CAP_ETIME	ENC1_CAP_ZPOS	ENC1_OVFL_NEG	ENC1_OVFL_POS	ENC0_CAP_ETIME	ENC0_CAP_ZPOS	ENC0_OVFL_NEG	ENC0_OVFL_POS

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15	MP1	Measurement Point 1 To clear this bit write a 1 at this position.	R/W	0
14	MP0	Measurement Point 0 To clear this bit write a 1 at this position.	R/W	0
13	ENC1_SIGN	Encoder 1 signal N To clear this bit write a 1 at this position.	R/W	0
12	ENC0_SIGN	Encoder 0 signal N To clear this bit write a 1 at this position.	R/W	0
11	CAP3	Captured register 3 To clear this bit write a 1 at this position.	R/W	0
10	CAP2	Captured register 2 To clear this bit write a 1 at this position.	R/W	0
9	CAP1	Captured register 1 To clear this bit write a 1 at this position.	R/W	0
8	CAP0	Captured register 0 To clear this bit write a 1 at this position.	R/W	0
7	ENC1_CAP_ETIME	Encoder1 captured edge time To clear this bit write a 1 at this position.	R/W	0
6	ENC1_CAP_ZPOS	Encoder1 captured null position To clear this bit write a 1 at this position.	R/W	0
5	ENC1_OVFL_NEG	Encoder1 overflow negative To clear this bit write a 1 at this position.	R/W	0
4	ENC1_OVFL_POS	Encoder1 overflow positive To clear this bit write a 1 at this position.	R/W	0
3	ENC0_CAP_ETIME	Encoder0 captured edge time To clear this bit write a 1 at this position.	R/W	0
2	ENC0_CAP_ZPOS	Encoder0 captured null position To clear this bit write a 1 at this position.	R/W	0
1	ENC0_OVFL_NEG	Encoder0 overflow negative To clear this bit write a 1 at this position.	R/W	0
0	ENC0_OVFL_POS	Encoder0 overflow positive To clear this bit write a 1 at this position.	R/W	0

ENC_CFG – Encoder Configuration Register 0x00163990

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																		ENC1_N_QUALIFIER		ENC1_COUNT_DIR		ENC1_FILTER_SR			reserved			ENC0_N_QUALIFIER		ENC0_COUNT_DIR		ENC0_FILTER_SR	

Bits	Name	Description	R/W	Default
31:14	reserved	-	R	0x00
13:12	ENC1_N_QUALIFIER	Encoder 1 N-qualifier 0 : none, sampled with every N pulse 1 : N-Signal channel 1 = 1 2 : MP0 = 1 3 : MP1 = 1	R/W	00
11	ENC1_COUNT_DIR	Encoder 1 count direction # 0 0: count up 1: count down	R/W	0
10:8	ENC1_FILTER_SR	Encoder 1 filter sample rate 0 : none 1 : 10 ns 2 : 20 ns 3 : 50 ns 4 : 100 ns 5 : 200 ns 6 : 500 ns 7 : 1 us	R/W	000
7:6	reserved	-	R/W	00
5:4	ENC0_N_QUALIFIER	Encoder 0 N-qualifier 0 : none, sampled with every N pulse 1 : N-Signal channel 1 = 1 2 : MP0 = 1 3 : MP1 = 1	R/W	00
3	ENC0_COUNT_DIR	Encoder 0 count direction 0 : count up 1 : count down	R/W	0
2:0	ENC0_FILTER_SR	Encoder 0 filter sample rate 0 : none 1 : 10 ns 2 : 20 ns 3 : 50 ns 4 : 100 ns 5 : 200 ns 6 : 500 ns 7 : 1 us	R/W	000

ENC_CMD – Encoder Command Register 0x00163998

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																					ENC1_RES_ZPOS	ENC1_RES_POS	ENC1_EN	reserved					ENC0_RES_ZPOS	ENC0_RES_POS	ENC0_EN

Bits	Name	Description	R/W	Default
31:11	reserved	-	R	0x00
10	ENC1_RES_ZPOS	Encoder1 reset null position	R/W	0
9	ENC1_RES_POS	Encoder1 reset position	R/W	0
8	ENC1_EN	Encoder1 enable	R/W	0
7:3	reserved	-	R	0x00
2	ENC0_RES_ZPOS	Encoder0 reset null position	R/W	0
1	ENC0_RES_POS	Encoder0 reset position	R/W	0
0	ENC0_EN	Encoder0 enable	R/W	0

ENC0_POS - Actual Position Encoder 0 0x001639a0
ENC1_POS - Actual Position Encoder 1 0x001639a8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																POSITION															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	POSITION	Actual position encoder	R/W	0

ENC0_NULL_POS - Sampled Null Position Encoder 0 0x001639a4
ENC1_NULL_POS - Sampled Null Position Encoder 1 0x001639ac

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																NULLPOSITION															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	NULLPOSITION	Last null position of encoder	R	0

ENC0_EDGE_TIME - System Time at Last Edge of Encoder 0 **0x001639b0**
ENC1_EDGE_TIME - System Time at Last Edge of Encoder 1 **0x001639b4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																EDGE_TIME															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	EDGE_TIME	System time at last edge of encoder	R	0

ENC_CFG_CAPT – Encoder Capture Configuration Register 0x00163994

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																CREG3_SRC		CREG3_CW		CREG2_SRC		CREG2_CW		CREG1_SRC		CREG1_CW		CREG0_SRC		CREG0_CW	

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:14	CREG3_SRC	Encoder capture register 3, source 0 : none 1 : system time ns 2 : position channel 0 3 : position channel 1	R/W	00
13:12	CREG3_CW	Encoder capture register 3, capture with 0 : mp0 positive edge 1 : mp0 negative edge 2 : mp1 positive edge 3 : mp1 negative edge	R/W	00
11:10	CREG2_SRC	Encoder capture register 2, source 0 : none 1 : system time ns 2 : position channel 0 3 : position channel 1	R/W	00
9:8	CREG2_CW	Encoder capture register 2, capture with 0 : mp0 positive edge 1 : mp0 negative edge 2 : mp1 positive edge 3 : mp1 negative edge	R/W	00
7:6	CREG1_SRC	Encoder capture register 1, source 0 : none 1 : system time ns 2 : position channel 0 3 : position channel 1	R/W	00
5:4	CREG1_CW	Encoder capture register 1, capture with 0 : mp0 positive edge 1 : mp0 negative edge 2 : mp1 positive edge 3 : mp1 negative edge	R/W	00
3:2	CREG0_SRC	Encoder capture register 0, source 0 : none 1 : system time ns 2 : position channel 0 3 : position channel 1	R/W	00
1:0	CREG0_CW	Encoder capture register 0, capture with 0 : mp0 positive edge 1 : mp0 negative edge 2 : mp1 positive edge 3 : mp1 negative edge	R/W	00

ENC_CAPT0 – Encoder Capture Register 0 0x001639b8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																ENC_CAPT															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	ENC_CAPT	Encoder capture register 0	R	0

ENC_CAPT1 – Encoder Capture Register 1 0x001639bc

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																ENC_CAPT1															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	ENC_CAPT1	Encoder capture register 1	R	0

ENC_CAPT2 – Encoder Capture Register 2 0x001639c0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																ENC_CAPT2															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	ENC_CAPT2	Encoder capture register 2	R	0

ENC_CAPT3 – Encoder Capture Register 3 0x001639c4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																ENC_CAPT3															

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:0	ENC_CAPT3	Encoder capture register 3	R	0

7.3 ADC – Analog Digital Converters

The ADC unit is controlled by a single register, which has different bit definitions for read and write accesses and is accessible at four different addresses, whereas each address is assigned to a specific xPEC channel (addresses are assigned in ascending order from xPEC0 to xPEC3).
When accessing the ADC register from the ARM CPU instead of an xPEC, any of the four register addresses can be used, however the corresponding xPEC unit must then be switched off!

Please note, that the external channels ADCx_IN0 to ADCx_IN3 correspond to the internal channel numbers 0, 2, 4 and 6 of ADC0 and 1 respectively (e.g. when using analog input ADC1_IN2, the ADC1_SEL parameter must be set to 0x4).

ADC – Analog Digital Converter **0x0017009c, 0x0017409c, 0x0017809c, 0x0017c09c**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
read access :																															
Reserved					ADC1_BUSY	ADC1_D[9:0]										reserved					ADC0_BUSY	ADC0_D[9:0]									
write access :																															
DONT_WRITE_ADC1	reserved													ADC1_SEL[2:0]		ADC1_START	ADC1_EN	DONT_WRITE_ADC0	reserved								ADC0_SEL[2:0]			ADC0_START	ADC0_EN

Bits	Name	Description	R/W	Default
read access :				
31:27	Reserved	-	R	0
26	ADC1_BUSY	ADC1 is busy. Bit must be polled till cleared to determine end of conversion	R	0
25:16	ADC1_D[9:0]	Sample value of ADC1	R	0
15:11	Reserved	-	R	0
10	ADC0_BUSY	ADC0 is busy. Bit must be polled till cleared to determine end of conversion	R	0
9:0	ADC0_D[9:0]	Sample value of ADC0	R	0
write access :				
31	DONT_WRITE_ADC1	Bit must be set, to configure ADC**0**.	W	-
30:21	Reserved	-	W	-
20:18	ADC1_SEL[2:0]	Channel selection of ADC1. -> only channels 0, 2, 4 and 6 are wired (correspond to channels 0-3 externally)	W	-
17	ADC1_START	Start conversion of ADC1	W	-
16	ADC1_EN	ADC1 unit enable	W	-
15	DONT_WRITE_ADC0	Bit must be set, to configure ADC**1**.	W	-

14:5	Reserved	-	W	-
4:2	ADC0_SEL[2:0]	Channel selection of ADC0. -> only channels 0, 2, 4 and 6 are wired (correspond to channels 0-3 externally)	W	
1	ADC0_START	Start conversion of ADC0	W	-
0	ADC0_EN	ADC0 unit enable	W	-

8 COMMUNICATION FUNCTIONS

The following table is a summary of registers related to communication functions.

ARM Address	Register Name	Short Description
0x00100010	PHY_CTRL	PHY Control Register
0x00100020	PHY_CLK_RATE_MUL_ADD	PHY Clock Rate Multiplier Add Value
0x00164000	PTR_FIFO_BASE	Pointer FIFO Base Address
0x00164080	PTR_FIFO_BOR_BASE	Pointer FIFO Upper Border Base Address
0x00164100	PTR_FIFO_RESET	Pointer FIFO Reset Vector
0x00164104	PTR_FIFO_FULL	Pointer FIFO Full Vector
0x00164108	PTR_FIFO_EMPTY	Pointer FIFO Empty Vector
0x0016410c	PTR_FIFO_OVER	Pointer FIFO Overflow Vector
0x00164110	PTR_FIFO_UNDER	Pointer FIFO Under-Run Vector
0x00164180	PTR_FIFO_FILL_LVL_BASE	Pointer FIFO Fill Level Base Address
0x00101000	CRC_VAL	Calculated CRC Value
0x00101004	CRC_IN_DATA	CRC Input Data
0x00101008	CRC_POLYNOMIAL	Polynomial for CRC Calculation
0x0010100c	CRC_CFG	CRC Configuration Register
0x00164400	IRQ_XP0	IRQs between XPEC0 and ARM
0x00164404	IRQ_XP1	IRQs between XPEC1 and ARM
0x00164408	IRQ_XP2	IRQs between XPEC2 and ARM
0x0016440c	IRQ_XP3	IRQs between XPEC3 and ARM

8.1 PHY – Controller for internal PHYs

PHY_CTRL - PHY Control Register **0x00100010**

This register contains all static connectors of the NEC Ethernet Phy. Usually the Phy read these values only during reset, which can be controlled by Bit31. Changing this register is only possible by the following sequence:

1: read out access key
2: write back access key
3: write desired value to this register

In total the programming sequence should be:

a: read access key, write access key, write new value with bit phy_reset=1
b: wait for synchronization(~16cc) and proper reset of phy(~40cc ? better check NEC document)
c: read access key, write access key, write new value with bit phy_reset=0

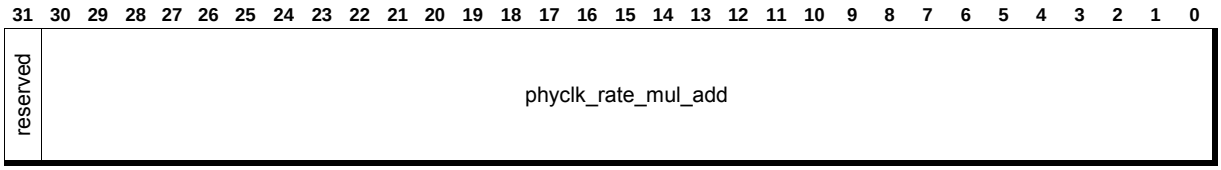
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

PHY_RESET	PHY_SIM_BYP	PHY_CLK_XLATIN	reserved	PHY1_ENABLE	PHY1_NP_MSG_CODE	PHY1_AUTOMDIX	PHY1_FIXMODE	PHY1_MODE	PHY0_ENABLE	PHY0_NP_MSG_CODE	PHY0_AUTOMIX	PHY0_FIXMODE	PHY0_MODE	PHY_ADDRESS

Bits	Name	Description	R/W	Default
31	PHY_RESET	Hardware reset for PHY 1: reset	R/W	0
30	PHY_SIM_BYP	PHY Power up Bypass (only used for simulation issues) 0: normal. 1: bypass.	R/W	0
29	PHY_CLK_XLATIN	0: phyclk_rate_mul_add (see below) for clock Ethernet PHY. 1: external oscillator input (25MHz) for clock Ethernet PHY.	R/W	1
28:22	reserved	-	R	0x00
21	PHY1_ENABLE	PHY1 enable	R/W	0
20:18	PHY1_NP_MSG_CODE	PHY1 Next Page Message Code (auto negotiation)	R/W	0x00
17	PHY1_AUTOMDIX	PHY1 Enables AutoMDIX state machine	R/W	0
16	PHY1_FXMODE	PHY1 100BASE-FX mode (phy_mode must be 01x)	R/W	0
15:13	PHY1_MODE	PHY1 Mode 000: 10BASE-T Half Duplex, Auto Negotiation disabled. 001: 10BASE-T Full Duplex. Auto-Negotiation disabled. 010: 100BASE-TX/FX Half Duplex. Auto-Negotiation disabled. CRS is active during Transmit & Receive. 011: 100BASE-TX/FX Full Duplex. Auto-Negotiation disabled. CRS is active during Receive. 100: 100BASE-TX Half Duplex is advertised. Auto-Negotiation enabled. CRS is active during Transmit & Receive. 101: Repeater mode. Auto-Negotiation enabled. 100BASETX Half Duplex is advertised. CRS is active during Receive. 110: Power Down mode. In this mode the PHY wake-up in Power-Down mode. 111: All capable. Auto-Negotiation enabled. AutoMDIX enabled.	R/W	110
12	PHY0_ENABLE	PHY0 enable	R/W	0
11:9	PHY0_NP_MSG_CODE	PHY0 Next Page Message Code (auto negotiation)	R/W	0x00
8	PHY0_AUTOMDIX	PHY0 Enables AutoMDIX state machine	R/W	0
7	PHY0_FXMODE	PHY0 100BASE-FX mode (phy_mode must be 01x)	R/W	0
6:4	PHY0_MODE	PHY0 Mode 000: 10BASE-T Half Duplex, Auto Negotiation disabled. 001: 10BASE-T Full Duplex. Auto-Negotiation disabled. 010: 100BASE-TX/FX Half Duplex. Auto-Negotiation disabled. CRS is active during Transmit & Receive. 011: 100BASE-TX/FX Full Duplex. Auto-Negotiation disabled. CRS is active during Receive. 100: 100BASE-TX Half Duplex is advertised. Auto-Negotiation enabled. CRS is active during Transmit & Receive. 101: Repeater mode. Auto-Negotiation enabled. 100BASETX Half Duplex is advertised. CRS is active during Receive. 110: Power Down mode. In this mode the PHY wake-up in Power-Down mode. 111: All capable. Auto-Negotiation enabled. AutoMDIX enabled.	R/W	110
3:0	PHY_ADDRESS	Bits 4:1 of phy mdio-address. Bit0 defines 1st or 2nd internal PHY	R/W	0x00

PHY_CLK_RATE_MUL_ADD - PHY Clock Rate Multiplier Add Value 0x00100020

This register contains the phyclk_rate_mul_add value. Changing this register is only possible by the following sequence:
1: read out access key
2: write back access key
3: write desired value to this register



Bits	Name	Description	R/W	Default
31	reserved	-	R/W	0
30:0	phyclk_rate_mul_add	This value is added each clk200 cycle to phyclk_rate_mul to generate phyclk. Change value according to formula: phyclk_rate_mul_add = [freq in MHz] / 200 * 2^31	R/W	0x10000000

8.2 PTR_FIFO – Pointer FIFO

The following registers are related to the Ethernet communication between the ARM CPU and the xPEC/xMAC which is realized over a special Memory with hardware supported Pointer FIFOs.

For further information, please refer to the netX_Product_Brief.

PTR_FIFO_BASE[0-31] - Pointer FIFO 0-31 Base Address 0x00164000-0x0016407c

The PTR_FIFO_BASE register provides the write/read interface for data into/out of the corresponding pointer FIFO.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIFO_DATA																															

Bits	Name	Description	R/W	Default
31:0	FIFO_DATA	Write access: write data to FIFO, read access: read data from FIFO	R/W	0x0

PTR_FIFO_BOR_BASE[0-31] - Pointer FIFO0-31 Upper Border 0x00164080-0x001640fc

Each address accesses the upper border of the appropriate FIFO in a 2048x32 bit RAM.
All upper borders should be ascending with the FIFO number. If a border between two FIFOs is moved, the adjacent FIFOs should be reset first.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR_FIFO_BOR_BASE																															

Bits	Name	Description	R/W	Default
31:0	PTR_FIFO_BOR_BASE	Last address of RAM used by appropriate FIFO = (first address – 1) of next FIFO FIFO 0 default depth : 1024 FIFO 1..30 default depth : 32 FIFO 31 default depth : 64	R/W	0x0

PTR_FIFO_RESET – Pointer FIFO Reset Vector 0x00164100

Pointer FIFO reset vector. Allows to reset each of the 32 FIFOs, i.e. set read and write pointer to lower border of FIFO, reset full, overflow, under run and set empty flag. Reset flag of adjacent FIFOs should be set before resizing the FIFO.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR_FIFO_RESET																															

Bits	Name	Description	R/W	Default
31:0	PTR_FIFO_RESET	Reset Vector, 1 bit per FIFO: 1: reset FIFO, 0: normal work mode	R/W	0x0

PTR_FIFO_FULL – Pointer FIFO Full Vector 0x00164104

Pointer FIFO full vector. Shows the FIFO_FULL flag of each FIFO.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR_FIFO_FULL																															

Bits	Name	Description	R/W	Default
31:0	PTR_FIFO_FULL	FIFO full vector, 1 bit per FIFO	R	0x0

PTR_FIFO_EMPTY – Pointer FIFO Empty Vector 0x00164108

Pointer FIFO empty vector. Shows the FIFO_EMPTY flag of each FIFO.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR_FIFO_EMPTY																															

Bits	Name	Description	R/W	Default
31:0	PTR_FIFO_EMPTY	FIFO empty vector, 1 bit per FIFO	R	0xffffffff

PTR_FIFO_OVER – Pointer FIFO Overflow Vector 0x0016410c

Pointer FIFO overflow vector. Shows the FIFO_OVERFLOW flag of each FIFO.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR_FIFO_OVER																															

Bits	Name	Description	R/W	Default
31:0	PTR_FIFO_OVER	FIFO overflow vector, 1 bit per FIFO	R	0x0

PTR_FIFO_UNDER – Pointer FIFO Under Run Vector 0x00164110

Pointer FIFO under run vector. Shows the FIFO_UNDERRUN flag of each FIFO.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR_FIFO_UNDER																															

Bits	Name	Description	R/W	Default
31:0	PTR_FIFO_UNDER	FIFO under run vector, 1 bit per FIFO	R	0x0

PTR_FIFO_FILL_LVL_BASE[0-31] – Pointer FIFO Fill Level 0-31 0x00164180-0x001641fc

Pointer FIFO fill-level table. Each of the following 32 addresses reads the FILL_LEVEL of the appropriate FIFO.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																					PTR_FIFO_FILL_LVL										

Bits	Name	Description	R/W	Default
31:11	reserved	-	R	0x00
10:0	PTR_FIFO_FILL_LVL	Actual number of words in appropriate FIFO (not valid, if FIFO had an overflow or underflow)	R	0x00

8.3 CRC – Configurable CRC-Generator

CRC_VAL – CRC Register 0x00101000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_VAL																															

Bits	Name	Description	R/W	Default
31:0	CRC_VAL	CRC value Upper bits should be masked, if crc_len smaller than 31	R/W	0x00

CRC_IN_DATA – CRC Input Data register 0x00101004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								CRC_IN_DATA							

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:0	CRC_IN_DATA	CRC input data	R/W	0x00

CRC_POLYNOMIAL – CRC Polynomial Register 0x00101008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_POLYNOMINAL																															

Bits	Name	Description	R/W	Default
31:0	CRC_POLYNOMINAL	CRC polynomial Most significant bit of polynomial is always one, thus not considered. Default: Ethernet CRC 32.	R/W	0x4c11db7

CRC_CFG – CRC Configuration Register 0x0010100c

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																					CRC_IN_MSB_LOW	CRC_NOF_BITS		CRC_DIRECT_DIV	CRC_SHIFT_DIV	CRC_LEN					

Bits	Name	Description	R/W	Default
31:11	reserved	-	R	0x00
10	CRC_IN_MSB_LOW	Swap crc_data_in, only usefull when calculating multiple bits in parallel (crc_nof_bits > 0): 1: msb of incoming bits is data_in[0], 0: msb is data_in[crc_nof_bits_m1] (msb=first bit in data-stream)	R/W	0
9:8	CRC_NOF_BITS	Number of bits to be calculated in parallel 00: 1 01: 2 10: 4 11: 8	R/W	00
7	CRC_DIRECT_DIV	1: calculate direct polynomial division without n zeros after frame, useful for parity calculation 0 : disable direct division	R/W	0
6	CRC_SHIFT_DIV	1 : shift CRC right 0 : shift CRC left	R/W	0
5:0	CRC_LEN	Length of CRC Polynom – 1	R/W	0x00

8.4 ARM_to_XPEC_IRQ

The following four registers indicate the interrupt requests between xPECs and ARM.

IRQ_XP0 – IRQs between XPEC0 and ARM Registers **0x00164400**
IRQ_XP1 – IRQs between XPEC1 and ARM Registers **0x00164404**
IRQ_XP2 – IRQs between XPEC2 and ARM Registers **0x00164408**
IRQ_XP3 – IRQs between XPEC3 and ARM Registers **0x0016440c**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARM_IRQ																XPEC_IRQ															

Bits	Name	Description	R/W	Default
31:16	ARM_IRQ	set by arm ; reset by xpec	R/W	0x00
15:0	XPEC_IRQ	set by xpec ; reset by arm	R/W	0x00

9 ARM System Control and Configuration Registers

These registers which are called CP15 registers are accessible using special ARM instructions.

The Caches, Tightly-Coupled Memory (TCM), Memory Management Unit (MMU) and most other system options are controlled by using CP15 registers.

For more details about the Caches, TCM, MMU and other system options see ARM926EJ-S Technical Reference Manual.

9.1 Addresses in an ARM926EJ-S System

- Virtual address (VA) in the ARM926EJ-S domain
- Modified virtual address (MVA) in the Cache and MMU domain
- Physical address (PA) in the AMBA domain

Example

This is an example of the address manipulation that occurs when the ARM926EJ-S core requests an instruction:

- The ARM926EJ-S core issues the virtual address of the instruction.
- The virtual address is translated using the FCSE PID (fast context switch extension process ID) value to the modified virtual address. The instruction cache (ICache) and memory management unit (MMU) find the modified virtual address (see "c13: Process ID register").
- If the protection check carried out by the MMU on the modified virtual address does not abort and the modified virtual address tag is in the ICache, the instruction data is returned to the ARM926EJ-S core. If the protection check carried out by the MMU on the modified virtual address does not abort but the cache misses (the MVA tag is not in the cache), the MMU translates the modified virtual address to produce the physical address. This address is given to the AMBA bus interface to perform an external access.

9.2 Accessing CP15 Registers

You can only access CP15 registers with MRC and MCR instructions in a privileged mode. CDP, LDC, STC, MCRR, and MRRC instructions, and unprivileged MRC or MCR instructions to CP15 cause the Undefined instruction exception to be taken.

The mnemonics for these instructions are:

```
MCR{cond} p15,opcode_1,<Rd>,CRn,CRm,opcode_2
MRC{cond} p15,opcode_1,<Rd>,CRn,CRm,opcode_2
```

If you try to read from a write-only register or write to a read-only register, you will have unpredictable results. In all instructions that access CP15:

The opcode_1 field should be zero, except when the values specified are used to select the operations you want. Using other values results in unpredictable behaviour.

The opcode_2 and CRm fields should be zero, except when the values specified are used to select the behaviour you want. Using other values results in unpredictable behaviour.

9.3 DTCM Address Space

The internal ARM data Tightly Coupled Memory (DTCM) is mapped to the physical address space from 0x10000000 to 0x10001fff (8kB).

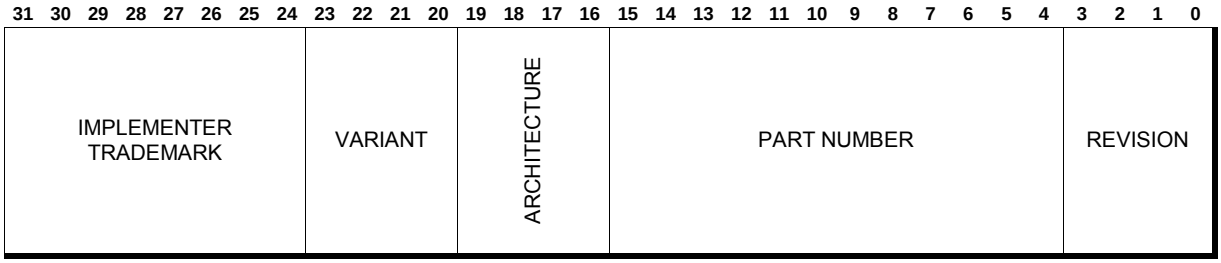
9.4 Register Description

The following registers are described in this section:

c0 ID Code Register, Cache Type Register, TCM Status Register
c1 Control Register
c2 Translation Table Base Register
c3 Domain Access Control Register
c5 Data Fault Status Register, Instruction Fault Status Register
c6 Fault Address Register
c7 Cache Operations Register
c8 TLB Operations Register
c9 Cache Lockdown Register, TCM Region Register
c10 TLB Lockdown Register
c13 FCSE Process ID Register, Context ID Register

Note:
c4, c11, c12 and c14 are reserved.

c0 – ID Code Register



Bits	Name	Description	R/W	Default
31:24	IMPLEMENTER TRADEMARK	ASCII code of implementer trademark	R	0x41
23:20	VARIANT	Variant	R	0x00
19:16	ARCHITECTURE	Architecture (ARMv5TEJ)	R	0x6
15:4	PART NUMBER	Part number	R	0x926
3:0	REVISION	Revision The revision value can be in the range 0x0 to 0x5, depending on the layout revision you are using..	R	0x03

This is a read-only register that returns the 32-bit device ID code.

You can access the ID Code Register by reading CP15 register c0 with the Opcode_2 field set to any value other than 1 or 2. For example:

```
MRC p15,0,<Rd>,c0,c0,0     ;returns ID
```

c0 – Cache Type Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved			CTYPE				S	DSIZE[11:0]												ISIZE[11:0]											

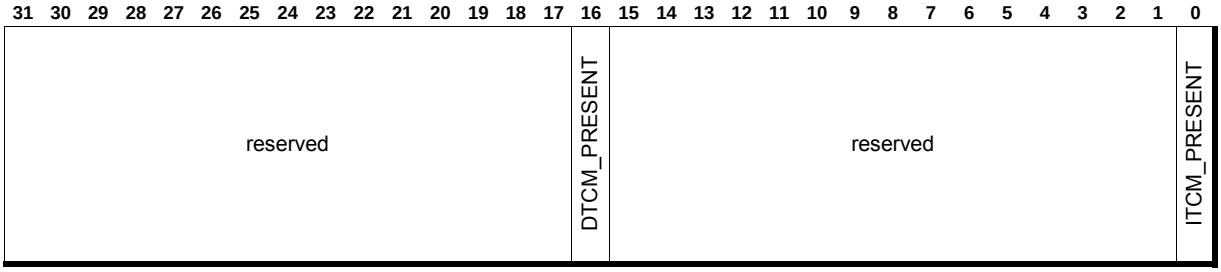
Bits	Name	Description	R/W	Default
31:29	reserved	-	-	000
28:25	CTYPE	Determines the cache type, and specifies whether the cache supports lockdown and how it is cleaned. Ctype encoding is shown below; all unused values are reserved. Value: 0b1110 Method: Writeback Cache cleaning: Register c7 operations (see "c7: Cache Operations register") Cache lockdown: Format C (see "c9: Cache Lockdown register")	R	1110
24	S	Specifies if the cache is a unified cache (S=0), or separate ICache and DCache (S=1). If S=0, the Isize and Dsize fields both describe the unified cache and must be identical. In the ARM926EJ-S processor, this bit is set to a 1 to denote separate caches.	R	1
23:12	DSIZE[11:0]	Specifies the size, line length, and associativity of the DCache Size: 8 KByte Line length : 32bit per line associativity : 4-way	R	000100010010
11:0	ISIZE[11:0]	Specifies the size, length, and associativity of the ICache. Size: 16 KByte Line length : 32 bit per line associativity : 4-way	R	000101010010

This is a read-only register that contains information about the size and architecture of the Instruction Cache (ICache) and Data Cache (DCache) enabling operating systems to establish how to perform such operations as cache cleaning and lockdown.

You can access the cache type register by reading CP15 register c0 with the Opcode_2 field set to 1. For example:

```
MRC p15,0,<Rd>,c0,c0,1     ;returns cache details
```

c0 – TCM Status Register



Bits	Name	Description	R/W	Default
31:17	reserved	-	R	0x00
16	DTCM_PRESENT	DTCM: 1 : present 0 : not present	R	1
15:1	reserved	-	R	0x00
0	ITCM_PRESENT	ITCM: 1 : present 0 : not present	R	0

This is a read-only register that enables operating systems to establish if TCM memories are present. See also TCM Region Register c9.

You can access the TCM Status Register by reading CP15 register c0 with the Opcode_2 field set to 2. For example:

```
MRC p15,0,<Rd>,c0,c0,2     ;returns TCM details
```

Note: Current revisions of netX100 and 500 do NOT provide an Instruction TCM!

c1 – Control Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reseved													reserved	reserved	reserved	L4	RR	V	I	reserved		R	S	B	reserved				C	A	M

Bits	Name	Description	R/W	Default
31:19	reserved	-	R	0x00
18	reserved	-	R	1
17	reserved	-	R	0
16	reserved	-	R	1
15	L4	Determines if the T bit is set when load instructions change the PC: 0 = loads to PC set the T bit 1 = loads to PC do not set T bit (ARMv4 behavior). For more details see the ARM Architecture Reference Manual.	R/W	0
14	RR	Replacement strategy for ICache and DCache: 0 = Random replacement 1 = Round-robin replacement.	R/W	0
13	V	Location of exception vectors: 0 = Normal exception vectors selected, address range = 0x0000 0000 to 0x0000 001C 1 = High exception vectors selected, address range = 0xFFFF 0000 to 0xFFFF 001C .	R/W	0
12	I	ICache enable/disable: 0 = ICache disabled 1 = ICache enabled.	R/W	0
11:10	reserved	-	R	00
9	R	ROM protection. This bit modifies the ROM protection system.	R/W	0
8	S	System protection. This bit modifies the MMU protection system.	R/W	0
7	B	Endianness: 0 = Little-endian operation 1 = Big-endian operation.	R/W	0
6:3	reserved	-	R	1111
2	C	DCache enable/disable: 0 = Cache disabled 1 = Cache enabled.	R/W	0
1	A	Alignment fault enable/disable: 0 = Data address alignment fault checking disabled 1 = Data address alignment fault checking enabled.	R/W	0
0	M	MMU enable/disable: 0 = disabled 1 = enabled.	R/W	0

Register c1 is the Control Register for the ARM926EJ-S processor. This register specifies the configuration used to enable and disable the caches and MMU. It is recommended that you access this register using a read-modify-write sequence.

For both reading and writing, the CRm and Opcode_2 fields Should Be Zero. To read and write this register, use the instructions:

```
MRC p15,0,<Rd>,c1,c0,0     ;read control register
MCR p15,0,<Rd>,c1,c0,0     ;write control register
```

All defined control bits are set to zero on reset.

For a more detailed description of the effects of Control Register on caches and on TCM see ARM926EJ-S (r0p4/r0p5) manual page 2-15 up to page 2-17.

c2 – Translation Table Base Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTBR																		reserved													

Bits	Name	Description	R/W	Default
31:14	TTBR	Translation table base	R/W	0x00
13:0	reserved	-	R	0x00

Register c2 is the Translation Table Base Register (TTBR), for the base address of the first-level translation table.

Reading from c2 returns the pointer to the currently active first-level translation table in bits [31:14]. Writing to register c2 updates the pointer to the first-level translation table from the value in bits [31:14] of the written value. Bits [13:0] should be zero.

You can use the following instructions to access the TTBR:

```
MRC p15,0,<Rd>,c2,c0,0    ;read TTBR
MCR p15,0,<Rd>,c2,c0,0    ;write TTBR
```

The CRm and Opcode_2 fields should be zero when writing to c2.

c3 – Domain Access Control Register

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Bits	Name	Description	R/W	Default
31:30	D15	00 : no access: Any access generates a domain fault. 01 : Client: Accesses are checked against the access permission bits in the section or page descriptor. 10 : reserved 11 : Manager: Accesses are not checked against the access permission bits so a permission fault cannot be generated.	R/W	00
29:28	D14	the same as above	R/W	00
27:26	D13	the same as above	R/W	00
25:24	D12	the same as above	R/W	00
23:22	D11	the same as above	R/W	00
21:20	D10	the same as above	R/W	00
19:18	D9	the same as above	R/W	00
17:16	D8	the same as above	R/W	00
15:14	D7	the same as above	R/W	00
13:12	D6	the same as above	R/W	00
11:10	D5	the same as above	R/W	00
9:8	D4	the same as above	R/W	00
7:6	D3	the same as above	R/W	00
5:4	D2	the same as above	R/W	00
3:2	D1	the same as above	R/W	00
1:0	D0	the same as above	R/W	00

Register c3 is the Domain Access Control Register consisting of 16 two-bit fields.

You can use the following instructions to access the Domain Access Control Register:

```
MRC p15,0,<Rd>,c3,c0,0     ;read domain access permissions
MCR p15,0,<Rd>,c3,c0,0     ;write domain access permissions
```

Each two-bit field defines the access permissions for one of the 16 domains (D15-D0). Reading from c3 returns the value of the Domain Access Control Register. Writing to c3 writes the value of the Domain Access Control Register.

c5 – Data Fault Status Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								DOMAIN				STATUS			

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:4	DOMAIN	Specifies which of the 16 domains (D15-D0) was being accessed when a data fault occurred.	R/W	undefined
3:0	STATUS	Type of fault generated For more details see ARM926EJ-S (r0p4/r0p5) Technical Reference Manual	R/W	undefined

The data fault status register (DFSR) contains the source of the last data fault. The DFSR is updated for alignment faults, and external aborts that occur while the MMU is disabled.

You can access the DFSR using the following instructions:

```
MRC p15,0,<Rd>,c5,c0,0    ;read DFSR
MCR p15,0,<Rd>,c5,c0,0    ;write DFSR
```

c5 – Instruction Fault Status Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																								DOMAIN				STATUS			

Bits	Name	Description	R/W	Default
31:8	reserved	-	R	0x00
7:4	DOMAIN	Specifies which of the 16 domains (D15-D0) was being accessed when a data fault occurred.	R/W	undefined
3:0	STATUS	Type of fault generated For more details see ARM926EJ-S (r0p4/r0p5) Technical Reference Manual	R/W	undefined

The Instruction Fault Status Register (IFSR) contains the source of the last instruction fault. The instruction-side FSR is intended for debug purposes only. The IFSR is updated for alignment faults, and external aborts that occur while the MMU is disabled.

You can access the IFSR using the following instructions:

```
MRC p15,0,<Rd>,c5,c0,1    ;read IFSR
MCR p15,0,<Rd>,c5,c0,1    ;write IFSR
```

c6 – Fault Address Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															

Bits	Name	Description	R/W	Default
31:0	ADDRESS	fault address	R/W	undefined

Register c6 accesses the Fault Address Register (FAR). The FAR contains the Modified Virtual Address of the access being attempted when a Data Abort occurred. The FAR is only updated for Data Aborts, not for Prefetch Aborts. The FAR is updated for alignment faults, and external aborts that occur while the MMU is disabled.

You can use the following instructions to access the FAR:

```
MRC p15,0,<Rd>,c6,c0,0     ;read FAR
MCR p15,0,<Rd>,c6,c0,0     ;write FAR
```

Writing c6 sets the FAR to the value of the data written. This is useful for a debugger to restore the value of the FAR to a previous state.

The CRm and Opcode_2 fields should be zero when reading or writing c6.

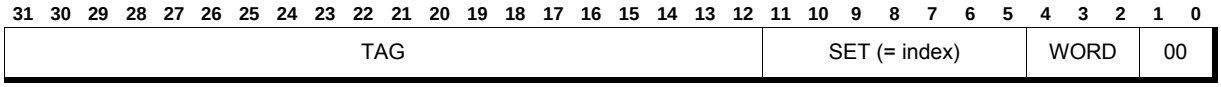
c7 – Cache Operation Register

Register c7 controls the caches and the write buffer. The function of each cache operation is selected by the Opcode_2 and CRm fields in the MCR instruction used to write to CP15 c7. Writing other Opcode_2 or CRm values is Unpredictable.
Reading from CP15 c7 is unpredictable, with the exception of the two test and clean operations.

You can use the following instructions to write to c7:

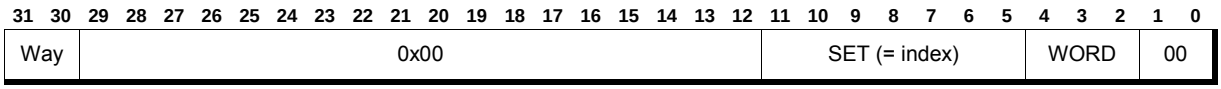
```
MCR p15,0,<Rd>,c7,c7,0     ;Invalidate ICache and DCache,
                           ;Rd should be zero
MCR p15,0,<Rd>,c7,c5,0     ;Invalidate ICache,
                           ;Rd should be zero
MCR p15,0,<Rd>,c7,c5,1     ;Invalidate ICache single entry,
                           ;Rd is MVA
MCR p15,0,<Rd>,c7,c5,2     ;Invalidate ICache single entry (Set/Way),
                           ;Rd is  Set/Way
MCR p15,0,<Rd>,c7,c13,1    ;Prefetch ICache line (MVA),
                           ;Rd is  MVA
MCR p15,0,<Rd>,c7,c6,0     ;Invalidate DCache, Rd should be zero
MCR p15,0,<Rd>,c7,c6,1     ;Invalidate DCache single entry(MVA),
                           ;Rd is MVA
```

The MVA format for Rd for the CP15 c7 MCR operations is below. The Tag, Set, and Word fields define the MVA. For all of the cache operations, Word should be zero.
MVA format:



For a 16KB cache, 4-way set associative, 8-word line, then:
• A = $\log_2$ associativity = $\log_2 4 = 2$
• S = $\log_2$ NSETS where:
NSETS= cache size in bytes/associativity/line length in bytes:
NSETS= 16384/4/32 = 128
Therefore:
S = $\log_2 128 = 7$

Set/Way format:



Test and clean operations:
The test and clean DCache instruction provides an efficient way to clean the entire DCache using a simple loop. The test and clean DCache instruction tests a number of lines in the DCache to determine if any of them are dirty. If any dirty lines are found, then one of those lines is cleaned. The test and clean DCache instruction also returns the status of the entire DCache in bit 30.

If the cache contains any dirty lines, bit 30 is set to 0. If the cache contains no dirty lines, bit 30 is set to 1. This means that you can use the following loop to clean the entire DCache:

```
tc_loop:  MRC p15,0,r15,c7,c10,3     ;test and clean
          BNE tc_loop
```

The test, clean, and invalidate DCache instruction is the same as test and clean DCache, except that when the entire cache has been cleaned, it is invalidated. This means that you can use the following loop to clean and invalidate the entire DCache:

```
tci_loop:  MRC p15,0,r15,c7,c14,3    ;test clean and invalidate
           BNE tci_loop
```

Note:
The test and clean DCache instruction `MRC p15,0,r15,c7,c10,3` is a special encoding that uses r15 as a destination operand. However, the PC is not changed by using this instruction. This MRC instruction also sets the condition code flags.

c8 – TLB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODIFIED VIRTUAL ADDRESS																						0x00									

This is a write-only register used to control the Translation Lookaside Buffer (TLB).
There is a single TLB used to hold entries for both data and instructions. The TLB is divided into two parts:

- a set-associative part
- a fully-associative part

The fully-associative part (also referred to as the lockdown part of the TLB) is used to store entries to be locked down. Entries held in the lockdown part of the TLB are preserved during an invalidate TLB operation. Entries can be removed from the lockdown TLB using an invalidate TLB single entry operation.
Six TLB operations are defined, and the function to be performed is selected by the Opcode_2 and CRm fields in the MCR instruction used to write CP15 c8. Writing other Opcode_2 or CRm values is Unpredictable. Reading from this register is Unpredictable.

You can use the following instructions to perform TLB operations:

```
MCR p15,0,<Rd>,c8,c7,0    ;Invalidate set-associative TLB,
                          ;<Rd> should be zero
MCR p15,0,<Rd>,c8,c7,1    ;Invalidate single entry,
                          ;<Rd> is MVA
MCR p15,0,<Rd>,c8,c5,0    ; Invalidate set-associative TLB,
                          ;<Rd> should be zero
MCR p15,0,<Rd>,c8,c5,1    ; Invalidate single entry,
                          ;<Rd> is MVA
MCR p15,0,<Rd>,c8,c6,0    ;Invalidate set-associative TLB,
                          ;<Rd> should be zero
MCR p15,0,<Rd>,c8,c6,1    ;Invalidate single entry,
                          ;<Rd> is MVA
```

The invalidate TLB operations invalidate all the unpreserved entries in the TLB. The invalidate TLB single entry operations invalidate any TLB entry corresponding to the Modified Virtual Address given in Rd, regardless of its preserved state. See TLB Lockdown Register for a description of how to preserve entries in the TLB.

The Modified Virtual Address for invalidate TLB single entry operations has the following format:

Note:
If either small or large pages are used, and these pages contain sub page access permissions that are different, then you must use four invalidate TLB single entry operations, with the MVA set to each sub page, to invalidate all information related to that page held in a TLB.

c9 – Cache Lockdown Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved																reserved												L BITS			

Bits	Name	Description	R/W	Default
31:16	reserved	-	R	0x00
15:4	reserved	-	R/W	0xfff
3	L bit for Way 3	Bits[3:0] are the L bits for each cache way: 0 : Allocation to the cache way is determined by the standard replacement algorithm (reset state) 1 : No allocation is performed to this cache way.	R/W	0
2	L bit for Way 2	the same as above	R/W	0
1	L bit for Way 1	the same as above	R/W	0
0	L bit for Way 0	the same as above	R/W	0

The Cache Lockdown Register uses a cache-way-based locking scheme (Format C) that enables you to control each cache way independently.

These registers enable you to control which cache ways of the four-way cache are used for the allocation on a line fill. When the registers are defined, subsequent line fills are only placed in the specified target cache way. This gives you some control over the cache pollution caused by particular applications, and provides a traditional lockdown operation for locking critical code into the cache.
A locking bit for each cache way determines if the normal cache allocation is allowed to access that cache way.
A maximum of three cache ways of the four-way associative cache can be locked, ensuring that normal cache line replacement is performed.
Note:
If no cache ways have L bits set to 0, then cache way 3 is used for all line fills.

You can use the following instructions to access the Cache Lockdown Register:

```
MRC p15,0,<Rd>,c9,c0,0    ;Read DCache Lockdown Register, Rd is L bits
MCR p15,0,<Rd>,c9,c0,0    ;Write DCache Lockdown Register, Rd is L bits
MRC p15,0,<Rd>,c9,c0,1    ;Read ICache Lockdown Register, Rd is L bits
MCR p15,0,<Rd>,c9,c0,1    ;Write ICache Lockdown Register, Rd is L bits
```

You must only modify the Cache Lockdown Register using a read-modify-write sequence. For example:

```
MRC p15,0,<Rn>,c9,c0,1    ;
ORR <Rn>,<Rn>,0x01        ;
MCR p15,0,<Rn>,c9,c0,1    ;
```

This sequence sets the L bit to 1 for way 0 of the ICache. All cache ways are available for allocation from reset.
You can use the cache lockdown and cache unlock procedures described in:

- Specific loading of addresses into a cache way
- Cache unlock procedure

Specific loading of addresses into a cache way:

The procedure to lock down code and data into way i of a cache with N ways using Format C involves making it impossible to allocate to any cache way other than the target cache way:
1. Ensure that no processor exceptions can occur during the execution of this procedure, for example by disabling interrupts. If this is not possible, all code and data used by any exception handlers must be treated as code and data as in steps 2 and 3.
2. If an ICache way is being locked down, ensure that all the code executed by the lockdown procedure is in an uncachable area of memory (including TCM) or in an already locked cache way.
3. If a DCache way is being locked down, ensure that all data used by the lockdown procedure is in an uncachable area of memory (including TCM) or is in an already locked cache way.
4. Ensure that the data/instructions that are to be locked down are in a cachable area of memory.
5. Ensure that the data/instructions that are to be locked down are not already in the cache. Use the register c7 clean and/or invalidate operations to ensure this.
6. Write to register c9, CRm == 0, setting L==0 for bit i and L==1 for all other ways. This enables allocation to the target cache way.
7. For each of the cache lines to be locked down in cache way i:
• If a DCache is being locked down, use an LDR instruction to load a word from the memory cache line to ensure that the memory cache line is loaded into the cache.
• If an ICache is being locked down, use the register c7 MCR prefetch ICache line (CRm == c13, Opcode2 == 1) to fetch the memory cache line into the cache.
8. Write to register c9, CRm == 0 setting L == 1 for bit i and restoring all the other bits to the values they had before the lockdown routine was started.

Cache unlock procedure:
To unlock the locked down portion of the cache, write to register c9 setting L == 0 for the appropriate bit. For example, the following sequence sets the L bit to 0 for way 0 of the ICache, unlocking way 0:

```
MRC p15,0,<Rn>,c9,c0,1    ;
BIC <Rn>,<Rn>,0x01        ;
MCR p15,0,<Rn>,c9,c0,1    ;
```

c9 – TCM Region Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE ADDRESS																				reserved						SIZE				reserved	ENABLE

Bits	Name	Description	R/W	Default
31:12	BASE ADDRESS	Base Address (physical address)	R/W	0x00
11:6	reserved	-	R	0x00
5:2	SIZE	0000 : **0 KByte, absent** 0100 : 8KByte	R	DTCM: 0100 ITCM: 0000
1	reserved	-	R	0
0	ENABLE	Enable bit: 0 : disabled 1 : enabled	R/W	0

The ARM926EJ-S processor supports physically-indexed, physically-tagged TCM. The TCM Region Register supports one region of instruction TCM and one region of data TCM. The minimum size of TCM region that can be supported is 4KB. The TCM Status Register indicates if TCM memories are attached (see TCM Status Register c0). The size of each TCM region is defined by the DRSIZE and IRSIZE input pins.
The data TCM is always disabled at reset. The instruction TCM is enabled at reset if the INITRAM pin is HIGH. This enables booting from the instruction TCM and sets the ITCM enable bit in the ITCM region register.

```
MRC p15,0,<Rd>,c9,c1,0     ;Read     data     TCM     Region     Register,
                           ;Rd is Base Address
MCR p15,0,<Rd>,c9,c1,0     ;Write     data     TCM     Region     Register,
                           ;Rd is Base Address
MRC p15,0,<Rd>,c9,c1,1     ;Read   instruction   TCM   Region   Register,
                           ;Rd is Base Address
MCR p15,0,<Rd>,c9,c1,1     ;Write   instruction   TCM   Region   Register,
                           ;Rd is Base Address
```

If either the data or instruction TCM is disabled, then the contents of the respective TCM are not accessed. If the TCM is subsequently re-enabled, the contents will not have been changed by the ARM926EJ-S processor.
For a Harvard arrangement, the instruction-side TCM must be accessible for both reads and writes during normal operation, and for loading code, or for debug activity. This enables accesses to literal pools, undefined instruction emulation, and parameter passing for SWI operations. You must insert an Instruction Memory Barrier (IMB) between a write to the instruction TCM and the instructions being read from the instruction TCM.

Instruction fetches from the data TCM are not possible. An attempt to fetch an instruction from an address in the data TCM space does not result in an access to the data TCM, and the instruction is fetched from main memory. These accesses can result in external aborts, because the address range might not be supported in main memory.

The instruction TCM must not be programmed to the same base address as the data TCM. If the two TCMs are of different sizes, the regions in physical memory must not overlap. If they do overlap, it is unpredictable which memory will be accessed.

Note: The base address value setting must be aligned to the TCM size.

Note: Current revisions of netX100 and 500 do NOT provide an Instruction TCM!

c10 – TLB Lockdown Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved			VICTIM			reserved																									P

Bits	Name	Description	R/W	Default
31:29	reserved	-	R	000
28:26	VICTIM	Victim	R/W	000
25:1	reserved	-	R	0x00
0	P	1: Means subsequent hardware page table walks place the TLB entry in the lockdown region at the entry specified by the victim, in the range 0 to 7. 0: Means subsequent hardware page table walks place the TLB entry in the set associative region of the TLB.	R/W	0

The TLB Lockdown Register controls where hardware page table walks place the TLB entry, in the set associative region or the lockdown region of the TLB, and if in the lockdown region, which entry is written. The lockdown region of the TLB contains eight entries.

TLB entries in the lockdown region are preserved so that invalidate TLB operations only invalidate the unpreserved entries in the TLB. That is, those in the set-associative region. Invalidate TLB single entry operations invalidate any TLB entry corresponding to the Modified Virtual Address given in Rd, regardless of their preserved state. That is, if they are in the lockdown or set-associative regions of the TLB. See TLB Operations Register c8 for a description of the TLB invalidate operations.

To program the TLB Lockdown Register you can use the following instructions:

```
MRC p15,0,<Rd>,c10,c0,0      ;Read data TLB lockdown victim
MCR p15,0,<Rd>,c10,c0,0      ;Write data TLB lockdown victim
```

Note:
It is not possible for a lockdown entry to entirely map either small or large pages, unless all the subpage access permissions are identical. Entries can still be written into the lockdown region, but the address range that is mapped only covers the subpage corresponding to the address that was used to perform the page table walk.

Code sequence example that locks down an entry to the current victim:

```
ADR r1,LockAddr           ;set r1 to the value of the address to be
                          ;locked down
MCR p15,0,r1,c8,c7,1      ;invalidate TLB single entry to ensure that
                          ;LockAddr is not already in the TLB
MRC p15,0,r0,c10,c0,0     ;read the lockdown register
ORR r0,r0,#1              ;set the preserve bit
MCR p15,0,r0,c10,c0,0     ;write to the lockdown register
LDR r1,[r1]               ;TLB will miss, and entry will be loaded
MRC p15,0,r0,c10,c0,0     ;read the lockdown register
                          ;(victim will have incremented)
BIC r0,r0,#1              ;clear preserve bit
MCR p15,0,r0,c10,c0,0     ;write to the lockdown register
```

c13 – FCSE PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FCSE PID							reserved																								

Bits	Name	Description	R/W	Default
31:25	FCSE PID	Fast Context Switch Extension Process Identifier Register	R/W	0x00
24:0	reserved	-	R	0x00

Fast Context Switch Extension Process Identifier Register

Addresses issued by the ARM9EJ-S core in the range 0 to 32MB are translated in accordance with the value contained in this register. Address A becomes A + (FCSE PID x 32MB). It is this modified address that is seen by the caches, MMU, and TCM interface. Addresses above 32MB are not modified. The FCSE PID is a seven-bit field, enabling 128 x 32MB processes to be mapped.

If the FCSE PID is 0, there is a flat mapping between the virtual addresses output by the ARM9EJ-S core and the modified virtual addresses used by the caches, MMU, and TCM interface. The FCSE PID is set to 0 at system reset.

If the MMU is disabled, then no FCSE address translation occurs.
FCSE translation is not applied for addresses used for entry based cache or TLB maintenance operations. For these operations VA = MVA.

Performing a fast context switch
You can perform a fast context switch by writing to CP15 register c13 with Opcode_2 = 0. The contents of the caches and the TLB do not have to be flushed after a fast context switch because they still hold valid address tags. The two instructions after the FCSE PID has been written have been fetched with the old FCSE PID, as the following code example shows:

```
{FCSE PID =0}
          MOV r0,#1:SHL:25          ;Fetched with FCSE PID =0
          MCR p15,0,r0,c13,c0,0     ;Fetched with FCSE PID =0
          A1                        ;Fetched with FCSE PID =0
          A2                        ;Fetched with FCSE PID =0
          A3                        ;Fetched with FCSE PID =1
```

Where A1, A2, and A3 are the three instructions following the fast context switch.

c13 – Context ID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONTEXT IDENTIFIER																															

Bits	Name	Description	R/W	Default
31:0	CONTEXT IDENTIFIER	You can use the process ID register to determine the process that is currently running. The process identifier is set to 0 at reset.	R/W	0x00

You can use the process ID register to determine the process that is currently running. The process identifier is set to 0 at reset.

Here are ARM instructions that you can use to access the Context ID Register:

```
MRC p15,0,<Rd>,c13,c0,1   ;Read context ID
MCR p15,0,<Rd>,c13,c0,1   ;Write context ID
```

10 Appendix A: Register Table

ARM Address	Register Name	Short Description
0x00100000	BOO_STAT	Bond Out Option Status Register
0x00100004	IO_CFG	IO Configuration Register
0x00100008	IO_CFG_MSK	IO Configuration Mask Register
0x0010000c	RESET_CTRL	Reset Control Register
0x00100034	NETX_REV	netX Revision Register
0x00100070	IO_CFG_ACCESS_KEY	IO Configuration Access Key Register
0x00100200	WDG_TRIG	Watchdog Trigger Register
0x00100204	WDG_CNTR	Watchdog Counter
0x00100208	WDG_IRQ_TIMEOUT	Watchdog Interrupt Timeout
0x0010020c	WDG_RESET_TIMEOUT	Watchdog Reset Timeout
0x001034d8	SYS_STAT	System Status
0x00100100	MEM_SRAM0_CTRL	Memory SRAM Control Register for Chip Select Area 0
0x00100104	MEM_SRAM1_CTRL	Memory SRAM Control Register for Chip Select Area 1
0x00100108	MEM_SRAM2_CTRL	Memory SRAM Control Register for Chip Select Area 2
0x00100140	MEM_SDRAM_CFG_CTRL	Memory SDRAM Configuration Control Register
0x00100144	MEM_SDRAM_TIMING_CTRL	Memory SDRAM Timing Control Register
0x00100148	MEM_SDRAM_MODE	Memory SDRAM Mode Register
0x0010014c	MEM_SDRAM_EXT_MODE	Memory SDRAM Extended Mode Register
0x00100180	MEM_PRIO_TIMESLOT_CTRL	Memory Priority Timeslot Control Register
0x00100184	MEM_PRIO_ACCESS_CTRL	Memory Priority Access Control Register
0x00103610	EXT_CONFIG_CS0	Extension Bus Configuration Chip Select 0
0x00103614	EXT_CONFIG_CS1	Extension Bus Configuration Chip Select 1
0x00103618	EXT_CONFIG_CS2	Extension Bus Configuration Chip Select 2
0x0010361c	EXT_CONFIG_CS3	Extension Bus Configuration Chip Select 3
0x001036bc	DPM_ARM_HS_CTRL15	Handshake Control Register 15
0x001036b8	DPM_ARM_HS_CTRL14	Handshake Control Register 14
0x001036b4	DPM_ARM_HS_CTRL13	Handshake Control Register 13
0x001036b0	DPM_ARM_HS_CTRL12	Handshake Control Register 12
0x001036ac	DPM_ARM_HS_CTRL11	Handshake Control Register 11
0x001036a8	DPM_ARM_HS_CTRL10	Handshake Control Register 10
0x001036a4	DPM_ARM_HS_CTRL9	Handshake Control Register 9
0x001036a0	DPM_ARM_HS_CTRL8	Handshake Control Register 8
0x0010369c	DPM_ARM_HS_CTRL3	Handshake Control Register 3
0x00103698	DPM_ARM_HS_CTRL6	Handshake Control Register 6
0x00103694	DPM_ARM_HS_CTRL5	Handshake Control Register 5
0x00103690	DPM_ARM_HS_CTRL4	Handshake Control Register 4
0x0010368c	DPM_ARM_HS_CTRL3	Handshake Control Register 3
0x00103688	DPM_ARM_HS_CTRL2	Handshake Control Register 2
0x00103684	DPM_ARM_HS_CTRL1	Handshake Control Register 1
0x00103680	DPM_ARM_HS_CTRL0	Handshake Control Register 0

0x0010367c	DPM_ARM_DB_MAP7	Data Block Mapping Address Register 7
0x00103678	DPM_ARM_DB_END7	Data Block 7 End Address
0x00103674	DPM_ARM_DB_MAP6	Data Block Mapping Address Register 6
0x00103670	DPM_ARM_DB_END6	Data Block 6 End Address
0x0010366c	DPM_ARM_DB_MAP5	Data Block Mapping Address Register 5
0x00103668	DPM_ARM_DB_END5	Data Block 5 End Address
0x00103664	DPM_ARM_DB_MAP4	Data Block Mapping Address Register 4
0x00103660	DPM_ARM_DB_END4	Data Block 4 End Address
0x0010365c	DPM_ARM_DB_MAP3	Data Block Mapping Address Register 3
0x00103658	DPM_ARM_DB_END3	Data Block 3 End Address
0x00103654	DPM_ARM_DB_MAP2	Data Block Mapping Address Register 2
0x00103650	DPM_ARM_DB_END2	Data Block 2 End Address
0x0010364c	DPM_ARM_DB_MAP1	Data Block Mapping Address Register 1
0x00103648	DPM_ARM_DB_END1	Data Block 1 End Address
0x00103644	DPM_ARM_DB_MAP0	Data Block Mapping Address Register 0
0x00103640	DPM_ARM_DB_END0	Data Block 0 End Address
0x00103638	DPM_ARM_IO_DATA1	Input / Output Data Register 1
0x00103634	DPM_ARM_IO_DRV_EN1	Input / Output Driver Enable Register 1
0x00103630	DPM_ARM_IO_MODE1	Input / Ouput Mode Register 1
0x00103628	DPM_ARM_IO_DATA0	Input / Output Data Register 0
0x00103624	DPM_ARM_IO_DRV_EN0	Input / Ouput Driver Enable Register 0
0x00103620	DPM_ARM_IO_MODE0	Input / Ouput Mode Register 0
0x0010360c	DPM_ARM_IF_CFG1	Interface Configuration Register 1
0x00103608	DPM_ARM_IF_CFG0	Interface Configuration Register 0
0x00103604	DPM_ARM_CLKOUT_CFG	Clockout Configuration Register
0x0010353c	DPM_ARM_HS_DATA15	Handshake Data Register 15
0x00103538	DPM_ARM_HS_DATA14	Handshake Data Register 14
0x00103534	DPM_ARM_HS_DATA13	Handshake Data Register 13
0x00103530	DPM_ARM_HS_DATA12	Handshake Data Register 12
0x0010352c	DPM_ARM_HS_DATA11	Handshake Data Register 11
0x00103528	DPM_ARM_HS_DATA10	Handshake Data Register 10
0x00103524	DPM_ARM_HS_DATA9	Handshake Data Register 9
0x00103520	DPM_ARM_HS_DATA8	Handshake Data Register 8
0x0010351c	DPM_ARM_HS_DATA7	Handshake Data Register 7
0x00103518	DPM_ARM_HS_DATA6	Handshake Data Register 6
0x00103514	DPM_ARM_HS_DATA5	Handshake Data Register 5
0x00103510	DPM_ARM_HS_DATA4	Handshake Data Register 4
0x0010350c	DPM_ARM_HS_DATA3	Handshake Data Register 3
0x00103508	DPM_ARM_HS_DATA2	Handshake Data Register 2
0x00103504	DPM_ARM_HS_DATA1	Handshake Data Register 1
0x00103500	DPM_ARM_HS_DATA0	Handshake Data Register 0
0x001034f0	DPM_ARM_INT_EN0	Interrupt Enable 0 Register
0x001034e0	DPM_ARM_INT_STAT0	Interrupt Status 0 Register
0x001034d8	DPM_ARM_SYS_STAT	System Status Register
0x001034cc	DPM_ARM_WDG_ARM_TRIG	Watchdog Trigger ARM

0x001034c8	DPM_ARM_WDG_ARM_TIMEOUT	Watchdog Timeout ARM, read only
0x001034c0	DPM_ARM_WDG_HOST_TIMEOUT	Watchdog Timeout Host
0x001034bc	DPM_ARM_CIS_MAP	Card Information Structure Mapping Address
0x0010323c	DPM_HOST_HS_DATA15	Handshake Data Register 15
0x00103238	DPM_HOST_HS_DATA14	Handshake Data Register 14
0x00103234	DPM_HOST_HS_DATA13	Handshake Data Register 13
0x00103230	DPM_HOST_HS_DATA12	Handshake Data Register 12
0x0010322c	DPM_HOST_HS_DATA11	Handshake Data Register 11
0x00103228	DPM_HOST_HS_DATA10	Handshake Data Register 10
0x00103224	DPM_HOST_HS_DATA9	Handshake Data Register 9
0x00103220	DPM_HOST_HS_DATA8	Handshake Data Register 8
0x0010321c	DPM_HOST_HS_DATA3	Handshake Data Register 3
0x00103218	DPM_HOST_HS_DATA6	Handshake Data Register 6
0x00103214	DPM_HOST_HS_DATA5	Handshake Data Register 5
0x00103210	DPM_HOST_HS_DATA4	Handshake Data Register 4
0x0010320c	DPM_HOST_HS_DATA3	Handshake Data Register 3
0x00103208	DPM_HOST_HS_DATA2	Handshake Data Register 2
0x00103204	DPM_HOST_HS_DATA1	Handshake Data Register 1
0x00103200	DPM_HOST_HS_DATA0	Handshake Data Register 0
0x001031f0	DPM_HOST_INT_EN0	Interrupt Enable 0
0x001031e0	DPM_HOST_INT_STAT0	Interrupt Status 0
0x001031dc	DPM_HOST_RES_REQ	Reset Request
0x001031d8	DPM_HOST_SYS_STAT	System Status
0x001031d4	DPM_HOST_TMR_START_VAL	Timer Start Value
0x001031d0	DPM_HOST_TMR_CTRL	Timer Control
0x001031c8	DPM_HOST_WDG_ARM_TIMEOUT	Watchdog ARM Timeout
0x001031c4	DPM_HOST_WDG_HOST_TRIG	Watchdog Host Trigger
0x001031c0	DPM_HOST_WDG_HOST_TIMEOUT	Watchdog Host Timeout, read only
0x00100800	GPIO_CFG0	GPIO 0 Configuration Register
0x00100804	GPIO_CFG1	GPIO 1 Configuration Register
0x00100808	GPIO_CFG2	GPIO 2 Configuration Register
0x0010080c	GPIO_CFG3	GPIO 3 Configuration Register
0x00100810	GPIO_CFG4	GPIO 4 Configuration Register
0x00100814	GPIO_CFG5	GPIO 5 Configuration Register
0x00100818	GPIO_CFG6	GPIO 6 Configuration Register
0x0010081c	GPIO_CFG7	GPIO 7 Configuration Register
0x00100820	GPIO_CFG8	GPIO 8 Configuration Register
0x00100824	GPIO_CFG9	GPIO 9 Configuration Register
0x00100828	GPIO_CFG10	GPIO 10 Configuration Register
0x0010082c	GPIO_CFG11	GPIO 11 Configuration Register
0x00100830	GPIO_CFG12	GPIO 12 Configuration Register
0x00100834	GPIO_CFG13	GPIO 13 Configuration Register
0x00100838	GPIO_CFG14	GPIO 14 Configuration Register
0x0010083c	GPIO_CFG15	GPIO 15 Configuration Register
0x00100840	GPIO_THRSH_CAPT0	GPIO 0 Threshold or Capture Register

0x00100844	GPIO_THRSH_CAPT1	GPIO 1 Threshold or Capture Register
0x00100848	GPIO_THRSH_CAPT2	GPIO 2 Threshold or Capture Register
0x0010084c	GPIO_THRSH_CAPT3	GPIO 3 Threshold or Capture Register
0x00100850	GPIO_THRSH_CAPT4	GPIO 4 Threshold or Capture Register
0x00100854	GPIO_THRSH_CAPT5	GPIO 5 Threshold or Capture Register
0x00100858	GPIO_THRSH_CAPT6	GPIO 6 Threshold or Capture Register
0x0010085c	GPIO_THRSH_CAPT7	GPIO 7 Threshold or Capture Register
0x00100860	GPIO_THRSH_CAPT8	GPIO 8 Threshold or Capture Register
0x00100864	GPIO_THRSH_CAPT9	GPIO 9 Threshold or Capture Register
0x00100868	GPIO_THRSH_CAPT10	GPIO 10 Threshold or Capture Register
0x0010086c	GPIO_THRSH_CAPT11	GPIO 11 Threshold or Capture Register
0x00100870	GPIO_THRSH_CAPT12	GPIO 12 Threshold or Capture Register
0x00100874	GPIO_THRSH_CAPT13	GPIO 13 Threshold or Capture Register
0x00100878	GPIO_THRSH_CAPT14	GPIO 14 Threshold or Capture Register
0x0010087c	GPIO_THRSH_CAPT15	GPIO 15 Threshold or Capture Register
0x00100880	GPIO_CNTR0_CTRL	GPIO counter 0 control register
0x00100884	GPIO_CNTR1_CTRL	GPIO counter 1 control register
0x00100888	GPIO_CNTR2_CTRL	GPIO counter 2 control register
0x0010088c	GPIO_CNTR3_CTRL	GPIO counter 3 control register
0x00100890	GPIO_CNTR4_CTRL	GPIO counter 4 control register
0x00100894	GPIO_CNTR0_MAX	GPIO counter 0 max values
0x00100898	GPIO_CNTR1_MAX	GPIO counter 1 max values
0x0010089c	GPIO_CNTR2_MAX	GPIO counter 2 max values
0x001008a0	GPIO_CNTR3_MAX	GPIO counter 3 max values
0x001008a4	GPIO_CNTR4_MAX	GPIO counter 4 max values
0x001008a8	GPIO_CNTR0_CNT	GPIO counter 0 current value
0x001008ac	GPIO_CNTR1_CNT	GPIO counter 1 current value
0x001008b0	GPIO_CNTR2_CNT	GPIO counter 2 current value
0x001008b4	GPIO_CNTR3_CNT	GPIO counter 3 current value
0x001008b8	GPIO_CNTR4_CNT	GPIO counter 4 current value
0x001008bc	GPIO_IRQ_MSK_SET	GPIO Interrupt Request Enable Mask
0x001008c0	GPIO_IRQ_MSK_RESET	GPIO Interrupt Request Disable Mask
0x001008c4	GPIO_SYSTIME_NS_CMP	GPIO System Time NS Compare Value
0x001008c8	GPIO_OUT	GPIO Output Register
0x001008cc	GPIO_IN	GPIO Input Register
0x001008d0	GPIO_IRQ	GPIO Interrupt Request
0x00100900	PIO_IN	PIO Input Register
0x00100904	PIO_OUT	PIO Output Register
0x00100908	PIO_OUT_EN	PIO Output Enable Register
0x00100a00	UART0_DATA	UART0 Data Register
0x00100a04	UART0_STAT	UART0 Status Register
0x00100a08	UART0_LINE_CTRL	UART0 Line Control Register
0x00100a0c	UART0_BAUD_DIV_MSB	UART0 Baud Rate Divisor MSB
0x00100a10	UART0_BAUD_DIV_LSB	UART0 Baud Rate Divisor LSB
0x00100a14	UART0_CTRL	UART0 Control Register

0x00100a18	UART0_FLAG	UART0 Flag Register
0x00100a1c	UART0_INT_ID	UART0 Interrupt Identification Register
0x00100a20	UART0_IRDA_LO_PWR_CNTR	UART0 IrDA Low Power Counter Register
0x00100a24	UART0_RTS_CTRL	UART0 RTS Control Register
0x00100a28	UART0_RTS_LEAD_CYC	UART0 RTS Leading Cycles
0x00100a2c	UART0_RTS_TRAIL_CYC	UART0 RTS Trailing cycles
0x00100a30	UART0_OUT_DRV_EN	UART0 UART Output Driver Enable Register
0x00100a34	UART0_BAUD_MODE_CTRL	UART0 Baud Rate Mode Control Register
0x00100a38	UART0_RX_FIFO_IRQ_LVL	UART0 Receive FIFO Interrupt Trigger Level
0x00100a3c	UART0_TX_FIFO_IRQ_LVL	UART0 Transmit FIFO Interrupt Trigger Level
0x00100a40	UART1_DATA	UART1 Data Register
0x00100a44	UART1_STAT	UART1 Status Register
0x00100a48	UART1_LINE_CTRL	UART1 Line Control Register
0x00100a4c	UART1_BAUD_DIV_MSB	UART1 Baud Rate Divisor MSB
0x00100a50	UART1_BAUD_DIV_LSB	UART1 Baud Rate Divisor LSB
0x00100a54	UART1_CTRL	UART1 Control Register
0x00100a58	UART1_FLAG	UART1 Flag Register
0x00100a5c	UART1_INT_ID	UART1 Interrupt Identification Register
0x00100a60	UART1_IRDA_LO_PWR_CNTR	UART1 IrDA Low Power Counter Register
0x00100a64	UART1_RTS_CTRL	UART1 RTS Control Register
0x00100a68	UART1_RTS_LEAD_CYC	UART1 RTS Leading Cycles
0x00100a6c	UART1_RTS_TRAIL_CYC	UART1 RTS Trailing cycles
0x00100a70	UART1_OUT_DRV_EN	UART1 UART Output Driver Enable Register
0x00100a74	UART1_BAUD_MODE_CTRL	UART1 Baud Rate Mode Control Register
0x00100a78	UART1_RX_FIFO_IRQ_LVL	UART1 Receive FIFO Interrupt Trigger Level
0x00100a7c	UART1_TX_FIFO_IRQ_LVL	UART1 Transmit FIFO Interrupt Trigger Level
0x00100a80	UART2_DATA	UART2 Data Register
0x00100a84	UART2_STAT	UART2 Status Register
0x00100a88	UART2_LINE_CTRL	UART2 Line Control Register
0x00100a8c	UART2_BAUD_DIV_MSB	UART2 Baud Rate Divisor MSB
0x00100a90	UART2_BAUD_DIV_LSB	UART2 Baud Rate Divisor LSB
0x00100a94	UART2_CTRL	UART2 Control Register
0x00100a98	UART2_FLAG	UART2 Flag Register
0x00100a9c	UART2_INT_ID	UART2 Interrupt Identification Register
0x00100aa0	UART2_IRDA_LO_PWR_CNTR	UART2 IrDA Low Power Counter Register
0x00100aa4	UART2_RTS_CTRL	UART2 RTS Control Register
0x00100aa8	UART2_RTS_LEAD_CYC	UART2 RTS Leading Cycles
0x00100aac	UART2_RTS_TRAIL_CYC	UART2 RTS Trailing cycles
0x00100ab0	UART2_OUT_DRV_EN	UART2 UART Output Driver Enable Register
0x00100ab4	UART2_BAUD_MODE_CTRL	UART2 Baud Rate Mode Control Register
0x00100ab8	UART2_RX_FIFO_IRQ_LVL	UART2 Receive FIFO Interrupt Trigger Level
0x00100abc	UART2_TX_FIFO_IRQ_LVL	UART2 Transmit FIFO Interrupt Trigger Level
0x00100c00	SPI_DATA	SPI Data Register
0x00100c04	SPI_STAT	SPI Status Register
0x00100c08	SPI_CTRL	SPI Control Register

0x00100c0c	SPI_INT_CTRL	SPI Interrupt Control Register
0x00100d00	I2C_CTRL	I2C Control Register
0x00100d04	I2C_DATA	I2C Data Register
0x00101100	SYS_TIME_NS	System Time Nanosecond Register
0x00101104	SYS_TIME_S	System Time Second Register
0x00101108	SYS_TIME_NS_BOR	System Time Nanoseconds Border Register
0x0010110c	SYS_TIME_NS_ADD_UP	System Time Nanoseconds Add Up Register
0x00101110	SYS_TIME_S_CMP	System Time Second Compare Register
0x00101114	SYS_TIME_S_CMP_EN	System Time Second Compare Enable Register
0x00101118	SYS_TIME_S_CMP_INT	System Time Second Compare Interrupt Register
0x00101200	RTC_1HZ_CNTR	RTC 1Hz COUNTER
0x00101204	RTC_32KHZ_CNTR_CUR_VAL	RTC 32KHz Counter Current Value
0x00101208	RTC_32KHZ_CNTR_LAT_VAL	RTC 32KHz Counter Latched Value
0x0010120c	RTC_IRQ2ISOLATE_CYC	Number of Clock_32kHz Cycles
0x00101210	RTC_INT_MSK	Isolated Area Interrupt Mask Register
0x00101214	RTC_INT_STAT	Isolated Area Interrupt Status Register
0x00101218	RTC_ISOLATED	Isolated Area is Currently Isolated
0x00104000	LCD_TIMING0	LCD Horizontal Axis Panel Control Register
0x00104004	LCD_TIMING1	LCD Vertical Axis Panel Control Register
0x00104008	LCD_TIMING2	LCD Clock and Signal Polarity Control Register
0x0010400c	LCD_TIMING3	LCD Line End Control Register
0x00104010	LCD_UP_PAN_BASE	LCD Upper Panel Frame Base Address Registers
0x00104014	LCD_LO_PAN_BASE	LCD Lower Panel Frame Base Address Registers
0x00104018	LCD_INT_MSK_SET_CLR	LCD Interrupt Mask Set/Clear Register
0x0010401c	LCD_CTRL	LCD Control Register
0x00104020	LCD_RAW_INT_STAT	LCD Raw Interrupt Status Register
0x00104024	LCD_MSK_INT_STAT	LCD Masked Interrupt Status Register
0x00104028	LCD_INT_CLR	LCD Interrupt Clear Register
0x0010402c	LCD_UP_PAN_CUR	LCD Upper Panel Current Address Value Registers
0x00104030	LCD_LO_PAN_CUR	LCD Lower Panel Current Address Value Registers
0x00104200	LCD_PALETTE_BASE	LCD Coulour Palette Register Base
0x001043fc	LCD_PALETTE_END	LCD Coulour Palette Registers End
0x00104fe0	CLCD_PERIPH_ID0	LCD Peripheral Identification Register 0
0x00104fe4	CLCD_PERIPH_ID1	LCD Peripheral Identification Register 1
0x00104fe8	CLCD_PERIPH_ID2	LCD Peripheral Identification Register 2
0x00104fec	CLCD_PERIPH_ID3	LCD Peripheral Identification Register 3
0x00104ff0	CLCD_PRIME_CELL_ID0	LCD Prime Cell Identification Register 0
0x00104ff4	CLCD_PRIME_CELL_ID1	LCD Prime Cell Identification Register 1
0x00104ff8	CLCD_PRIME_CELL_ID2	LCD Prime Cell Identification Register 2
0x00104ffc	CLCD_PRIME_CELL_ID3	LCD Prime Cell Identification Register 3
0x00120000	USB_ID	USB ID Register
0x00120004	USB_CTRL	USB Control Register
0x00120008	USB_FRM_TMR	USB Frame Timer Register
0x0012000c	USB_MAIN_EV	USB Main Event Register
0x00120010	USB_MAIN_EV_MSK	USB Main Event Mask Register

0x00120014	USB_PIPE_EV	USB Pipe Event Register
0x00120018	USB_PIPE_EV_MSK	USB Pipe Event Mask Register
0x00120024	USB_PIPE_SEL	USB Pipe Select Register
0x0012002c	USB_PORT_STAT	USB Port Status Register
0x00120030	USB_PORT_CTRL	USB Port Control Register
0x00120034	USB_PORT_STAT_CHG_EV	USB Port Status Change Event Register
0x00120038	USB_PORT_STAT_CHG_EV_MSK	USB Port Status Change Event Mask Register
0x00120040	USB_PIPE_CTRL	USB Pipe Control Register
0x00120044	USB_PIPE_CFG	USB Pipe Configuration Register
0x00120048	USB_PIPE_ADDR	USB Pipe Address Register
0x0012004c	USB_PIPE_STAT	USB Pipe Status Register
0x00120050	USB_PIPE_DATA_PTR	USB Pipe Data Pointer Register
0x00120054	USB_PIPE_DATA_TOT	USB Pipe Total Bytes Register
0x00120058	USB_PIPE_ALT_DATA_PTR	USB Pipe Alternative Data Pointer Register
0x0012005c	USB_PIPE_ALT_DATA_TOT	USB Pipe Alternative Data Total Bytes Register
0x00120060	USB_DBG_CTRL	USB Debug Control Register
0x00120064	USB_DBG_PID	USB Debug PID Register
0x00120068	USB_DBG_STAT	USB Debug Status Register
0x0012006c	USB_TEST	USB Test Register
0x00120080	USB_MAIN_CFG	USB Main Configuration Register
0x00120084	USB_MODE_CFG	USB Mode Configuration Register
0x00120088	USB_CORE_CTRL	USB Core Control and Status Register
0x00130000	USB_FIFO	FIFO for USB-Interface
0x001ff000	VIC_IRQ_STAT	IRQ Status Register
0x001ff004	VIC_FIQ_STAT	FIQ Status Register
0x001ff008	VIC_RAW_INT_STAT	Raw Interrupt Status Register
0x001ff00c	VIC_INT_SEL	Interrupt Select Register
0x001ff010	VIC_INT_EN	Interrupt Enable Register
0x001ff014	VIC_INT_EN_CLR	Interrupt Enable Clear Register
0x001ff018	VIC_SWI	Software Interrupt Register
0x001ff01c	VIC_SWI_CLR	Software Interrupt Clear Register
0x001ff020	VIC_PROT_EN	Protection Enable Register
0x001ff030	VIC_VECT_ADDR	Vector Address Register
0x001ff034	VIC_DFLT_VECT_ADDR	Default Vector Address Register
0x001ff100	VIC_VECT_ADDR0	Vector Address Register 0
0x001ff104	VIC_VECT_ADDR1	Vector Address Register 1
0x001ff108	VIC_VECT_ADDR2	Vector Address Register 2
0x001ff10c	VIC_VECT_ADDR3	Vector Address Register 3
0x001ff110	VIC_VECT_ADDR4	Vector Address Register 4
0x001ff114	VIC_VECT_ADDR5	Vector Address Register 5
0x001ff118	VIC_VECT_ADDR6	Vector Address Register 6
0x001ff11c	VIC_VECT_ADDR7	Vector Address Register 7
0x001ff120	VIC_VECT_ADDR8	Vector Address Register 8
0x001ff124	VIC_VECT_ADDR9	Vector Address Register 9
0x001ff128	VIC_VECT_ADDR10	Vector Address Register 10

0x001ff12c	VIC_VECT_ADDR11	Vector Address Register 11
0x001ff130	VIC_VECT_ADDR12	Vector Address Register 12
0x001ff134	VIC_VECT_ADDR13	Vector Address Register 13
0x001ff138	VIC_VECT_ADDR14	Vector Address Register 14
0x001ff13c	VIC_VECT_ADDR15	Vector Address Register 15
0x001ff200	VIC_VECT_CTRL0	Vector Control Register 0
0x001ff204	VIC_VECT_CTRL1	Vector Control Register 1
0x001ff208	VIC_VECT_CTRL2	Vector Control Register 2
0x001ff20c	VIC_VECT_CTRL3	Vector Control Register 3
0x001ff210	VIC_VECT_CTRL4	Vector Control Register 4
0x001ff214	VIC_VECT_CTRL5	Vector Control Register 5
0x001ff218	VIC_VECT_CTRL6	Vector Control Register 6
0x001ff21c	VIC_VECT_CTRL7	Vector Control Register 7
0x001ff220	VIC_VECT_CTRL8	Vector Control Register 8
0x001ff224	VIC_VECT_CTRL9	Vector Control Register 9
0x001ff228	VIC_VECT_CTRL10	Vector Control Register 10
0x001ff22c	VIC_VECT_CTRL11	Vector Control Register 11
0x001ff230	VIC_VECT_CTRL12	Vector Control Register 12
0x001ff234	VIC_VECT_CTRL13	Vector Control Register 13
0x001ff238	VIC_VECT_CTRL14	Vector Control Register 14
0x001ff23c	VIC_VECT_CTRL15	Vector Control Register 15
0x0016295c	PWM0_CFG	PWM 0 Configuration Register
0x0016395c	PWM1_CFG	PWM 1 Configuration Register
0x00162960	PWM0_STAT	PWM 0 Status Register
0x00163960	PWM1_STAT	PWM 1 Status Register
0x00162964	PWM0_TP	PWM 0 Period
0x00163964	PWM1_TP	PWM 1 Period
0x00162968	PWM0_TU	PWM 0 Channel U Low Phase Width
0x00163968	PWM1_TU	PWM 1 Channel U Low Phase Width
0x0016296c	PWM0_TV	PWM 0 Channel V Low Phase Width
0x0016396c	PWM1_TV	PWM 1 Channel V Low Phase Width
0x00162970	PWM0_TW	PWM 0 Channel W Low Phase Width
0x00163970	PWM1_TW	PWM 1 Channel W Low Phase Width
0x00162974	PWM0_TD	PWM 0 Dead Time Counter Preload
0x00163974	PWM1_TD	PWM 1 Dead Time Counter Preload
0x00162980	PWM0_CNT	Actual Counter Motor PWM 0 Period
0x00163980	PWM1_CNT	Actual Counter Motor PWM 1 Period
0x00162988	PWM0_STRTIME	Captured System Time at Start Point of Motor PWM 0 Period
0x00163988	PWM1_STRTIME	Captured System Time at Start Point of Motor PWM 1 Period
0x00162978	RPWM0_TP	Resolver PWM 0 Period
0x00163978	RPWM1_TP	Resolver PWM 1 Period
0x0016297c	RPWM0_TR	Resolver PWM 0 Pulse
0x0016397c	RPWM1_TR	Resolver PWM 1 Pulse
0x00162984	RPWM0_CNT	Actual Counter Resolver PWM 0 Period
0x00163984	RPWM1_CNT	Actual Counter Resolver PWM 1 Period

0x0016298c	RPWM0_STRTIME	Captured System time at Start Point of Resolver PWM 0 Period
0x0016398c	RPWM1_STRTIME	Captured System time at Start Point of Resolver PWM 1 Period
0x0016399c	ENC_STAT	Encoder Position and Capture Status
0x00163990	ENC_CFG	Encoder Configuration Register
0x00163998	ENC_CMD	Encoder Command Register
0x001639a0	ENC0_POS	Actual Position Encoder 0
0x001639a8	ENC1_POS	Actual Position Encoder 1
0x001639a4	ENC0_NULL_POS	Sampled Null Position Encoder 0
0x001639ac	ENC1_NULL_POS	Sampled Null Position Encoder 1
0x001639b0	ENC0_EDGE_TIME	System Time at Last Edge of Encoder 0
0x001639b4	ENC1_EDGE_TIME	System Time at Last Edge of Encoder 1
0x00163994	ENC_CFG_CAPT	Encoder Capture Configuration Register
0x001639b8	ENC_CAPT0	Encoder Capture Register 0
0x001639bc	ENC_CAPT1	Encoder Capture Register 1
0x001639c0	ENC_CAPT2	Encoder Capture Register 2
0x001639c4	ENC_CAPT3	Encoder Capture Register 3
0x0017c09c	ADC	Analog Digital Converter
0x00100010	PHY_CTRL	PHY Control Register
0x00100020	PHY_CLK_RATE_MUL_ADD	PHY Clock Rate Multiplier Add Value
0x00164000	PTR_FIFO_BASE	Pointer FIFO Base Address
0x00164080	PTR_FIFO_BOR_BASE	Pointer FIFO Upper Border Base Address
0x00164100	PTR_FIFO_RESET	Pointer FIFO Reset Vector
0x00164104	PTR_FIFO_FULL	Pointer FIFO Full Vector
0x00164108	PTR_FIFO_EMPTY	Pointer FIFO Empty Vector
0x0016410c	PTR_FIFO_OVER	Pointer FIFO Overflow Vector
0x00164110	PTR_FIFO_UNDER	Pointer FIFO Under-Run Vector
0x00164180	PTR_FIFO_FILL_LVL_BASE	Pointer FIFO Fill Level Base Address
0x00101000	CRC_VAL	Calculated CRC Value
0x00101004	CRC_IN_DATA	CRC Input Data
0x00101008	CRC_POLYNOMIAL	Polynomial for CRC Calculation
0x0010100c	CRC_CFG	CRC Configuration Register
0x00164400	IRQ_XP0	IRQs between XPEC0 and ARM
0x00164404	IRQ_XP1	IRQs between XPEC1 and ARM
0x00164408	IRQ_XP2	IRQs between XPEC2 and ARM
0x0016440c	IRQ_XP3	IRQs between XPEC3 and ARM

Document History

01/03/06	Corrected IOC_CR and IOC_MR Register description	JL
12/04/07	Revised chapter 2.2 (IOC_CR and IOC_MR Register)	JL
16/04/07	Revised chapter 2.3 (RESET)	JL
17/04/07	Revised chapter 7 (changed description)	JL
20/04/07	Revised register descriptions of DPM_ARM_IF_CFG1/0	JL