An Expresso family device

OXFORD SEMICONDUCTOR

Data Sheet

PCI EXPRESS

OXPCIe952
PCI Express Bridge to Dual Serial & Parallel Port

Features

- Flexible Configuration
  - Dual UART
  - Single UART plus IEEE 1284 parallel port
- PCI Express® End-point Controller
  - Single lane with integrated SerDes
  - PCI Express base spec 1.1 compliant
  - PCI Power Management 1.2 compliant
  - MSI/MSI-X compatible
  - DMA/bus mastering facility for both UARTs
  - ASPM (L0S, L1) Link power management
- Serial Ports
  - High performance Oxford 950 UARTs
  - Asynchronous baud rates up to 15 Mbps
  - 128-byte deep transmit/receive FIFOs
  - 9, 8, 7, 6 & 5-bit data framing
  - Flexible clock prescaler from 1 to 31.875
  - Automated in-Band, Xon/Xoff flow control
  - Automated hardware flow control
  - Advanced FIFO fill management
  - RS232, RS422, RS485 and IrDA operation
  - Programmable RS485 Turn-around delay
  - 450 through 950 software compatibility
- Parallel Port
  - IEEE 1284-compliant SPP/EPP/ECP
- General
  - ExpressCard™, Mini CARD™ & AIC compatible
  - 8 user-configurable GPIOs /PWMs
  - Device parameters configurable via EEPROM
  - 3.3-V operation
  - 1.8-V, 2.5-V or 3.3-V UART & GPIO I/O voltage
  - 120-pin TFBGA package
  - Industrial temperature range -40°C to 85°C
  - Broad operating system support including device drivers for Windows Vista™, Windows® XP, Windows 2000, Windows CE & Linux

Description

The OXPCIe952 is a single-chip solution for PCI Express-based high-performance serial and parallel connectivity that provides a combination of rich features and user configurability to enable highly differentiated end products.

The device combines a fully integrated, single-lane PCI Express end-point controller and SerDes with two high-performance Oxford Semiconductor 950 UARTs, an IEEE1284 compliant SPP/EPP/ECP parallel port and user-defined GPIOs/PWMs. Configurable as either two serial channels, or one serial channel and one IEEE 1284 parallel port, the device accommodates popular Add-in Card formats and with comprehensive power management support and the reassurance of industrial temperature range, the OXPCIe952 is the ideal choice for power and temperature sensitive ExpressCard™ and Mini CARD™ applications.

The OXPCIe952 achieves outstanding performance by combining the class-leading 15 Mbps asynchronous data rates and 128-byte transmit and receive FIFOs of the Oxford Semiconductor 950 UART, with advanced system management features, such as MSI/MSI-X interrupt handling and bus master DMA, to maximize data throughput while substantially reducing CPU and system overheads. Each 950 UART has a full modem interface and includes advanced features such as hardware- accelerated out-of-band and in-band flow control, readable FIFO levels and RS485 turnaround delay for further performance optimization, while the flexible clock prescaler provides scope for the widest possible range of baud rates.

The device can be configured at power-up using an external EEPROM to take advantage of a range of possible device and in-system customizations that are easily defined and programmed using Oxide, the Oxford Semiconductor graphical development tool.

The advanced features of the device are supported by dedicated Oxford Semiconductor device drivers that cover a broad range of operating systems including Windows Vista/XP/2K/CE & Linux, are quality assured through exhaustive testing and are WHQL approved.

Contents

Features 1

Description 1

Contents 2

Device Modes 3

OXPCIe952 Pin Descriptions 4

Configuration Space & Base Address Registers 10
- OXPCIe952 Configuration Space - - - - - - - - - - - - - - - - -10
- Base Address Register Allocation - - - - - - - - - - - - - - - - -12

System Overview 14
- OXPCIe952 Clocking and Reset Scheme - - - - - - - - - - - - -14
- Personality Application -14
- Interrupt Management -15
- PCI Express Interface -16
- Power Management -16
- Power Supply Management -16

OXPCIe952 Functions 17
- UART Function -17
- Reset Configuration -27
- Transmitter & Receiver FIFOs - - - - - - - - - - - - - - - - - -28
- Line Control & Status -31
- UART Interrupts -34
- Modem Interface -39
- Automatic Flow Control -42
- Additional Features -45
- Parallel Port Function -59
- GPIO Function -62

EEPROM Interface & Programming Capabilities 71
- Overview -71
- EEPROM Zone Allocation -73

Operating Conditions 76
- Maximum Ratings -76
- Electrical Characteristics -77
- AC Electrical Characteristics - - - - - - - - - - - - - - - - - - -78

Package Mechanical Drawings 80

Ordering Information 81

Contacting Oxford Semiconductor 82

Revision Information 82

Device Modes

The OXPCIe952 allows up to two serial devices or one serial and one parallel device to be efficiently connected to a PCIexpress-enabled host PC. The OXPCIe952 mode pins enable a broad range of connectivity options, including the choice of using the UARTs in either PCIE native mode, using Oxford Semiconductor UART device drivers, or in legacy mode, using standard operating system drivers. Note that the OXPCIe952 parallel port is intended for use with standard operating system drivers.

Table 1 illustrates all the physical mode settings for the OXPCIe952 and hence the combination of functions supported at any one time. As well as the mode settings (MODE[2:0] pins) there are two additional configuration pins (UART_EN and GPIO_EN). These additional pins independently enable/disable UART and GPIO functionality. Note that they are applicable to native-mode configurations only.

Table 1 Physical Mode Settings for the OXPCIe952

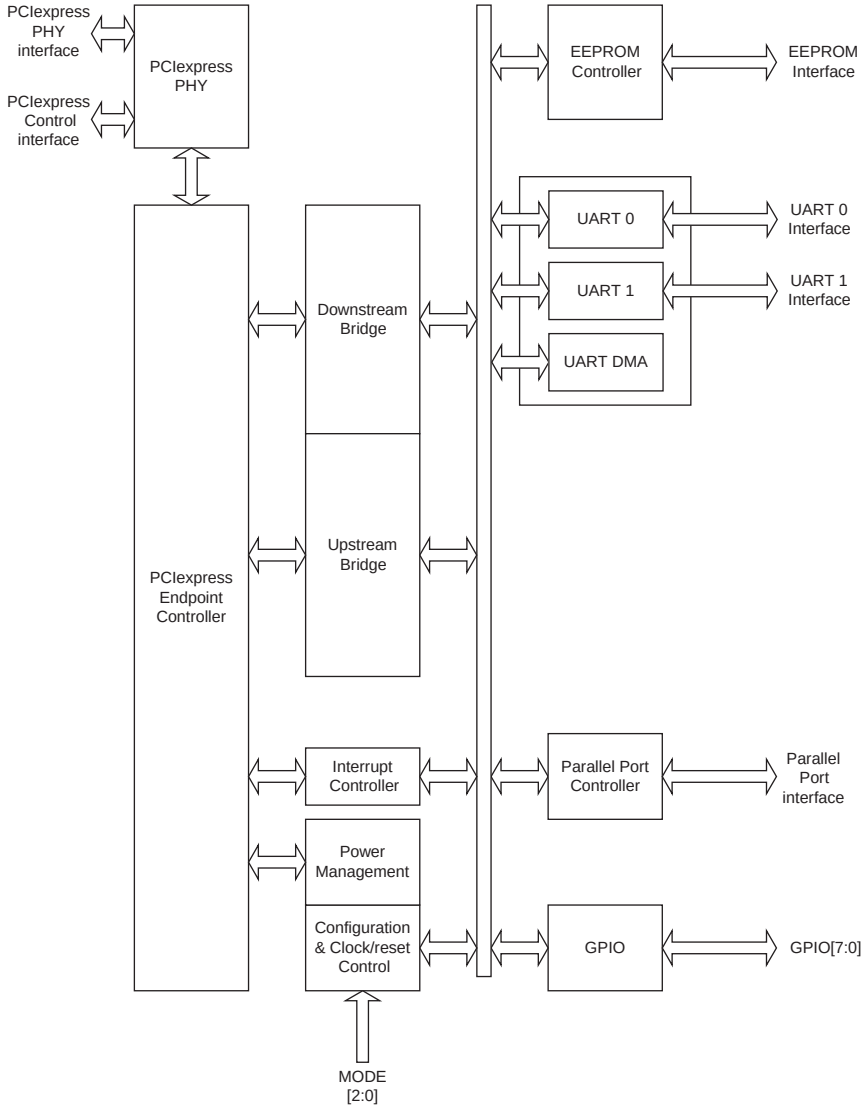
UART_EN Pin	GPIO_EN Pin	MODE[2:0] Pins	Function 0	Function 1	Function 2	Function 3
n/a	1(4)	000	PPORT(1)	1 legacy UART(1)	GPIO(2)	not visible
1(3)	1(4)	001	PPORT(1)	not visible	GPIO(2)	1 native UART(2)
n/a	1(4)	010	1 legacy UART(1)	not visible	GPIO(2)	not visible
0	1	011	GPIO(2)	not visible	not visible	not visible
1	0	011	1 native UART(2)	not visible	not visible	not visible
1	1	011	GPIO(2)	1 native UART(2)	not visible	not visible
n/a	1(4)	100	1 legacy UART(1)	1 legacy UART(1)	GPIO(2)	not visible
0	1	101	GPIO(2)	not visible	not visible	not visible
1	0	101	2 native UARTs(2)	not visible	not visible	not visible
1	1	101	GPIO(2)	2 native UARTs(2)	not visible	not visible

Notes:

1 These functions are driven by standard operating system drivers.
2 These functions are driven by Oxford Semiconductor drivers.
3 Setting this pin to logic 0 disables the UART function.
4 Setting this pin to logic 0 disables the GPIO function.

Figure 1 on page 4 shows the OXPCIe952 architecture.

Figure 1 OXPCIe952 Block Diagram



OXPCIe952 Pin Descriptions

Table 2 on page 5 lists the OXPCIe952 pins and descriptions.

Table 2 OXPCIe952 Pin Descriptions (Sheet 1 of 4)

Pin	No Bits	Type	Name	Description
EEPROM (4 pins)[1]				
M4	1	M_O_6	EECK	EEPROM serial clock (745KHz).
M3	1	M_O_6	EECS	EEPROM chip select (active high).
N3	1	M_I_6	EEDI	EEPROM data in.
N5	1	M_O_6	EEDO	EEPROM data out.
GPIO (8 pins)[2]				
D1, D2, D3, C2, B1, A1, B2, C3	8	M_B_6	GPIO_[7:0]	General Purpose IO.
PCIexpress Control (3 pins)[3]				
A4	1	5_O_T	nWAKE	PCIexpress edge connector wake (WAKE#).
B4	1	5_O_T	nCLKREQ	PCIexpress edge connector clock request (CLKREQ#).
A3	1	5_I_S	nPERST	PCIexpress edge connector reset (PERST#).
UART (2*8 = 16 pins)[1]				
M9, L13	2	M_I	nDCD[1:0]	Data carrier detect (active low).
N10, H12	2	M_I	nDSR[1:0]	Data set ready (active low). If automated nDSR flow control is enabled, upon deassertion of the nDSR pin, the transmitter will complete the current character and enter the idle mode until the nDSR pin is reasserted. Note: flow control characters are transmitted regardless of the state of the nDSR pin.
M8, L12	2	M_I	SIN[1:0]	Serial data inputs. IrDA mode => IrDA_In. UART IrDA data input when IrDA mode is enabled (i.e. when MCR[6] of the corresponding channel is set in enhanced mode).
N9, K12	2	M_O_3	nRTS[1:0]	Request to send (active low). If automated nRTS flow control is enabled, the nRTS pin is deasserted and reasserted whenever the receiver FIFO reaches or falls below the programmed thresholds, respectively.
M10, L11	2	M_O_3	SOUT[1:0]	Serial data outputs. IrDA mode => IrDA_Out. UART IrDA data output when IrDA mode is enabled (i.e. when MCR[6] of the corresponding channel is set in enhanced mode).

Table 2 OXPCIe952 Pin Descriptions (Sheet 2 of 4)

Pin	No Bits	Type	Name	Description
N12, J12	2	M_I	nCTS[1:0]	Clear to send (active low). If automated nCTS flow control is enabled, upon deassertion of the nCTS pin, the transmitter will complete the current character and enter the idle mode until the nCTS pin is reasserted. Note: flow control characters are transmitted regardless of the state of the nCTS pin.
N11, M13	2	M_O_3	nDTR[1:0]	Data terminal ready (active low). If automated nDTR flow control is enabled, the nDTR pin is asserted and deasserted if the receiver FIFO reaches or falls below the programmed thresholds, respectively. RS485 half-duplex mode => 485_En. This pin may be programmed to reflect the state of the transmitter empty bit to automatically control the direction of the RS485 transceiver buffer (see register ACR[4:3] on page 47).
M11, J13	2	M_I	nRI[1:0]	Ring indicator (active low).
Parallel Port (18 pins) Note 3				
B11,A11,B10,A10, B9, A9,C9,C8	8	5_B_8	P_DATA[7:0]	Data bus
C11	1	5_I	nERR	SPP mode => nFault (active low). Set low by the peripheral to indicate that an error has occurred.
C12	1	5_I	SLCT	SPP mode => Select (active high). Set high by the peripheral to indicate that the peripheral is online.
C13	1	5_I	BUSY	SPP mode => Busy (active high). Driven high by the peripheral to indicate that he peripheral is not ready to receive data. EPP mode => nWait (active low). Driven inactive as a positive acknowledgment from the peripheral that transfer of data or address is completed. It should be driven active as an indication that the device is ready for the next address or data transfer.
B12	1	5_I	PE	SPP mode => PError (active high). Driven high by the peripheral to indicate that the peripheral has encountered an error in its paper path.
A12	1	5_I	nACK	SPP mode => nAck. Pulsed low by the peripheral to acknowledge transfer of the data byte from the host. EPP mode => Intr (active low). Pulsed low by the peripheral to interrupt the host.

Table 2 OXPCIe952 Pin Descriptions (Sheet 3 of 4)

Pin	No Bits	Type	Name	Description
B8	1	5_O_8_T 5_O_8	nINIT	SPP mode => nInit (active low). Pulsed low in conjunction with IEEE 1284 Active low to reset the interface and force a return to Compatibility Mode idle phase. EPP mode => nInit (active low). When driven low this signal initiates a termination cycle that results in the interface returning to Compatibility Mode.
B7	1	5_O_8_T 5_O_8	nAFD	SPP mode => nAutoFd (active low). Set low by the host to put printer into auto line feed mode. EPP mode => nDStrb (active low). Used to denote a data cycle.
A8	1	5_O_8_T 5_O_8	nSTB	SPP mode => nStrobe (active low). Set active low to transfer data into the input latch of the peripheral. Data is valid while nStrobe is low. EPP mode => nWrite (active low). Set low to denote an address or data write operation to the peripheral. Set high to denote an address or data read operation from the peripheral.
C6	1	5_O_8_T 5_O_8	nSLIN	SPP mode => nSelectIn (active low). Set low by host to select peripheral. EPP mode => nAStrb (active low). Used to denote an address cycle.
A7	1	5_O_8	DIR	Line driver bidirectional control. Logic ‘1’ => data is output. Logic ‘0’ => data is input.
PCIexpress PHY (7 pins)				
K1	1	O	TX_P	High speed differential transmit line.
K2	1	O	TX_N	High speed differential transmit line.
H1	1	I	RX_P	High speed differential receive line.
H2	1	I	RX_N	High speed differential receive line.
F1	1	I	REFCLK_P	Differential reference clock input.
F2	1	I	REFCLK_N	Differential reference clock input.
G3	1	A	REXT	Reference resistor connection. 191Ω 1% 100ppm/deg C precision resistor to ground.
M2	1	A	PCIe_TEST	PCIexpress PHY test pin (active high). Should be tied to VSS for normal operation.
Integrated 3.3V -> 1.2V PMU (5 pins)				
D12	1	3_I	SHUTDOWN	PMU shutdown pin (active high). Should be tied to VSS for normal operation.
D11	1	5_O_8	POWERGOOD	PMU 1.2V output supply status good flag (active high).

Table 2 OXPCIe952 Pin Descriptions (Sheet 4 of 4)

Pin	No Bits	Type	Name	Description
G13,G12	2	O	VOUT	PMU buck 1.2V switch output.
E13	1	I	FB	Feedback signal for PMU comparator.
MISC (6 pins)				
B6	1	5_I	TEST(3)	PCIe952 test pin (active high). Should be tied to VSS for normal operation.
A6,A5,B5	1	5_I	MODE[2:0](3)	Mode configuration pins
N6	1	M_I	GPIO_EN(1)	GPIO enable pin (active high)
M6	1	M_I	UART_EN(1)	UART enable pin (active high)
B3	1	5_I_S	nPOWER_RST(3)	Power on reset input. See section "Resets" on page 14 for timing of this signal.
Power and Ground (51 pins)				
E2, J3	2	P	VP_1V2	PCIexpress PHY 1.2V power supply.
E1, K3	2	P	VP_3V3	PCIexpress PHY 3.3V power supply.
G1, G2, J1	3	P	GND	PCIexpress PHY ground.
E11	1	P	VINQ	PMU quiet supply for switcher.
F13, F12	2	P	VIN[1:0]	PMU supply for DC-DC.
E12	1	P	VSSQ_D	PMU digital ground for switcher.
D13	1	P	VSSQ_A	PMU analogue ground for switcher.
B13, C7, A2,	3	P	VDDIO0	IO power supply for 3V (5V tolerant) interface (parallel port, mode, config)
C1	1	P	VDDIO1	IO power supply for GPIO interface (1.8V, 2.5V or 3.3V)
N2, N4, N8, M12, K13,	5	P	VDDIO2	Common IO power supply for UART[1:0] interfaces, EEPROM interface and GPIO_EN, UART_EN (1.8V, 2.5V or 3.3V)
C4, C10, H11, K11, L4, L6, L8, L10	8	P	VDDCORE	Core power supply (1.2V)
A13, C5, F3, F11, G11, H3, J2, J11, L1, L2, L3, L5, L7, L9, N1, N7, N13	17	P	VSS	Digital I/O and core ground
E3, M1, M5, M7, H13	5	P	VGG	3.3V reference for Multi-voltage IO

Notes:

1 Powered by multi voltage power supply VDDIO2
2 Powered by multi voltage power supply VDDIO1
3 Powered by 3.3V power supply VDDIO0

4 Type Key: format is [(W_)X(Y)(_Z(A))] where the following conventions apply:

W-Tolerance[1]		X-Type		Y -Pull		Z-Drive[2]		A-Other	
5	5V	I	Input	U	Pull up	4	4mA	T	Tristate
3	3.3V	O	Output	D	Pull down	6	6mA		Normal
2	2.5V	B	Bidirectional		None	8	8mA	S	Schmitt
M	Multivoltage: @ 1.8V is 2.5V tolerant. @2.5V is 3.3V tolerant.	A	Analogue						
		P	Power						

Notes:

1 When parameter W value is M, denotes multi-voltage cell. The power group can be set to 1.8, 2.5 or 3.3V.

2 When parameter W value is M and Z is 6, output drive depends on VDDIO. If 1.8V then drive value is 6mA. If 2.5V or 3.3V then output drive value is 10mA.

Configuration Space & Base Address Registers

The register descriptions for the OXPCIe952 individual functions are located in the sections describing those interfaces later in this document. The OXPCIe952 configuration space and base address register locations are described below.

OXPCIe952 Configuration Space

Figure 2 shows the OXPCIe952 configuration space, which is allocated for each function and is always 32 bits wide.

Figure 2 OXPCIe952 Configuration Space (Per Function)

31 – 16	15 – 0	
Device ID	Vendor ID	00h
Not Targeted		04h
Classcode	Revision ID	08h
Note Targeted		0Ch
BAR 0		10h
BAR1		14h
BAR2		18h
BAR[5:3] Disabled		1Ch
		20h
		24h
Not Targeted		28h
Subsystem ID	Subsystem Vendor ID	2Ch
Not Targeted		30h
		34h
		38h
		3Ch

Device ID

Each OXPCIe952 function provides a unique device ID, as shown in Table 3 on page 11. The device ID and subsystem device ID fields are derived from MODE[2:0] pins, the GPIO_EN enable pin and UART_EN enable pin. The number of the target function is also incorporated.

Table 3 Device ID Field

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Description	1	1	0	0	0	0	0	1	0	MODE[2:0](1)			UART_EN	GPIO_EN	function number(1)	

Notes: 1 #1 As defined in "Device Modes" on page 3.

Vendor ID

The Vendor ID defaults to **1415h**. This can be set to a customer-specific value using the Oxford Semiconductor Oxide utility.

Class Code

Each supported PCI Express function on the OXPCIe952 has a default class-code definition, as shown in Table 4.

Table 4 Default Class Code Definition

Category	Class Code
Parallel port	0x070102
UART	0x070002
GPIO	0x088000

Revision ID

The Revision ID defaults to **0000h**. This can be set to a customer-specific value using the Oxford Semiconductor Oxide utility.

Subsystem ID

The Subsystem ID defaults to the same value as the Device ID. This can be set to a customer-specific value using the Oxford Semiconductor Oxide utility.

Subsystem Vendor ID

The Subsystem Vendor ID defaults to the same value as the Vendor ID. This can be set to a customer-specific value using the Oxford Semiconductor Oxide utility.

Device Serial Number—PCI Express DSN

A 64-bit IEEE-EUI64 is assigned to the OXPCIe952; it can be read using the DSN capability of the configuration space in Function 0. The hexadecimal encoding is shown in Table 5.

Table 5 Device Serial Number

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
63:40	Reserved (Oxford Semiconductor organizationally unique identifier (OUI))	W	R	0x0030E0
39:0	Reserved	W	R	0x1111000300

Base Address Register Allocation

Each OXPCIe952 function has access to a unique set of base address registers (BARs).

Parallel Port BAR Allocation

The parallel port is always a PCIe legacy-mode device. Being a legacy device, the BAR allocation is mandated by standard operating system legacy driver software. Any function with a PPORT has a BAR map as shown in Table 6.

Table 6 Parallel Port BAR Allocation

BAR	Type	Size	Target Base Address at Module
0	IO	8	0x000 (PPORT)
1	IO	4	0x008 (PPORT)
2 to 5	n/a	0	n/a

Legacy UART BAR Allocation

When the UART(s) are enabled in legacy form, their allocated function has a BAR map as shown in Table 7.

Table 7 Legacy UART BAR Allocation

BAR	Type	Size	Target Base Address at Module
0	IO	8	0x000 (UART)
1 to 5	n/a	0	n/a

Native UART BAR Allocation

When the UART(s) are enabled in native form, their allocated function has access to three BARs as follows:

- UARTs and UARTs DMA (UDMA) access a 16-Kbyte memory space via BAR 0
- PCIe interface uses BAR1 to support MSI-X
- BAR 2 is used to support EEPROM programming via the Oxford Semiconductor Oxide utility

The dual-mapping of the full 2-Mbyte memory space is necessary because the UART, unlike GPIO, supports MSI-X, which consumes BAR1 and leaves BAR2 available for the device configuration support. This is shown in Table 8.

Table 8 UART BAR Allocation

BAR	Type	Size	Target Base Address at Module
0	MEM	16K	0x000 UART & UDMA[(1)]
1	MEM	2M	All visible Modules & MSI-X (Used for MSI-X)
2	MEM	2M	All visible Modules & MSI-X (Used for EEPROM)
3 to 5	n/a	0	n/a

Note:

1 See Table 14 for full details on register mapping

GPIO BAR Allocation

When the GPIO is enabled, its allocated function has access to two BARs as follows:

- GPIO accesses a 1-Kbyte memory space accessed via BAR 0
- BAR 1 is used to support EEPROM programming via the Oxford Semiconductor Oxide utility

Table 9 shows the GPIO BAR allocation.

Table 9 GPIO BAR Allocation

BAR	Type	Size	Target Base Address at Module
0	MEM	1K	0x000 GPIO
1	MEM	2M	ALL visible Modules & MSI-X (Used for EEPROM)
2 to 5	n/a	0	n/a

Note:

1 See Table 40 for full details on register mapping

System Overview

This section gives an overall view of the OXPCIe952. Subsequent sections deal with the OXPCIe952 external interfaces.

OXPCIe952 Clocking and Reset Scheme

Clock Sources

The 100-MHz clock supplied via the PCI Express edge connector is the only external clock source for the OXPCIe952. The internal system clock, supplied by the PCI express PHY PLL, is carefully managed within the device. A number of techniques are used to ensure a low internal clock toggle rate, which contributes to the low overall power consumption of the device.

The system clock is dynamically replaced by a very low frequency, internally generated, standby clock, which the device uses when the device goes into $D3_{COLD}$ (as defined in the PCI Bus Power Management Interface Specification 1.2). On exiting $D3_{COLD}$ the low frequency clock is seamlessly replaced by the PCI Express PHY PLL generated clock.

Resets

The PCI Express power management system requires the use of *sticky registers*, which are not cleared on receiving the external reset-pulse from the PCI Express edge connector nPERST. They maintain their context while the auxiliary PCI Express power supply, 3v3aux, is present.

The sticky registers, however, must be initialized at power-up time to their default values, as specified by the PCI Express 1.1 standard, for which the OXPCIe952 PCI Express core provides two reset domains. This necessitates an additional power-on reset signal, nPOWER_RST, which is used to signal the initial device power-up. In addition, nPOWER_RST gates the nPERST signal, so that the whole device remains fully in reset until the power-on reset has finished.

Personality Application

The OXPCIe952 *personality* is specified by a combination of its mode pins, the GPIO enable (GPIO_EN) and UART enable (UART_EN) pins and the contents of the external configuration EEPROM.

At power up, the OXPCIe952 performs a two-pass personality assignment. In the first pass the I/O pins are interrogated for configuration information, the device is put into the appropriate functional mode and the required functions are made available. The second pass involves reading the EEPROM and transferring the information contained in it to appropriate device registers.

The EEPROM is organized into five separate zones, each dedicated to enabling parameter customization of a specific part of the OXPCIe952, as shown in Table 10.

Table 10 OXPCIe952 EEPROM Zones

EEPROM Zone	OXPCIe952 Function Access
Zone 0	EEPROM Content Description
Zone 1	PCIexpress configuration space & PHY access
Zone 2	UART
Zone 3	Parallel port
Zone 4	GPIO
Zone 5	Not used

Take care when modifying the default configuration of the PCI Express registers—incorrect settings may make the device inaccessible to the host system.

Interrupt Management

The OXPCIe952 fully supports MSI and MSI-X. The MSI capability is as defined in the *PCI Local Bus Specification, Revision 3.0*. The MSI-X capability is as defined in the MSI-X ECN for the *PCI Local Bus Specification, Revision 3.0*.

The OXPCIe952 is capable of generating both level- and edge-triggered interrupt messaging. By default the legacy PCI-based INTx level sensitive wired **OR** interrupt emulation is the main mechanism for sending interrupts from the device to the host system. Newer operating systems, such as Windows Vista and Linux 2.6 support MSI/MSI-X as the native interrupt mechanism of PCI Express, which is specified in the PCI standard but is not supported by older operating systems including Windows XP. The OXPCIe952 supports both styles of interrupts.

Table 11 shows which interrupt type is supported by specific OXPCIe952 functions.

Table 11 Interrupt Types Per Function

Mapping	Legacy INT_x Support	MSI—Single Vector	MSI-X
PPORT legacy mode	yes	yes	no
UART native mode	yes	no	yes
UART legacy Mode	yes	yes	no
GPIO native mode	yes	yes	no

Each function has full MSI or MSI-X capability as mandated by the PCI Express specification.

The global registers supporting UART and GPIO interrupts are shown in Table 14 and Table 40 respectively.

Table 12 shows the location of the UART MSI-X registers, which are reserved for system use.

Table 12 UART MSI-X Vector Location

Address	Bits	Type	Description
0x2000	16	RW	PBA table for the UART module MSI-X
0x3000	128	RW	Table Offset entry for the UART MSI-X Vector

PCI Express Interface

The OXPCIe952 provides a fully-featured X1 (single-lane) PCI EXPRESS interface which is fully compliant with both the *PCI EXPRESS Base Specification, Revision 1.1* and the *PCI Bus Power Management Interface Specification, Revision 1.2.*

Power Management

The OXPCIe952 includes the complete PCI Express Power management capabilities as defined in the PCI Bus Power Management Interface Specification, Revision. 1.2, including full support for D0 and D3 device states and PME message support.

Power Supply Management

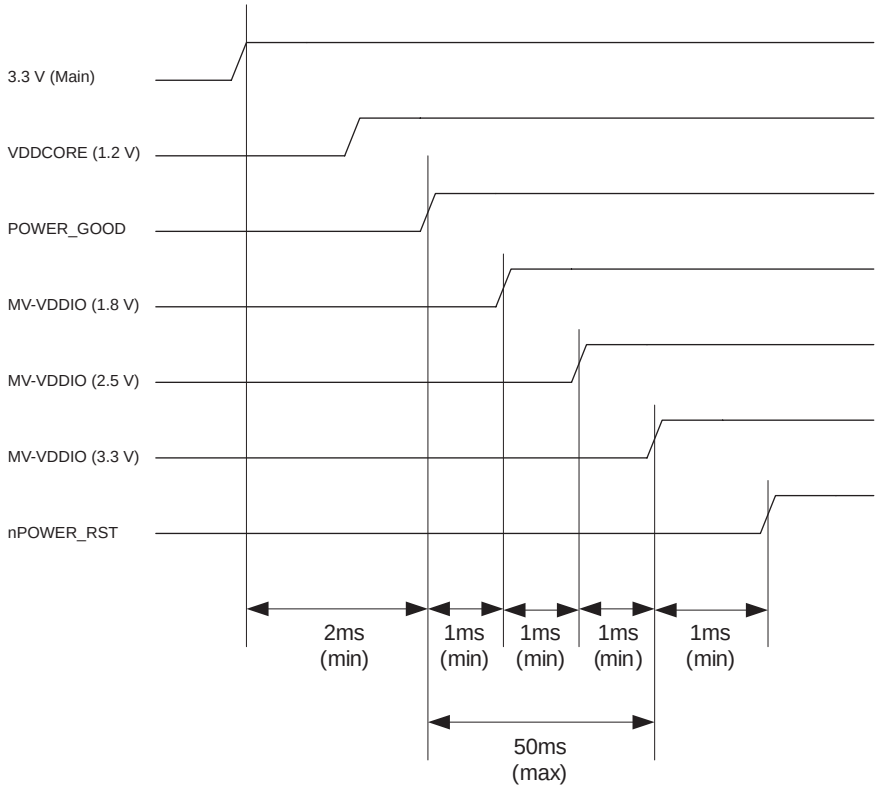
All OXPCIe952 power supply requirements can be met from a single 3.3-V power source, which is typically provided by the PCI Express edge connector.

The UART and GPIO interfaces can be independently configured to operate at 1.8V, 2.5V or 3.3V by connecting the power supply pins associated with the appropriate functional pin grouping to the required interface voltage. In cases other than 3.3V, the system designer must ensure that a suitable voltage source is available.

The OXPCIe952 low-power 1.2V internal core is supplied via the integral power supply system, PMU, which provides high efficiency energy conversion from 3.3V to 1.2V.

It is recommended that sequencing of the multi-voltage IO power supplies should be performed. For power up, after the core voltage is stable (flagged by the POWER_GOOD pin) the IO supplies must be sequenced with the lowest supply (1.8V) first to the highest supply last (3.3V). Once all supplies are stable the power on reset (nPOWER_RST) pin can be de-asserted to allow start of operation for the OXPCIe952. This sequencing (with timing) is given in Figure 3 on page 17.

Figure 3 Power-up Sequence



For the OXPCIe952 there are only two multi-voltage domains (VDDIO0 and VDDIO1). However, even if both VDDIO0 and VDDIO1 are to be at 3.3 V, they should still be enabled after the 1.2-V core supply is stable and so should not be directly connected to the main 3.3-V supply from the PCIexpress connector.

OXPCIe952 Functions

UART Function

In native mode, the two OXPCIe952 UARTs are enumerated as a single device function by the Oxford Semiconductor Multifunction Driver. In legacy operation, each UART is enumerated to a separate function.

The UARTs in the OXPCIe952 are based on that used in the well-proven OX16C95x UARTs, but with a number of significant enhancements.

UART Modes

The OXPCIe952 UART is designed to be used with Oxford Semiconductor UART drivers. These operate the UART exclusively in 950 mode.

However, the UART is register-compatible with the 16C450, 16C550, 16C654 and 16C750 UARTs. The operation of the 950 depends on a number of modes which are referenced throughout this data sheet. The FIFO depth and compatibility modes are tabulated in Table 13.

Table 13 UART Mode Configuration

UART Mode	FIFO Size	FiCR[0]	Enhanced Mode (EFR[4]=1)	FiCR[5] (guarded with LCR[7] = 1)
450	1	0	X	X
550	16	1	0	0
650	128	1	1	X
750	128	1	0	1
950*	128	1	1	X

Note:
1 950 mode configuration is identical to 650 configuration

450 Mode

After a hardware reset, bit 0 of the FiCR is cleared, making the 950 compatible with the 16C450. The transmitter and receiver FIFOs (referred to as the Transmit Holding Register and Receiver Holding Register) have a depth of one. This is referred to as byte mode. When FiCR[0] is cleared, all other mode selection parameters are ignored.

550 Mode

After a hardware reset, writing 1 to FiCR[0] sets the FIFO size to 16, providing compatibility with 16C550 devices.

750 Mode

Writing a 1 to FiCR[0] increases the FIFO size to 16. In a similar fashion to 16C750, the FIFO size can be further increased to 128 by writing a 1 to FiCR[5]. Note that access to FiCR[5] is protected by LCR[7]; i.e., to set FiCR[5], software should first set LCR[7] to remove the guard temporarily. When FiCR[5] is set, software should clear LCR[7] for normal operation.

16C750 additional features over the 16C550 are available as long as the UART is not put into enhanced mode (i.e. EFR[4] should be 0). The features are as follows:

- Deeper FIFOs
- Automatic RTS/CTS out-of-band flow control

650 Mode

The 950 is compatible with the 16C654 when EFR[4] is set, i.e., the device is in enhanced mode. Because 650 software drivers usually put the device into enhanced mode, running 650 drivers on the 950 results in 650 compatibility with 128-deep FIFOs, as long as FiCR[0] is set. Note that the 650 emulation mode of the 950 provides 128-byte deep FIFOs whereas the standard 16C654 has only 32-byte FIFOs.

650 mode has the same enhancements as the 16C750 over the 16C550, but they are enabled by different registers.

There are also additional enhancements over those of the 16C750 in this mode. They are as follows:

- Automatic in-band flow control
- Special character detection
- Infra-red IrDA format transmit and receive mode
- Transmit trigger levels
- Optional clock prescaler

950 Mode

The additional features offered in 950 mode generally only apply when the UART is in enhanced mode (EFR[4]=1). Provided FiCR[0] is set, in enhanced mode the FIFO size is 128.

950 mode and 650 mode configuration are identical; however, additional 950-specific features are enabled using the Additional Control Register (ACR). In addition to larger FIFOs and higher baud rates, the enhancements of the 950 over the 16C654 are as follows:

- Selectable arbitrary trigger levels for the receiver and transmitter FIFO interrupts
- Improved automatic flow control using selectable arbitrary thresholds
- nDSR/nDTR automatic flow control
- Transmitter & receiver can be optionally disabled
- Software reset of device
- Readable FIFO fill levels
- Optional generation of an RS-485 buffer enable signal
- Four-byte device identification (0x16C9500D)
- Readable status for automatic in-band and out-of-band flow control
- Flexible *M N*/8 clock prescaler

- Fixed 62.5-MHz clock input derived from the PCI Express 100MHz clock
- Programmable sample clock to allow data rates up to 15.625 Mbps
- 9-bit data mode

The 950 trigger levels are enabled when ACR[5] is set (bits 4 to 7 of FiCR are ignored). Then arbitrary trigger levels can be defined in RTL, TTL, FCL and FCH registers. The Additional Status Register (ASR) offers flow control status for the local and remote transmitters. FIFO levels are readable using RFL and TFL registers.

The UART has a flexible prescaler capable of dividing the system clock by any value between 1 and 31.875 in steps of 0.125. It divides the system clock by an arbitrary value in M N/8 format, where M and N are 5- and 3-bit binary numbers programmed in CPR[7:3] and CPR[2:0] respectively. This arrangement offers a great deal of flexibility when choosing how to synthesize arbitrary baud rates.

It is also possible to define the oversampling rate used by the transmitter and receiver clocks. The 16C450/16C550 and compatible devices employ 16 times oversampling, i.e., there are 16 clock cycles per bit. However, the 950 can employ oversampling rates from 4 to 16 by programming the TCR register. TCR defaults to 0x00 after a reset, which corresponds to a 16-cycle sampling clock. Writing 0x01, 0x02 or 0x03 also results in a 16-cycle sampling clock. To program any value from 4 to 15 it is necessary to write this value into TCR. For example, to set the device to a 13 cycle sampling clock, write 0x0D to TCR.

The 950 also offers 9-bit data frames for multi-drop industrial applications.

Extending CPR for Legacy UART

When operating in Legacy UART mode, the system drivers will assume the industry standard 1.8432MHz reference clock. As the Reference clock for the UARTs in the OXPCIe952 is 62.5MHz, all DLL/DLM values will be incorrect if no pre-scaling is done by the UART. In order to correct this effect the CPR register must divide the reference clock by 33.90842 which is approximated to 33.875 by default after a reset. As such the CPR has been extended to support a larger 6 bit integer range by using bit-0 of CPR2 to represent bit-6 of the integer portion of the clock pre-scalar

In order to maintain backward compatibility with software that only understands CPR, any writes to CPR will clear bit 0 of CPR2, thus returning the total CPR divider back to a range of 1 to 31.875 as in PCI products.

Writes to CPR2 do not affect CPR.

Thus CPR will default to 0xF and CPR2 will default to 0x1, giving a combined divisor of 33.875.

UART BAR Detailed Breakdown

When the OXPCIe952 UART function is enabled, the UART BAR actually accesses several internal locations over its 16-Kbyte aperture, as shown in Table 14.

Table 14 UART & UART DMA BAR Allocation

Bar Offset	Description
0x0000	Class code & Rev-ID
0x0004	Decimal Number of UARTs
0x0008	Global UART IRQ Status
0x000C	Global UART IRQ Enable
0x0010	Global UART IRQ Disable
0x0014	Global UART Wake Enable
0x0018	Global UART Wake Disable
0x001C..0x0FFF	Reserved
0x1000..0x10FF	UART[0] registers(1)
0x1100..0x110F	UART[0] DMA channels(2)
0x2000..0x10FF	UART[1] registers(1)
0x2100..0x110F	UART[1] DMA channels(2)

Notes:

1 See "OXPCIe952 UART Register Map" on page 23 and "OXPCIe952 UART Enhancements" on page 54 for details

2 See section "UART DMA" on page 56 for details

A breakdown of the individual registers is given in Tables 15 to 21.

Table 15 Class Code & Revision ID (Bar Offset 0x0000)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:8	Classcode	-	R	0x070002
7:0	Revision ID	W	R	0x0

Table 16 exemplifies the situation where MODE[2:0] pins have been set for two UARTs enabled.

Table 16 Decimal number of UARTs (Bar Offset 0x0004)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:2	Reserved	-	-	0x0000000
1:0	UART[1:0] enabled status. Bits are set when UART(s) are enabled. Value is dependant on MODE[2:0] pin setting.	W	R	See note 1

Note:

1 Reset value is either 0x01 or 0x02 depending on the MODE pins, unless overwritten by EEPROM to another smaller value.

Table 17 Global UART IRQ Status (Bar Offset 0x0008)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:2	Reserved	-	-	0x0000000
1:0	UART[1:0] IRQ status	-	R	XX

Table 18 Global UART IRQ Enable (Bar Offset 0x000C)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:2	Reserved	-	-	0x0000000
1:0	UART[1:0] IRQ enable status	W	RW	00

Table 19 Global UART IRQ Disable (Bar Offset 0x0010)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:2	Reserved	-	-	0x0000000
1:0	UART[1:0] IRQ disable status	W	RW	11

Table 20 Global UART Wake Enable (Bar Offset 0x0014)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:1	Reserved	-	-	0x000000
0	Global UART wake enable	W	RW	1

Table 21 Global UART Wake Disable (Bar Offset 0x0018)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:1	Reserved	-	-	0x000000
0	Global UART wake disable	W	RW	0

OXPCIe952 UART Register Map

Table 22 illustrates the OXPCIe952 UART directly-mapped address decoding regions. The baseline UART registers reside in the standard 0x00..0x07 region. All enhancements start at offset 0x80.

Table 22 UART Direct Access Register Map

Register Bank	Mapped Address Range
Baseline 550 registers	0x00 .. 0x07
Blank region	0x08 .. 0x7F
Extended 550 registers	0x80 .. 0x87
Blank region	0x88 .. 0x8F
Baseline 650 registers	0x90 .. 0x97
Blank region	0x98 .. 0x9F
Baseline 950 registers	0xA0 .. 0xA7
Blank region	0xA8 .. 0xBF
Baseline Index Control Registers	0xC0 .. 0xFF

Table 23 UART Standard 550 Compatible Registers (Start Address 0x00) (Sheet 1 of 2)

Register Name	Address Offset	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
THR[(1)]	0x00	W	Data to be transmitted							
RHR[(1)]	0x00	R	Data received							
IER[(1)(2)] 650/950 Mode	0x01	R/W	CTS interrupt mask	RTS interrupt mask	Special Char. Detect	Unused	Modem interrupt mask	Rx Stat interrupt mask	THRE interrupt mask	RxRDY interrupt mask
550/750 Mode			Unused							
FiCR[(3)] 650 mode	0x02	W	RHR Trigger Level		THR Trigger Level		DMA Mode/Tx Trigger Enable	Flush THR	Flush RHR	Enable FIFO
750 mode			RHR Trigger Level		FIFO Size	Unused				
950 mode			Unused							
ISR[(3)]	0x02	R	FIFOs enabled		Interrupt priority (Enhanced mode)		Interrupt priority (All modes)			Interrupt pending

Table 23 UART Standard 550 Compatible Registers (Start Address 0x00) (Sheet 2 of 2)

Register Name	Address Offset	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LCR[4]	0x03	R/W	Divisor latch access	Tx break	Force parity	Odd / even parity	Parity enable	Number of stop bits	Data length	
MCR[3][4] 550/750 Mode	0x04	R/W	Unused		CTS & RTS Flow Control	Internal Loop Back Enable	OUT2 (Int En)	OUT1	RTS	DTR
650/950 Mode			Baud prescale	IrDA mode	XON-Any					
LSR[3][5] Normal	0x05	R	Data Error	Tx Empty	THR Empty	Rx Break	Framing Error	Parity Error	Overrun Error	RxRDY
9-bit data mode								9th Rx data bit		
MSR[3]	0x06	R	DCD	RI	DSR	CTS	Delta DCD	Trailing RI edge	Delta DSR	Delta CTS
SPR[3] Normal	0x07	R/W	Temporary data storage register and Indexed control register offset value bits							
9-bit data mode			Unused							9th Tx data bit
Additional Standard Registers – These registers require divisor latch access bit (LCR[7]) to be set to 1.										
DLL	0x00	R/W	Divisor latch bits [7:0] (Least significant byte)							
DLM	0x01	R/W	Divisor latch bits [15:8] (Most significant byte)							

Register Access Notes:

1. Requires LCR[7] = 0
2. Requires ACR[7] = 0
3. Requires that last value written to LCR was not 0xBF
4. To read this register ACR[7] must be = 0
5. To read this register ACR[6] must be = 0

Table 24 UART Device Driver Speed-up Registers (Start Address 0x80) (Sheet 1 of 2)

Register Name	Address Offset	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
THR/RHR	000	R/W	Shadow of THR/RHR							
THR/RHR	001	R/W	Shadow of THR/RHR							
THR/RHR	010	R/W	Shadow of THR/RHR							
THR/RHR	011	R/W	Shadow of THR/RHR							
LSR	100		Shadow of LSR (Standard 550 Register)							
ISR	101		Shadow of ISR (Standard 550 Register)							
RFL	110		Shadow of Rx Fifo Fill Level (950 Register RFL)							

Table 24 UART Device Driver Speed-up Registers (Start Address 0x80) (Sheet 2 of 2)

Register Name	Address Offset	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TFL	111		Shadow of Tx Fifo Fill Level (950 Register TFL)							

Table 25 UART 650 Compatible Registers (Start Address 0x90)

Register Name	Address Offset	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
To access these registers LCR must be set to 0xBF										
EFR	010	R/W	CTS flow control	RTS Flow control	Special char detect	Enhance mode	In-band flow control mode			
XON1 9-bit mode	100	R/W	XON Character 1							
			Special character 1							
XON2 9-bit mode	101	R/W	XON Character 2							
			Special Character 2							
XOFF1 9-bit mode	110	R/W	XOFF Character 1							
			Special character 3							
XOFF2 9-bit mode	111	R/W	XOFF Character 2							
			Special character 4							

Table 26 UART 950 Specific Registers (Start Address 0xA0)

Register Name	Address Offset	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ASR(1)(3)(4)	001	R/W(4)	Tx Idle	FIFO size	FIFO-SEL	Special Char Detect	DTR	RTS	Remote Tx Disabled	Tx Disabled
RFL(3)	011	R	Number of characters in the receiver FIFO							
TFL(2)(3)	100	R	Number of characters in the transmitter FIFO							
ICR(2)(5)(6)	101	R/W	Data read/written depends on value written to the SPR prior to accessing this register (see Table 27)							

Register Access Notes:

1. Requires LCR[7] = 0
2. Requires that last value written to LCR was not 0xBF
3. Requires ACR[7] = 1
4. Only bits 0 and 1 of this register can be written
5. To read this register ACR[6] must be = 1
6. This register acts as a window through which to read and write registers in the Indexed Control Register set

Table 27 UART Indexed Control Register Set (Sheet 1 of 2)

Register Name	SPR Offset	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Indexed Control Register Set										
ACR	0x00	R/W	Additional Status Enable	ICR Read Enable	950 Trigger Level Enable	DTR definition and control		AutoDSR Flow Control Enable	Tx Disable	Rx Disable
CPR	0x01	R/W	lower 5 Bit “integer” part of 6 bit clock prescaler (see CPR2 below)					3 Bit “fractional” part of clock prescaler		
TCR	0x02	R/W	Unused				per data-bit over-sampling resolution, range 4..15 default is 16 times			
CPR2	0x03	R/W	Unused							Bit 6 of CPR
TTL	0x04	R/W	Unused	Transmitter Interrupt Trigger Level (0-127)						
RTL	0x05	R/W	Unused	Receiver Interrupt Trigger Level (1-127)						
FCL	0x06	R/W	Unused	Automatic Flow Control Lower Trigger Level (0-127)						
FCH	0x07	R/W	Unused	Automatic Flow Control Higher Trigger level (1-127)						
ID1	0x08	R	Hardwired ID byte 1 (0x16)							
ID2	0x09	R	Hardwired ID byte 1 (0xC9)							
ID3	0x0A	R	Hardwired ID byte 1 (0x50)							
REV	0x0B	R	Hardwired revision byte							
CSR	0x0C	W	Writing 0x00 to this register will reset the UART internals							
NMR	0x0D	R/W	Unused		9th Bit SChar 4	9th Bit Schar 3	9th Bit SChar 2	9th Bit SChar 1	9th Bit Int. En.	9 Bit Enable
MDM	0x0E	R/W	Unused				Δ DCD Interrupt disable	Trailing RI edge disable	Δ DSR Interrupt disable	Δ CTS Interrupt disable
RFC	0X0F	R	FiCR[7]	FiCR[6]	FiCR[5]	FiCR[4]	FiCR[3]	FiCR[2]	FiCR[1]	FiCR[0]
GDS[13]	0X10	R	Unused							Good Data Status
DMS	0x11	R/W	Force internal TxRdy inactive	Force internal RxRdy inactive	Unused				Internal TxRdy status (R)	Internal RxRdy status (R)
PIDX	0x12	R	Hardwired Port Index							
RS485_ DLYEN	0x14	R/W								RS_DEL

Table 27 UART Indexed Control Register Set (Sheet 2 of 2)

Register Name	SPR Offset	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RS485_ DLYCNT	0x15	R/W	RS485 delay bit count				RS485 delay phase count			
WDM	0x17	R/W	Unused			Disable Wakeup sensitivity	Δ DCD Interrupt disable	Trailing RI edge disable	Δ DSR Interrupt disable	Δ CTS Interrupt disable
Indexed Control Register Set – Alternative Baud Rate Control Registers										
A_LATCH	0x18	R/W	To access addresses 0x19 to 0x1E in this alternative register set the value 0xEB must be written to this location.							
A_ENABLE	0x19	R/W					Enables A_TCR	Enable both A_CPRx	Enables A_DLM	Enables A_DLL
A_DLL	0x1A	R/W	Alternative DLL register.							
A_DLM	0x1B	R/W	Alternative DLM register.							
A_CPR	0x1C	R/W	Alternative CPR register.							
A_TCR	0x1D	R/W	Alternative TCR register.							
A_CPR2	0x1E	R/W	Alternative CP2 register							

Reset Configuration

Host Reset

After a hardware reset or soft reset (bit 7 of COR register), all writable registers are reset to 0x00, with the following exceptions:

1. DLL—reset to 0x01
2. CPR—reset to 0x0F
3. CPR2—reset to 0x01

The state of read-only registers following a hardware reset is as follows:

RHR[7:0]:	Indeterminate
RFL[6:0]:	0000000_2
TFL[6:0]:	0000000_2
LSR[7:0]:	0x60 signifying that both the transmitter and the transmitter FIFO are empty
MSR[3:0]:	0000_2
MSR[7:4]:	Dependent on modem input lines DCD, RI, DSR and CTS respectively
ISR[7:0]:	0x01, i.e. no interrupts are pending

ASR[7:0]: $1xx00000_2$

RFC[7:0]: 00000000_2

GDS[7:0]: 00000001_2

DMS[7:0]: 00000010_2

The reset state of output signals is given in Table 28:

Table 28 Output Signal Reset State

Signal	Reset State
SOUT	Inactive High
nRTS	Inactive High
nDTR	Inactive High

Software Reset

An additional feature available in the 950 core is software resetting of the serial channel. The software reset is available using the CSR register. Software reset has the same effect as a hardware reset. To reset the UART, write 0x00 to the CSR.

Transmitter & Receiver FIFOs

The transmitter and receiver holding registers (FIFOs), are referred to as THR and RHR respectively.

In normal operation, when the transmitter finishes transmitting a byte it removes the next data from the top of the THR and transmits it. If the THR is empty, it waits until data is written into it. If THR is empty and the last character transmission is complete (i.e. the transmitter shift register is empty) the transmitter is said to be idle. Similarly, when the receiver finishes receiving a byte, it transfers it to the bottom of the RHR. If the RHR is full, an overrun condition occurs (see "Line Status Register—LSR" on page 33).

Data is written into the bottom of the THR queue and read from the top of the RHR queue completely asynchronously to the operation of the transmitter and receiver.

FIFO size depends on the setting of the FiCR register. In byte mode, FIFOs only accept one byte at a time before indicating that they are full; this is compatible with the 16C450. In a FIFO mode, the size of the FIFOs is either 16 (compatible with the 16C550) or 128.

Data written to the THR when it is full is lost. Data read from the RHR when it is empty is invalid. The empty or full status of the FIFOs is indicated in LSR (see "Line Status Register—LSR" on page 33). Interrupts can be generated or DMA signals can be used to transfer data to/from the FIFOs. The number of items in each FIFO may also be read back from the transmitter FIFO level (TFL) and receiver FIFO level (RFL) registers (see "FIFO Fill levels—TFL & RFL" on page 46).

FIFO Control Register—FiCR

FiCR[0]: Enable FIFO mode

logic 0 ⇒	Byte mode
logic 1 ⇒	FIFO mode

This bit should be enabled before setting the FIFO trigger levels.

FiCR[1]: Flush RHR

logic 0 ⇒	No change
logic 1 ⇒	Flushes the contents of the RHR

This is only operative in a FIFO mode. The RHR is flushed automatically whenever changing between Byte mode and a FIFO mode. This bit returns to zero after clearing the FIFOs.

FiCR[2]: Flush THR

logic 0 ⇒	No change
logic 1 ⇒	Flushes the contents of the THR in the same manner as FiCR[1] does for the RHR

DMA Transfer Signalling:

FiCR[3]: DMA signaling mode / Tx trigger level enable

logic 0 ⇒	DMA mode 0
logic 1 ⇒	DMA mode 1

DMA signals are not bonded out in the OXPCIe952, so this control only affects the transmitter trigger level in DMA mode 0.

FiCR[5:4]: THR trigger level

Generally in 450, 550, extended 550 and 950 modes these bits are unused (see section 8.1 for mode definition). In 650 mode they define the transmitter interrupt trigger levels and in 750 mode FiCR[5] increases the FIFO size.

450, 550 and extended 550 modes:

The transmitter interrupt trigger levels are set to 1 and FiCR[5:4] are ignored.

650 mode:

In 650 mode the transmitter interrupt trigger levels are set to the values given in Table 29.

Table 29 Transmit Interrupt Trigger Levels

FiCR[5:4]	Transmit Interrupt Trigger level
00	16
01	32
10	64
11	112

These levels only apply to enhanced mode and DMA mode 1 (FiCR[3] = 1), otherwise the trigger level is set to 1. A transmitter empty interrupt will be generated (if enabled) if the TFL falls below the trigger level.

750 mode:

In 750 compatible non-enhanced (EFR[4]=0) mode, transmitter trigger level is set to 1, FiCR[4] is unused and FiCR[5] defines the FIFO depth as follows:

FiCR[5]=0	transmitter and receiver FIFO size is 16 bytes
FiCR[5]=1	transmitter and receiver FIFO size is 128 bytes

In non-enhanced mode FiCR[5] is only writable when LCR[7] is set. Note: in enhanced mode, the FIFO size is also increased to 128 bytes when FiCR[0] is set.

950 mode:

Setting ACR[5] to 1 enables arbitrary transmitter trigger level setting using the TTL register (see "Transmitter Trigger Level—TTL" on page 48), so FiCR[5:4] are ignored.

FiCR[7:6]: RHR trigger level

In 550, extended 550, 650 and 750 modes, the receiver FIFO trigger levels are defined using FiCR[7:6]. The interrupt trigger level and upper flow control trigger level where appropriate are defined by L2 in Table 30 on page 31. L1 defines the lower flow control trigger level where applicable. Separate upper and lower flow control trigger levels introduce a hysteresis element in in-band and out-of-band flow control (see "Automatic Flow Control" on page 42).

Table 30 Receiver Trigger Levels

FiCR [7:6]	Mode					
	650 FIFO Size 128		750 FIFO Size 128		550 FIFO Size 16	
	L1	L2	L1	L2	L1	L2
00	1	16	1	1	n/a	1
01	16	32	1	32	n/a	4
10	32	112	1	64	n/a	8
11	112	120	1	112	n/a	14

In byte mode (450 mode) the trigger levels are all set to 1.

In all cases, a receiver data interrupt is generated (if enabled) if the Receiver FIFO Level (RFL) reaches the upper trigger level L2.

950 mode:

When 950 trigger levels are enabled (ACR[5]=1), more flexible trigger levels can be set by writing to the TTL, RTL, FCL and FCH (see "Additional Features" on page 45) hence ignoring FiCR[7:6].

Line Control & Status

False Start Bit Detection

On the falling edge of a start bit, the receiver waits for 1/2 bit and then resynchronizes the receiver's sampling clock to the centre of the start bit. The start bit is valid if the SIN line is still low at this mid-bit sample and the receiver proceeds to read in a data character. Verifying the start bit prevents the receiver from assembling a false data character due to a low going noise spike on the SIN input.

When the first stop bit is sampled, the received data is transferred to the RHR and the receiver waits for a low transition on SIN signifying the next start bit.

The receiver continues to receive data even if the RHR is full or has been disabled (see "Additional Control Register—ACR" on page 47) in order to maintain framing synchronization. The only difference is that the received data is not transferred to the RHR.

Line Control Register—LCR

The LCR specifies the data format that is common to both transmitter and receiver. Writing 0xBF to LCR enables access to the EFR, XON1, XOFF1, XON2 and XOFF2, DLL and DLM registers. This value (0xBF) corresponds to an unused data format. Writing the value 0xBF to LCR will set LCR[7] but leaves LCR[6:0] unchanged. Therefore, the data format of the transmitter and receiver data is not affected. Write the desired LCR value to exit from this selection.

LCR[1:0]: Data length

LCR[1:0] Determines the data length of serial characters. Note however, that these values are ignored in 9-bit data framing mode, i.e. when NMR[0] is set.

Table 31 LCR Data Length Configuration

LCR[1:0]	Data Length
00	5 bits
01	6 bits
10	7 bits
11	8 bits

LCR[2]: Number of stop bits

LCR[2] defines the number of stop bits per serial character.

Table 32 LCR StopBit Number Configuration

LCR[2]	Data Length	No. Stop Bits
0	5,6,7,8	1
1	5	1.5
1	6,7,8	2

LCR[5:3]: Parity type

The selected parity type will be generated during transmission and checked by the receiver, which may produce a parity error as a result. In 9-bit mode parity is disabled and LCR[5:3] is ignored.

Table 33 LCR Parity Configuration

LCR[5:3]	Parity Type
xx0	No parity bit
001	Odd parity bit
011	Even parity bit
101	Parity bit forced to 1
111	Parity bit forced to 0

LCR[6]: Transmission break

logic 0 ⇒	Break transmission disabled
logic 1 ⇒	Forces the transmitter data output SOUT low to alert the communication terminal, or send zeros in IrDA mode

The software driver must ensure that the break duration is longer than the character period for it to be recognized remotely as a break rather than data.

LCR[7]: Divisor latch enable

logic 0 ⇒	Access to DLL & DLM registers disabled
logic 1 ⇒	Access to DLL & DLM registers enabled

Line Status Register—LSR

This register provides the status of data transfer to CPU.

LSR[0]: RHR data available

logic 0 ⇒	RHR is empty: no data available
logic 1 ⇒	RHR is not empty: data is available to be read

LSR[1]: RHR overrun error

logic 0 ⇒	No overrun error
logic 1 ⇒	Data was received when the RHR was full. An overrun error has occurred. The error is flagged when the data would normally have been transferred to the RHR

LSR[2]: Received data parity error

logic 0 ⇒	No parity error in normal mode or 9th bit of received data is 0 in 9-bit mode
logic 1 ⇒	Data has been received that did not have correct parity in normal mode or 9th bit of received data is ‘1’ in 9-bit mode

The flag is set when the data item in error is at the top of the RHR, and cleared following a read of the LSR. In 9-bit mode, LSR[2] is no longer a flag and corresponds to the 9th bit of the received data in RHR.

LSR[3]: Received data framing error

logic 0 ⇒	No framing error
logic 1 ⇒	Data was received with an invalid stop bit

This status bit is set and cleared in the same manner as LSR[2]. When a framing error occurs, the UART tries to re-synchronize by assuming that the error was due to sampling the start bit of the next data item.

LSR[4]: Received break error

logic 0 ⇒	No receiver break error
logic 1 ⇒	The receiver received a break

A break condition occurs when the SIN line goes low (normally signifying a start bit) and stays low throughout the start, data, parity and first stop bit. (Note: the SIN line is sampled at the bit rate). A zero character with associated break flag set is transferred to the RHR, and the receiver then waits until the SIN line returns high. The LSR[4] break flag is set when this data item gets to the top of the RHR and is cleared following a read of the LSR.

LSR[5]: THR empty

logic 0 ⇒	Transmitter FIFO (THR) is not empty.
logic 1 ⇒	Transmitter FIFO (THR) is empty.

LSR[6]: Transmitter and THR empty

logic 0 ⇒	The transmitter is not idle
logic 1 ⇒	THR is empty and the transmitter has completed the character in shift register and is in idle mode. (I.e. set whenever the transmitter shift register and the THR are both empty.)

LSR[7]: Receiver data error

logic 0 ⇒	Either there are no receiver data errors in the FIFO or it was cleared by an earlier read of LSR
logic 1 ⇒	At least one parity error, framing error or break indication in the FIFO

In 450 mode LSR[7] is permanently cleared, otherwise this bit is set when an erroneous character is transferred from the receiver to the RHR. It is cleared when the LSR is read. **Note that in 16C550 this bit is only cleared when all of the erroneous data are removed from the FIFO**. In 9-bit data framing mode parity is permanently disabled, so this bit is not affected by LSR[2].

UART Interrupts

The serial interrupt on the OXPCIe952 is routed to the OXPCIe952 interrupt control, regardless of MCR[3].

Interrupt Enable Register—IER

Serial channel interrupts are enabled using the Interrupt Enable Register (IER).

IER[0]: Receiver data available interrupt mask

logic 0 ⇒	Disable the receiver ready interrupt.
logic 1 ⇒	Enable the receiver ready interrupt

IER[1]: Transmitter empty interrupt mask

logic 0 ⇒ Disable the transmitter empty interrupt

logic 1 ⇒ Enable the transmitter empty interrupt

IER[2]: Receiver status interrupt

Normal mode:

logic 0 ⇒ Disable the receiver status interrupt

logic 1 ⇒ Enable the receiver status interrupt

9-bit data mode:

logic 0 ⇒ Disable receiver status & address bit interrupt

logic 1 ⇒ Enable receiver status & address bit interrupt

In 9-bit mode (i.e. when NMR[0] is set) reception of a character with the address-bit (9th bit) set can generate a level 1 interrupt if IER[2] is set.

IER[3]: Modem status interrupt mask

logic 0 ⇒ Disable the modem status interrupt

logic 1 ⇒ Enable the modem status interrupt

IER[4]: Reserved

IER[5]: Special character interrupt mask

9-bit data framing mode:

logic 0 ⇒ Disable the special character receive interrupt

logic 1 ⇒ Enable the special character receive interrupt

In 9-bit data mode, The receiver can detect up to four special characters programmed in Special Character 1 to 4. When IER[5] is set, a level 5 interrupt is asserted when a match is detected.

650/950 modes (non-9-bit data framing):

logic 0 ⇒ Disable the special character receive interrupt

logic 1 ⇒ Enable the special character receive interrupt

In 16C654-compatible mode, when the device is in enhanced mode (EFR[4]=1), this bit enables the detection of special characters. It enables both the detection of XOFF characters (when in-band flow control is enabled via EFR[3:0]) and the detection of the XOFF2 special character (when enabled via EFR[5]).

750 mode (non-9-bit data framing):

logic 0 ⇒ Disable alternative sleep mode

logic 1 ⇒ Enable alternative sleep mode whereby the internal clock of the channel is switched off.

IER[6]: RTS interrupt mask

logic 0 ⇒ Disable the RTS interrupt

logic 1 ⇒ Enable the RTS interrupt

This enable is only operative in enhanced mode (EFR[4]=1). In non-enhanced mode, RTS interrupt is permanently enabled.

IER[7]: CTS interrupt mask

logic 0 ⇒ Disable the CTS interrupt

logic 1 ⇒ Enable the CTS interrupt

This enable is only operative in Enhanced mode (EFR[4]=1). In non-Enhanced mode, CTS interrupt is permanently enabled.

Interrupt Status Register—ISR

The source of the highest priority interrupt pending is indicated by the contents of the interrupt status register ISR. There are nine sources of interrupt at six levels of priority (1 is the highest) as tabulated below:

Table 34 Interrupt Status Identification Codes

Level	Interrupt Source	ISR[5:0][3]
-	No interrupt pending[1]	000001
1	Receiver status error or Address-bit detected in 9-bit mode	000110
2a	Receiver data available	000100
2b	Receiver time-out	001100
3	Transmitter THR empty	000010
4	Modem status change	000000
5[2]	In-band flow control XOFF or Special character (XOFF2) or Special character 1, 2, 3 or 4 or bit 9 set in 9-bit mode	010000
6[2]	CTS or RTS change of state	100000

Notes:

1 ISR[0] indicates whether any interrupts are pending.

2 Interrupts of priority levels 5 and 6 cannot occur unless the UART is in Enhanced mode.

3 ISR[5] is only used in 650 & 950 modes. In 750 mode, it is 0 when FIFO size is 16 and 1 when FIFO size is 128. In all other modes it is permanently set to 0.

Interrupt Description

Level 1

Receiver status error interrupt (ISR[5:0]='000110'):

Normal (non-9-bit) mode:

This interrupt is active whenever any of LSR[1], LSR[2], LSR[3] or LSR[4] are set. These flags are cleared following a read of the LSR. This interrupt is masked with IER[2].

9-bit mode:

This interrupt is active whenever any of LSR[1], LSR[2], LSR[3] or LSR[4] are set. The receiver error interrupt due to LSR[1], LSR[3] and LSR[4] is masked with IER[3]. The 'address-bit' received interrupt is masked with NMR[1]. The software driver can differentiate between receiver status error and received address-bit (9th data bit) interrupt by examining LSR[1] and LSR[7]. In 9-bit mode LSR[7] is only set when LSR[3] or LSR[4] is set and it is not affected by LSR[2] (i.e. 9th data bit).

Level 2a

Receiver data available interrupt (ISR[5:0]='000100'):

This interrupt is active whenever the receiver FIFO level is above the interrupt trigger level.

Level 2b

Receiver time-out interrupt (ISR[5:0]='001100'):

A receiver time-out event, which may cause an interrupt, will occur when all of the following conditions are true:

- The UART is in a FIFO mode
- There is data in the RHR.
- There has been no read of the RHR for a period of time greater than the time-out period.
- There has been no new data received and written into the RHR for a period of time greater than the time-out period. The time-out period is four times the character period (including start and stop bits) measured from the centre of the first stop bit of the last data item received.

Reading the first data item in RHR clears this interrupt.

Level 3

Transmitter empty interrupt (ISR[5:0]='000010'):

This interrupt is set when the transmit FIFO level falls below the trigger level. It is cleared on an ISR read of a level 3 interrupt or by writing more data to the THR so that the trigger level is exceeded. Note that when 950 mode trigger levels are enabled (ACR[5]=1) and the transmitter trigger level of zero is selected (TTL=0x00), a transmitter empty interrupt is only asserted when both the transmitter FIFO and transmitter shift register are empty and the SOUT line has returned to idle marking state.

Level 4

Modem change interrupt (ISR[5:0]='000000'):

This interrupt is set by a modem change flag (MSR[0], MSR[1], MSR[2] or MSR[3]) becoming active due to changes in the input modem lines. This interrupt is cleared following a read of the MSR.

Level 5

Receiver in-band flow control (Xoff) detect interrupt,
Receiver special character (Xoff2) detect interrupt,
Receiver special character 1, 2, 3 or 4 interrupt or
9th Bit set interrupt in 9-bit mode (ISR[5:0]=010000):

A level 5 interrupt can only occur in enhanced-mode when any of the following conditions are met:

- A valid Xoff character is received while in-band flow control is enabled.
- A received character matches Xoff2 while special character detection is enabled.
- A received character matches special character 1, 2, 3 or 4 in 9-bit mode (see "Nine-bit Mode Register—NMR" on page 50).

It is cleared on an ISR read of a level 5 interrupt.

Level 6

CTS or RTS changed interrupt (ISR[5:0]=100000):

This interrupt is set whenever either of the nCTS or nRTS pins changes state from low to high. It is cleared on an ISR read of a level 6 interrupt.

Sleep Mode

For a channel to go into sleep mode, all of the following conditions must be met:

1. Sleep mode enabled (IER[4]=1 in 650/950 modes, or IER[5]=1 in 750 mode)
2. The transmitter is idle, i.e. the transmitter shift register and FIFO are both empty
3. SIN is high
4. The receiver is idle
5. The receiver FIFO is empty (LSR[0]=0)
6. The UART is not in loopback mode (MCR[4]=0)
7. Changes on modem input lines have been acknowledged (i.e. MSR[3:0]=0000)
8. No interrupts are pending

A read of IER[4] (or IER[5] if a 1 was written to that bit instead) shows whether the power-down request was successful. The UART will retain its programmed state whilst in power-down mode.

The channel automatically exits power-down mode when any of the conditions 1 to 7 becomes false. It may be woken manually by clearing IER[4] (or IER[5] if the alternate sleep mode is enabled).

Sleep mode operation is not available in IrDA mode.

Modem Interface

Modem Control Register—MCR

MCR[0]: DTR

logic 0 ⇒	Force nDTR output to inactive (high)
logic 1 ⇒	Force nDTR output to active (low)

nDTR can be used for automatic out-of-band flow control when enabled using ACR[4:3] (see "Additional Control Register—ACR" on page 47).

MCR[1]: RTS

logic 0 ⇒	Force nRTS output to inactive (high)
logic 1 ⇒	Force nRTS output to active (low)

nRTS can be used for automatic out-of-band flow control when enabled using EFR[6] (see "Automatic Out-Of-Band Flow Control" on page 45).

MCR[2]: OUT1

logic 0 ⇒	Force nOUT1 output low when loopback mode is disabled
logic 1 ⇒	Force nOUT1 output high

nOUT1 is not bonded out in the OXPCIe952, but is used internally for loopback testing.

MCR[3]: OUT2

logic 0 ⇒	Force nOUT2 output low when loopback mode is disabled
logic 1 ⇒	Force nOUT2 output high

nOUT2 is not bonded out in the OXPCIe952, but is used internally for loopback testing.

MCR[4]: Loopback mode

logic 0 ⇒	Normal operating mode
logic 1 ⇒	Enable local loop-back mode (diagnostics)

In local loop-back mode, the transmitter output (SOUT) and the modem outputs (nDTR, nRTS) are set inactive (high), and the receiver inputs SIN, nCTS, nDSR, nDCD, and nRI are all disabled. Internally the transmitter output is connected to the receiver input and nDTR, nRTS, nOUT1 and nOUT2 are connected to modem status inputs nDSR, nCTS, nRI and nDCD respectively.

In this mode, the receiver and transmitter interrupts are fully operational. The modem control interrupts are also operational, but the interrupt sources are now the lower four bits of the MCR instead of the four modem status inputs. The interrupts are still controlled by the IER.

MCR[5]: Enable XON-Any in enhanced mode or enable out-of-band flow control in non-enhanced mode

650/950 modes (enhanced mode):

logic 0 ⇒	XON-Any is disabled
logic 1 ⇒	XON-Any is enabled

In enhanced mode (EFR[4]=1), this bit enables the Xon-Any operation. When Xon-Any is enabled, received data is accepted as a valid Xon (see "Automatic In-band Flow Control" on page 44).

750 mode (non-enhanced mode):

logic 0 ⇒	CTS/RTS flow control disabled
logic 1 ⇒	CTS/RTS flow control enabled

In non-enhanced mode, this bit enables the CTS/RTS out-of-band flow control.

MCR[6]: IrDA mode

logic 0 ⇒	Standard serial receiver & transmitter data format
logic 1 ⇒	Data is transmitted & received in IrDA format

This function is only available in enhanced mode. It requires a 16x clock to function correctly.

MCR[7]: Baud rate prescaler select

logic 0 ⇒	Normal (divide by 1) baud rate generator prescaler selected
logic 1 ⇒	Divide-by-*M N*/8 baud rate generator prescaler selected

Where *M* & *N* are programmed in CPR (ICR offset 0x01). After a hardware reset, CPR defaults to 0x20 (divide-by-4) and MCR[7] is reset to 0. User writes to this flag only take effect in enhanced mode. See "Enhanced Features Register—EFR" on page 42.

Modem Status Register—MSR

MSR[0]: Delta nCTS

Indicates that the nCTS input has changed since the last time the MSR was read.

MSR[1]: Delta nDSR

Indicates that the nDSR input has changed since the last time the MSR was read.

MSR[2]: Trailing edge nRI

Indicates that the nRI input has changed from low to high since the last time the MSR was read.

MSR[3]: Delta nDCD

Indicates that the nDCD input has changed since the last time the MSR was read.

MSR[4]: CTS

This bit is the complement of the nCTS input. It is equivalent to RTS (MCR[1]) during internal loop-back mode.

MSR[5]: DSR

This bit is the complement of the nDSR input. It is equivalent to DTR (MCR[0]) during internal loop-back mode.

MSR[6]: RI

This bit is the complement of the nRI input. In internal loop-back mode it is equivalent to the internal OUT1.

MSR[7]: DCD

This bit is the complement of the nDCD input. In internal loop-back mode it is equivalent to the internal OUT2.

Automatic Flow Control

Automatic in-band flow control, automatic out-of-band flow control and special character detection features can be used in enhanced mode and are software-compatible with the 16C654. Alternatively, 16C750-compatible automatic out-of-band flow control can be enabled in non-enhanced mode. In 950 mode, in-band and out-of-band flow controls are compatible with 16C654, with the addition of fully programmable flow control thresholds.

Enhanced Features Register—EFR

Writing 0xBF to LCR enables access to the EFR and other Enhanced mode registers. This value corresponds to an unused data format. Writing 0xBF to LCR will set LCR[7] but leaves LCR[6:0] unchanged. Therefore, the data format of the transmitter and receiver data is not affected. Write the desired LCR value to exit from this selection.

In-band transmit and receive flow control is disabled in 9-bit mode.

EFR[1:0]: In-band receive flow control mode

When in-band receive flow control is enabled, the UART compares the received data with the programmed Xoff character(s). When this occurs, the UART disables transmission as soon as any current character transmission is complete. The UART then compares the received data with the programmed Xon character(s). When a match occurs, the UART re-enables transmission (see "Flow Control Levels—FCL & FCH" on page 49).

For automatic in-band flow control, EFR[4] must be set. The combinations of software receive flow control can be selected by programming EFR[1:0] as follows:

logic [00] ⇒	In-band receive flow control is disabled
logic [01] ⇒	Single character in-band receive flow control enabled, recognizing Xon2 as the XON character and Xoff2 as the XOFF character
logic [10] ⇒	Single character in-band receive flow control enabled, recognizing Xon1 as the Xon character and Xoff1 and the Xoff character

logic [11] ⇒ The behavior of the receive flow control depends on the configuration of EFR[3:2]. Single-character in-band receive flow control is enabled, accepting both Xon1 and Xon2 as valid Xon characters and both Xoff1 and Xoff2 as valid Xoff characters when EFR[3:2] = 01 or 10. EFR[1:0] should not be set to 11 when EFR[3:2] is 00

EFR[3:2]: In-band transmit flow control mode

When in-band transmit flow control is enabled, Xon/Xoff characters are inserted into the data stream whenever the RFL passes the upper trigger level and falls below the lower trigger level respectively.

For automatic in-band flow control, EFR[4] must be set. The combinations of software transmit flow control can then be selected by programming EFR[3:2] as follows:

logic [00] ⇒ In-band transmit flow control is disabled

logic [01] ⇒ Single character in-band transmit flow control enabled, using Xon2 as the Xon character and Xoff2 as the Xoff character

logic [10] ⇒ Single character in-band transmit flow control enabled, using Xon1 as the Xon character and Xoff1 as the Xoff character

Logic[11] ⇒ The value EFR[3:2] = 11 is reserved for future use and should not be used

EFR[4]: Enhanced mode

logic 0 ⇒ Non-enhanced mode. Disables IER[7:4], ISR[5:4], FiCR[5:4], MCR[7:5] and in-band flow control. Whenever this bit is cleared, the setting of other bits of EFR is ignored

logic 1 ⇒ Enhanced mode. Enables the enhanced mode functions. These functions include enabling IER[7:4], FiCR[5:4], MCR[7:5]. For in-band flow control the software driver must set this bit first. If this bit is set, out-of-band flow control is configured with EFR[7:6], otherwise out-of-band flow control is compatible with 16C750

EFR[5]: Enable special character detection

logic 0 ⇒ Special character detection is disabled

logic 1 ⇒ While in Enhanced mode (EFR[4]=1), the UART compares the incoming receiver data with the XOFF2 value. Upon a correct match, the received data will be transferred to the RHR and a level 5 interrupt (XOFF or special character) will be asserted if level 5 interrupts are enabled (IER[5] set to 1)

EFR[6]: Enable automatic RTS flow control

logic 0 ⇒ RTS flow control is disabled (default)

logic 1 ⇒ RTS flow control is enabled in enhanced mode (i.e. EFR[4] = 1), where the nRTS pin is forced inactive high if the RFL reaches the upper flow control threshold. This is released when the RFL drops below the lower threshold. The 650 and 950 software drivers should use this bit to enable RTS flow control. The 750 compatible driver uses MCR[5] to enable RTS flow control

EFR[7]: Enable automatic CTS flow control

logic 0 ⇒	CTS flow control is disabled (default)
logic 1 ⇒	CTS flow control is enabled in enhanced mode (i.e. EFR[4] = 1), where the data transmission is prevented whenever the nCTS pin is held inactive high. The 650 and 950 software drivers should use this bit to enable CTS flow control. The 750 compatible driver uses MCR[5] to enable CTS flow control

Special Character Detection

In enhanced mode (EFR[4]=1), when special character detection is enabled (EFR[5]=1) and the receiver matches received data with Xoff2, the received special character flag ASR[4] is set and a level 5 interrupt is asserted, if enabled by IER[5]. This flag is cleared following a read of ASR. The received status (i.e. parity and framing) of special characters does not have to be valid for these characters to be accepted as valid matches.

Automatic In-band Flow Control

When in-band receive flow control is enabled, the UART compares the received data with Xoff1 or Xoff2 characters to detect an Xoff condition. When this occurs, the UART disables transmission as soon as any current character transmission is complete. Status bits ISR[4] and ASR[0] are set. A level 5 interrupt occurs, if enabled by IER[5]. The UART then compares all received data with Xon1 or Xon2 characters to detect an Xon condition. When this occurs, the UART re-enables transmission and status bits ISR[4] and ASR[0] are cleared.

Any valid Xon/Xoff characters are not written into the RHR, apart from when special character detection is enabled and an Xoff2 character is received that is a valid Xoff. In this instance, the character is written into the RHR.

The received status (i.e., parity and framing) of Xon/Xoff characters does not have to be valid for these characters to be accepted as valid matches.

When the Xon-Any flag (MCR[5]) is set, any received character is accepted as a valid Xon condition and the transmitter is re-enabled. The received data is transferred to the RHR.

When in-band transmit flow control is enabled, the RFL is sampled whenever the transmitter is idle (briefly, between characters, or when the THR is empty) and an Xon/Xoff character may be inserted into the data stream if needed. Initially, remote transmissions are enabled and hence ASR[1] is clear. If ASR[1] is clear and the RFL has passed the upper trigger level (i.e. is above the trigger level), Xoff is sent and ASR[1] is set. If ASR[1] is set and the RFL falls below the lower trigger level, Xon is sent and ASR[1] is cleared.

If transmit flow control is disabled after an Xoff has been sent, an Xon is sent automatically.

Automatic Out-Of-Band Flow Control

Automatic RTS/CTS flow control is selected by different means, depending on whether the UART is in enhanced or non-enhanced mode. In non-enhanced mode, MCR[5] enables both RTS and CTS flow control. In enhanced mode, EFR[6] enables automatic RTS flow control and EFR[7] enables automatic CTS flow control. This allows software compatibility with both 16C654 and 16C750 drivers.

When automatic CTS flow control is enabled and the nCTS input becomes active, the UART disables transmission as soon as any current character transmission is complete. Transmission is resumed whenever the nCTS input becomes inactive.

When automatic RTS flow control is enabled, the nRTS pin is forced inactive when the RFL reaches the upper trigger level and returns to active when the RFL falls below the lower trigger level. The automatic nRTS flow control is ANDed with MCR[1] and hence is only operational when MCR[1]=1. This allows the software driver to override the automatic flow control and disable the remote transmitter regardless by setting MCR[1]=0 at any time.

Automatic DTR/DSR flow control behaves in the same manner as RTS/CTS flow control but is enabled by ACR[3:2], regardless of whether or not the UART is in enhanced mode.

Additional Features

Additional Status Register—ASR

ASR[0]: Transmitter disabled

logic 0 ⇒	The transmitter is not disabled by in-band flow control
logic 1 ⇒	The receiver has detected an Xoff, and has disabled the transmitter

This bit is cleared after a hardware reset or channel software reset. The software driver may write 0 to this bit to re-enable the transmitter if it was disabled by in-band flow control. Writing 1 to this bit has no effect.

ASR[1]: Remote transmitter disabled

logic 0 ⇒	The remote transmitter is not disabled by in-band flow control
logic 1 ⇒	The transmitter has sent an Xoff character, to disable the remote transmitter. (Cleared when a subsequent Xon is sent)

This bit is cleared after a hardware reset or channel software reset. The software driver may write 0 to this bit to re-enable the remote transmitter (an XON is transmitted). Writing 1 to this bit has no effect.

The remaining bits (ASR[7:2]) of this register are read only

ASR[2]: RTS

This is the complement of the actual state of the nRTS pin when the device is not in loopback mode. The driver software can determine whether the remote transmitter is disabled by nRTS out-of-band flow control by reading this bit. In loopback mode this bit reflects the flow control status rather than the pin's actual state.

ASR[3]: DTR

This is the complement of the actual state of the nDTR pin when the device is not in loopback mode. The driver software can determine whether the remote transmitter is disabled by nDTR out-of-band flow control by reading this bit. In loopback mode this bit reflects the flow control status rather than the pin's actual state.

ASR[4]: Special character detected

logic 0 ⇒	No special character has been detected
logic 1 ⇒	A special character has been received and is stored in the RHR

This can be used to determine whether a level 5 interrupt was caused by receiving a special character rather than an Xoff. The flag is cleared following the read of the ASR.

ASR[5]: RESERVED

This bit is unused in the OXPCIe952 and reads 0.

ASR[6]: FIFO size

logic 0 ⇒	FIFOs are 16 deep if FiCR[0] = 1
logic 1 ⇒	FIFOs are 128 deep if FiCR[0] = 1

If FiCR[0] = 0, the FIFOs are 1 deep.

ASR[7]: Transmitter Idle

logic 0 ⇒	Transmitter is transmitting
logic 1 ⇒	Transmitter is idle

This bit reflects the state of the internal transmitter. It is set when both the transmitter FIFO and shift register are empty.

FIFO Fill levels—TFL & RFL

The TFL and RFL registers can be accessed in both the 950-specific registers and the OXPCIe952-specific registers; the latter is the most direct means of access. The number of characters stored in the THR and RHR can be determined by reading the TFL and RFL registers respectively.

Additional Control Register—ACR

The ACR register is located at offset 0x00 of the ICR

ACR[0]: Receiver disable

logic 0 ⇒	The receiver is enabled, receiving data and storing it in the RHR
logic 1 ⇒	The receiver is disabled. The receiver continues to operate as normal to maintain the framing synchronization with the receive data stream, but received data is not stored into the RHR. In-band flow control characters continue to be detected and acted upon. Special characters are not detected

Changes to this bit are only recognized following the completion of any data reception pending.

ACR[1]: Transmitter disable

logic 0 ⇒	The transmitter is enabled, transmitting any data in the THR
logic 1 ⇒	The transmitter is disabled. Any data in the THR is not transmitted but is held. However, in-band flow control characters may still be transmitted

Changes to this bit are only recognized following the completion of any data transmission pending.

ACR[2]: Enable automatic DSR flow control

logic 0 ⇒	Normal. The state of the nDSR line does not affect the flow control
logic 1 ⇒	Data transmission is prevented whenever the nDSR pin is held inactive high

This bit provides another automatic out-of-band flow control facility using the nDSR line.

ACR[4:3]: nDTR line configuration

The nDTR pin is defined as follows:

logic [00] ⇒	nDTR is compatible with 16C450, 16C550, 16C654 and 16C750 (i.e. normal)
logic [01] ⇒	nDTR pin is used for out-of-band flow control. It is forced inactive high if the RFL reaches the upper flow control threshold. nDTR line is re-activated when the RFL drops below the lower threshold (see "Flow Control Levels—FCL & FCH" on page 49)
logic [10] ⇒	nDTR pin is configured to drive the active low enable pin of an external RS485 buffer. In this configuration the nDTR pin will be forced low whenever the transmitter is not empty (LSR[6]=0), otherwise nDTR pin is high
logic [11] ⇒	nDTR pin is configured to drive the active-high enable pin of an external RS485 buffer. In this configuration, the nDTR pin will be forced high whenever the transmitter is not empty (LSR[6]=0), otherwise nDTR pin is low

If the user sets ACR[4], the nDTR line is controlled by the status of the transmitter empty bit of LCR. When ACR[4] is set, ACR[3] is used to select active-high or active-low enable signals. In half-duplex systems using RS485 protocol, this facility enables the nDTR line to directly control the enable signal of external 3-state line driver buffers. When the transmitter is empty the nDTR would go inactive once the SOUT line returns to its idle marking state.

ACR[5]: 950 mode trigger levels enable

logic 0 ⇒	Interrupts and flow control trigger levels are as described in FiCR register and are compatible with 16C654/16C750 modes
logic 1 ⇒	950 specific enhanced interrupt and flow control trigger levels defined by RTL, TTL, FCL and FCH are enabled

ACR[6]: ICR read enable

logic 0 ⇒	The LSR is readable
logic 1 ⇒	The ICRs are readable

Setting this bit maps the ICR set to the LSR location for reads. During normal operation this bit should be cleared.

ACR[7]: Additional status enable

logic 0 ⇒	Access to the ASR, TFL and RFL registers is disabled
logic 1 ⇒	Access to the ASR, TFL and RFL registers is enabled

When ACR[7] is set, the MCR and LCR registers are no longer readable but remain writable, and the TFL and RFL registers replace them in the memory map for read operations. The IER register is replaced by the ASR register for all operations. The software driver may leave this bit set during normal operation, since MCR, LCR and IER do not generally need to be read.

Transmitter Trigger Level—TTL

The TTL register is located at offset 0x04 of the ICR.

Whenever 950 trigger levels are enabled (ACR[5]=1), bits 4 and 5 of FiCR are ignored and an alternative arbitrary transmitter interrupt trigger level can be defined in the TTL register. This 7-bit value provides a fully programmable transmitter interrupt trigger facility. In 950 mode, a priority level 3 interrupt occurs indicating that the transmitter buffer requires more characters when the interrupt is not masked (IER[1]=1) and the transmitter FIFO level falls below the value stored in the TTL register. The value 0 (0x00) has a special meaning. In 950 mode when the user writes 0x00 to the TTL register, a level 3 interrupt only occurs when the FIFO and the transmitter shift register are both empty and the SOUT line is in the idle marking state. This feature is particularly useful to report back the empty state of the transmitter after its FIFO has been flushed away.

Receiver Interrupt. Trigger Level—RTL

The RTL register is located at offset 0x05 of the ICR.

Whenever 950 trigger levels are enabled (ACR[5]=1), bits 6 and 7 of FiCR are ignored and an alternative arbitrary receiver interrupt trigger level can be defined in the RTL register. This 7-bit value provides a fully programmable receiver interrupt trigger facility as opposed to the limited trigger levels available in 16C654 and 16C750 devices. It enables the system designer to optimize the interrupt performance hence minimizing the interrupt overhead.

In 950 mode, a priority level 2 interrupt occurs indicating that the receiver data is available when the interrupt is not masked (IER[0]=1) and the receiver FIFO level reaches the value stored in this register.

Flow Control Levels—FCL & FCH

The FCL and FCH registers are located at offsets 0x06 and 0x07 of the ICR respectively

Enhanced software flow control using Xon/Xoff and hardware flow control using nRTS/nCTS and nDTR/nDSR are available when 950 mode trigger levels are enabled (ACR[5]=1). Improved flow control threshold levels are offered using Flow Control Lower trigger level (FCL) and Flow Control Higher trigger level (FCH) registers to provide a greater degree of flexibility when optimizing the flow control performance. Generally, these facilities are only available in enhanced mode.

In 650 mode, in-band flow control is enabled using the EFR register. An Xoff character is transmitted when the receiver FIFO exceeds the upper trigger level defined by FiCR[7:6] as described in section 8.4.2. An Xon is then sent when the FIFO is read down to the lower fill level. The flow control is enabled and the appropriate mode selected using EFR[3:0].

In 950 mode, the flow control thresholds defined by FiCR[7:6] are ignored. In this mode threshold levels are programmed using FCL and FCH. When in-band flow control is enabled (defined by EFR[3:0]) and the RFL) reaches the value programmed in the FCH register, an Xoff is transmitted to stop the flow of serial data. The flow is resumed when the receiver FIFO fill level falls to below the value programmed in the FCL register, at which point an Xon character is sent. The FCL value of 0x00 is illegal.

For example if FCL and FCH contain 64 and 100 respectively, Xoff is transmitted when the receiver FIFO contains 100 characters, and Xon is transmitted when sufficient characters are read from the receiver FIFO such that 63 characters remain.

CTS/RTS and DSR/DTR out-of-band flow control use the same trigger levels as in-band flow control. When out-of-band flow control is enabled, nRTS (or nDTR) line is de-asserted when the receiver FIFO level reaches the upper limit defined in the FCH and is re-asserted when the receiver FIFO is drained below the lower limit defined in FCL. When 950 trigger levels are enabled (ACR[5]=1), the nCTS flow control functions as in 650 mode and is configured by EFR[7]. However, when EFR[6] is set, nRTS is automatically de-asserted when RFL reaches FCH and re-asserted when RFL drops below FCL.

nDSR flow control is configured with ACR[2]. nDTR flow control is configured with ACR[4:3].

Nine-bit Mode Register—NMR

The NMR register is located at offset 0x0D of the ICR

The 950 offers 9-bit data framing for industrial multi-drop applications. 9-bit mode is enabled by setting bit 0 of the nine-bit mode register (NMR). In 9-bit mode the data length setting in LCR[1:0] is ignored. Furthermore as parity is permanently disabled, the setting of LCR[5:3] is also ignored.

The receiver stores the 9th bit of the received data in LSR[2] (where parity error is stored in normal mode). Note that the 950 provides a 128-deep FIFO for LSR[3:1]. The transmitter FIFO is 9-bit wide and 128 deep. The user should write the 9th (MSB) data bit in SPR[0] first and then write the other 8 bits to THR.

As parity mode is disabled, LSR[7] is set whenever there is an overrun, framing error or received break condition. It is unaffected by the contents of LSR[2] (Now the received 9th data bit).

In 9-bit mode, in-band flow control is disabled regardless of the setting of EFR[3:0] and the Xon1/Xon2/Xoff1 and Xoff2 registers are used for special character detection.

Interrupts in 9-Bit Mode:

While IER[2] is set, on receiving a character with status error, a level 1 interrupt is asserted when the character and the associated status are transferred to the FIFO.

The 950 can assert an optional interrupt if a received character has its 9th bit set. As multi-drop systems often use the 9th bit as an address bit, the receiver is able to generate an interrupt upon receiving an address character. This feature is enabled by setting NMR[2]. This results in a level 1 interrupt being asserted when the address character is transferred to the receiver FIFO.

In this case, as long as there are no errors pending, i.e. LSR[1], LSR[3], and LSR[4] are clear, 0 can be read back from LSR[7] and LSR[1], thus differentiating between an address interrupt and receiver error or overrun interrupt in 9-bit mode. Note: if an overrun or error interrupt actually occurs, an address character may also reside in the FIFO. In this case, the software driver should examine the contents of the receiver FIFO as well as process the error.

The above facility produces an interrupt for recognizing any address characters. Alternatively, users can configure the OXPCIe952 UART to match the receiver data stream with up to four programmable 9-bit characters and assert a level 5 interrupt after detecting a match. The interrupt occurs when the character is transferred to the FIFO (See below).

NMR[0]: 9-bit mode enable

logic 0 ⇒	9-bit mode is disabled
logic 1 ⇒	9-bit mode is enabled

NMR[1]: Enable interrupt when 9th bit is set

logic 0 ⇒	Receiver interrupt for detection of an address character (i.e. 9th bit set) is disabled
logic 1 ⇒	Receiver interrupt for detection of an address character (i.e. 9th bit set) is enabled and a level 1 interrupt is asserted.

Special Character Detection

While the UART is in both 9-bit mode and enhanced mode, setting IER[5] enables detection of up to four address characters. The least significant eight bits of these four programmable characters are stored in special characters 1 to 4 (Xon1, Xon2, Xoff1 and Xoff2 in 650 mode) registers and the 9th bit of these characters are programmed in NMR[5] to NMR[2] respectively.

NMR[2]: Bit 9 of Special Character 1
NMR[3]: Bit 9 of Special Character 2
NMR[4]: Bit 9 of Special Character 3
NMR[5]: Bit 9 of Special Character 4
NMR[7:6]: Reserved

Bits 6 and 7 of NMR are always cleared and reserved for future use.

Modem Interrupt Disable Mask—MDM

The MDM register is located at offset 0x0E of the ICR.

This register is cleared after a hardware reset to maintain compatibility with 16C550. It allows the user to mask interrupts from individual modem lines or the serial input line.

UART power saving is controlled from outside the UART. See "Sleep Mode" on page 39.

MDM[0]: Disable delta CTS interrupt

logic 0 ⇒	Delta CTS is enabled. It can generate a level 4 interrupt when enabled by IER[3]
logic 1 ⇒	Delta CTS is disabled. It cannot generate an interrupt

MDM[1]: Disable delta DSR interrupt

logic 0 ⇒	Delta DSR is enabled. It can generate a level 4 interrupt when enabled by IER[3]
logic 1 ⇒	Delta DSR is disabled. It cannot generate an interrupt

MDM[2]: Disable Trailing edge RI interrupt

logic 0 ⇒	Trailing-edge RI is enabled. It can generate a level 4 interrupt when enabled by IER[3]
logic 1 ⇒	Trailing-edge RI is disabled. It cannot generate an interrupt

MDM[3]: Disable delta DCD interrupt

logic 0 ⇒	Delta DCD is enabled. It can generate a level 4 interrupt when enabled by IER[3]
logic 1 ⇒	Delta DCD is disabled. It cannot generate an interrupt

MDM[7:4]: Reserved

These bits must be set to 0.

Modem Wakeup Disable Mask–WDM

The WDM register is located at offset 0x17 of the ICR.

This register is set after a hardware reset to allow modem wakeup from the RI input only, to maintain compatibility to 450 legacy UART drivers. The register allows the user to mask individual wakeups in the UART power saving operation.

WDM[0]: Disable delta CTS wakeup

Logic 0 =>	Delta CTS is enabled. Delta CTS can wakeup the UART when it is asleep under auto-sleep operation
(default) Logic 1 =>	Delta CTS is disabled. It cannot wakeup the UART

WDM[1]: Disable delta DSR wakeup

Logic 0 =>	Delta DSR is enabled. Delta DSR can wakeup the UART when it is asleep under auto-sleep operation
(default) Logic 1 =>	Delta DSR is disabled. It cannot wakeup the UART

WDM[2]: Disable trailing edge RI wakeup

(default) Logic 0 =>	Trailing edge RI is enabled. Trailing edge RI can wakeup the UART when it is asleep under auto-sleep operation
Logic 1 =>	Trailing edge RI is disabled. It cannot wakeup the UART

WDM[3]: Disable delta DCD wakeup

Logic 0 =>	Delta DCD is enabled. Delta DCD can wakeup the UART when it is asleep under auto-sleep operation
(default) Logic 1 =>	Delta DCD is disabled. It cannot wakeup the UART

WDM[4]: Disable wakeup sensitivity

(default) Logic 0 =>	Enables wakeups based on the sensitivity settings of WDM[3:0]
Logic 1 =>	Disables all modem wakeups

WDM[7:5]: Reserved

These bits must be set to 0.

Readable FiCR—RFC

The RFC register is located at offset 0x0F of the ICR

This read-only register returns the current state of the FiCR register (FiCR is write-only). This register is included for diagnostic purposes.

Good-data Status Register—GDS

The GDS register is available directly in OXPCIe952 specific register space at 0x0F. It is also accessible at offset 0x10 of the ICR.

Good data status is set when the following conditions are true:

- ISR reads level0 (no interrupt), level2 or 2a (receiver data) or level3 (THR empty) interrupt
- LSR[7] is clear i.e. no parity error, framing error or break in the FIFO
- LSR[1] is clear i.e. no overrun error has occurred

GDS[0]: Good Data Status
GDS[7:1]: Reserved

DMA Status Register—DMS

The DMS register is located at offset 0x11 of the ICR. This register is unused in the OXPCIe952 except for test purposes.

Port Index Register—PIDX

The PIDX register is located at offset 0x12 of the ICR. This read-only register gives the UART index.

Alternative UART Baud Rate Control Registers: A_LATCH, A_ENABLE, A_DLL, A_DLM, A_CPR, A_TCR

Table 27 shows the OXPCIe952-specific alternative UART baud rate control registers, which can be used instead of the standard versions of these registers. This facility prevents legacy drivers with knowledge of the 950 UART register set from making incorrect assumptions about the OXPCIe952 crystal frequency and synthesize incorrect baud rate values by writing to the standard DLL, DLM, CPR and TCR registers.

OXPCIe952 UART Enhancements

The UART in the OXPCIe952 has a number of features in addition to the standard 950 mode, which enable it to exploit the high data throughput of the PCI Express interface and to minimize the host CPU utilization of the UART function. These features are fully supported by Oxford Semiconductor UART driver software.

FIFO Speed-Up Register (address 0x80..0x83)

The FIFO Speed-Up register provides software with the option of accessing the Transmit and Receive Holding Registers (THR/RHR) or Transmit and Receive FIFOs) in byte, half-word or word form. It enables, for example, the host machine to make a single 32-bit word access to the location, which the OXPCIe952 UART internally converts into four sequential byte accesses to THR/RHR to give four bytes of THR or RHR data. For a 32-bit read (assuming N+1 bytes data exist in FIFO), the relations are:

- FIFO[N-3] to bits 31-24
- FIFO[N-2] to bits 23-16
- FIFO[N-1] to bits 15-8
- FIFO[N] to bits 7-0

See Table 24 on page 24 for further details.

ISR Rapid Read Register (address 0x84..0x87)

The ISR Rapid Read register provides software with the option of accessing the LSR, ISR, RFL and TFL registers collectively in byte, half-word or word form. It enables, for example, the host machine to make a single 32-bit word access to the location, which then returns four bytes of key information and helps to reduce the host CPU overhead in servicing UART interrupts. Reading the register performs the same destructive read operations as if the LSR or ISR had been read directly. See Table 24 for further details.

Direct Access to All Registers

With a full 256-byte address range (0x00..0xFF), the OXPCIe952 provides direct access to its full register set, including registers which traditionally are only available indirectly via paging. See Table 22 on page 23 for the UART Direct Access Register Map. For compatibility with older UART drivers, the original paged concept (650/950/ICR registers) is fully supported.

Direct access cannot be performed to DLL and DLM registers. Please refer to Table 23 on page 23 for details of how to access to these registers.

Extended CPR for Legacy UART Modes

To give greater baud rate flexibility with its PCI Express–derived, 62.5-MHz clock source, the OXPCIe952 UART features an extended Clock Prescaler (CPR). The standard 5-bit integer and 3-bit fractional components reside at the standard locations, but are augmented by a sixth (MS) integer bit at bit 0 of a separate location, CPR2.

To maintain backward compatibility with software that is unaware of CPR2, any writes to CPR clear bit 0 of CPR2 and return the total CPR divider to its standard range.

RS485 Hold-Off Control

The RS485_DLYEN register is located at offset 0x14 of the ICR. Bit 0 of this register enables programmable RS485 switch off delay time.

RS485_DLYEN[0]: Enable RS485 Delay

logic 0 ⇒	Normal 950 style operation
logic 1 ⇒	When RS485 signalling is enabled, the turn off time at the end of transmission can be programmed from the commencement of the stop bit. There is a minimum delay of 2 clock cycles (of the transmitter phase clock) plus the delay programmed into the RS485_DLYCNT register. The delay comprises number of bits, and a phase. The phase represents the oversampling from the TCR register. Care should be taken that the phase programmed is possible. For example if TCR is set to 4 clocks/bit then only phases 0, 1, 2 and 3 are valid.

RS485 Programmable Delay Register 'RS485_DLYCNT'

The RS485 Programmable Delay Register 'RS485_DLYCNT' is located at offset 0x15_of the ICR. This register sets the programmable RS485 switch off delay time, provided the feature has been enabled via RS485_DLYEN[0].

RS485_DLYCNT[3:0]: RS485 Phase Delay

The number of phases required for the delay. A phase is the count of transmitter clock cycles per bit (programmable in the TCR). Each bit commences transmission on phase 0. It should be ensured that this phase is consistent with the TCR for correct operation.

RS485_DLYCNT[7:4]: RS485 Bit Delay

The number of bit periods required for the delay. Zero indicates the final stop bit.

UART DMA

The UART DMA function provides a fast bridge between the UART and the PCI Express interface. DMA compresses the complete filling or emptying of the UART Receive or Transmit FIFOs into a single PCI Express burst transfer.

In 950 mode the UART runs with FIFO levels of 128 bytes for both transmit and receive. The current device driver sets the IRQ fill level thresholds slightly below the maximum (112). When a FIFO fill level IRQ is dispatched, the CPU reads the various IRQ status registers to determine what operation to perform. Only the FIFO load/unload operation is performed under DMA.

The UART DMA controller is started by writing a non-zero value to the DMA Length register. The directional control of the DMA transfer is merged with the 32-bit DMA Transfer Length register to allow driver software to reduce host accesses to a minimum. The DMA operation begins immediately after software has programmed a non-zero value into this register and it cannot be stopped by software.

When DMA operation is under way, the DMA Status register reflects the state of the transfer. Software polls this register to determine whether the DMA transfer has completed. If an internal error occurs during the burst, the DMA transfer is aborted and the DMA error flag and the done flag are set. This register bit is a write-1-to-clear field and must be cleared before initiating a new transfer. In the event of an error condition the DMA Transfer Length register indicates the number of bytes that have been loaded/unloaded from the UART. By clearing the DMA error flag, the DMA Transfer Length register is automatically reset to zero. Software cannot reset this register directly when an error is outstanding.

The DMA core can work in 32- or 64-bit host systems. When programming the target host address registers, software must ensure that both fields are initialized correctly. If a 32-bit access is required, the upper 32-bit register must be set to zeros. Failure to do so causes an illegal 64-bit transaction to be dispatched. The DMA function uses the upper register to select between 32- or 64-bit operations.

The target-address specified by software can have byte alignment. Non-DWORD starting addresses perform 8- and/or 16-bit transfers initially to realign to 32 bits and then burst in DWORD form until the end of the transfer. The last transfer is always DWORD in size.

The Transmit and Receive DMAs differ significantly. For this reason the two operations are discussed below.

Rx FIFO Unloading Flow using DMA Core

When the UART device driver ISR has determined the number of bytes to be extracted from the FIFO, the ISR writes a 32-/64-bit target address into the DMA core followed by the transfer length in bytes. Writing to the DMA Transfer Length register activates the DMA channel.

During the DMA operation the ISR can continue to perform other tasks. The software must poll the DMA Status register to determine the status of the transfer. When the Status register indicates' done', the DMA transfer is considered complete and all data is assumed to be present in system memory. The ISR can now safely transfer the data back to user-space without worrying about memory coherency.

Allocation of the static driver DMA common buffer must be done with cache operations disabled. If this is not possible, a kernel cache-flush must be issued after the DMA status indicates transfer completion. The flush call incurs kernel overheads which would render the DMA performance improvement useless.

Tx FIFO Unloading Flow using DMA Core

When the UART device driver ISR has determined the number of bytes to be written to the FIFO, the ISR writes a 32-/64-bit target address into the DMA core followed by the transfer length in bytes. Writing to the DMA Transfer Length register activates the DMA channel.

During the DMA operation the ISR can continue to perform other tasks. The software must poll the DMA Status register to determine the status of the transfer. When the Status register indicates done, the DMA transfer is considered complete and all data is assumed to have been transferred to the transmit FIFO.

When all data has arrived and the UART is loaded, the done flag is set and the DMA transfer is complete.

A DMA transfer cannot be stopped after it has started.

UART DMA Registers

The OXPCIe952 UART DMA facility is supported by the registers shown in Table 35.

Table 35 UART DMA Registers (Bar Offset 0x1100, 0X2100)

Register Name	Offset Address	Bits	Type	Description
DMA Address Low	0x00	32	RW	Lower 32 bits [31:0] of host system target address
DMA Address High	0x04	32	RW	Upper 32 bits [63:32] of host system target address
DMA Transfer Length	0x08	32	RW	DMA transfer length and direction control
DMA Status	0x0C	4	RW1C	DMA status

The encoding of the **DMA Transfer Length** register (offset 0x08) is shown in Table 36.

Table 36 UART DMA Transfer Length Register

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31	DMA transfer direction 0 = transmit 1 = receive	W	RW	0
30:13	Reserved		R	0x00000
12:0	DMA length (usable range 1 to 128)	W	RW	0x0000

The encoding of the **DMA Status register** (offset 0x0C) is shown in Table 37.

Table 37 UART DMA Status Register

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:3	Reserved		R	0x00000000
2	DMA done (auto-cleared by hardware if a new DMA starts)		RW1C	1
1	DMA error		RW1C	0
0	DMA active		R	0

UART Baud Rate

The UART baud rates are generated using the OXPCIe952 PCI Express derived 62.5-MHz clock. Table 38 shows typical settings used to achieve industry-standard UART baud rates with a high degree of accuracy. The table also shows options for setting non-standard high-speed baud rates.

Table 38 Typical OXPCIe952 UART Baud Rate Settings

Target Baud Rate	TCR Reg(Bin)	CPR Bits(7:3)	CPR Bits(2:0)	Divisor DLM	Divisor DLL	Actual Baud Rate	Dev Per Bit %	Dev per 10-Bit Word %	
1,200	0100	00100	000	0C	B6	1200.45	0.037	0.3712	Standard
2,400	0100	00100	000	06	5B	2400.89	0.037	0.3712	Standard
4,800	0100	00100	000	03	2D	4804.74	0.099	0.9856	Standard
9,600	0100	00100	000	01	96	9621.31	0.221	2.2144	Standard
19,200	0100	00100	000	00	CB	19242.61	0.221	2.2144	Standard
38,400	0100	00100	000	00	66	38296.57	-0.270	-2.7008	Standard
57,600	0100	00100	000	00	44	57444.85	-0.270	-2.7008	Standard
115,200	0100	00100	000	00	22	114889.71	-0.270	-2.7008	Standard
230,400	0100	00100	000	00	11	229779.41	-0.270	-2.7008	Standard
460,800	0100	00001	000	00	22	459558.82	-0.270	-2.7008	Standard
921,600	0100	00001	000	00	11	919117.65	-0.270	-2.7008	Standard
1,843,200	0100	00100	010	00	02	1838235.29	-0.270	-2.7008	Standard
3,686,400	0100	00100	010	00	01	3676470.59	-0.270	-2.7008	Standard
7,812,500	0100	00001	000	00	02	7812500.00	0.000	0	Non-standard
8,928,571	0111	00001	000	00	01	8928571.43	0.000	4.8E-05	Non-standard
10,416,666	0110	00001	000	00	01	10416666.67	0.000	0	Non-standard
12,500,000	0101	00001	000	00	01	12500000.00	0.000	0	Non-standard
15,625,000	0100	00001	000	00	01	15625000.00	0.000	0	Non-standard

Parallel Port Function

Parallel Port Overview

The OXPCIe952 offers a compact, low power, IEEE-1284 compliant host-interface parallel port designed to interface to many peripherals such as printers, scanners and external drives. It supports compatibility modes, SPP, NIBBLE, PS2, EPP and ECP modes. The parallel port is always used in PCIexpress legacy mode. The register set is compatible with the Microsoft® register definition.

A sideband control signal is provided to control line drivers in applications where true 5.0-V signalling is required. By default, the parallel port uses 3.3-V output signalling, but its inputs tolerate 5.0-V signalling.

Operation and Mode selection

The system can access the parallel port via one 8-byte and one 4 byte block of I/O space; BAR0 contains the address of the basic parallel port registers, BAR1 contains the address of the upper registers. These are referred to as the lower block and upper block in this section. The 4 byte upper block can be located at an address 0x400 above the 8 byte lower block, allowing generic PC device drivers to be used to configure the port, because the addressable registers of legacy parallel ports always have this relationship.

SPP Mode

SPP (output-only) is the standard implementation of a simple parallel port. In this mode, the PD lines always drive the value in the PDR register. All transfers are done under software control. Input must be performed in nibble mode.

Generic device driver-software may use the address in I/O space encoded in BAR0 of function 1 to access the parallel port. The default configuration allocates 8 bytes to BAR0 in I/O space.

PS2 Mode

This mode is also referred to as bi-directional or compatible parallel port. To use the PS2 mode, the mode field of the Extended Control Register (ECR[7:5]) must be set to 001 using the negotiation steps defined by the IEEE1284 specification. PS2 operation is similar to SPP mode but, in this mode, directional control of the parallel port data lines (PD[7:0]) is possible by setting & clearing DCR[5], the data direction bit.

EPP Mode

To use the enhanced parallel port (EPP) the mode bits (ECR[7:5]) must be set to 100 using the negotiation steps defined by the IEEE1284 specification.

The EPP address and data port registers are compatible with the IEEE 1284 definition. A write or read in relation to an EPP port register is passed through the parallel port to access the external peripheral. In EPP mode, the STB#, INIT#, AFD# and SLIN# pins change from open-drain outputs to active push-pull (totem pole) drivers (as required by IEEE 1284) and the pins ACK#, AFD#, BUSY, SLIN# and STB# are redefined as INTR#, DATASTB#, WAIT#, ADDRSTB# and WRITE# respectively.

An EPP port access begins with the host reading from or writing to one of the EPP port registers. The device automatically buffers the data between the I/O registers and the parallel port depending on whether it is a read or a write cycle. When the peripheral is ready to complete the transfer it takes the WAIT# status line high. This allows the host to complete the EPP cycle.

If a faulty or disconnected peripheral fails to respond to an EPP cycle the host never senses a rising edge on WAIT#, and subsequently locks up. A built-in time-out facility is provided to prevent this from happening. It uses an internal timer which aborts the EPP cycle and sets a flag in the DSR register to indicate the condition. When the parallel port is not in EPP mode the timer is switched off to reduce current consumption. The host time-out period is 10ms as specified in the IEEE-1284 specification.

The register set is compatible with the Microsoft register definition. Assuming that the upper block is located 0x400 above the lower block, the registers are found at offset 0x000-0x007h and 0x0400-0x0402.

The OXPCIe952 supports version 1.7 of the EPP protocol.

ECP Mode

To use the Extended Capabilities Port (ECP) mode, the mode field of the Extended Control Register (ECR[7:5]) must be set to 011 using the negotiation steps as defined by the IEEE1284 specification.

ECP mode is compatible with the Microsoft register definition for ECP, and the IEEE-1284 bus protocol and timing.

ECP mode supports the decompression of run-length-encoded (RLE) data, in hardware. The RLE received data is expanded automatically by the correct number, into the ECP receiver FIFO. RLE on data to be transmitted is not available in hardware. This needs to be handled in software, if required.

Assuming that the upper block is located 0x0400 above the lower block, the ECP registers are found at offset 0x000-0x007 and 0x0400-0x0402.

Parallel Port Register Description

The parallel port registers are described below. Table 39 on page 62 shows an example where the upper block is 0x0400 above the lower block.

Table 39 Parallel Port Register Set [2]

Register Name	Address Offset	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SPP (Compatibility Mode) Registers										
PDR	000h	R/W	Parallel Port Data Register							
DSR (EPP mode)	001h	R	nBUSY	ACK#	PE	SLCT	ERR#	INT#	1	Timeout
(Other modes)	001h	R	nBUSY	ACK#	PE	SLCT	ERR#	INT#	1	1
DCR	002h	R/W	0	0	DIR	INT_EN	nSLIN#	INIT#	nAFD#	nSTB#
EPPA[1]	003h	R/W	EPP Address Register							
EPPD1[1]	004h	R/W	EPP Data 1 Register							
EPPD2[1]	005h	R/W	EPP Data 2 Register							
EPPD3[1]	006h	R/W	EPP Data 3 Register							
EPPD4[1]	007h	R/W	EPP Data 4 Register							
EcpDFifo	400h	R/W	ECP Data FIFO							
TFifo	400h	R/W	Test FIFO							
CnfgA	400h	R	Configuration A Register – always 90h							
CnfgB	401h	R	0	int	000000					
ECR	402h	R/W	Mode[2:0]			Reserved – Must write 00001 Reads return FIFO status and Service Interrupt status				
-	403h	-	Reserved							

Notes:
1 These registers are only available in EPP mode.
2 For registers in this table, prefix 'n' denotes that a signal is inverted at the connector. Suffix '#' denotes active-low signalling

The reset state of PDR, EPPA and EPPD1-4 is not determinable (i.e. 0xXX). The reset value of DSR is 'XXXXX111'. DCR and ECR are reset to 0000XXXX and 00010101 respectively.

GPIO Function

GPIO Overview

Eight dedicated GPIO pins on the OXPCIe952 device provide the system designer with a flexible and configurable control and sensing interface that can be dynamically or statically controlled for proprietary use. The OXPCIe952 GPIO interface is supported by Oxford Semiconductor software drivers, which give user applications access to all functions supported by the GPIO module.

Each GPIO bit in the block is completely independent of the other seven.

GPIO Modes

Each GPIO pin can be independently programmed to operate in one of four modes as follows:

- Input mode
- Output mode
- Open drain mode
- Pulse-width-modulated (PWM) output mode

In input mode, the external GPIO pin level is always reported, and any CPU writes to the GPIO output register have no effect. When selected as an input pin, the GPIO block can additionally assign interrupt capabilities to it. GPIO pins set as inputs can be configured to act as system wake-up lines.

Interrupts and/or wake-ups can be generated following various trigger events:

- Rising edge
- Falling edge
- Any edge
- High level
- Low level

In output mode, the external GPIO pin is driven hard to either high or low levels depending on the value present in the output register.

In open-drain mode, the external pin is left in a floating (Hi-Z) state if the output register is set to logic 0. When set to logic 1, the external pin is pulled down hard to logic 0.

In PWM output mode, the external GPIO pin is driven in a pulse-width-modulated pattern. Three dedicated registers are used to set the high and low durations of the PWM cycle and an 8-bit pre-scalar register. The values are scaled in steps based on the prescale value and each has a resolution of 12 bits for low and 12 bits for high.

The GPIO module uses a 62.5-MHz baseline clock, which switches to a much lower frequency standby clock when the OXPCIe952 is put into standby mode, which means that the GPIO PWM output frequency should be considered indeterminate.

Clearing the level-based interrupt mode while the level is still active causes a single clock cycle-wide inactive pulse to be generated, which allows the GPIO function to generate a new edge-based MSI when enabled.

GPIO BAR Detailed Breakdown

The GPIO BAR accesses various internal locations over its 1-Kbyte aperture, as shown in Table 40.

Table 40 GPIO BAR Details[(1)]

BAR Offset	Description
0x000	Class code & rev ID
0x004	Decimal number of GPIOs
0x008	Global GPIO IRQ status
0x00C	Global GPIO IRQ enable
0x010	Global GPIO IRQ disable
0x014	Global GPIO wake enable
0x018	Global GPIO wake disable
0x01C..0x0FF	Reserved (returns zero)
0x100..0x198	All GPIO registers[Note 1]
0x140..0x1FF	Reserved (returns zero)
0x200...0x23F	Reserved (returns zero)
0x240..0x3FF	Reserved (returns zero)

Note:
1 See "GPIO Registers" on page 65 for details.

Table 41 Class Code & Revision ID (Bar Offset 0x000)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:8	Classcode	-	R	0x070002
7:0	Revision ID	W	R	0x0

Table 42 Decimal Number of GPIOs (Bar Offset 0x004)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:5	Reserved	-	R	0x00000000
4:0	Number of GPIOs enabled (in decimal)	W	R	0x8

Table 43 Global GPIO IRQ Status (Bar Offset 0x008)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:8	Reserved	-	R	0x0000000
7:0	GPIO[7:0] IRQ status	-	R	0x00

Table 44 Global GPIO IRQ Enable (Bar Offset 0x00C)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:8	Reserved	-	R	0x0000000
7:0	GPIO[7:0] IRQ enable	W	RW	0x00

Table 45 Global GPIO IRQ Disable (Bar Offset 0x010)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:8	Reserved	-	R	0x0000000
7:0	GPIO[7:0] IRQ disable	W	RW	0xFF

Table 46 Global GPIO Wake Enable (Bar Offset 0x014)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:1	Reserved	-	R	0x0000000
0	Global GPIO wake enable bit	W	RW	1

Table 47 Global GPIO Wake Disable (Bar Offset 0x018)

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:1	Reserved	-	R	0x0000000
0	Global GPIO wake disable bit	W	RW	0

GPIO Registers

The GPIO register set is described in this section. Table 48 on page 66 shows the complete GPIO register set. Each individual GPIO bit has its own dedicated configuration, PWM prescaler and PWM timing registers. Additionally, a number of registers for writing, reading and interrupt controlling are shared by all eight GPIO pins. This means that functions can be accessed through two registers: either the GPIO configuration registers (which allows access to multiple functions for a single GPIO); or the individual function registers (which allow access to a single function for multiple GPIO pins). For example, the input value of GPIO 0 can be read from bit 13 of GPIO[0] Configuration Register or bit 0 of GPIO[7:0] Input Value Register.

Table 48 GPIO Registers (Bar Offset 0x100)

Address offset	Bits	Type	Description
0x100	16	RW	GPIO[0] Configuration
0x104	8	RW	GPIO[0] PWM Prescaler
0x108	32	RW	GPIO[0] PWM Timing
0x110	16	RW	GPIO[1] Configuration
0x114	8	RW	GPIO[1] PWM Prescaler
0x118	32	RW	GPIO[1] PWM Timing
0x120	16	RW	GPIO[2] Configuration
0x124	8	RW	GPIO[2] PWM Prescaler
0x128	32	RW	GPIO[2] PWM Timing
0x130	16	RW	GPIO[3] Configuration
0x134	8	RW	GPIO[3] PWM Prescaler
0x138	32	RW	GPIO[3] PWM Timing
0x140	16	RW	GPIO[4] Configuration
0x144	8	RW	GPIO[4] PWM Prescaler
0x148	32	RW	GPIO[4] PWM Timing
0x150	16	RW	GPIO[5] Configuration
0x154	8	RW	GPIO[5] PWM Prescaler
0x158	32	RW	GPIO[5] PWM Timing
0x160	16	RW	GPIO[6] Configuration
0x164	8	RW	GPIO[6] PWM Prescaler
0x168	32	RW	GPIO[6] PWM Timing
0x170	16	RW	GPIO[7] Configuration
0x174	8	RW	GPIO[7] PWM Prescaler
0x178	32	RW	GPIO[7] PWM Timing
0x180	8	RW	GPIO[0:7] Output Value
0x184	8	R	GPIO[0:7] Input Value
0x190	8	RW1C	GPIO[0:7] Interrupt Status
0x194	8	RW1S	GPIO[0:7] Interrupt Enable
0x198	8	RW1S	GPIO[0:7] Interrupt Disable

GPIO Configuration Registers

Table 49 on page 67 shows the bit assignment used by the eight GPIO Configuration registers, which reside at locations 0x00, 0x10, 0x20, 0x30, 0x40, 0x50, 0x60 and 0x70 respectively.

Table 49 GPIO Configuration Registers

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:14	Reserved		R	0x00000
13	GPIO input value 1 = input at logic ‘1’ 0 = input at logic ‘0’		R	0
12	GPIO output value 1 = output driving logic ‘1’ 0 = output driving logic ‘0’		RW	0
11	GPIO wake-up status 1 = active 0 = inactive		RW1C	0
10	GPIO interrupt status 1 = active 0 = inactive		RW1C	0
9	GPIO interrupt disable WR 1 = disable 0 = do not change current settings RD 1 = GPIO interrupt is disabled 0 = GPIO interrupt is enabled	W	RW	0
8	GPIO interrupt enable WR 1 = enable 0 = do not change current settings RD 1 = GPIO interrupt is enabled 0 = GPIO interrupt is disabled	W	RW	0
7	GPIO wake-up enable 1 = enabled 0 = disabled	W	RW	0
6:4	GPIO interrupt and wake-up mode 000 = Rising edge 001 = Falling edge 010 =High level (continuous assertion) 011 =Low level (continuous assertion) 1xx =Any edge	W	RW	000
3:2	Reserved		R	00
1:0	GPIO pin mode 00 =Input mode 01 =Output mode 10 =Open drain mode 11 = PWM output mode	W	RW	00

Notes:

1 Writing 00 or 11 to bit[9..8] (GPIO interrupt disable and GPIO interrupt enable) does not change current settings.

2 Bit 10 (interrupt status) is RW1C (read/write 1 to clear) type, which means that the IRQ status bit can only be cleared by writing 1 to it. If the interrupt is level-based and the IRQ event criterion is still met, a new interrupt is generated immediately after clearing this bit.

3 Bit 11, wake-up status, is also RW1C type.

GPIO PWM Prescaler Registers

Table 50 shows the bit assignment used by the eight PWM Prescaler registers, which reside at locations 0x04, 0x14, 024, 0x34, 0x44, 0x54, 0x64 and 0x74. Each of these registers uniquely assigns a PWM prescaler ratio to a specific GPIO pin. The ratio is only applicable when the GPIO mode is set to PWM output mode. The 8-bit value in bits 7 to 0 equates to a scaling factor of the reference frequency which can is selected via bit 31.The reference time-base is either 62.5-MHz or (62.5-MHz / 65536) ~953 Hz.

In conjunction with the settings available in the PWM timing registers, the combined divisors allow frequencies from pulses at 31-MHz to less than 1 pulse an hour.

GPIO PWM prescaler registers are not intended to be modified while there is an active PWM pulse. In PWM mode, changing the prescaler register for a currently-active GPIO produces an indeterminate pulse pattern.

Table 50 PWM Prescaler Register Bit Assignments

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31	Slow clock mode 1 = Use internal 954Hz clock as time base for pre-scalar 0 = Use internal 62.5MHz clock as time base for pre-scalar	W	RW	0
30:8	Reserved		R	0x000000
7:0	Pre-scalar divide reference time-base (range 1 to 255)	W	RW	0x00

Notes:

1 A zero pre-scale turns PWM off.
2 Period = time-base * pre-scale * (PWM low + PWM high).
3 Fastest frequency = 16 nS * 1 * (1 + 1) = 32 nS (31.25 MHz).
4 Slowest frequency = (16 nS * 65536) * 255 * (8191 + 8191) = 4380 S (0.00028 Hz).

GPIO PWM Timing Registers

Table 51 on page 69 shows the bit assignment used by the eight PWM timing registers, which reside at locations 0x08, 0x18, 0x28, 0x38, 0x48, 0x58, 0x68 & 0x78.

Each register uniquely assigns PWM timing parameters to a specific GPIO pin. The parameters are only applicable when the GPIO mode is set to PWM output mode. If any field is set to zero, the PWM output does not toggle. PWM low level and high level timing fields can be safely changed while there is an active PWM pulse on the GPIO channel concerned.

Table 51 PWM Timing Registers

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:29	Reserved		R	000
28:16	PWM low-level timing in steps of pre-scalar interval (range 1 to 8191)[(1)]	W	RW	0x0000
15:13	Reserved		R	000
12:0	PWM high-level timing in steps of pre-scalar interval (range 1 to 8191)[(1)]	W	RW	0x0000

Note:
1 A zero in either the low- or high-level timing field turns PWM off.

GPIO[7:0] Output Value Register

Table 52 shows the bit assignment for the GPIO[7:0] Output Value register, which resides at location 0x80.

Table 52 GPIO[7:0] Output Value Register Bit Assignment

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:8	Reserved		R	0x000000
7:0	GPIO[7:0]: Open drain or Output mode value status. The following values represent a single register bit per pin: Open drain mode[(1)] 1 = output driven to logic ‘1’ 0 = floating output Output mode[(2)] 1 = output driven to logic ‘1’ 0 = output driven to logic ‘0’		RW	0x00

Notes:
1 GPIO configuration register bits [1:0] = ‘10’.
2 GPIO configuration register bits [1:0] = ‘01’.

GPIO[7:0] Input Value Register

Table 53 shows the bit assignment for the GPIO[7:0] Input Value register, which resides at location 0x84.

Table 53 GPIO[7:0] Input Value Register Bit Assignment

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:8	Reserved		R	0x000000
7:0	GPIO[7:0]: Input value. The following values represent a single register bit per pin: 1 = Input value is logic ‘1’ 0 = Input value is logic ‘0’		R	0x00

GPIO[7:0] Interrupt Status Register

Table 54 shows the bit assignment for the GPIO[7:0] Interrupt Status register, which resides at location 0x90.

This register is RW1C type, which means that IRQ status bits can only be cleared by writing 1 to the appropriate bit. If the interrupt is level-based and the IRQ event criteria is still met, a new interrupt is generated immediately after clearing this register.

Table 54 GPIO[7:0] Interrupt Status Value Registers Bit Assignment

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:8	Reserved		R	0x000000
7:0	GPIO[7:0]: Interrupt status. The following values represent a single register bit per pin: 1 = active 0 = inactive		R	0x00

GPIO[7:0] Interrupt Enable Register

Table 55 shows the bit assignment for the GPIO[7:0] Interrupt Enable register, which resides at location 0x94.

This register is RW1S type, which means that IRQ enable bits can only be set by writing 1 to the appropriate bit. Reading the register returns 1 for the bits that are enabled, and 0 for those that are disabled.

Table 55 GPIO[7:0] Interrupt Enable Register Bit Assignment

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:8	Reserved		R	0x000000
7:0	GPIO[7:0]: interrupt enable. The following values represent a single register bit per pin: 1 = enable 0 = do not change	W	RW1S	0x00

GPIO[7:0] Interrupt Disable Register

Table 56 on page 71 shows the bit assignment for the GPIO Interrupt Disable register, which resides at location 0x98.

This register is RW1S type, which means that IRQ disable bits can only be set by writing 1 to the appropriate bit. Reading the register returns 1 for the bits that are disabled, and 0 for those that are enabled.

Table 56 GPIO Interrupt Disable Register Bit Assignment

Bits	Description	Read/Write		Reset
		EEPROM	PCIexpress	
31:8	Reserved		R	0x000000
7:0	GPIO[7:0]: interrupt disable. The following values represent a single register bit per pin: 1 = disable 0 = do not change	W	RW1S	0x00

EEPROM Interface & Programming Capabilities

Overview

The OXPCIe952 EEPROM read and programming capability allows developers to customize the default personality the device exhibits after system reset by bootstrapping any of the device registers available in the chosen mode option. The OXPCIe952 dedicated EEPROM controller performs this task in two passes. The first pass uses data in an internal ROM store containing mandatory personalization of the PCI Express core for a specific mode setting. The second pass involves accessing the off-chip EEPROM for the customer personality application.

The OXPCIe952 is fully supported by the Oxford Semiconductor Oxide utility. This deploys the zone concept currently used with other Oxford Semiconductor products, by which each external EEPROM zone accesses a specific peripheral interface using a peek and poke mechanism. In addition, a more automatic device read/write capability is provided on the OXPCIe952.

The automatic EEPROM read/writing capability allows software developers to apply upgrades to the final product more easily, or handle peek and poke debugging in a development environment. The system software can set up an operation and then inform the EEPROM interface to execute it. The OXPCIe952 internal EEPROM control function then performs the task in the background, with system software polling the EEPROM controller until the command is complete—for a read operation, software may then read the data register; for a write operation, software may then start another operation. EEPROM devices need to be opened and closed for writing. To aid this, the auto-write enable/disable operation is done by setting a command bit and then polling the status flag as if it had been an EEPROM read or write operation.

For further customer solution protection, the EEPROM write feature can optionally be masked to prevent accidental or malicious alteration. The boot-EEPROM can write to a write-protect register to stop host software from modifying the contents of the external EEPROM.

Boot ROM: Phase 1

At power-on, device mode settings determine the basic personality of the OXPCIe952. The new personality must be imprinted before the external EEPROM makes any other modifications, because the first phase enables specific functions and BARs within the PCI Express core on the OXPCIe952.

The boot ROM is programmed with information to configure PCI Express registers for all functions present in the device. Functions depend on the settings of the mode, GPIO enable and UART enable pins. No functions require BAR 3, BAR 4 or BAR 5, so they are permanently disabled.

The registers modified by personality assignment are listed below:

- VENDOR-ID, DEVICE-ID
- CLASS-CODE & REVISION ID
- SUBSYSTEM-VENDOR-ID, SUBSYSTEM-DEVICE-ID
- All applicable BAR regions (mask and type)

Figure 2 on page 10 shows the OXPCIe952 configuration space, which is allocated for each function. All targets are 32 bits wide, so the ROM is always formatted with 32-bit data.

Boot ROM: Phase 2

The second phase of bootstrapping uses the Microwire interface to upload customer configurations from an off-chip EEPROM device. The EEPROM interface operates with a divisor of 70 (decimal) to produce a clock rate of ~890 KHz. The slowest supported Microwire device is 1 MHz—at this rate an EEPROM access takes approximately 31 µS.

For safety, the first operation of the phase 2 configuration is to perform a dummy read to the EEPROM to detect whether a device is present. This also detects the number of address bits on the device. Booting occurs when these two metrics have been extracted.

The first location of the EEPROM must have a Zone 0 header; and the ID field extracted by the phase 2 configuration must match. If it does not, the EEPROM image is considered corrupt and the phase 2 boot aborts, setting the bootstrap error status bit. If the EEPROM image passes this first check but is still corrupt, the booting sequence continues blindly, accessing the off chip EEPROM until the address register loops back round to zero. At this point it recognizes that the image is corrupt and abandons configuration, setting the bootstrap error status bit.

CRC16

The EEPROM image ends with a CRC16 value. The value is calculated on every word in the EEPROM that is read out by the EEPROM interface. The CRC16 value is initialized to all Fs and uses the PCI Express DLLP CRC16 polynomial:

$$100B = x^{16} + x^{12} + x^{3} + x + 1$$

If the CRC16 word fails to match the expected CRC16, the bootstrap is considered to have failed and the bootstrap error bit is set. However the device will attempt to continue to function using the settings read from the EEPROM.

CRC generation starts with bit 0 of first header word and progresses to bit 15 of the header word. CRC generation then moves to bit 0 of word 1 and so on.

EEPROM Zone Allocation

The basic format of the EEPROM image for bootstrapping and using the Oxford Semiconductor EEPROM programming utilities is shown in Table 10.

The first location in the EEPROM is the Zone 0 header, as shown in Table 57. The EEPROM controller validates the EEPROM contents by checking the ID field of the Zone 0 header word.

Table 57 EEPROM Zone 0 Format

Bits	Description
15:5	These bits denote the ID-field and must be encoded with the following bit sequence: 1001 0111 110
4	1—Zone 1 present 0—Zone 1 does not exist
3	1—Zone 2 present 0—Zone 2 does not exist
2	1—Zone 3 present 0—Zone 3 does not exist
1	1—Zone 4 present 0—Zone 4 does not exist
0	1—Zone 5 present 0—Zone 5 does not exist

Each zone uses the same method for capturing the configuration data. If the EEPROM attempts to program illegal states, undefined operations within the device may occur. Read-only registers cannot be overwritten.

The EEPROM content is formatted in pairs of information. The first word is the target address/control, and the second word is the target data. Table 58 shows the format of the address word.

Table 58 EEPROM Address Word Format

Bits	Description
15	1—last address/data pair of this zone 0—more address/data pairs to come in this zone
14	1—16-bit write 0—8-bit write
13:0	Target address in zone

Bit 15 indicates whether more updates are expected for this zone, and bit 14 permits either 8-bit or 16-bit values to be written to the target register. Target data can either be 8 or 16 bits wide. Because the EEPROM is always 16 bits wide, 8-bit writes use the lower eight bits of EEPROM data and the upper bits are ignored.

The address mapping relationships for each zone type are described below.

Zone 1: PCI Express Configuration Space

Each PCI Express function configuration space can use up to 11 address bits. Because each function requires access, the function-number is encoded into the address field as shown in Table 59.

Table 59 PCI Express EEPROM Address Word Format

Bits	Description
15	1—last address/data pair of this zone 0—more address/data pairs to come in this zone
14	1—16-bit write 0—8-bit write
13:12	Target function
11	1—target PHY 0—target PCI Express core.
10:0	Target register

Table 60 on page 75 shows which PCI Configuration registers are writable from the EEPROM for each function.

Table 60 PCI Configuration Registers Writable from the EEPROM for each Function

Offset	Bits	Description
0x02	7:0	Device ID bits 7 to 0
0x03	7:0	Device ID bits 15 to 8
0x06	3:0	Must be '0000'
0x06	4	Extended Capabilities
0x06	7:5	Must be '000'
0x09	7:0	Class Code bits 7 to 0
0x0A	7:0	Class Code bits 15 to 8
0x0B	7:0	Class Code bits 23 to 16
0x2E	7:0	Subsystem ID bits 7 to 0
0x2F	7:0	Subsystem ID bits 15 to 8
0x3D	7:0	Interrupt pin
0x42	7:0	Power Management Capabilities bits 7 to 0
0x43	7:0	Power Management Capabilities bits 15 to 8

Zone 2: UART

This zone permits the UART and its DMA controller to be accessed. The address field is formatted to make the distinction between each of the target modules, as shown in Table 61. The global UART control registers can also be reached.

Table 61 UART EEPROM Address Word Format

Bits	Description
15	1—last address/data pair of this zone 0—more address/data pairs to come in this zone
14	1—16-bit write 0—8-bit write
13:12	0—UART is the target instance 1—global UART registers 2 or 3—DMA is the target instance
11:8	UART instance being selected (range 0 to 1).
7:0	Target UART register

Zone 3: Parallel Port

Register mapping is one-to-one to the parallel port, in that the SPP and EPP registers reside at 0x000 and the ECP registers start at 0x008. See "Parallel Port Function" on page 59 for further details.

The allocation of the address field is shown in Table 62 on page 76.

Table 62 Parallel Port EEPROM Address Word Format

Bits	Description
15	1—Last pair of this zone 0—More pairs to come in this zone
14	1—16-bit write 0—8-bit write
13:4	Reserved—must be set to zero
3:0	Parallel port register

Zone 4: GPIO

The first GPIO block resides between 0x00..0xFF in accordance with the detailed GPIO module design documents. The slave GPIO resides at offset 0x100..0x1FF. The address format is shown in Table 63.

Table 63 Parallel Port EEPROM Address Word Format

Bits	Description
15	1—last address/data pair of this zone 0—more address/data pairs to come in this zone
14	1—16-bit write 0—8-bit write
13	Reserved; must be set to zero
12:0	Target register

Zone 5: Not Used

Operating Conditions

Maximum Ratings

Table 64 shows the device absolute maximum device ratings.

Table 64 Absolute Maximum Device Ratings

Symbol	Parameter	Rating		Units
		Min	Max	
V_{DD}3V3	3.3V DC supply voltage	3.0	3.6	V
V_{DD}1V2	1.2V DC core supply voltage	1.08	1.32	V
V_{DDIO} (3V3)	multi-voltage IO DC supply voltage @3.3V	3.0	3.6	V
V_{DDIO} (2V5)	multi-voltage IO DC supply voltage @2.5V	2.25	2.75	V
V_{DDIO} (1V8)	multi-voltage IO DC supply voltage @1.8V	1.71	2.16	V
V_{DDPMUO}	PMU DC output voltage	1.14	1.26	V
T_{OP}	Operational temperature range	-40	85	°C

Devices should be stored as per conditions indicated in JEDEC Standard J-STD-033.

Electrical Characteristics

Table 65 shows the multi-voltage device I/O buffer electrical characteristics.

Table 65 Multivoltage (3.3V, 2.5V, 1.8V) Device I/O Buffer Electrical Characteristics

Symbol	Parameter	Condition	Rating			Units
			Min	Typ	Max	
V_{MSSIO}	I/O Ground		-0.3		0.3	V
V_{MIH}	Input high voltage		$0.8*V_{DDIO}$		V_{DDIO}	V
V_{MIL}	Input low voltage		V_{SSIO}		$0.2*V_{DDIO}$	V
I_{MI}	Input leakage current				±1	µA
V_{MIMAX} (1.8)	Input voltage tolerance @1.8V			2.5		V
V_{MIMAX} (2.5)	Input voltage tolerance @ 2.5V			3.3		V
V_{MOH}	Output high voltage		$V_{DDIO} - (0.15*V_{DDIO})$		V_{DDIO}	V
V_{MOL}	Output low voltage		V_{SSIO}		$V_{SSIO} + (0.15*V_{DDIO})$	V

Table 66 shows the 5-V tolerant device I/O buffer electrical characteristics.

Table 66 3.3V (5V tolerant) Only Device I/O Buffer Electrical Characteristics

Symbol	Parameter	Condition	Rating			Units
			Min	Typ	Max	
V_{IH}	Input high voltage		2.0		5.5	V
V_{IL}	Input low voltage		-0.3		0.8	V
V_T	Threshold point		1.17		1.23	V
V_T+	Schmitt trig. low to high threshold point		1.51		1.59	V
V_{T-}	Schmitt trig. High to low threshold point		0.92		0.98	V
I_I	Input leakage current				±10	µA
I_{OZ}	Tristate output leakage current	V_O = 3.3V or 0V			±10	µA
V_{IMAX}	Maximum input voltage				5.5V	V
V_{OH}	Output high voltage		2.4			V
V_{OL}	Output low voltage				0.4	V

AC Electrical Characteristics

Parallel Port

Table 67 shows the EPP Mode dynamic characteristics.

VDDIO0 = 3V to 3.6V, VDDCORE = 1.08V to 1.32V; Tamb = -40°C to +85°C, unless otherwise specified. Typical values are VDDIO0 = 3.3V; VDDCORE = 1.2V; Tamb = 25°C, unless otherwise specified.

Table 67 EPP Mode Dynamic Characteristics

Symbol	Parameter	Condition	Min	Max	Unit
T_H	Host response time	Output load is 50pF	138	165	ns
T_D	Minimum data setup time (ECP/EPP modes only)	-	0	-	ns

Figure 4 shows the EPP Mode Data Write Phase timing.

Figure 4 EPP Mode Data Write Phase

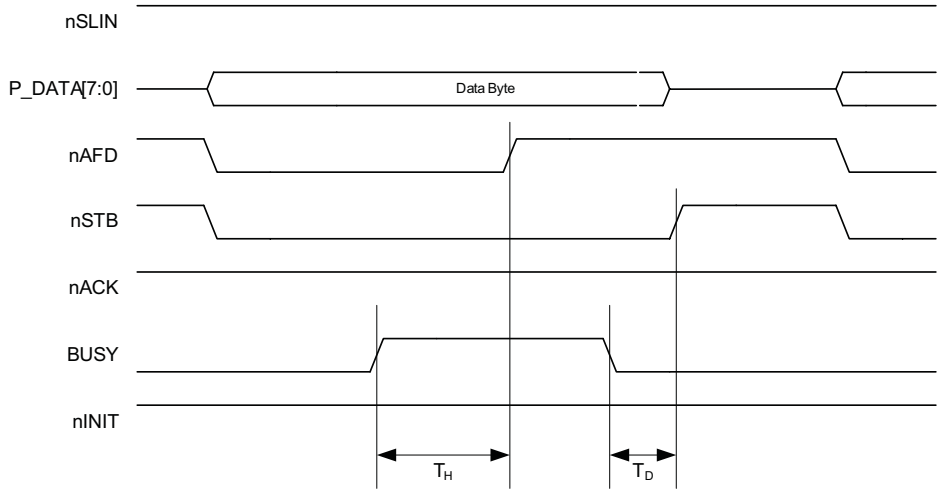


Figure 5 shows the EPP Mode Data Read Phase timing.

Figure 5 EPP Mode Data Read Phase

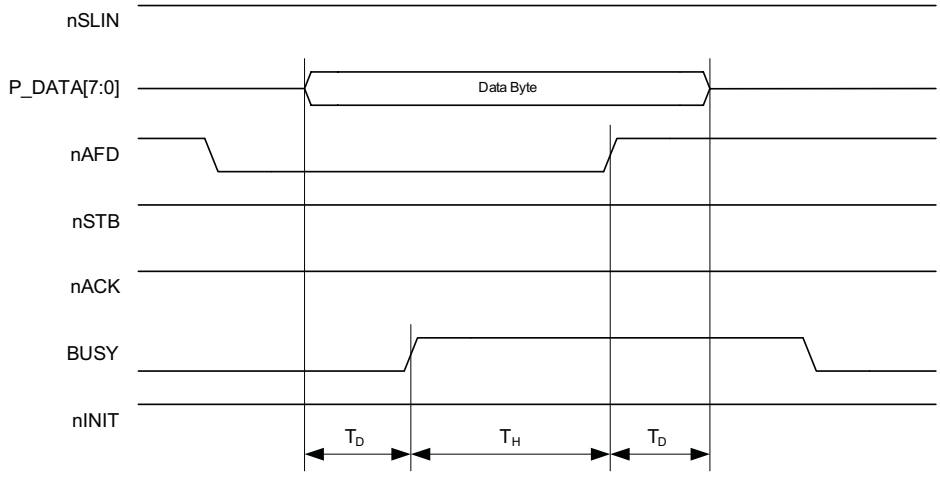
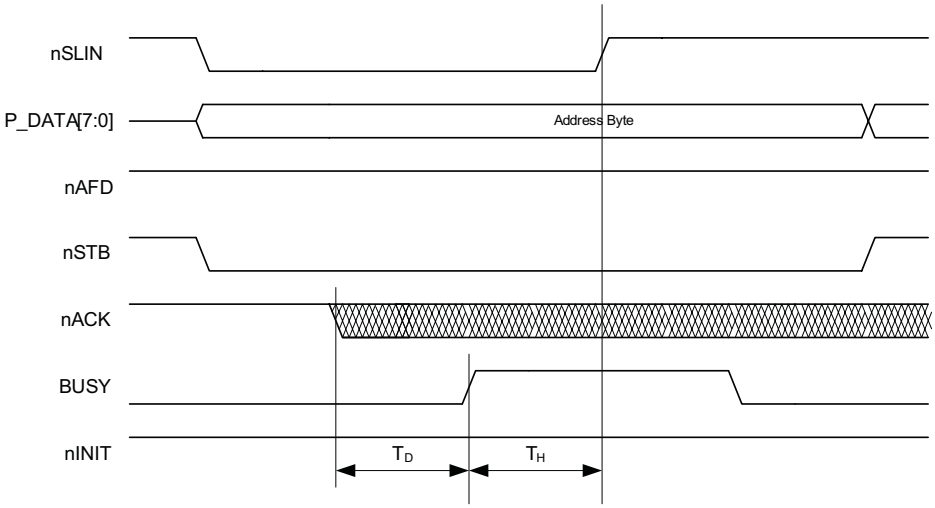


Figure 6 shows the EPP Mode Address Write Phase timing.

Figure 6 EPP Mode Address Write Phase



EEPROM (Microwire) Interface

Table 68 shows the EEPROM (Microwire) interface dynamic characteristics. All timing shown is with respect to the rising edge of EECK.

VDDIO2 = 3V to 3.6V, VDDCORE = 1.08V to 1.32V; Tamb = -40°C to +85°C; unless otherwise specified.

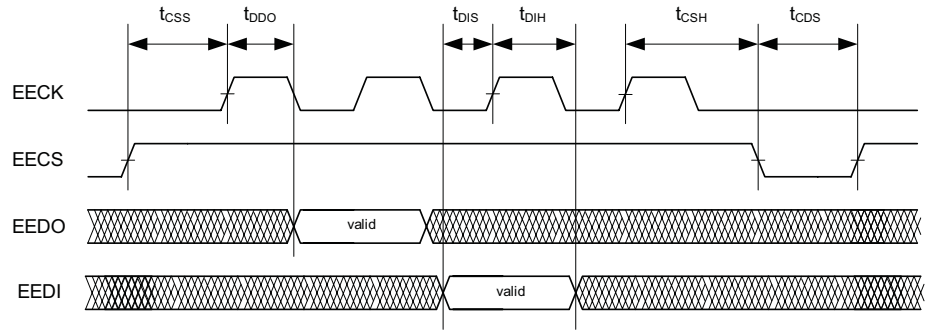
Typical values are VDDIO2 = 3.3V; VDDCORE = 1.2V; Tamb = 25°C; unless otherwise specified.

Table 68 EEPROM (Microwire) Interface Dynamic Characteristics

Symbol	Parameter	Condition	Min	Max	Unit
t_{CCS}	Rising EECS to EECK	Output load is 15pF	656	-	ns
t_{CSH}	EECK to falling EECS	Output load is 15pF	1344		ns
t_{CDS}	EECS deselect	Output load is 15pF	1344	-	ns
t_{DIS}	EEDI setup	-	48		ns
t_{DIH}	EEDI hold	-	0	-	ns
t_{DDO}	EEDO output delay	Output load is 15pF	656	688	ns
t_{CK}	Clock Period	-	1344	-	ns

Figure 5 shows the EEPROM (Microwire) interface timing.

Figure 7 EEPROM (Microwire) Interface Timing



Package Mechanical Drawings

Figure 8 shows the top and side view of the 120-pin device package.

Figure 8 120-Pin T-fpBGA Package-Top & Side View

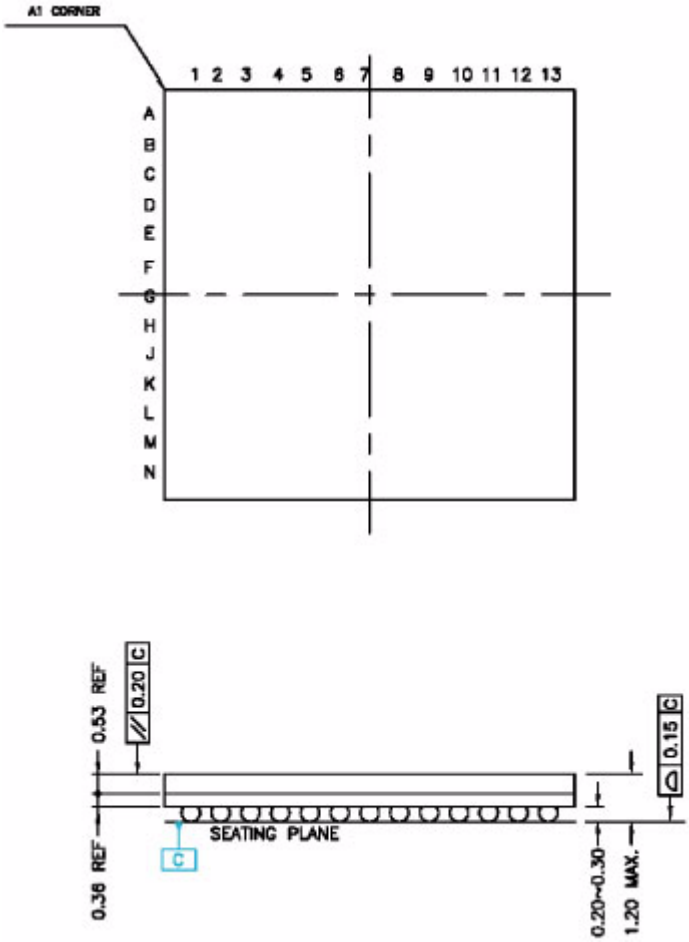
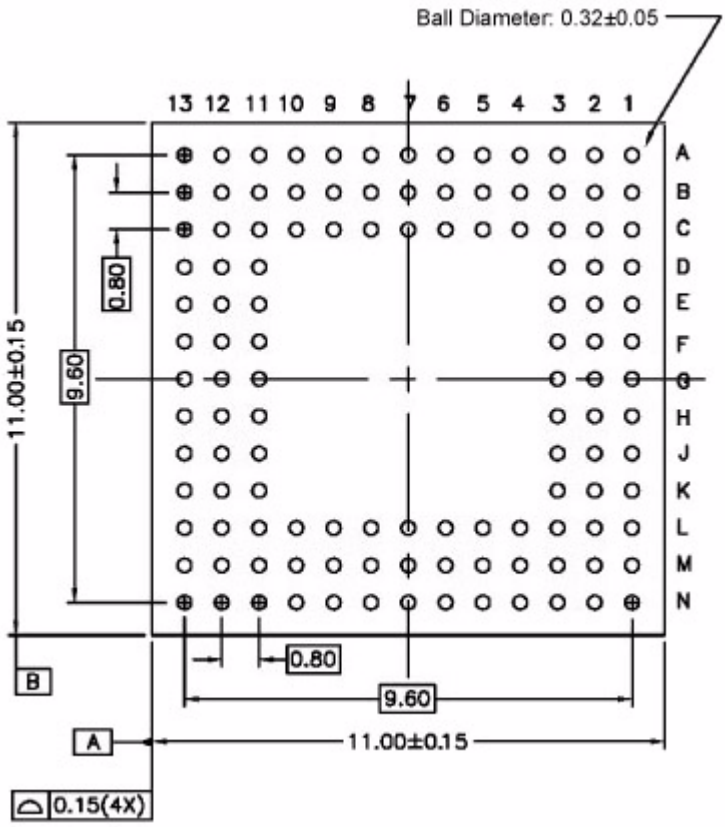


Figure 9 shows the bottom view of the 120-pin device package.

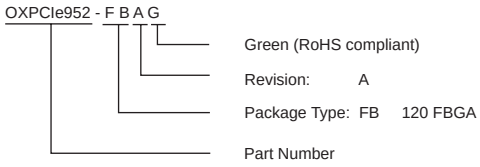
Figure 9 120-Pin T-fpBGA Package-Bottom



Ordering Information

The order code for the Oxford Semiconductor OXPCIe952 is TBD.

The following conventions are used to identify Oxford Semiconductor products:



Contacting Oxford Semiconductor

Oxford Semiconductor contact details:

Oxford Semiconductor, Inc.
1900 McCarthy Boulevard, Suite 210
Milpitas, CA 95035
USA

Website: http://www.oxsemi.com
Email: sales@oxsemi.com

Alternatively, you can contact your local representative.

Revision Information

Table 69 documents the revisions of this guide.

Table 69 Revision Information

Revision	Modification
October 2007	First publication
November 2007	Minor text revisions
January 2008	AC Electrical Characteristics (Tables 67-68) and waveform diagrams (Figures 4-7) added

Windows is a trademark of Microsoft, Inc., registered in the US and other countries.
All other trademarks are the property of their respective owners

© Oxford Semiconductor, Inc. 2008

The content of this manual is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Oxford Semiconductor, Inc. Oxford Semiconductor, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in this book.