



OX16C954 High Performance Four Channel UART with 128 byte FIFOs

Features

- Four full-duplex asynchronous channels
- IBM PC/AT compatible
- Fully software compatible with industry standard 16C450, 16C550, 16C654, 16C750 and 16C952 UARTs
- Pin compatible with TL16C554 and ST16C654
- 128-byte deep FIFO per transmitter and receiver
- 3-bit wide (128 deep) status FIFO per receiver
- Baud rates up to 12.5 Mbps in normal mode and 50Mbps in external 1x clock mode
- Readable FIFO levels
- Flexible clock prescaler to offer clock division in the range of 1 to 31.875 in steps of 0.125
- Isochronous mode using external 1x baud rate clock up to 50Mbps.
- Detection of stale data in the receiver FIFO
- 6 modem lines per channel; 4 input, 2 output (CTS#,DSR#,RI#,DCD#,DTR#,RTS#)
- Automated in-band (software) flow control using Xon/Xoff in both directions
- Programmable Xon/Xoff characters
- Automated out-of-band (hardware) flow control using CTS#/RTS# and DSR#/DTR#
- Readable in-band and out-of-band flow control status
- Arbitrary trigger levels for receiver and transmitter FIFO interrupts and automatic in-band and out-of-band flow control
- Transmitter empty interrupt (shift register and FIFO both empty)
- Infra-red (IrDA) receiver and transmitter
- Transmitter and receiver disable
- RS485 buffer enable signals
- Software channel reset
- Four bytes of device ID
- Sleep mode (low operating current)
- System clock up to 50 MHz.
- 68 PLCC package

Description

The OX16C954 is a four-channel ultra-high performance UART offering data rates up to 12.5Mbps per channel and 128-deep FIFOs on transmitters and receivers. Deep FIFOs reduce CPU overhead and allow the high data rate to be utilised. Each channel operates independently from the other channels.

It is software compatible with the widely used industry-standard 16C450, 16C550 devices as well as the OX16C952, ST16C654 and the TL16C750 devices. It is pin-compatible with the TL16C554 and ST16C654 devices.

In addition to increased performance and FIFO size, the OX16C954 also provides enhanced features including improved flow control. Automated software flow control using Xon/Xoff and automated hardware flow control using CTS#/RTS# and DSR#/DTR# prevent FIFO overrun. Flow control and interrupt thresholds are fully programmable and readable enabling

programmers to fine-tune the performance of their system.

FIFO levels are readable to facilitate fast driver applications in poll mode. The addition of software reset enables recovery from unforeseen error conditions and drivers may restart gracefully. A user programmable special character may also be detected.

The OX16C954 is ideally suited to PC applications, such as high-speed COM1, COM2, COM3, COM4 add-in cards which enable PC users to take advantage of the maximum performance of analogue modems and ISDN terminal adapters. It is also suitable for any piece of equipment requiring high speed RS232/RS422/RS485 interfaces. Fabricated in 0.5µm process, OX16C954 has a low operating current and a sleep mode for battery operated applications.

Contents

Features	1
Description	1
Contents	2
1 Performance Comparison	4
Improvements of the OX16C954 over previous generations of PC UART:	4
2 Block Diagram	5
3 Pin Information	6
3.1 Pin Diagram	6
3.2 Pin Descriptions	7
4 Operating Conditions	10
4.1 Absolute Maximum Rating	10
4.2 Recommended Operating Condition	10
5 DC Electrical Characteristics	10
6 AC Electrical Characteristics	11
Data bus timing for Intel mode:	11
Data bus timing for Motorola mode:	11
7 Timing Waveforms	12
8 Serial Ports	14
8.1 General Mode Selection	14
8.1.1 450 Mode	14
8.1.2 550 Mode	14
8.1.3 Extended 550 Mode	14
8.1.4 750 Mode	14
8.1.5 650 Mode	15
8.1.6 950 Mode	15
8.2 Register Memory Map	16
8.2.1 UART Register Description Table	17
8.3 Reset Configuration	20
8.3.1 Hardware Reset	20
8.3.2 Software Reset	20
8.4 Specific Features of the OX16C954	21
8.4.1 Additional Control Register 'ACR' (ICR offset 0x00)	21
8.4.2 Additional Status Register ('ASR')	22
8.4.3 Transmitter and Receiver FIFO levels 'TFL' and 'RFL'	23
8.4.4 Receiver Interrupt Trigger Level Register 'RTL' (ICR offset 0x05)	23
8.4.5 Transmitter Interrupt Trigger Level Register 'TTL' (ICR offset 0x04)	23
8.4.6 Flow Control Trigger Levels 'FCL' and 'FCH' (ICR offsets 0x06 and 0x07)	23
8.4.7 Device Identification (ICR offsets 0x08, 0x09, 0x0A and 0x0B)	24
8.4.8 Times Clock Register 'TCR' (ICR offset 0x02)	24

8.5	Flow Control	25
8.5.1	Enhanced Features Register ('EFR')	25
8.5.2	Special Character Detection Operation	26
8.5.3	Automatic In-band Flow Control Operation	26
8.5.4	Automatic Out-of-band Flow Control Operation	27
8.6	Baud Rate Generation	28
8.6.1	Clock Prescaler and Sampling Clock Configuration	28
8.6.2	1.8432MHz Clock	29
8.6.3	7.372MHz Clock	29
8.6.4	14.7456MHz Clock	29
8.6.5	18.432MHz Clock	30
8.6.6	32.0MHz Clock	30
8.6.7	33.0MHz Clock	31
8.6.8	40.0MHz Clock	31
8.6.9	50.0MHz Clock	31
8.6.10	External 1x Clock Mode	32
8.6.11	Very high speed baud rates up to 12Mbps	32
8.6.12	Clock oscillator	33
8.7	FIFOs	34
8.7.1	Transmitter and Receiver Holding Registers	34
8.7.2	FIFO Control Register ('FCR')	34
8.7.3	DMA Transfer Signalling:	35
8.8	Transmitter and Receiver	36
8.8.1	Receiver Framing, Sampling and False Start Bit Detection	36
8.8.2	Line Control Register ('LCR')	36
8.8.3	Line Status Register ('LSR')	37
8.9	Interrupts	38
8.9.1	Interrupt Enable Register ('IER')	38
8.9.2	Interrupt Status Register (ISR):	39
8.9.3	Interrupt Description	40
8.9.4	Sleep Mode	40
	Note that OX16C942 does not offer the sleep mode operation in IrDA mode.	41
8.10	Modem interface	41
8.10.1	Modem Control Register (MCR):	41
8.10.2	Modem Status Register ('MSR')	42
8.11	Scratch Pad Register ('SPR')	43
9	Package Information	44
	68 Pin Plastic Leaded Chip Carrier (68 PLCC)	44
10	Ordering Information	45

1 Performance Comparison

The performance of OX16C954 is compared with 16C454, 16C554, 16C654 and 16C750 devices in the table below.

Feature	OX16C954	16C454	16C554	16C654	16C750
Serial channels	4	4	4	4	1
External 1x baud rate clock	yes	no	no	no	no
Max baud rate in normal mode	12.5 Mbps	115 kbps	115 kbps	1.5 Mbps	1 Mbps
Max baud rate in 1x clock mode	50 Mbps	n/a	n/a	n/a	n/a
FIFO depth	128	1	16	64	64
Sleep mode	yes	no	no	yes	yes
Auto Xon/Xoff flow	yes	no	no	yes	no
Auto CTS#/RTS# flow	yes	no	no	yes	yes
Auto DSR#/DTR# flow	yes	no	no	no	no
No. of Rx interrupt thresholds	128	1	4	4	4
No. of Tx interrupt thresholds	128	1	1	4	1
No. of flow control thresholds	128	n/a	n/a	4	n/a
Transmitter empty interrupt	yes	no	no	no	no
Readable status of flow control	yes	n/a	no	no	no
Readable FIFO levels	yes	n/a	no	no	no
Clock prescaler options	248	n/a	n/a	2	n/a
Rx/Tx disable	yes	no	no	no	no
Software reset	yes	no	no	no	no
Device ID	yes	no	no	no	no
RS485 buffer enable	yes	no	no	no	no
Infra-red (IrDA)	yes	no	no	yes	no

Improvements of the OX16C954 over previous generations of PC UART:

Deeper FIFOs:

OX16C954 offers 128-byte deep FIFOs for each of the transmitters and receivers.

Higher data rates:

Transmission and reception baud rates up to 12.5Mbps. A flexible clock prescaler offers division ratios of 1 to 31 7/8 in steps of 1/8 using a divide-by-“M N/8” circuitry. The flexible prescaler allows users to select from a wide variety of input clock frequencies as well as access to higher baud rates whilst maintaining compatibility with existing software drivers (see section 8.6).

External 1x clock:

Transmitters and receiver can accept an external 1x clock on the RI# input. In 1x mode, asynchronous data may be transmitted and received at speeds up to 50 Mbps (see section 8.6).

Automatic flow control:

Each channel can be programmed independently to automatically handle either or both in-band (software) flow control (transmitting and receiving Xon/Xoff characters) and out-of-band (hardware) flow control using the RTS#/CTS# or DSR#/DTR# modem control lines.

Special character detection:

The receiver can be programmed to generate an interrupt upon reception of a particular character value.

Power-down:

Each channel can be placed in a sleep state to conserve power.

Readable FIFO levels:

Fast driver applications are facilitated using readable FIFO levels.

Selectable trigger levels:

The receiver FIFO threshold can be arbitrarily programmed. The transmitter FIFO threshold and thresholds for automatic flow control can be programmed to operate at a variety of trigger levels.

Additional control:

The transmitter and receiver can be disabled independently.

Additional status:

Software drivers are able to read the status of in-band and out-of-band automatic flow controls, and distinguish between Xoff and special character received interrupts.

Software reset:

Each channel may be reset independently by software to recover from unforeseen or unusual error conditions.

Transmitter empty interrupt:

The transmitter can generate an interrupt when the FIFO and shift register are both empty.

RS485 buffer enable:

The function of the DTR# pin may be re-assigned to buffer enable signal for RS485 line driver in half-duplex mode (see ACR[4:3] in section 8.4.1).

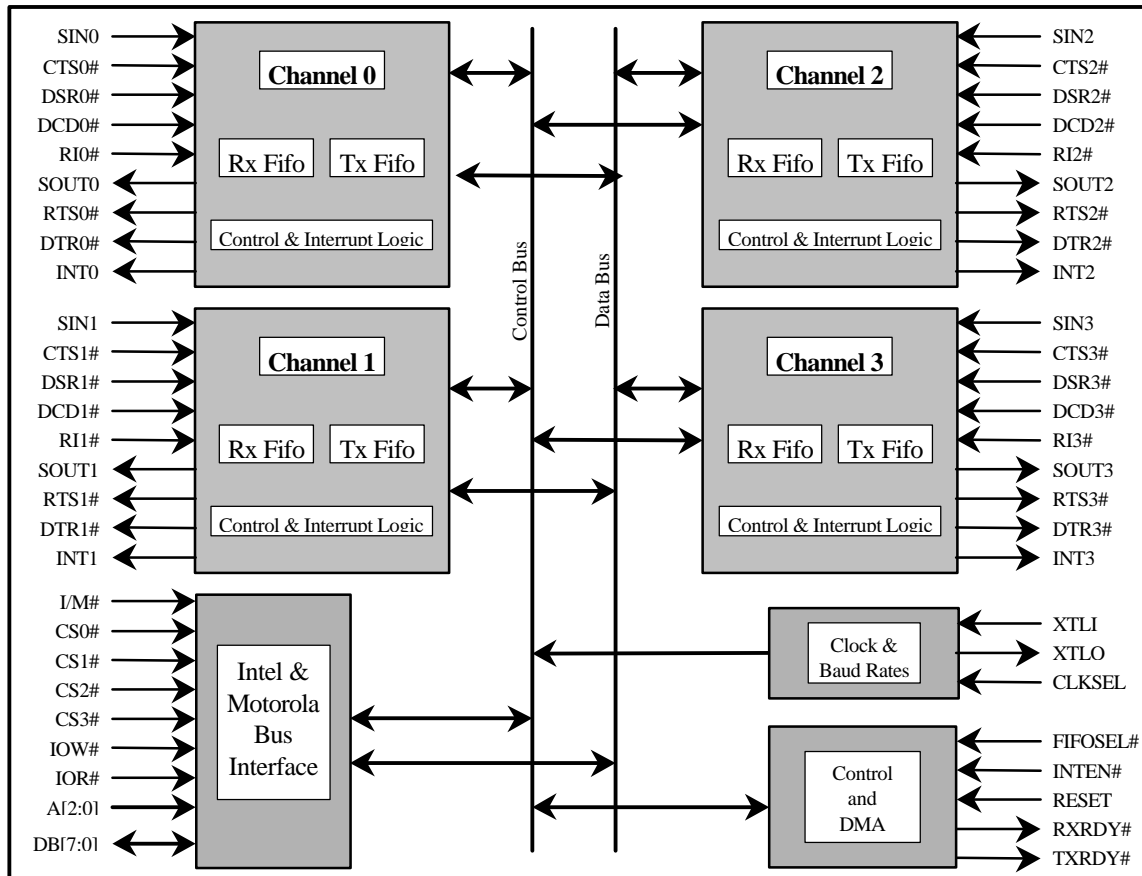
Device ID:

A four-byte device ID register is available to identify the OX16C954 device to software drivers.

Infra-red 'IrDA' interface:

Each channel contains an IrDA compliant modulator/demodulator.

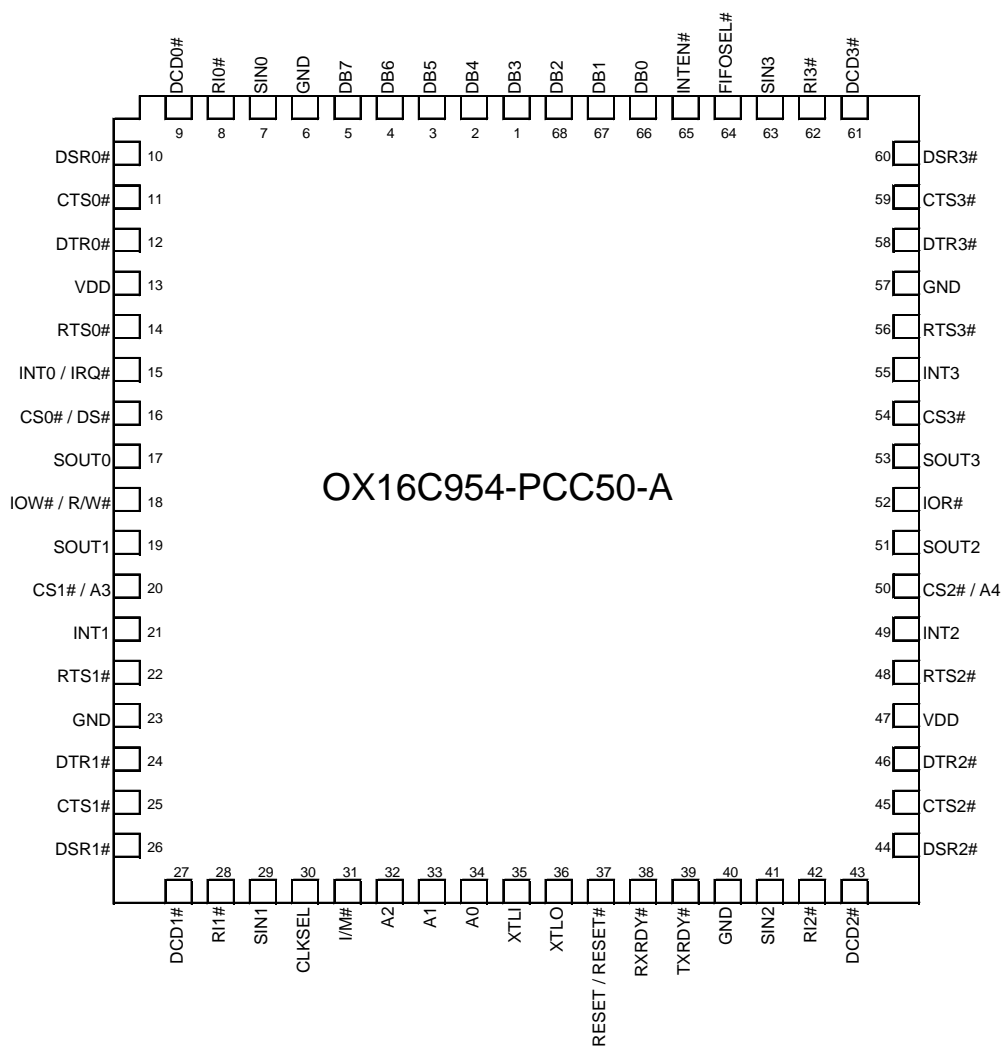
2 Block Diagram



OX16C954 Block Diagram

3 Pin Information

3.1 Pin Diagram



3.2 Pin Descriptions

Pin No.	Dir ¹	Name	Description
Clock			
36	O	XTLO	Crystal oscillator output.
35	I	XTLI	Crystal oscillator input or external clock pin. The maximum frequency is 50 MHz.
30	I	CLKSEL	This pin is provided to select an internal clock prescaler. In 16C554 devices this pin is a VDD. When CLKSEL pin is high the internal prescaler is bypassed (a 1.8432MHz clock is assumed). Connect this pin to GND to enable the internal clock prescaler (see section 8.6). The complement of this pin is loaded in bit 7 of the MCR register after a hardware reset.
Processor Interface Pins in Intel Mode (I/M# = '1')			
37	I	RESET	Active-high Hardware Reset. The configuration of OX16C954 after a hardware reset is described in section 8.3. This pin exhibits a small hysteresis to provide noise immunity.
16, 20, 50, 54	I	CS0#, CS1#, CS2#, CS3#	Active-low Chip-Select for channel 0 to channel 3 respectively in Intel bus mode respectively.
32 to 34	I	A[2:0]	Address lines to select channel registers.
5 to 168 to 66	I/O	DB[7:0]	Eight-bit 3-state data bus.
18	I	IOW#	Active-low write strobe in Intel bus mode.
	I	R/W#	Read-not-write signal in Motorola bus mode.
52	I	IOR#	Active-low read strobe in Intel bus mode. In Motorola bus mode this pin is unused and should be connected to VDD or GND.
Processor Interface Pins in Motorola Mode (I/M# = '0')			
37	I	RESET#	Active-low Hardware Reset. The configuration of OX16C954 after a hardware reset is described in section 8.3. This pin exhibits a small hysteresis to provide noise immunity.
16	I	DS#	Active-low Data-Strobe. In Motorola bus individual registers are accessed using DS#, R/W# and A[4:0].
52, 54	I	UNUSED	In Motorola bus mode these pin are unused and should be connected to VDD or GND.
50, 20	I	A4, A3	The A[4:3] combination selects individual channels as follows: 00 = Channel0 01 = Channel1 10 = Channel2 11 = Channel3
32 to 34	I	A[2:0]	Address lines to select channel registers.
5 to 168 to 66	I/O	DB[7:0]	Eight-bit 3-state data bus.
18	I	R/W#	Read-not-write signal. This signal should be high during read cycles and low during write cycles.
Serial Port Pins			
17, 19, 51, 53	O	SOUT0 - SOUT3	Transmitter serial data output.

Pin No.	Dir ¹	Name	Description
	O	IrDA_Out0 - IrDA_Out3	These pins are redefined to IrDA output when IrDA mode is enabled, i.e. MCR[6] is set in Enhanced mode.
14, 22, 48, 56	O	RTS0# - RTS3#	Active-low Request-To-Send output. Whenever the automated RTS# flow control is enabled, the RTS# pin is de-asserted and re-asserted if the receiver FIFO reaches or falls below a pair of programmed flow control thresholds, respectively.
12, 24, 46, 58	O	DTR0# - DTR3#	Active-low modem Data-Terminal-Ready output. Whenever the automated DTR# flow control is enabled, the DTR# pin is asserted and de-asserted if the receiver FIFO reaches or falls below a pair of programmed flow control thresholds, respectively.
		485_En0 - 485_En3	In RS485 half-duplex mode, the DTR# pin may be programmed to reflect the state of the transmitter empty bit (or it's inverse) to automatically control the direction of the RS485 transceiver buffer.
7, 29, 41, 63	I	SIN0 - SIN3	Receiver serial data input.
		IrDA_In0 - IrDA_In3	These pins are redefined to IrDA input when IrDA mode is enabled, i.e. MCR[6] is set in Enhanced mode.
11, 25, 45, 59	I	CTS0# - CTS3#	Active-low Clear-To-Send input. Whenever the automated CTS# flow control is enabled and the CTS# pin is de-asserted, the transmitter will complete the current character and enter the idle mode until the CTS# pin is re-asserted. However, flow control characters are transmitted regardless of the state of the CTS# pin.
10, 26, 44, 60	I	DSR0# - DSR3#	Active-low modem Data-Set-Ready input. Whenever the automated CTS# flow control is enabled and the DSR# pin is de-asserted, the transmitter will complete the current character and enter the idle mode until the DSR# pin is re-asserted. However, flow control characters are transmitted regardless of the state of the DSR# pin.
9, 27, 43, 61	I	DCD0# - DCD3#	Active-low modem Data-Carrier-Detect input.
8, 28, 42, 62	I	RI0# - RI3#	Active-low modem Ring-Indicator input.
		Ext_CK0 - Ext_CK3	When external 1x baud rate clock is selected (i.e. CPR = 0x00), the transmitter and receiver clock for a particular channel is supplied by its RI# pin.
Interrupt & DMA Pins			
39	O	TXRDY#	Signal for DMA transfer of transmitter data. There are two modes of DMA signalling described in section 8.7. This pin is the wire "OR-ed" function of TXRDY# signal from all channels.
38	O	RXRDY#	Signal for DMA transfer of received data. There are two modes of DMA signalling described in section 8.7. This pin is the wire "OR-ed" function of RXRDY# signal from all channels.
15	O	INT0	Channel 0 interrupt in Intel bus mode. Each serial channel has a 3-state interrupt output (enabled by MCR[3] and INTEN# pin) which goes active (high) when an interrupt condition occurs. The interrupt is disabled after a hardware reset.
	OD	IRQ#	Device interrupt in Motorola mode. This pin goes active (low) when the interrupt signal from any of the channels is asserted, otherwise it is in high-impedance state.
21, 49, 55	O	INT1, INT2, INT3	Interrupt pins for channels 1, 2 and 3.

Pin No.	Dir ¹	Name	Description
Miscellaneous Pins			
65	ID	INTEN#	Active-low Interrupt enable. When this pin is left open or connected to GND, the three-state interrupts are enabled according to the setting of MCR[3]. If this pin is high interrupts are enabled regardless of the state of MCR[3]. This pin is ignored in Motorola bus mode.
31	IU	I/M#	Intel or Motorola bus interface select. In 16C554 this pin is unconnected. When this pin is tied high or left open, the Intel bus interface is selected. When this pin is tied low, the Motorola bus interface is selected where RESET, IOW#, CS0#, CS1# are CS2# are re-assigned and CS3#, IOR# and INTEN# are unused. All interrupt outputs (INT0 to INT3) are wired "OR-ed" to IRQ#. This pin contains an internal pull-up.
64	I	FIFOSEL#	For backward compatibility with 16C550, 16C650 and 16C750 devices the FIFO depth is 16 when FIFOSEL# is high and 128 when FIFOSEL# is low. The unlatched state of this pin is readable by software. The FIFO size may be set to 128 by writing a 1 in FCR[5] when LCR[7] is set or by putting the device into Enhanced mode, thus overriding the state of the FIFOSEL# pin. Pin 64 is a VDD in 16C554 and 16C654 devices.
Power and Ground			
6, 23, 40, 57		GND	Ground (0 Volts). All GND pins should be tied to ground.
13, 47		VDD	Power supply. The VDD pins should be tied to 5 Volts.

Note 1 : Direction key:

I *Input*
IU *Input with pull-up*
ID *Input with pull-down*
O *Output*
I/O *Bi-directional*
OD *Open drain*

Note: All unused input pins should be tied to VDD or GND and must not be left floating. For high speed operation (XTAL > 10MHz), the card designers are recommended to follow the guidelines for high-speed digital design such as maintaining PCB tracks as short as possible, using a multi-layer PCB with separate power and ground planes, and using good-quality de-coupling capacitors.

4 Operating Conditions

4.1 Absolute Maximum Rating

Symbol	Parameter	Min.	Max.	Units
V _{DD}	DC supply voltage	-0.3	7.0	V
V _{IN}	DC input voltage	-0.3	V _{DD} + 0.3	V
I _{IN}	DC input current		+/- 10	mA
T _{STG}	Storage temperature	-40	125	°C

4.2 Recommended Operating Condition

Symbol	Parameter	Commercial		Industrial		Units
		Min	Max	Min	Max	
V _{DD}	DC supply voltage	4.75	5.25	4.5	5.5	V
T _O	Operating Temperature range	0	70	-40	85	°C

5 DC Electrical Characteristics

Symbol	Parameter	Condition	Min.	Max.	Units
V _{DD}	Supply voltage	Commercial Industrial	4.75 4.5	5.25 5.5	V
V _{IH}	Input high voltage	TTL Interface ^{Note1} TTL Schmitt trigger	2.0 2.4		V
V _{IL}	Input low voltage	TTL Interface ^{Note1} TTL Schmitt trigger		0.8 0.6	V
C _{IL}	Capacitance of input buffers			5.0	pF
C _{OL}	Capacitance of output buffers			10.0	pF
I _{IH}	Input high leakage current	V _{in} = V _{DD}	-10	10	μA
I _{IL}	Input low leakage current	V _{in} = V _{SS}	-10	10	μA
V _{OH}	Output high voltage	I _{OH} = 1 μA	V _{DD} - 0.05		V
V _{OH}	Output high voltage	I _{OH} = 4 mA ^{Note2}	2.4		V
V _{OL}	Output low voltage	I _{OL} = 1 μA		0.05	V
V _{OL}	Output low voltage	I _{OL} = 4 mA ^{Note2}		0.4	V
I _{OZ}	3-state output leakage current		-10	10	μA
I _{ST}	Static current	V _{in} = V _{DD} or V _{SS}	90	150	μA
I _{CC}	Operating supply current in normal mode ^{Note3}	f _{CK} = 1.8432 MHz	1.0	1.5	mA
		f _{CK} = 7.372 MHz	3.3	5.0	
		f _{CK} = 50.00 MHz	10.0	20.0	
	Operating supply current in sleep mode ^{Note3}	f _{CK} = 1.8432 MHz	0.9	1.2	
		f _{CK} = 7.372 MHz	2.8	3.5	
		f _{CK} = 50.00 MHz	11.0	14.0	

Note 1: All input buffers are TTL with the exception of RESET which is a Schmitt trigger buffer.

Note 2: I_{OH} and I_{OL} are 12 mA for DB[7:0] and 4 mA for all other outputs.

Note 3: For further details on operating current please refer to the 'OX16C95x Test Document'.

6 AC Electrical Characteristics

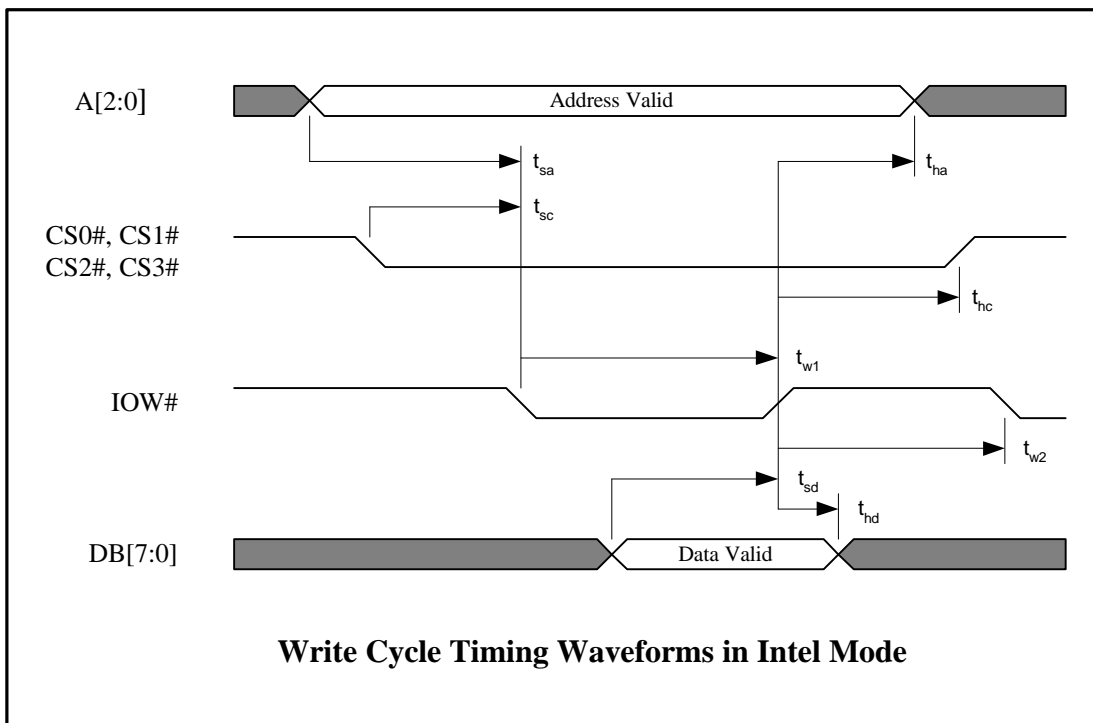
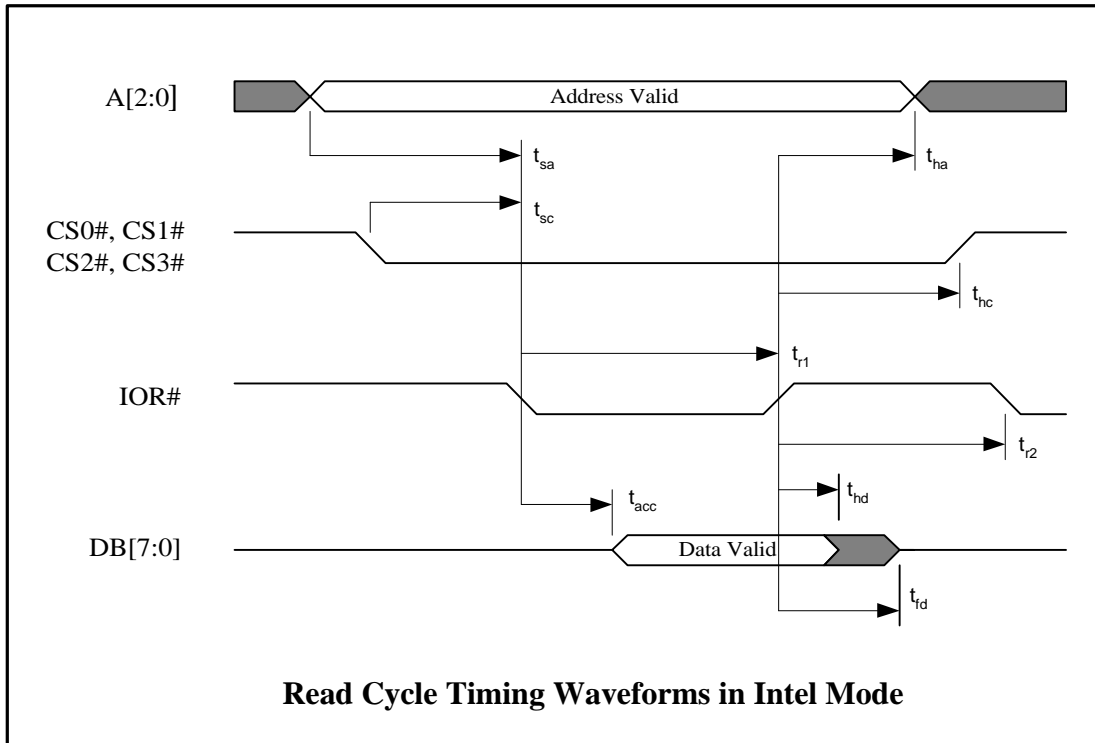
Data bus timing for Intel mode:

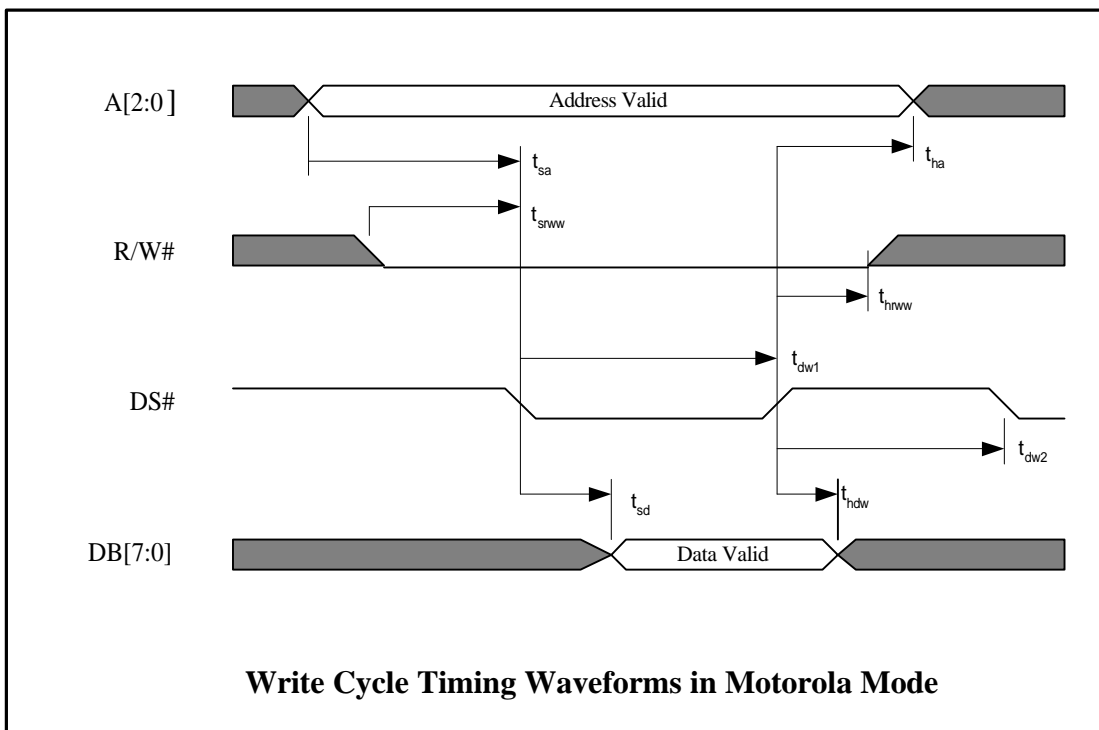
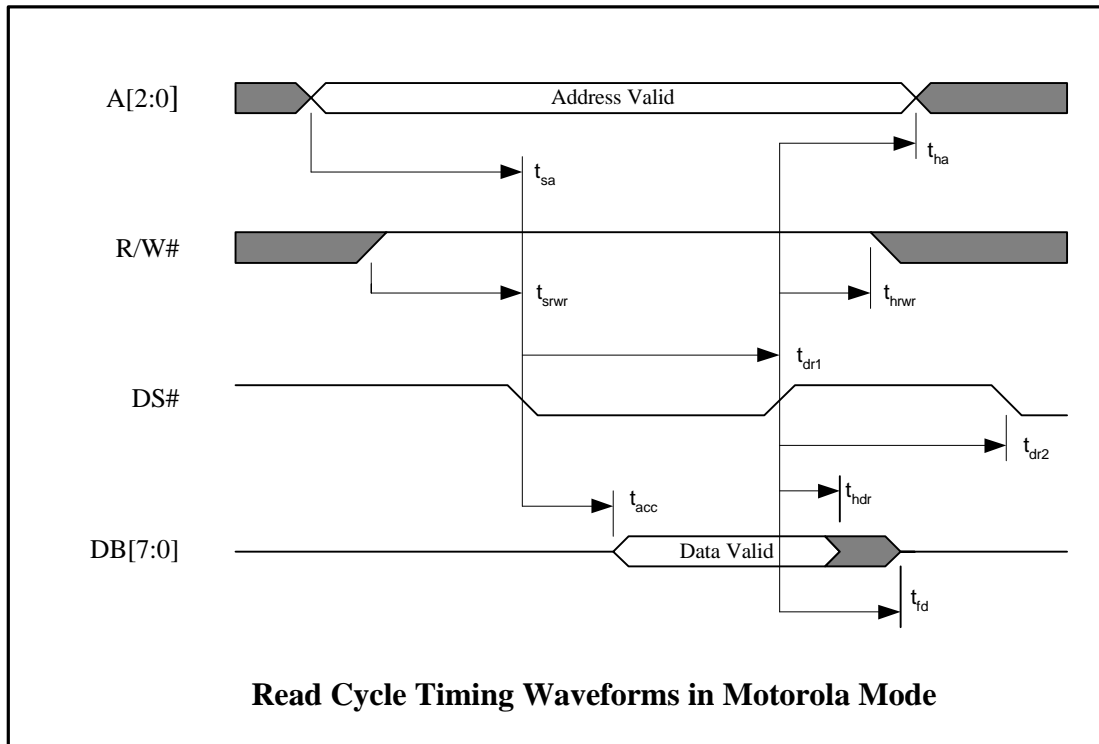
Symbol	Parameter	Commercial		Industrial		Units
		Min	max	Min	Max	
t _{sa}	Address set-up time to IOR# or IOW# falling	0		0		ns
t _{ha}	Address hold time after IOR# or IOW# rising	0		0		ns
t _{sc}	Chip-select set-up time to IOR# or IOW# falling	0		0		ns
t _{hc}	Chip-select hold time after IOR# or IOW# rising	0		0		ns
t _{r1}	Pulse duration of IOR#	47		50		ns
t _{r2}	Delay time between IOR# rising and IOR# or IOW# falling	25		27		
t _{acc}	Data valid after IOR# falling (access time)		46		49	ns
t _{hd}	Data valid (hold) after IOR# rising	2		2		
t _{fd}	Data bus floating after IOR# rising		5		6	ns
t _{w1}	Pulse duration of IOW#	29		31		ns
t _{w2}	Delay time between IOW# rising and IOR# or IOW# falling	25		26		ns
t _{sd}	Data set-up time to IOW# rising	3		3		ns
t _{hd}	Data hold time after IOW# rising	4		4		ns

Data bus timing for Motorola mode:

Symbol	Parameter	Commercial		Industrial		Units
		Min	max	Min	Max	
t _{sa}	Address set-up time to DS# falling	6		6		ns
t _{ha}	Address hold time after DS# rising	0		0		ns
t _{srwr}	R/W# set-up time to DS# falling (read cycle)	4		4		ns
t _{hrwr}	R/W# hold time after DS# rising (read cycle)	2		2		ns
t _{dr1}	Pulse duration of DS# (read cycle)	48		50		ns
t _{dr2}	Delay to start of next read/write cycle (read cycle)	27		28		ns
t _{acc}	Read data valid after DS# falling (access time)	47		49		ns
t _{dhr}	Read data hold data after DS# rising	2		2		ns
t _{fd}	Data bus float after DS# rising		6		6	ns
t _{srww}	R/W# set-up time to DS# falling (write cycle)	4		4		ns
t _{hrww}	R/W# hold time after DS# rising (write cycle)	2		2		ns
t _{dw1}	Pulse duration of DS# (write cycle)	24		26		ns
t _{dw2}	Delay to start of next read/write cycle (write cycle)	26		28		ns
t _{sd}	Write data set-up time to DS# rising	3		3		ns
t _{hdw}	Write data valid after DS# rising	4		4		ns

7 Timing Waveforms





8 Serial Ports

The OX16C954 consists of four independent UARTs, which are referred to as channel 0, channel 1, channel 2 and channel 3. The following is a description of a single channel.

8.1 General Mode Selection

The OX16C954 device is backward-compatible with the 16C450, 16C550, 16C654 and 16C750 UARTs. The operation of the OX16C954 depends on a number of mode settings. These modes are referred to throughout this data sheet. The FIFO depth and compatibility modes are tabulated below:

FCR[0]	Enhanced mode (EFR[4]=1)	FCR[5] (guarded with LCR[7] = 1)	FIFOSEL#	FIFO size	Value in ASR[6]	Compatibility mode
0	X	X	X	1	X	450
1	0	0	1	16	0	550
1	1	X	X	128	1	650
1	0	1	1	128	1	750
1	0	X	0	128	1	Extended 550

8.1.1 450 Mode

After a hardware reset bit 0 of FIFO Control Register ('FCR') is cleared, hence OX16C954 is compatible with the 16C450. The transmitter and receiver FIFOs (referred to as the 'Transmit Holding Register' and 'Receiver Holding Register' respectively) have a depth of one. This is referred to as 'Byte mode'. When FCR[0] is cleared, all other mode selection parameters are ignored.

8.1.2 550 Mode

Connect pin 64 (FIFOSEL#) to VDD. Writing a 1 to FCR[0] would then increase the FIFO size to 16, thus providing compatibility with 16C550 devices. Since pin 64 is VDD in 16C554 devices, replacing a 16C554 with OX16C954 would result in a 550 compatible device with 16 deep FIFOs.

8.1.3 Extended 550 Mode

Connect pin 64 (FIFOSEL#) to ground. Writing a 1 to FCR[0] would then increase the FIFO size to 128, thus providing a 550 device with 128 deep FIFOs.

Note that for full compatibility with 550 or extended 550 modes, the software driver should not enable the Enhanced mode or set FCR[5] while LCR[7] is '1'. The UART is in Enhanced mode when bit 4 of the Enhanced Features Register ('EFR') is set. Existing 550 drivers comply with these requirements.

8.1.4 750 Mode

The 16C654 and 16C750 devices are not compatible with one another. For compatibility with 16C750 software drivers, pin 64 (FIFOSEL#) should be connected to VDD. Writing a 1 to FCR[0] would then increase the FIFO size to 16. In a similar fashion to the 16C750, the FIFO size can be further increased to 128 by writing a 1 to FCR[5]. Note that in order to differentiate between 650 and 750 software drivers, when FIFOSEL# is high access to FCR[5] is protected by LCR[7]. In other words, to set FCR[5], the driver should first write a '1' LCR[7] to remove the guard temporarily. Once FCR[5] is set, the software driver should clear LCR[7] for normal operation.

The 16C750 additional features over the 16C550 are available as long as the UART is not put into Enhanced mode i.e. EFR[4] = '0'. These features are:

1. Deeper FIFOs
2. Automatic RTS/CTS out-of-band flow control
3. Sleep mode

8.1.5 650 Mode

The OX16C954 is compatible with the 16C654 when EFR[4] is set, i.e. the device is in Enhanced mode. As 650 (or 654) software drivers usually put the device in Enhanced mode, running 650 drivers on the OX16C954 device would result in 650 compatibility with 128 deep FIFOs, as long as FCR[0] is set. This is regardless of the state of the FIFOSEL# pin.

The 16C654 has the same enhancements over the 16C550 as the 16C750 has, but these are enabled using different registers. There are additional enhancements in the 16C650 over the 16C750. These enhancements are:

1. Automatic in-band flow control
2. Special character detection
3. Infra-red "IrDA-format" transmit and receive mode
4. Transmit trigger levels
5. Optional clock prescaler

8.1.6 950 Mode

The additional features offered in OX16C954 (950 mode) generally only apply when the UART is in Enhanced mode (EFR[4]='1'). Provided that FCR[0] is set, in Enhanced mode the FIFO size is 128 regardless of the state of the FIFOSEL# pin. The 950 specific features are enabled with various settings in the Additional Control Register 'ACR' (see section 8.4). In addition to larger FIFOs and higher baud rates, the enhancements of the 16C954 over the 16C650 are:

1. Selectable arbitrary trigger levels for the receiver and transmitter FIFO interrupts.
2. Improved automatic flow control using selectable arbitrary thresholds.
3. DSR#/DTR# automatic flow control.
4. Disabling of transmitter and receiver.
5. Software reset of transmitter and receiver.
6. Readable FIFO levels.
7. RS485 buffer enable signal
8. Four-byte device ID register.
9. Readable status of automatic in-band and out-of-band flow control.
10. External 1x clock (see section 8.6).
11. Flexible "M N/8" clock prescaler (see section 8.6).
12. Programmable sample clock to allow data rates up to 12.5 Mbps (see section 8.6).

The 950 trigger levels are enabled when ACR[5] is set where bits 4 to 7 of FCR are ignored. Then arbitrary trigger levels can be defined in RTL, TTL, FCL and FCH registers (see section 8.4). The Additional Status Register ('ASR') offers flow control status for the local and remote transmitters. FIFO levels are readable using 'RFL' and 'TFL' registers.

Each channel has a flexible prescaler capable of dividing the system clock by any value between 1 and "31 7/8" (i.e. 31.875) in steps of 1/8 (= 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 an input clock frequency to synthesise arbitrary baud rates. The default division value is 4 to provide backward compatibility with 16C650 devices.

The user may select an external 1x (or bit rate) clock by writing the reserved value 0x00 to the CPR register. In this mode the bit rate clock is supplied externally using the RI# input pin.

It is also possible to define the over-sampling rate used by the transmitter and receiver clocks. The 16C450/16C550 and compatible devices employ 16 times over-sampling, where there are 16 clock cycles per bit. However, OX16C954 can employ any over-sampling rate from 4 to 16 by programming the TCR register. This allows the data rates to be increased to 12.5 Mbps using a 50 MHz clock, or 460.8 Kbps using a 1.8432MHz clock. The default value after a reset for this register is 0x00 which corresponds to a 16 cycle sampling clock. Writing 0x01, 0x02 or 0x03 will also result in a 16 cycle sampling clock. To programme the value to any value from 4 to 15 it is necessary to write this value into the TCR i.e. to set the device to a 13 cycle sampling clock it would be necessary to write 0x0D to TCR. For further information see sections 8.4.8 and 8.6.

8.2 Register Memory Map

The three address lines select the various registers in each UART. Since there are more than 8 registers, selection of the registers is also dependent on the state of the Line Control Register 'LCR' and Additional Control Register 'ACR':

1. LCR[7]=1 enables the divider latch registers.
2. 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.
3. ACR[7]=1 enables the 'additional' features registers. Unlike the other selectors, this bit can normally be left set as this will only disable the reading of registers that rarely need to be read.
4. ACR[6]=1 enables reading the ICR registers as described below.

Register Name		A[2:0]	Access	Special Access Conditions	Reset Value
Transmit Holding Register	THR	000	W	LCR[7] = 0	n/a
Receiver Holding Register	RHR	000	R	LCR[7] = 0	0xXX
Divisor Latch LSB	DLL	000	R/W	LCR[7] = 1	0x01
Interrupt Enable Register	IER	001	R/W	LCR[7] = 0 & ACR[7] = 0	0x00
Additional Status Register	ASR	001	R/W	LCR[7] = 0 & ACR[7] = 1	0xX0
Divisor Latch MSB	DLM	001	R/W	LCR[7] = 1	0x00
FIFO Control Register	FCR	010	W	LCR ≠ 0xBF	0x00
Interrupt Status Register	ISR	010	R	LCR ≠ 0xBF	0x01
Enhanced Features Register	EFR	010	R/W	LCR = 0xBF	0x00
Line Control Register	LCR	011	R/W	write : none read : ACR[7] = 0	0x00
Receiver FIFO Level	RFL	011	R	ACR[7] = 1	0x00
Modem Control Register	MCR	100	R/W	write : LCR ≠ 0xBF read: LCR ≠ 0xBF & ACR[7] = 0	X0000000
Transmitter FIFO Level	TFL	100	R	LCR ≠ 0xBF & ACR[7] = 1	0x00
XON1 Definition Register	XON1	100	R/W	LCR = 0xBF	0x00
Indexed Control Register	ICR	101	R/W	write : LCR ≠ 0xBF read : LCR ≠ 0xBF & ACR[6] = 1	0x00
Line Status Register	LSR	101	R	LCR ≠ 0xBF & ACR[6] = 0	0x60
XON2 Definition Register	XON2	101	R/W	LCR = 0xBF	0x00
Modem Status Register	MSR	110	R	LCR ≠ 0xBF	0xX0
XOFF1 Definition Register	XOFF1	110	R/W	LCR = 0xBF	0x00
Scratch Pad Register	SPR	111	R/W	LCR ≠ 0xBF	0x00
XOFF2 Definition Register	XOFF2	111	R/W	LCR = 0xBF	0x00

8.2.1 UART Register Description Table

These registers are selected when CS0# (CS1#, CS2# or CS3#) are low.

General Registers:

Register Name	Address	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Standard Registers (550 Compatible Registers)										
THR ¹	000	W	Data to be transmitted; Bits [7:0]							
RHR ¹	000	R	Data received; Bits [7:0]							
IER ^{1,2} Enhanced mode	001	R/W	CTS interrupt mask	RTS interrupt mask	Special Char. Detect	Sleep mode	Modem interrupt mask	Rx Stat interrupt mask	THRE interrupt mask	RxRDY interrupt mask
			0	0	alternate sleep mode	Sleep mode	Modem interrupt mask	Rx Stat interrupt mask	THRE interrupt mask	RxRDY interrupt mask
FCR ^{3,15} 650 mode	010	W	RHR Trigger Level		THR Trigger Level		DMA mode	Flush THR	Flush RHR	Enable FIFO
			RHR Trigger Level		FIFO Size	Unused	DMA mode	Flush THR	Flush RHR	Enable FIFO
			Unused			DMA mode	Flush THR	Flush RHR	Enable FIFO	
ISR ³	010	R	FIFOs enabled		Interrupt priority bits [5:4] (Enhanced mode)		Interrupt priority bits [3:1] (All modes)			interrupt pending
LCR ⁴	011	R/W	divisor latch access (DLAB)	Tx break	Force parity	Odd / even parity	Parity enable	Number of stop bits	data length bits [1:0]	
MCR ^{3,4,5} Non-Enhanced mode	100	R/W	0	0	CTS & RTS Flow Control	Loop Back	OUT2 (Int En)	OUT1	RTS	DTR
			Baud prescale	IrDA mode	XON-Any	Loop Back	OUT2 (Int En)	OUT1	RTS	DTR
LSR ^{3,6}	101	R	Data Error	Tx Empty	THR Empty	Rx Break	Framing Error	Parity Error	Overrun Error	RxRDY
MSR ³	110	R	DCD	RI	DSR	CTS	Delta DCD	Trailing RI edge	Delta DSR	Delta CTS
SPR ³ Normal	111	R/W	Temporary data storage register bits [7:0] or Indexed control register offset value bits [7:0]							
Additional Standard Registers – These registers require LCR[7] (DLAB) to be set to 1.										
DLL	000	R/W	Divisor latch bits [7:0] (Least significant byte)							
DLM	001	R/W	Divisor latch bits [15:8] (Most significant byte)							

650 Compatible Registers - 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 Bits [3:0]
XON1	100	R/W	XON Character 1: Bits [7:0]				
XON2	101	R/W	XON Character 2: Bits [7:0]				
XOFF1	110	R/W	XOFF Character 1: Bits [7:0]				
XOFF2	111	R/W	XOFF Character 2: Bits [7:0]				

Specific Registers:

Register Name	Address	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
954 Specific Registers											
ASR ^{1,7,8}	001	R/W	Tx Idle [R]	FIFO size [R]	FIFO-SEL [R]	special char detect [R]	DTR [R]	RTS [R]	remote Tx disabled [R/W]	Tx disabled [R/W]	
RFL ⁹	011	R	Number of characters in the receiver FIFO bits [7:0]								
TFL ^{3,9}	100	R	Number of characters in the transmitter FIFO bits [7:0]								
ICR ^{3,10,14}	101	R/W	Data read/written depends on the value written to the SPR prior to the access of this register (see table below)								
Register Name	SPR Offset ¹¹	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
Indexed Control Register Set ¹²											
ACR	0x00	R/W	Additional status enable	ICR read enable	952 mode trigger levels enable	DTR# defn and control bit 1	DTR# defn and control bit 0	auto DSR Flow control enable	Tx disable	Rx disable	
CPR	0x01	R/W	5 bit "integer" part of the clock prescaler					3 bit "fractional" part of the clock prescaler			
TCR	0x02	R/W	0	0	0	0	4 bit N-times clock selection bits [3:0]				
TTL	0x04	R/W	0	7 bit transmitter interrupt trigger level bits [6:0]							
RTL	0x05	R/W	0	7 bit receiver interrupt trigger level [6:0]							
FCL	0x06	R/W	0	7 bit flow control lower trigger level [6:0]							
FCH	0x07	R/W	0	7 bit flow control higher trigger level [6:0]							
ID1	0x08	R	0	0	0	1	0	1	1	0	
ID2	0x09	R	1	1	0	0	1	0	0	1	
ID3	0x0A	R	0	1	0	1	0	1	0	0	
REV	0x0B	R	0	0	0	0	0	0	0	0	
CSR	0x0C	W	Writing 0x00 to this register will reset the applicable channel								
CTR ¹³	0xFF	R/W	unused	unused	unused	unused	divert baud out	Transfer FIFO loop	enable FIFO loop	enable baud test	

Note 1 : Requires LCR[7] to be set to '0'.

Note 2 : Requires ACR[7] to be set to '0'.

Note 3 : Requires that LCR is not set to 0xBF.

Note 4 : To read this register ACR[7] to be set to '0'.

Note 5 : MCR[7] is reset to the complement of the CLKSEL input pin.

Note 6 : To read this register ACR[6] to be set to '0'.

Note 7 : Requires ACR[7] to be set to '1'.

Note 8 : Only bit 0 and bit 1 of ASR are writable.

Note 9 : To read this register ACR[7] to be set to '1'.

Note 10 : To read this register ACR[6] to be set to '1'.

Note 11 : The SPR offset column indicates the value which must be written into the SPR prior to reading/writing from any of the indexed control registers. Offsets which are not listed in the table are reserved for future use and should not be used.

Note 12 : To read or write to any of the Indexed Control Registers use the following procedure.

Writing to ICR registers:

1. Ensure that the last value written to LCR was not 0xBF (reserved value). If it is difficult to keep a track of the last value written to LCR, then read the contents of LCR (address 011b) and write it back into LCR.
2. Write the desired offset to SPR (address 111b).
3. Write the desired value to ICR (address 101b).

Reading from ICR registers:

1. Ensure that the last value written to LCR was not 0xBF (see point 1 above).
2. Write 0x00 offset to SPR to select ACR.
3. Set bit 6 of ACR (ICR read enable) by writing x1xxxxxb to address 101b. Ensure that other bits in ACR are not changed. Software drivers should keep a copy of the contents of the ACR elsewhere since reading ICR involves overwriting ACR!
4. Write the desired offset to SPR (address 111b).
5. Read the desired value to ICR (address 101b).
6. Write 0x00 offset to SPR to select ACR.
7. Write x0xxxxxb to ICR (address 101b) to re-enable reading LSR.

This example will write 0xAB to FCL.

- i. Ensure that the last value written to LCR is not 0xBF
- ii. Write 0x06 to SPR (address 111b)
- iii. Write 0xAB to ICR (address 101b)

This example will read the value in CPR.

- i. Ensure that the last value written to LCR is not 0xBF
- ii. Write 0x00 to SPR (address 111b)
- iii. Write x1xxxxxb to ICR (address 101b) to set bit 6 of ACR
- iv. Write 0x01 to SPR (address 111b)
- v. Read from ICR (address 101b)
- vi. Write 0x00 to SPR (address 111b)
- vii. Write x0xxxxxb to ICR (address 101b) to re-enable reading LSR

Note 13 : This register exists for test purposes only. Writing to this register will produce unpredictable results!

Note 14 : The Indexed Control Register ('ICR') facilitates access to a number of additional control registers using an offset value defined in the Scratch Pad Register ('SPR'). These registers provide enhanced selection of interrupt levels and flow control. This revision of the device does not use all of the maximum possible 256 available offset values. The remaining offset values are reserved for future revisions of this device and should not be used. The Indexed Control Registers are described in the table above. To write to and read from registers in the ICR group follow the steps in Note 12 (see above). The user must not write any values to the CTR register as it is reserved for testing.

Note 15 : The FCR register is unused in the 450-compatible mode. In 550 and extended 550-compatible modes, the transmitter trigger level is set to 1 and FCR[5:4] is ignored. For 450, 550, Extended-550, 650 and 750 modes see section 8.1. When ACR[5] is set, the 950-mode trigger levels are enabled and FCR[7:4] is ignored (see TTL, RTL, FCH, FCL in section 8.4).

8.3 Reset Configuration

8.3.1 Hardware Reset

After a hardware reset, all writable registers are reset to 0x00, with the following exceptions:

1. DLL which is reset to 0x01.
2. MCR[7] is reset to the complement of the CLKSEL input pin value (see section 8.6).
3. CPR is reset to 0x20.

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

RHR[7:0]: Indeterminate
 RFL[6:0]: '0000000'
 TFL[6:0]: '0000000'
 LSR[7:0]: 0x60 signifying that both the transmitter and the transmitter FIFO are empty
 MSR[3:0]: '0000'
 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'

The reset state of signals for each serial channel's outputs are tabulated below:

Signal ¹	Reset state
SOUT _n	Inactive High
RTSn#	Inactive High
DTRn#	Inactive High
INT _n / IRQ#	High-impedance (inactive)
RXRDY#	Inactive High
TXRDY#	Active low (signifying that THR is able to receive data).

Note1: n denotes channel 0, 1, 2 or 3.

8.3.2 Software Reset

An additional enhancement feature available in the OX16C954 device is software resetting of the serial channel. The software reset is available using the CSR register.

Channel reset:

The channel reset command has the same effect as a hardware reset except it only affects the selected serial channel. To reset a given serial channel write 0x00 to the Channel Software Reset register ('CSR').

8.4 Specific Features of the OX16C954

8.4.1 Additional Control Register 'ACR' (ICR offset 0x00)

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 synchronisation 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 will not be detected.

Changes to this bit will only be recognised 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 will only be recognised following the completion of any data transmission pending.

ACR[2]: Enable automatic DSR flow control

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

This bit provides another automatic out-of-band flow control facility using DSR# line.

ACR[4:3]: DTR# line configuration

- logic [00] ⇔ DTR# is compatible with 16C450, 16C550, 16C650 and 16C750 (i.e. normal).
 logic [01] ⇔ DTR# pin is used for out-of-band flow control. It will be forced inactive high if the Receiver FIFO Level ('RFL') reaches the upper flow control threshold. DTR# line will be re-activated (=0) when the RFL drops below the lower threshold (see FCL & FCH below).
 logic [10] ⇔ DTR# pin is configured to drive the active-low enable pin of an external RS485 buffer. In this configuration the DTR# pin will be forced low whenever the transmitter is not empty (LSR[6]=0), otherwise DTR# pin is high.
 logic [11] ⇔ DTR# pin is configured to drive the active-high enable pin of an external RS485 buffer. In this configuration, the DTR# pin will be forced high whenever the transmitter is not empty (LSR[6]=0), otherwise DTR# pin is low.

If the user sets ACR[4], then the DTR# 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 DTR# line to directly control the enable signal of external 3-state line driver buffers. When the transmitter is empty the DTR# 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 FCR register and are compatible with 16C650/16C750 modes.
 logic 1 ⇔ 16C954 specific enhanced interrupt and flow control trigger levels defined by TCR, RTL, TTL and FCL are enabled.

ACR[6]: ICR read enable

- logic 0 ⇔ The Line Status Register is readable.
 logic 1 ⇔ The Indexed Control Registers are readable.

Bit 6 of ACR will substitute the indexed control registers for LSR in the memory map (see section 8.2). During normal operation this bit should be cleared.

ACR[7]: Additional status enable

- logic 0 ⇔ The ASR, TFL and RFL registers are not enabled.
 logic 1 ⇔ The ASR, TFL and RFL registers are enabled.

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

8.4.2 Additional Status Register ('ASR')**ASR[0]: Transmitter disabled by in-band flow control**

- logic 0 ⇔ The transmitter is not disabled by in-band flow control.
 logic 1 ⇔ The receiver has received an XOFF which has disabled the transmitter.

This bit is cleared after a hardware reset or channel software reset. The software driver may write a 0 to this bit to re-enable the transmitter if it was disabled by in-band flow control. Writing a 1 to this bit is illegal.

ASR[1]: Remote transmitter disabled by in-band flow control

- 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, which has not yet been cancelled by a subsequent XON.

This bit is cleared after a hardware reset or channel software reset. The software driver may write a 0 to this bit to re-enable the remote transmitter where an XON is transmitted. Writing a 1 to this bit is illegal.

ASR[2]: RTS

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

ASR[3]: DTR

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

ASR[4]: Special character detected

- logic 0 ⇔ No special character is 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]: FIFOSEL

This bit reflects the complement of the FIFOSEL# pin.

ASR[6]: FIFO size

- logic 0 ⇔ FIFOs are 16 deep if FCR[0] = 1.
 logic 1 ⇔ FIFOs are 128 deep if FCR[0] = 1.

Note: If FCR[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 internal transmitter idle bit.

8.4.3 Transmitter and Receiver FIFO levels 'TFL' and 'RFL'

The number of characters stored in the THR and RHR can be determined by reading the TFL and RFL registers respectively. As the UART clock is asynchronous with respect to the processor, it is possible for the levels to change during a read of these FIFO levels. It is therefore recommended that the levels are read twice and compared to check that the values obtained are valid. The values should be interpreted as follows:

1. The number of characters in the THR is no greater than the value read back from TFL.
2. The number of characters in the RHR is no less than the value read back from RFL.

8.4.4 Receiver Interrupt Trigger Level Register 'RTL' (ICR offset 0x05)

Whenever 950 trigger levels are enabled (ACR[5]=1), bits 6 and 7 of FCR 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 16C650 and 16C750 devices. It enables the system designer to optimise the interrupt performance hence minimising the interrupt overhead.

In 16C950 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 equals the value stored in this register.

8.4.5 Transmitter Interrupt Trigger Level Register 'TTL' (ICR offset 0x04)

Whenever 950 trigger levels are enabled (ACR[5]=1), bits 4 and 5 of FCR are ignored and an alternative arbitrary transmitter trigger level can be defined in the TTL register. This 7-bit value provides a fully programmable transmitter interrupt trigger facility. Note that to program an arbitrary transmitter trigger level, the UART should be configured in FIFO mode 1 (FCR[0]=1 and FCR[3]=1), otherwise the trigger level is set to one. 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.

8.4.6 Flow Control Trigger Levels 'FCL' and 'FCH' (ICR offsets 0x06 and 0x07)

Enhanced software flow control using XON/XOFF and hardware flow control using RTS#/CTS# and DTR#/DSR# 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 optimising 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. One or two XOFF characters may be transmitted when the receiver FIFO exceeds the upper trigger level defined by FCR[7:6] as described in section 8.6. The flow control is enabled and the appropriate mode is selected using EFR[3:0].

In 950 mode, the flow control thresholds defined by FCR[7:6] are ignored. In this mode while the UART is in Enhanced mode (EFR[4]=1), arbitrary threshold levels are programmed using FCL and FCH. When flow control is enabled by EFR[3:0] and the receiver FIFO level ('RFL') reaches the value programmed in the FCH register, one or two XOFFs may be transmitted to stop the flow of serial data as defined by EFR[3:0]. The flow may be resumed by sending one or two XON characters (as defined in EFR[3:0]) when the receiver FIFO is drained below a lower limit defined in the FCL register. The FCL value of 0x00 is illegal. For example if FCL and FCH contain 0x04 and 0x67, XOFF is transmitted when the receiver FIFO contains 103 characters, and XON is transmitted when the FIFO is drained to 3 characters.

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, RTS# (or DTR#) 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 a lower limit defined in FCL. When 950 trigger level is enabled (ACR[5]=1), the CTS# flow control functions as in 650 mode and is configured by EFR[7]. However, RTS# is automatically de-asserted and re-asserted when EFR[6] is set and RFL reaches FCH and drops below FCL. DSR# flow control is configured with ACR[2]. DTR# flow control is configured with ACR[4:3].

8.4.7 Device Identification (ICR offsets 0x08, 0x09, 0x0A and 0x0B)

The OX16C954 offers four bytes of device identification. The device ID registers may be read using offset values 0x08 to 0x0B of the Indexed Control Register. Registers ID1, ID2 and ID3 identify the device as an OX16C954 and return 0x16, 0xC9 and 0x54 respectively. The REV register resides at offset 0x0B of ICR and identifies the revision of OX16C954. This register returns 0x00 for revision A of the OX16C954.

8.4.8 Times Clock Register 'TCR' (ICR offset 0x02)

The OX16C954 has the facility to operate at baud-rates up to 12.5 Mbps. These speeds are enabled through the use of the Times Clock Register, TCR. The 16C450, 16C550, and compatible devices use a 16x over-sampling clock to sample the serial bit stream. This means that the highest baud rate is one sixteenth of the system clock. The OX16C952 enables the device drivers to select lower over-sampling clock rates to increase the available baud rates from a given system clock through the TCR register.

To maintain compatibility with 16C550, at reset the TCR register is set to 0x00 to select the 16x over-sampling clock used by 16C550. In other words, this will select an over-sampling clock that uses 16 clock cycles per bit. The current version of OX16C952 offers a selection of 4x and 16x over-sampling clocks. The table below indicates how the value in the register corresponds to the number of clock cycles per bit. TCR[3:0] is used to program the clock. TCR[7:4] are unused and will return "0000" if read. Please refer to section 8.6 (8.6.11 in particular) to see how the configuration of these register affects the baud rate generation calculations.

TCR[3:0]	Clock cycles per bit
0000 to 0011	16
0100	4
0101	Reserved
0110	Reserved
0111	Reserved
1000	Reserved
1001	Reserved
1010	Reserved
1011	Reserved
1100	Reserved
1101	Reserved
1110	Reserved
1111	Reserved

The features of the TCR do not require the device to be in 650 or 950 mode although only drivers that have been written to take advantage of the 950 mode features will actually be able to change this register. Writing 0x01 to the TCR will not switch the device into 1x isochronous mode (TCR has no effect in isochronous mode). If 0x01, 0x10 or 0x11 is written to TCR the device will operate in 16x mode. A read of the TCR will always return the last value that was written to it irrespective of mode of operation.

8.5 Flow Control

Automatic in-band flow control, automatic out-of-band flow control and special character detection features can be used when in Enhanced mode and are software compatible with the 16C654. Alternatively, automatic out-of-band flow control can be enabled when in non-Enhanced mode that is 16C750 compatible. In 950 mode, in-band and out-of-band flow controls are compatible with 16C654 with the exception of additional flow control thresholds.

8.5.1 Enhanced Features Register ('EFR')

Writing 0xBF to LCR enables access to the EFR and other Enhanced mode 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.

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 will disable transmission as soon as any pending transmission has completed. The UART then compares the received data with the programmed XON character(s). When this occurs, the UART will re-enable transmission (see section 8.5.3).

For automatic in-band flow control the software driver must first set bit 4 for EFR. Then, 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, recognising XON2 as the XON character and XOFF2 as the XOFF character.
- logic [10] ⇔ Single character in-band receive flow control enabled, recognising XON1 as the XON character and XOFF1 and the XOFF character.

Important Note: When EFR[1:0] = "11" the behaviour of the receive flow control is dependent on the configuration of EFR[3:2].

- logic [11] ⇔ Double character in-band receive flow control enabled, accepting XON1 directly followed by XON2 as the XON sequence and XOFF1 directly followed by XOFF2 as the XOFF sequence when EFR[3:2] = "00".
- logic [11] ⇔ Single character in-band receive flow control enabled, accepting both XON1 and XON2 as valid XON characters and both XOFF1 and XOFF2 as valid XOFF characters when EFR[3:2] = "01".
- logic [11] ⇔ Single character in-band receive flow control enabled, accepting both XON1 and XON2 as valid XON characters and both XOFF1 and XOFF2 as valid XOFF characters when EFR[3:2] = "10".
- logic [11] ⇔ Double character in-band receive flow control enabled, accepting XON1 directly followed by XON2 as the XON sequence and XOFF1 directly followed by XOFF2 as the XOFF sequence when EFR[3:2] = "11".

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

When in-band transmit flow control is enabled, XON/XOFF character(s) may be inserted into the data stream whenever the RFL passes the upper trigger level and falls below the lower trigger level respectively (see section 8.5.3).

For automatic in-band flow control, the software driver must first set bit 4 for EFR. Then, the combinations of software transmit flow control can 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] ⇔ Double character in-band transmit flow control enabled, using XON1 directly followed by XON2 as the XON sequence and XOFF1 directly followed by XOFF2 as the XOFF sequence.

EFR[4]: Enhanced mode

- logic 0 ⇔ Non-Enhanced mode. Disables IER bits 4-7, ISR bits 4-5, FCR bits 4-5, MCR bits 5-7 and in-band flow control. When in non-Enhanced mode, the UART is assumed to operate either in standard 16C550 mode or in 16C750 compatibility mode. Whenever this bit is cleared, the setting of other bits of EFR are ignored.
- logic 1 ⇔ Enhanced mode. Enables the Enhanced Mode functions. These functions include enabling IER bits 4-7, FCR bits 4-5, MCR bits 5-7. For in-band flow control the software driver must set this bit first. When this bit is set, the UART is assumed to be in 650 compatibility mode or in 950 mode. If this bit is set, out-of-band flow control is configured with EFR bits 6-7, 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]=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 RTS# pin will be forced inactive high if the RFL reaches the upper flow control threshold. This will be 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 CTS# 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.

8.5.2 Special Character Detection Operation

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[3] will be set and a level 5 interrupt is asserted if level 5 interrupts are enabled (IER[5]=1). This flag will be 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.

8.5.3 Automatic In-band Flow Control Operation

When in-band receive flow control is enabled, the UART will compare the received data with XOFF1 and/or XOFF2 characters to detect an XOFF condition. When this occurs, the UART will disable transmission as soon as any pending transmission has completed. Status bits ISR[4] and ASR[0] will be set. A level 5 interrupt will occur if enabled by IER[5]. The UART will then compare the received data with XON1 and/or XON2 characters to detect an XON condition. When this occurs, the UART will re-enable transmission and status bits ISR[4] and ASR[0] will be cleared.

If single character receiver flow control is in operation, any valid XON/XOFF characters will not be written into the RHR. If double character receiver flow control is in operation, the two characters forming a valid XON/XOFF sequence will both be written into the RHR. Any invalid XON/XOFF characters received are treated as data and transferred to the RHR. Two examples of this are:

1. Receiving an XON when comparing internal data with XOFF.
2. Receiving XOFF1 followed by a character that is not XOFF2 when looking for a double character XOFF sequence.

An exception to the filtering of valid single XON/XOFF characters occurs if special character detection is enabled and an XOFF2 character is received that is a valid XOFF. In this instance, the character will be 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 will get re-enabled. The received data will be transferred to the RHR.

When in-band transmit flow control is enabled, the RFL will be sampled whenever the transmitter is idle (briefly, between characters, or when the THR is empty) and an XON/XOFF character/sequence may be inserted into the data stream. 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 will be sent and ASR[1] will be set. If ASR[1] is set and the RFL falls below the lower trigger level, XON will be sent and ASR[1] will be cleared.

If transmit flow control is disabled after an XOFF has been sent, an XON will be sent automatically.

8.5.4 Automatic Out-of-band Flow Control Operation

Automatic RTS/CTS flow control is selected by different means, depending on whether the UART is in Enhanced or non-Enhanced mode. When in non-Enhanced mode, MCR[5] enables both RTS and CTS flow control. When in Enhanced mode, EFR[6] enables automatic RTS flow control and EFR[7] enables automatic CTS flow control. This allows software compatibility with both 16C650 and 16C750 drivers.

When automatic CTS flow control is enabled, the UART will disable transmission as soon as any pending transmission has completed when the CTS# input is active. Transmission is resumed whenever the CTS# input becomes inactive.

When automatic RTS flow control is enabled, the RTS# pin will be forced inactive when the RFL reaches the upper trigger level and will return to active when the RFL falls below the lower trigger level. The automatic RTS# 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 the UART is in Enhanced mode or not.

8.6 Baud Rate Generation

8.6.1 Clock Prescaler and Sampling Clock Configuration

The UART contains a programmable baud rate generator that is capable of taking any clock input from DC to 50MHz and dividing it by any 16-bit divisor number from 1 to 65535 written into the DLM (MSB) and DLL (LSB) registers. The transmitter and receiver baud rate is:

$$\text{Baud Rate} = \frac{\text{Input Clock Frequency}}{(\text{SC} * \text{Divisor} * \text{Prescalar})}$$

where

SC = the sample clock value as defined by TCR[3:0] (see section 8.4.8)

Divisor = DLL + (256 * DLM)

Prescalar = 1 when MCR[7] = '0' else

$$\text{Prescalar} = M + \frac{N}{8}$$

where M = CPR[7:3] and N = CPR[2:0]

After a hardware reset, MCR[7] is set to the complement of CLKSEL (pin 30). Since in 16C554 compatible systems pin 30 is VDD, CLKSEL is tied high and hence MCR[7] is defaulted to '0' and a prescaler value of '1' is selected by default (see the above equations). Software drivers may set or clear MCR[7] to over-ride CLKSEL. Since access to MCR[7] is restricted to Enhanced mode only, the driver should first set EFR[4] and then set or clear MCR[7] as required.

If CLKSEL is connected to ground or MCR[7] is set by the software driver, the internal clock prescaler is enabled. The prescaler divides the system clock by any value in the range of 1 to "31 7/8" in steps of 1/8. The divisor take the form "M N/8", where M is a 5 bit value defined in CPR[7:3] and N is a 3 bit value defined in CPR[2:0]. The CPR is the Clock Prescaler Register and it resides at offset 0x01 of the Indexed Control Register ('ICR'). To write to CPR first ensure that the last value written to LCR was not the reserved value 0xBF. Then software driver should write 0x01 to the Scratch Pad Register (address 111b) to set the offset and then access CPR using the ICR register (address 101b).

Upon a hardware reset, CPR defaults to 0x20 (division-by-4). Hence compatibility with existing 16C550 baud rate divisor table is maintained using either a 1.8432MHz clock with CLKSEL (pin 30) connected to VDD, or a 7.372MHz clock with CLKSEL connected to ground. The OX16C954 device can accept any input clock frequency up to 50 MHz. When the prescaler is by-passed, the 7.372MHz clock can offer baud rates up to 460.8Kbps. For higher baud rates use a higher frequency clock, e.g. 14.7456MHz, 18.432MHz, 32MHz, 40MHz or 50.0MHz. Indeed any reasonably fast clock may be employed. The flexible prescaler allows system designers to use clocks that do not divide to popular baud rates directly. When using a non-standard clock frequency, compatibility with existing 16C550 software drivers may be maintained with a minor software patch to program the on-board prescaler.

Upon a hardware reset the TCR is reset to 0x00 which means that a 16x clock will be used to evaluate the bit-period. This is configurable to select 4x and 16x clocks and allows data rates up to 12.5 Mbps. The TCR is best employed for data rates over 3.125 Mbps although it can be used at any data rate. All the tables below assume that TCR is set to 0x00 and so is utilising a 16x clock. To convert the baud-rates in the tables to take into account the value in TCR it is necessary to multiply the baud-rate in the table by 16 then divide it by the value that has been written into TCR (If 0x00, 0x01, 0x02 or 0x03 has been written to TCR then the device will operate with a 16x clock i.e. the baud-rates would be as indicated in the tables).

The baud rate divisor values for 1.8432MHz, 7.372MHz and 14.7456MHz clocks are tabulated below. The 'standard-divisor' column contains values used in 16C550 compatible software drivers.

8.6.2 1.8432MHz Clock

When using a 1.8432MHz clock connect CLKSEL (pin 30) to VDD. After a hardware reset, bit 7 of MCR is cleared thus by-passing the prescaler. Then run existing 16C550 software drivers. The 550 compatible baud rate divisors are tabulated below. They offer standard baud rate from 50 to 115.2Kbps. The maximum baud rate may be increased to 460.8 Kbps by utilising the TCR register.

8.6.3 7.372MHz Clock

When using a 7.372MHz clock connect CLKSEL (pin 30) to ground. After a hardware reset, bit 7 of MCR is set thus enabling the prescaler. Since CPR defaults to 0x20 (i.e. division-by-4), compatibility with any existing 550 software driver is maintained. The 7.372MHz clock can offer higher baud rates using a minor software patch. An alternate set of baud rate values up to 460.8Kbaud is available by clearing MCR[7], thus by-passing the prescaler. Note that access MCR[7] is only possible in the Enhanced mode. To clear MCR[7], first place the device in the Enhanced mode by setting EFR[4]. Then write a '0' to MCR[7]. The maximum baud rate may be increased to 1.8432 Mbps by utilising the TCR register.

Clock	1.8432MHz	7.372MHz		14.7456MHz			
		550 compatible divisors		550 compatible divisors		Non-compatible divisor ³	
Divisor ¹	CPR=0x20 ² MCR[7]=0	CPR=0x20 MCR[7]=1	CPR=0x20 ² MCR[7]=0	CPR=0x40 MCR[7]=1	CPR=0x20 ² MCR[7]=0	Other Divisor	CPR=0x20 ² MCR[7]=0
2304	50	50	200	50	400	1536	600
1536	75	75	300	75	600	768	1200
768	150	150	600	150	1200	384	2400
384	300	300	1200	300	2400	192	4800
192	600	600	2400	600	4800	96	9600
96	1200	1200	4800	1200	9600	64	14.4K
48	2400	2400	9600	2400	19.2K	48	19.2K
32	3600	3600	14.4K	3600	28.8K	32	28.8K
24	4800	4800	19.2K	4800	38.4K	24	38.4K
16	7200	7200	28.8K	7200	57.6K	16	57.6K
12	9600	9600	38.4K	9600	76.8K	12	76.8K
6	19.2K	19.2K	76.8K	19.2K	153.6K	8	115.2K
3	38.4K	38.4K	153.6K	38.4K	307.2K	4	230.4K
2	57.6K	57.6K	230.4K	57.6K	460.8K	2	460.8K
1	115.2K	115.2K	460.8K	115.2K	921.6K	1	921.6K

Note 1: This is a standard divisor table used by 16C550 compatible software drivers.

Note2: Whenever MCR[7] is cleared, the CPR value is ignored.

Note3: These divisor values are not compatible with standard 16C550 values. The divisor values are programmed in the DLL and DLM registers using the equation: Divisor = DLL + 256*DLM.

8.6.4 14.7456MHz Clock

The 14.7456MHz clock offers compatibility with the existing software drivers while providing higher popular baud rates, e.g. 230.4K, 460.8K and 921.6K. Higher baud rates are offered using an alternate baud rate table. This clock is the slowest frequency capable of offering 921.6Kbaud. Hence, it offers the best speed/power performance for fast popular baud rates. When using a 14.7456MHz clock, the standard 16C550 compatible divisor table is available by programming the prescaler to perform a divide-by-8 operation using a simple software patch. To divide the clock by 8 write 0x40 to CPR using offset 0x01 of ICR. Enable the Enhanced mode (EFR[4]=1) and then set MCR[7]. The baud rate table for 14.7456MHz clock is shown above. For faster baud rates the software driver should ensure that MCR[7] is cleared to by-pass the prescaler. Then the standard 550-compatible divisor table offers 430.8k and 921.6K baud rates as shown in the table above.

The 14.7456MHz clock can also offer a full set of popular baud rates when using a non-standard set of divisor values suggested above (see non-compatible divisor column). This set offers 115.2K, 230.4K, 460.8K and 921.6K as well as the slower popular baud rates. Use the above equation and program new divisor values in the DLL and DLM registers. The maximum baud rate may be increased to 3.6864 Mbps by utilising the TCR register.

Baud rates in excess of 921.6K require faster clocks unless the TCR register is utilised. The 18.423MHz, 32MHz, 33MHz, 40MHz and 50MHz clock prescaling requirements are discussed below. These examples clearly demonstrate that given any fast clock, the prescaler can offer compatibility with existing 16C550 software drivers whilst providing alternate baud rate tables with fast and popular baud rate values up to 921.6Kbps. Non-standard values up to 3.125Mbps are also available.

8.6.5 18.432MHz Clock

For 18.432MHz clock, select divide-by-10 prescaling (CPR=0x50) to emulate a 1.8432MHz clock and maintain compatibility with existing drivers. For faster baud rates by-pass the prescaler and program the alternate divisor table using DLL and DLM register (see the above equation). This table gives popular baud rates up to 230.4K as well as 576K and 1.152Mbaud. The maximum baud rate may be increased to 4.608 Mbps by utilising the TCR register.

18.432 MHz						50.0 MHz					
CPR = 0x50 MCR[7]=1			CPR = 0x20 ² MCR[7]=0			CPR = 0xD9 MCR[7]=1			CPR = 0x20 ² MCR[7]=0		
Divisor ¹	Baud Rate	% Err	Other divisor ³	Baud Rate	% Err	Divisor ¹	Baud Rate	% Err	Other divisor ³	Baud Rate	% Err
2304	50	0	3840	300	0	2304	300	0.006	2604	1200	0.01
1536	75	0	1920	600	0	768	1200	0.006	1302	2400	0.01
768	150	0	960	1200	0	384	2400	0.006	651	4800	0.01
384	300	0	480	2400	0	192	4800	0.006	326	9600	0.15
192	600	0	240	4800	0	96	9600	0.006	217	14.4K	0.01
96	1200	0	120	9600	0	64	14.4K	0.006	163	19.2K	0.15
48	2400	0	80	14.4K	0	48	19.2K	0.47	109	28.8K	0.45
32	3600	0	60	19.2K	0	32	28.8K	0.47	94	33.4K	0.47
24	4800	0	40	28.8K	0	24	38.4K	0.47	56	56K	0.35
16	7200	0	34	33.4K	0.1	16	57.6K	0.47	27	115.2K	0.47
12	9600	0	20	57.6K	0	12	76.8K	0.47	6	520.8K	0
6	19.2K	0	10	115.2K	0	8	115.2K	0.47	5	625K	0
3	38.4K	0	5	230.4K	0	4	230.4K	0.47	3	1.042M	0
2	57.6K	0	2	576K	0	2	460.8K	0.47	2	1.563M	0
1	115.2K	0	1	1.152M	0	1	921.5K	0.47	1	3.125M	0

Note 1: This is a standard divisor table used by 16C550 compatible software drivers.

Note2: Whenever MCR[7] is cleared, the CPR value is ignored.

Note3: These divisor values are not compatible with standard 16C550 values. The divisor values are programmed in the DLL and DLM registers using the equation: $Divisor = DLL + 256 * DLM$

8.6.6 32.0MHz Clock

For 32.0MHz clock, select divide-by-17.375 prescaling ("17 3/8" giving CPR=0x8B) to emulate a 1.8432MHz clock and maintain compatibility with existing drivers. This division yields a negligible bit rate error of 0.08%. The 32.0Mhz alternate baud rate table offers values up to 2.0Mbps (see table below). The maximum baud rate may be increased to 8.0 Mbps by utilising the TCR register.

For fast and popular baud rates up to 460.8K program the prescaler to divide by 4.375 ("4 3/8" giving CPR=0x23). This value of prescaler divides the 32.0MHz clock to 7.372MHz with an acceptable error of 0.8%.

This set of baud rate values uses the standard 16C550 divisor table. For a full set of available baud rates see the 7.732MHz table shown above.

8.6.7 33.0MHz Clock

For 33.0MHz clock, select divide-by-17.875 prescaling (“17 7/8” giving CPR=0x8F) to emulate a 1.8432MHz clock and maintain compatibility with existing drivers. This division yields a small bit rate error of 0.16%.

For popular baud rates up to 921.6K program the prescaler to divide by 2.25 (“2 2/8” giving CPR=0x12). This value of prescaler divides the 33.0MHz clock to 14.7456MHz with an acceptable error of 0.16%. This set of baud rate values offers all of the popular values including 115.2K, 230.4K, 460.8K and 921.6Kbps.

Another option to obtain fast popular baud rates but avoid programming a new divisor table is to divide the 33.0MHz clock to 7.372MHz. Software drivers may then use the standard 16C550 divisor table. Program 0x24 in CPR to divide the 33.0MHz clock by 4.5 “4 4/8”, giving a prescaling error of 0.5%.

8.6.8 40.0MHz Clock

For 40.0MHz clock, select divide-by-21.75 prescaling (“21 6/8” giving CPR=0xAE) to emulate a 1.8432MHz clock and maintain compatibility with existing device drivers. This division yields an acceptable bit rate error of 0.22%. The alternate set of baud rates tabulated below for 40.0MHz clock offers non-standard fast baud rates up to 2.5Mbps. The maximum baud rate may be increased to 10 Mbps by utilising the TCR register.

For fast and popular baud rates up to 460.8K program the prescaler to divide by 5.375 (“5 3/8” giving CPR=0x2B). This value of prescaler divides the 40.0MHz clock to 7.372MHz with 0.94% error which is acceptable. This set of baud rates uses the standard 16C550 divisor table (see the 7.732MHz table above).

32.0 MHz, CPR = 0x08, MCR[7]=0			40.0 MHz, CPR = 0x08, MCR[7]=0		
Non-compatible divisor values	Baud Rate	% Err	Non-compatible divisor values	Baud Rate	% Err
3333	600	0.01	4167	600	0.01
1667	1200	0.02	2083	1200	0.02
833	2400	0.04	1042	2400	0.03
417	4800	0.08	521	4800	0.03
208	9600	0.16	260	9600	0.16
139	14.4K	0.08	174	14.4K	0.22
104	19.2K	0.16	130	19.2K	0.16
69	28.8K	0.64	87	28.8K	0.22
60	33.4K	0.20	75	33.4K	0.20
36	56K	0.80	45	56K	0.80
35	57.6K	0.80	43	57.6K	0.93
10	200K	0	10	250K	0
5	400K	0	5	500K	0
4	500K	0	4	625K	0
2	1M	0	2	1.25M	0
1	2M	0	1	2.5M	0

8.6.9 50.0MHz Clock

For 50.0MHz clock, select a divide by 27.125 (i.e. “27 1/8” giving CPR=0xD9) to emulate a 1.8432MHz clock and maintain compatibility with existing drivers. This division yields a negligible bit rate error of 0.006%. The alternate set of baud rates tabulated above for 50.0MHz clock offers non-standard fast baud rates up to 3.125Mbps. However, this set of baud rates requires a non-standard divisor table as shown above. The maximum baud rate may be increased to 12.5 Mbps by utilising the TCR register.

For popular baud rates up to 921.6K program the prescaler to divide by 3.375 (“3 3/8” giving CPR=0x1B). This value of prescaler divides the 50.0MHz clock to 14.7456MHz with a 0.5% error which is acceptable. Then use the non-compatible divisors shown in the 14.7456MHz table above. This set of baud rates offers all of the popular baud rates including 115.2K, 230.4K, 460.8K and 921.6Kbps. Note that as well as programming the prescaler, the driver is required to program the new divisor values in the DLL and DLM registers.

Another option to obtain fast popular baud rates but avoid programming a new divisor table is to divide the 50.0MHz clock to 7.372MHz. Software drivers may then use the standard 16C550 divisor table. Program 0x36 in CPR to divide the 50.0Mhz clock by 6.75 “6 6/8”, giving a prescaling error of 0.5%.

8.6.10 External 1x Clock Mode

The CPR setting 0x00 is reserved for selecting the 1x external clock, otherwise known as the Isochronous mode. In this mode an external 1x clock (or bit rate clock) is applied to the RI# pin. The transmitter and receiver will use this 1x clock applied to the RI# pin. In 1x mode, asynchronous framing is maintained using start-bit, parity and stop-bit, however serial transmission and reception is synchronised to the external 1x clock. In this mode asynchronous data may be transmitted at baud rates up to 50Mbps. To configure OX16C954 in 1x clock mode, first put the UART in Enhanced mode by setting EFR[4]. Then write 0x00 to CPR. Finally set bit 7 of MCR.

8.6.11 Very high speed baud rates up to 12Mbps

The 16C550 and other compatible devices such as 16C650 and 16C750 use a 16 times (16x) over-sampling channel clock. The 16x over-sampling clock means that the channel clock runs at 16 times the selected serial bit rate. It limits the highest baud rate to 1/16 of the system clock when using a divisor latch value of unity. However, the 16C954 UART is designed in a manner to enable it to accept other multiplications of the bit rate clock. It can use 4x and 16x clock as programmed in the TCR as long as the clock (oscillator) frequency error, stability and jitter are within reasonable parameters. The TCR register resides at offset 0x02 of ICR (see section 8.4.8).

N-Times Clock	TCR Value	System Clock (MHz)							
		1.8432	7.372	14.7456	18.432	32	33	40	50
16	0x00	115,200	460,750	921,600	1,152,000	2,000,000	2,062,500	2,500,000	3,125,000
4	0x04	460,800	1,843,000	3,686,400	4,608,000	8,000,000	8,250,000	10M	12.5M

For very high-speed operation up to 12.5Mbps, the TCR can be programmed to use a 4x clock. For example given a 50MHz clock and using a divisor value of 1 programmed in DLL and DLM register, 12.5Mbps can be obtained by writing 0x04 to TCR. In another example a 1.8432MHz clock can provide baud rates up to 460.8Kbps by writing 0x04 to TCR.

The maximum baud-rates available for each of the system clock frequencies mentioned earlier at all of the allowable values of TCR are indicated in the following table. These are the values in bits-per-second (bps) that are obtained if the divisor latch = 0x01 and the Prescaler is set to 1.

The N-Times clock feature is designed for use only when the device has to transmit at speeds faster than those possible with the standard 16x clock with any given system clock. Therefore, if your system clock is fixed at 50MHz but you never need baud-rates in excess of 3.125Mbps then it is recommended that the value of N-Times clock should always be the default value of 16 (TCR = 0x00).

For applications where the designer must select the appropriate system clock for the application it is recommended that a faster system clock be used and a higher value of the N-times clock be used.

Example 1:

An application requires a maximum baud-rate of 460kbps. It is recommended that a 7.372MHz clock be used with an N-Times clock value of 16 rather than a 1.8432MHz clock with an N-Times clock value of 4. This approach would allow for some future-proofing of the design because the application could run up to 4 times faster by modifying the controlling software and not needing to change the oscillator.

However for lower power applications or where there is only a 1.8432MHz clock is available, The software driver should select 4 time over-sampling (TCR=0x04). In this case a 1.8432MHz clock can provide baud rates up to 460Kbps.

Example 2:

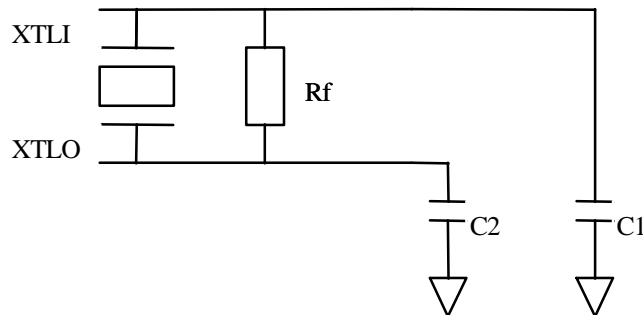
Whenever possible it is recommended that the N-Times clock should be set to 16 (TCR = 0x00).

If the divisor latch = 28 and the N-Times clock = 4 at 50Mhz then the baud-rate is 446.4kbps (assuming the clock pre-scalar = 1).

It is recommended that the divisor latch should be set to 7 and the N-Times clock be set to 16. This would result in exactly the same baud-rate.

8.6.12 Clock oscillator

The OX16C954 may be clocked by a crystal connected to XTLI and XTLO or directly from a clock source connected to the XTLI pin. The oscillator circuit is shown below.



Freq Range (MHz)	C1 (pF)	C2 (pF)	Rf (Ω)
1-50	33 – 68	33 – 68	100K - 10M

Note: For better stability use a smaller value of Rf. Increase Rf to reduce power consumption. Oxford Semiconductor's reference design uses 33pF values for C1 and C2 and 2.2M for Rf.

8.7 FIFOs

8.7.1 Transmitter and Receiver Holding Registers

Both the transmitter and receiver have associated holding registers (FIFOs), referred to as the transmitter holding register (THR) and receiver holding register (RHR) respectively. In normal operation, when the transmitter finishes transmitting a byte it will remove the next data from the top of the THR and proceed to transmit it. If the THR is empty, it will wait until data is written into it. Similarly, when the receiver finishes receiving a byte, it will transfer it to the bottom of the RHR. If the RHR is full, an overrun condition will occur (see section 8.8). Data can be 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. The size of the FIFOs is dependent on the setting of the FCR register. When in Byte mode, these FIFOs only accept one byte at a time before indicating that they are full; this is compatible with the 16C450. When 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 are indicated in the LSR (see section 8.8). 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 be read back from the transmitter FIFO level (TFL) and receiver FIFO level (RFL) registers (see section 8.4).

8.7.2 FIFO Control Register ('FCR')

FCR[0]: Enable FIFO mode

logic 0 ⇔ Byte mode.
logic 1 ⇔ FIFO mode.

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

FCR[1]: Flush RHR

logic 0 ⇔ No change.
logic 1 ⇔ Flushes the contents of the RHR. This is only operative when already in a FIFO mode. The RHR is automatically flushed whenever changing between Byte mode and a FIFO mode. This bit will return to zero after clearing the FIFOs.

FCR[2]: Flush THR

logic 0 ⇔ No change.
logic 1 ⇔ Flushes the contents of the THR, in the same manner as FCR[1] does for the RHR.

FCR[3]: Select DMA signalling mode

logic 0 ⇔ FIFO mode '0'.
logic 1 ⇔ FIFO mode '1'.

Refer to the 'DMA Transfer Signalling' section below. In FIFO mode 0, the transmitter trigger level is set to 1, thus ignoring FCR[5:4].

FCR[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 FCR[5] increase the FIFO size. In Byte mode (FCR[0]=0), non-Enhanced mode (EFR[4]=0) or FIFO mode 0 (FCR[3]=0), these bits are ignored and the transmitter trigger level is set to 1. When 950 thresholds are enabled (ACR[5]=1), these bits are ignored again and the trigger level is defined in the TTL register.

450, 550 and extended 550 modes:

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

650 mode:

In 650 mode the transmitter interrupt trigger levels are set to the following values:

FCR[5:4]	Trigger level
00	16
01	32
10	64
11	112

These levels only apply when in Enhanced mode and in FIFO mode 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, FCR[4] is unused and FCR[5] defines the FIFO depth as follows:

FCR[5]=0 transmitter and receiver FIFO sizes are 16 bytes.

FCR[5]=1 transmitter and receiver FIFO sizes are 128 bytes.

In non-Enhanced mode and when FIFSEL# pin is high, FCR[5] is only writable when LCR[7] is set. Note that in Enhanced mode, the FIFO size is increased to 128 bytes when FCR[0] is set.

950 mode:

In 950 mode (ACR[5]=1), the transmitter trigger level may be set to any arbitrary value programmed in the TTL register (see section 8.4), hence FCR[5:4] is ignored.

FCR[7:6]: RHR trigger level

In 550, extended 550, 650 and 750 modes, the receiver FIFO trigger levels are defined using FCR[7:6]. The interrupt, upper and lower trigger levels are tabulated below:

FCR[7:6]	FIFO size = 128			FIFO size = 16		
	Interrupt trigger level	Upper trigger level	Lower trigger level	Interrupt trigger level	Upper trigger level	Lower trigger level
00	16	16	1	1	1	1
01	32	32	16	4	4	1
10	112	112	32	8	8	1
11	120	120	112	14	14	1

In Byte mode (450 mode) the trigger levels are all set to 1. A receiver full interrupt will be generated (if enabled) if the Receiver FIFO Level ('RFL') reaches the upper trigger level. Separate upper and lower trigger levels introduce a hysteresis element in in-band and out-of-band flow control (see section 8.5). Note that the Interrupt trigger level is the same as the upper trigger level. When 950 trigger levels are enabled (ACR[5]=1), more flexible trigger levels can be set by writing to the RTL, FCL and FCH (see Section 8.4) hence ignoring FCR[7:6].

8.7.3 DMA Transfer Signalling:

The TXRDY# pin has no hysteresis and is simply activated using a comparison operation. When the UART is in FIFO mode 0 (Byte mode), the TXRDY# output pin is low whenever THR is empty, otherwise it is high. When in FIFO mode 1, the TXRDY# pin is high when the THR is full, otherwise it is low.

When the UART is in FIFO mode 0 (Byte mode), RXRDY# output pin is high when RHR empty, otherwise it is low. When in FIFO mode 1, the operation is as follows:

1. RXRDY# is set low when RFL has reached the upper trigger level or a time-out event has occurred (see section 8.8). It remains low as long as RHR is not empty.
2. RXRDY# is set high when RHR is empty. It remains high until condition 1 occurs.

8.8 Transmitter and Receiver

8.8.1 Receiver Framing, Sampling and False Start Bit Detection

On the falling edge of the start bit, the receiver will wait for 1/2 bit and re-synchronise the receiver's sampling clock onto 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 will proceed 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.

Once the first stop bit has been sampled, the received data is transferred to the RHR and the receiver will then wait for a low transition on SIN signifying the next start bit.

The receiver will continue receiving data even if the RHR is full or the receiver has been disabled (see section 8.4) in order to maintain framing synchronisation. The only difference is that the received data does not get transferred to the RHR.

8.8.2 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]	Data length
00	5 bits
01	6 bits
10	7 bits
11	8 bits

LCR[2]: Number of stop bits

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.

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.

It is the responsibility of the software driver to ensure that the break duration is longer than the character period (for it to be recognised remotely as a break rather than data).

LCR[7]: Divisor latch enable

- logic 0 ⇔ Access to DLL and DLM registers disabled.
- logic 1 ⇔ Access to DLL and DLM registers enabled.

8.8.3 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 be flagged when the data would normally have been transferred to the RHR.

LSR[2]: Received data parity error

- logic 0 ⇔ No parity error.
- logic 1 ⇔ Data has been received that did not have correct parity.

The flag will be set when the data item in error is at the top of the RHR. It is cleared following a read of the LSR.

LSR[3]: Received data framing error

- logic 0 ⇔ No framing error.
- logic 1 ⇔ Data has been received that did not have a valid stop bit.

This status bit is set and cleared in the same manner as LSR[2]. When a framing error occurs, the UART will try to re-synchronise 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 that the SIN line is sampled at the bit rate). One zero character with associated break flag set will be transferred to the RHR and the receiver will then wait until the SIN line returns high. The LSR[4] break flag will be set when this data item gets to the top of the RHR and it is cleared following a read of the LSR.

LSR[5]: THR empty

- logic 0 ⇔ Transmitter holding register (FIFO) is not empty.
- logic 1 ⇔ Transmitter holding register (FIFO) is empty.

LSR[6]: Transmitter and THR empty

- logic 0 ⇔ THR is not empty or transmitter is still transmitting.
- logic 1 ⇔ THR is empty and the transmitter has completed the character in shift register and is in idle mode.

This flag is 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 will be 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].

8.9 Interrupts

In Intel mode ($I/M\#=1$) interrupts from serial channel 0 to channel 3 are asserted on pins INTO to INT3 respectively. The serial channel interrupt outputs are 3-state output pins and are placed in high-impedance state after a hardware reset. Bit 3 of the MCR for each channel enables the corresponding interrupt pin. When MCR[3] is low the interrupt pin is in the high-impedance state, otherwise it is a forcing active high pin which goes high whenever there is an interrupt pending and goes low when the interrupt is cleared.

In Motorola mode ($I/M\#=0$) interrupts from all channels are wire-ORed internally and asserted on the open-drain output called IRQ# (pin 15). This pin goes active (low) when the interrupt signal from any of the channels is asserted, otherwise it is in high-impedance state.

8.9.1 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

- logic 0 ⇔ Disable the receiver status interrupt.
- logic 1 ⇔ Enable the receiver status interrupt.

IER[3]: Modem status interrupt mask

- logic 0 ⇔ Disable the modem status interrupt.
- logic 1 ⇔ Enable the modem status interrupt.

IER[4]: Sleep mode

- logic 0 ⇔ Disable sleep mode.
- logic 1 ⇔ Enable sleep mode whereby the internal clock of the channel is switched off.

Sleep mode is described in section 8.9.4. Note that sleep operation does not function in IrDA mode.

IER[5]: Special character interrupt mask (Enhanced mode) or alternate sleep mode (750 mode)

650/950 modes:

- logic 0 ⇔ Disable the received special character interrupt.
- logic 1 ⇔ Enable the received special character interrupt.

In 16C650 compatible mode when the device is in Enhanced mode ($EFR[4]=1$), this bit enables the detection of special character. 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]). Note that 950 drivers also operate in Enhanced mode.

750 mode:

logic 0 ⇔ Disable alternate sleep mode.

logic 1 ⇔ Enable alternate sleep mode whereby the internal clock of the channel is switched off.

In 16C750 compatible mode (i.e. non-Enhanced mode), this bit is used as an alternate sleep mode and has the operation effect as IER[4]. Sleep mode is described in section 8.9.4. Note that alternate sleep mode does not function in IrDA mode.

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 masked.

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 masked.

8.9.2 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:

Priority level	Interrupt source	ISR[7:6] ³		ISR[5] ⁵			ISR[4:0] ^{1,2}
		Byte Mode	FIFO Mode	750 mode		650 mode	
				16 Byte FIFO	128 Byte FIFO		
-	No interrupt pending	00	11	0	1	0	00001
1	Receiver status error	00	11	0	1	0	00110
2a	Receiver data available	00	11	0	1	0	00100
2b	Receiver time-out	00	11	0	1	0	01100
3	Transmitter THR empty	00	11	0	1	0	00010
4	Modem status change	00	11	0	1	0	00000
5 ⁴	In-band flow control XOFF detected or Special character (XOFF2) detected	00	11	0	1	0	10000
6 ⁴	CTS or RTS change of state	00	11	0	1	1	00000

Note1: ISR[0] indicates whether any interrupts are pending.

Note2: ISR[5:1] indicates which interrupt is the highest one pending.

Note3: ISR[6:7] bits are not used and are set to 0 in BYTE mode and 1 in FIFO modes.

Note4: Interrupts of priority levels 5 and 6 cannot occur unless the UART is in Enhanced mode.

Note5: In 750 mode, ISR[5] is '0' when FIFO size is 16 or '1' when FIFO size is 128.

8.9.3 Interrupt Description

Level 1: Receiver status error interrupt (ISR[5:0]='000110'):

This interrupt is active whenever any of LSR[1], LSR[2], LSR[3] and LSR[4] are high. These flags are cleared following a read of the LSR.

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 the following conditions are true:

1. The UART is in a FIFO mode
2. There is data in the RHR.
3. There has been no read of the RHR for a period of time greater than the time-out period.
4. 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.

This interrupt is cleared by reading the first data item in the RHR.

Level 3: Transmitter empty interrupt (ISR[5:0]='000010'):

This interrupt is set when the transmit FIFO level is 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 16C954 mode trigger levels are enabled (ACR[5]=1) and the transmitter trigger level of zero is selected (TTL=0x00), a transmitter empty interrupt will only be 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 or Receiver special character (XOFF2) detect interrupt (ISR[5:0]='010000'):

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

1. A valid XOFF character is received, disabling the transmitter. It can only be set if in-band flow control is enabled.
2. A received character matches XOFF2. It can only be set if special character detection is enabled (EFR[5]=1).

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 any of the CTS# or RTS# pins changes state from low to high. It is cleared on an ISR read of a level 6 interrupt.

8.9.4 Sleep Mode

For a channel to actually go into sleep mode, all of the following conditions must all be met when either sleep mode is enabled in Enhanced mode (IER[4]=1), or alternate sleep mode is enabled in non-Enhanced mode (EFR[5]=1):

1. The transmitter is idle, i.e. the transmitter shift register and FIFO are both empty.
2. SIN is high.
3. The receiver is idle.
4. The receiver FIFO is empty (LSR[0]=0).

5. The UART is not in loopback mode (MCR[4]=0).
6. Changes on modem input lines have been acknowledged (i.e. MSR[3:0]=0000).
7. 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 (see section 8.4). The UART will not lose any programmed bits whilst in power-down mode.

The channel will automatically exit power-down mode when any of the conditions 1 to 7 becomes false. It may be woken manually by writing a 0 to IER[4] (or IER[5] if a 1 was written to that bit instead).

Note that OX16C942 does not offer the sleep mode operation in IrDA mode.

8.10 Modem interface

8.10.1 Modem Control Register (MCR):

MCR[0]: DTR

- logic 0 ⇔ Force DTR# output to inactive (high).
- logic 1 ⇔ Force DTR# output to active (low).

Note that DTR# can be used for automatic out-of-band flow control when enabled using ACR[4:3] (see section 8.4.1).

MCR[1]: RTS

- logic 0 ⇔ Force RTS# output to inactive (high).
- logic 1 ⇔ Force RTS# output to active (low).

Note that RTS# can be used for automatic out-of-band flow control when enabled using EFR[6] (see section 8.5).

MCR[2]: OUT1

MCR2 has no effect on the operation.

- logic 0 ⇔ Internal OUT1 is low.
- logic 1 ⇔ Internal OUT1 is high.

MCR[3]: External interrupt enable (OUT2)

- logic 0 ⇔ External interrupt is in high-impedance state and thus disabled. Internal OUT2 is low.
- logic 1 ⇔ External interrupt is enabled and operating in normal active (forcing) mode. Internal OUT2 is high.

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) is set high (marking condition), and the receiver inputs SIN, CTS#, DSR#, DCD#, and RI# are all disabled. Internally the transmitter output is connected to the receiver input and DTR#, RTS#, OUT1# (complement of internal OUT1) and OUT2# (complement of internal OUT2) are connected to modem control inputs DSR#, CTS#, RI# and DCD# 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 Modem Control Register instead of the four modem control 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:

logic 0 ⇔ Xon-Any is disabled.

logic 1 ⇔ Xon-Any is enabled.

In 16C650 compatible mode when the device is in Enhanced mode (EFR[4]=1), this bit enables the Xon-Any operation. When Xon-Any is enabled, any received data will be accepted as a valid XON (see in-band flow control, section 8.5.3). Note that 950 drivers also operate in Enhanced mode.

750 mode:

logic 0 ⇔ CTS/RTS flow control disabled.

logic 1 ⇔ CTS/RTS flow control enabled.

In 16C750 compatible mode (i.e. non-Enhanced mode), this bit enables the CTS/RTS out-of-band flow control.

MCR[6]: IrDA mode

logic 0 ⇔ Standard serial receiver and transmitter data format.

logic 1 ⇔ Data will be transmitted and 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, defaults to 0x20 (divide-by-4) and MCR[7] is loaded with the complement of the CLKSEL pin. User writes to this flag will only take effect in the Enhanced mode. See section 8.6.1.

8.10.2 Modem Status Register ('MSR')**MSR[0]: Delta CTS#**

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

MSR[1]: Delta DSR#

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

MSR[2]: Trailing edge RI#

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

MSR[3]: Delta DCD#

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

MSR[4]: CTS

This bit is the compliment of the CTS# input. It is equivalent to RTS (MCR[1]) during local loop-back mode.

MSR[5]: DSR

This bit is the compliment of the DSR# input. It is equivalent to DTR (MCR[0]) during local loop-back mode.

MSR[6]: RI

This bit is the compliment of the RI# input. In local loop-back mode it is equivalent to the internal OUT1.

MSR[7]: DCD

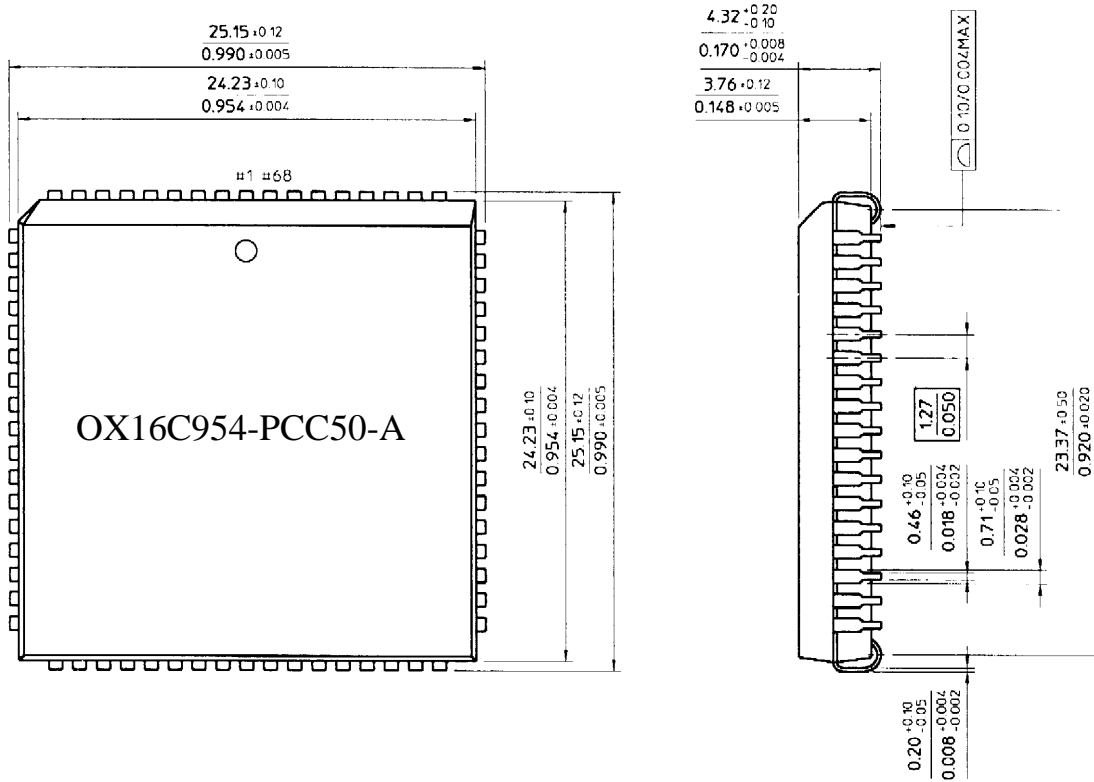
This bit is the compliment of the DCD# input. In local loop-back mode it is equivalent to the internal OUT2.

8.11 Scratch Pad Register ('SPR')

The scratch pad register does not affect operation of the rest of the UART in any way and can be used for temporary data storage. The register may also be used to define an offset value to access the registers in the Indexed Control Register set. For more information on Indexed Control registers see section 8.1.

9 Package Information

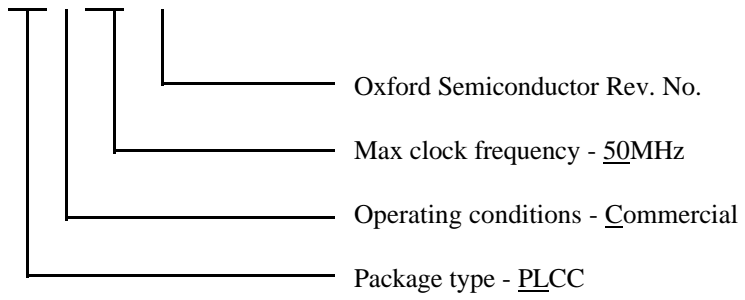
68 Pin Plastic Leaded Chip Carrier (68 PLCC)



10 Ordering Information

Order code:

OX16C954-PCC50-A



Contact Details:

Oxford Semiconductor Ltd
69 Milton Park
Abingdon
Oxon OX14 4RX
United Kingdom

Tel: +44 (0)1235 824900
Fax: +44 (0)1235 821141

Sales e-mail: sales@oxsemi.com
Tech support e-mail: support@oxsemi.com
Web site: <http://www.oxsemi.com>

©Copyright Oxford Semiconductor Ltd 1998

Oxford Semiconductor Ltd believes the information contained in this document to be accurate and reliable. However, it is subject to change without notice. No responsibility is assumed by Oxford Semiconductor for its use, nor for infringement of patents or other rights of third parties. No part of this publication may be reproduced, or transmitted in any form or by any means without the prior consent of Oxford Semiconductor Ltd. Oxford Semiconductor's terms and conditions of sale apply at all times.