NEC

Preliminary User's Manual

System-on-Chip Lite

Gate Array with ARM7TDMI Subsystem, 32-Bit RISC CPU Core

Hardware

µPD65977S1-xxx-B6

Document No. A15402EE1V0UM00
Date Published August 2001

Printed in Germany

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

Strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it once, when it has occurred. Environmental control must be adequate. When it is dry, humidifier should be used. It is recommended to avoid using insulators that easily build static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work bench and floor should be grounded. The operator should be grounded using wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with semiconductor devices on it.

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

No connection for CMOS device inputs can be cause of malfunction. If no connection is provided to the input pins, it is possible that an internal input level may be generated due to noise, etc., hence causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using a pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND with a resistor, if it is considered to have a possibility of being an output pin. All handling related to the unused pins must be judged device by device and related specifications governing the devices.

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

Power-on does not necessarily define initial status of MOS device. Production process of MOS does not define the initial operation status of the device. Immediately after the power source is turned ON, the devices with reset function have not yet been initialized. Hence, power-on does not guarantee out-pin levels, I/O settings or contents of registers. Device is not initialized until the reset signal is received. Reset operation must be executed immediately after power-on for devices having reset function.

ARM, THUMB, ARM7TDMI and ARM7 are either registered trademarks or trademarks of ARM Limited in the UK and/or other countries.
All other brand names or product names are the property of their respective holders.

The related documents in this publication may include preliminary versions. However, preliminary versions are not marked as such.

The export of this product from Japan is regulated by the Japanese government. To export this product may be prohibited without governmental license, the need for which must be judged by th customer. The export or re-export of this product from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

M5 2000.03

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

NEC Electronics Inc. (U.S.)
Santa Clara, California
Tel: 408-588-6000
800-366-9782
Fax: 408-588-6130
800-729-9288

NEC Electronics (Germany) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 02
Fax: 0211-65 03 490

NEC Electronics (UK) Ltd.
Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Italiana s.r.l.
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

NEC Electronics (Germany) GmbH
Benelux Office
Eindhoven, The Netherlands
Tel: 040-2445845
Fax: 040-2444580

NEC Electronics (France) S.A.
Vélizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

NEC Electronics (France) S.A.
Spain Office
Madrid, Spain
Tel: 91-504-2787
Fax: 91-504-2860

NEC Electronics (Germany) GmbH
Scandinavia Office
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Singapore Pte. Ltd.
United Square, Singapore 1130
Tel: 65-253-8311
Fax: 65-250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-2719-2377
Fax: 02-2719-5951

NEC do Brasil S.A.
Electron Devices Division
Rodovia Presidente Dutra, Km 214
07210-902-Guarulhos-SP Brasil
Tel: 55-11-6465-6810
Fax: 55-11-6465-6829

J99.1

Preface

Readers

This manual is intended for users who want to understand the functions of the System-on-Chip Lite (nickname SoCLite).

Purpose

This manual presents the hardware manual of SoCLite.

Organization

This user's manual describes the following sections:

- Pin function
- CPU function
- Internal peripheral function
- Test function

Legend

Symbols and notation are used as follows:

Weight in data notation	:	Left is high-order column, right is low order column
Active low notation	:	n (small letter n before or after signal name)
Memory map address:	:	High order at high stage and low order at low stage
Note	:	Explanation of (Note) in the text
Caution	:	Item deserving extra attention
Remark	:	Supplementary explanation to the text
Numeric notation	:	Binary . . . xxxx or xxx_b Decimal . . . $xxxx_d$ Hexadecimal . . . xxxxH or 0x xxxx

Prefixes representing powers of 2 (address space, memory capacity)

K (kilo): $2^{10} = 1024$

M (mega): $2^{20} = 1024^2 = 1.048.576$

G (giga): $2^{30} = 1024^3 = 1.073.741.824$

Data Type:

Word ... 32 bits

Halfword ... 16 bits

Byte ... 8 bits

Related Documents The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

Document Name	Document No.
System-on-Chip Lite Data Sheet	A15647EE1V0DS00
EA-9HD Family Block Library	A13052EJ6V0BL00
ARM7TDMI Data Sheet	ARM DDI 0029E

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 General

The System-on-Chip Lite ("SoCLite") is based on standard ASIC technology and consists of two blocks: an ARM7TDMI based subsystem and a sea-of-gates area. The ARM subsystem is fully designed and verified as a supermacro. It frees the user from the task of developing a complete RISC computer system. The sea-of-gates area allows the user to implement custom logic or special peripheral functions.

The SoCLite is designed for embedded control applications. To maintain flexibility, SoCLite is not realized as an ASSP (Application Specific Standard Product), this means that it can be used for a wide range of different applications. Once the customer functions are implemented into the sea-of-gates, it becomes a custom SoC.

The ARM7TDMI based subsystem of SoCLite offers a basic combination of general purpose peripheral functions, like a serial communication interface (UART), Timer (32-bit), Interrupt Controller, Watchdog, internal RAM and ROM (only as boot-ROM).

(1) SoCLite CPU

The SoCLite CPU is the ARM7TDMI, an ARM7 32-bit RISC processor core with the THUMB extension, on-chip debugging and 32 x 8 multiplier.
THUMB offers 32-bit RISC performance at 16-bit system cost through a “compression” of the original ARM instruction set, resulting in excellent code density and thus saving memory space. During execution the THUMB instructions are “decompressed” on the fly into full 32-bit ARM instructions. It is possible to select between ARM and THUMB modes during execution.

(2) Bus System

The ARM subsystem includes a fully AMBA compliant bus system structure. Two main buses - AHB and APB - connect the different macros. The AHB is a high-speed multimaster bus for connection to high-speed macros like CPU and memory controller. The APB is a lower speed bus for peripherals like UART, timer, etc. Both buses are 32-bit wide. The APB and any additional signals required for interrupts and reset are made available to the User Defined Logic (UDL).

(3) External Memory Interface Function

The SoCLite memory controller supports static memory-mapped devices including RAM, ROM, Flash and burst ROM. The SoCLite contains a non multiplexed external bus interface, including an address range per chip select of 64 Mbytes with a 32-bit external memory data path.

(4) On-chip Memory

The ARM subsystem features an internal 8 Kbyte RAM and a small 2 Kbyte ROM. The ROM contains a bootstrap loader program, selectable via an external pin, for device start-up.

(5) Interrupt Controller

The interrupt controller supports up to 32 interrupts: 29 interrupts from the UDL (User Defined Logic) and 3 from the ARM subsystem. All interrupts are priority controlled, individually or globally maskable and selectable by triggering the IRQ of the ARM7TDMI core.

(6) Peripherals

The ARM subsystem comprises a simple UART and a timer consisting of a 32-bit down counter with load register. A second timer is used as a watchdog timer, generating a reset on overflow.

(7) Clock Generation

The SoCLite clock generation has two parts: an oscillator and a programmable PLL.
The recommended max. operating frequency to the ARM subsystem is 35 MHz.

(8) User Defined Logic (UDL)

The sea-of-gates area for the User Defined Logic has a size of 190K raw gates. The UDL area is connected via the APB bus with the ARM subsystem. The UDL is for user-developed functions. These can be additional peripheral blocks or hardwired logic functions for reducing the CPU load. The custom logic is implemented into the SoCLite chip using the NEC Gate Array design flow. In addition NEC supports FPGA conversion services for customers not familiar with standard ASIC design flow.

1.2 Device Features

- CPU
  - Core: ARM7TDMI
  - Format: Little endian
  - Min. instruction execution time: 29 ns (@ ϕ = 35 MHz)
  - General registers: 32 bits x 37

- Instruction Set:
  - ARM (32-bit instruction set) and THUMB (16-bit instruction set)
  - Signed multiplication
    (32 bits x 32 bits $\rightarrow$ 32 bits: 5 clocks; 32 bits x 32 bits $\rightarrow$ 64 bits: 6 clocks
  - Saturated operation instructions (with overflow/underflow detection function)
  - 32-bit shift instructions: 1 clock
  - data processing instructions
  - Load/store instructions
  - Signed load instructions

- Internal Memory
  - internal ROM (boot ROM only): 2 Kbyte
  - internal RAM: 8 Kbyte

- Clock Generator
  - PLL output: 6,25 MHz to 115 MHz
  - ARM subsystem: max. 35 MHz
  - Crystal frequency range: 2 MHz $\leq \phi_{CRYSTAL} \leq$ 16 MHz

- Bus Control Unit
  - Address/data separated bus (26-bit address/ 32-bit data bus)
  - 32/16/8-bit bus sizing function

- Serial Interface
  - UART mode: 1 channel

- Timers
  - 32-bit timer with load register: 1 channel
  - 12-bit Watchdog timer: 1 channel

- Interrupt Controller: 32 prioritized interrupt sources

- Built-in Power Saving Modes: Pause mode

- Debug Functionality: On-chip debug capability via JTAG

- UDL Pins: 133

- Power Supply Voltage Range: 3.0 V $\leq V_{DD} \leq$ 3.6 V

- Temperature Range: Ta = - 40 to + 85°C

- Package: 256-pin Plastic BGA, (27 mm x 27 mm)

- Sea-of-Gates: 190K raw gates for UDL

- Process: CMOS-9HD

1.3 Application Fields

The SoCLite is intended to be used in the industrial segment e.g. industrial bus systems, control, POS and for telecommunication applications like terminals, business phones, home communication and xDSL. It is also an excellent choice for other applications where a combination of a predefined ARM subsystem and a user defined logic is required.

1.4 Ordering Information

Part Number	Package
µPD65977S1-xxx-B6	256-pin PBGA (1.27 mm pitch)

1.5 Pin Configuration

Figure 1-1: Pin Configuration of the SoCLite Chip

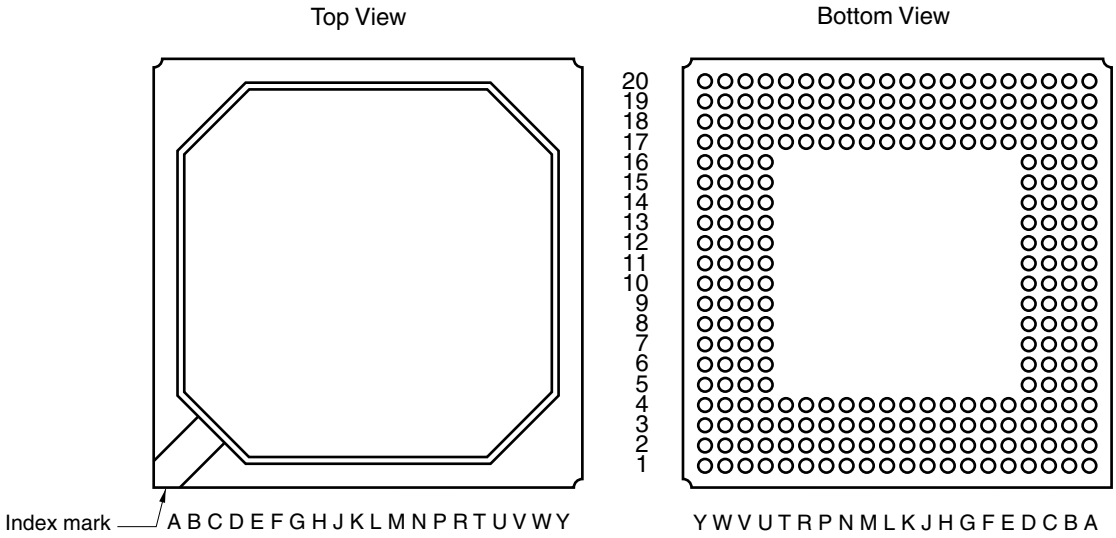


Table 1-1: Pin Configuration

Pin Number	Pin Name	Pin Number	Pin Name	Pin Number	Pin Name	Pin Number	Pin Name
A1	V_{SS0}	D5	nXCS5	L1		U17	V_{SS7}
A2	XDATA31	D6	V_{DD11}	L2		U18	
A3	XDATA29	D7	nXCS0	L3		U19	
A4	XDATA27	D8	V_{SS12}	L4		U20	
A5	nXCS2	D9	XADDR22	L17	VDD7	V1	
A6	XDATA25	D10	XADDR20	L18	IC1**Note1**	V2	
A7	XDATA24	D11	V_{DD10}	L19	nXBLS1	V3	
A8	XDATA22	D12	XADDR16	L20	nXBLS0	V4	
A9	XDATA20	D13	V_{SS11}	M1		V5	
A10	XDATA18	D14	PLLOD0	M2		V6	
A11	XDATA17	D15	V_{DD9}	M3		V7	
A12	XDATA15	D16	IC1**Note1**	M4		V8	
A13	XDATA14	D17	V_{SS10}	M17	JTAG_TDO	V9	
A14	XDATA13	D18	XADDR10	M18	JTAG_TDI	V10	
A15	XDATA12	D19	XADDR9	M19	TXD	V11	
A16	IC1**Note1**	D20	XDATA8	M20	RXD	V12	
A17	AV_{SS}	E1		N1		V13	

Table 1-1: Pin Configuration

Pin Number	Pin Name	Pin Number	Pin Name	Pin Number	Pin Name	Pin Number	Pin Name
A18	DV_{DD}	E2		N2		V14	
A19	X1	E3		N3		V15	
A20	nRST	E4		N4	V_{SS3}	V16	
B1		E17	XADDR8	N17	V_{SS8}	V17	
B2	TCLK	E18	XADDR7	N18	JTAG_TRST	V18	
B3	XDATA30	E19	XDATA7	N19	JTAG_TMS	V19	
B4	XDATA28	E20	XDATA6	N20	JTAG_TCK	V20	
B5	nXCS3	F1		P1		W1	
B6	XDATA26	F2		P2		W2	
B7	XADDR24	F3		P3		W3	
B8	XDATA23	F4	V_{DD0}	P4		W4	
B9	XDATA21	F17	V_{DD8}	P17		W5	
B10	XDATA19	F18	XADDR6	P18		W6	
B11	XADDR17	F19	XADDR5	P19		W7	
B12	XDATA16	F20	XDATA5	P20		W8	
B13	XADDR13	G1		R1		W9	
B14	XADDR11	G2		R2		W10	
B15	PLLOD1	G3		R3		W11	
B16	DV_{SS}	G4		R4	V_{DD2}	W12	
B17	EA1	G17	XADDR4	R17	V_{DD6}	W13	
B18	X2	G18	XADDR3	R18		W14	
B19	TEST[Note2]	G19	XDATA4	R19		W15	
B20	XDATA11	G20	XDATA3	R20		W16	
C1		H1		T1		W17	
C2		H2		T2		W18	
C3	nXCS7	H3		T3		W19	
C4	nXCS6	H4	V_{SS2}	T4		W20	
C5	nXCS4	H17	V_{SS9}	T17		Y1	
C6	nXCS1	H18	XADDR2	T18		Y2	
C7	XADDR25	H19	XDATA2	T19		Y3	
C8	XADDR23	H20	XDATA1	T20		Y4	
C9	XADDR21	J1		U1		Y5	
C10	XADDR19	J2		U2		Y6	

Table 1-1: Pin Configuration

Pin Number	Pin Name	Pin Number	Pin Name	Pin Number	Pin Name	Pin Number	Pin Name
C11	XADDR18	J3		U3		Y7	
C12	XADDR15	J4		U4	V_{SS4}	Y8	
C13	XADDR14	J17	XADDR1	U5		Y9	
C14	XADDR12	J18	XADDR0	U6	V_{DD3}	Y10	
C15	IC1[Note1]	J19	XDATA0	U7		Y11	
C16	AV_{DD}	J20	nXWAIT	U8	V_{SS5}	Y12	
C17	EA0	K1		U9		Y13	
C18	DBGEN	K2		U10	V_{DD4}	Y14	
C19	XDATA10	K3		U11		Y15	
C20	XDATA9	K4	V_{DD1}	U12		Y16	
D1		K17	nXOE	U13	V_{SS6}	Y17	
D2		K18	nXWEN	U14		Y18	
D3		K19	nXBLS3	U15	V_{DD5}	Y19	
D4	V_{SS1}	K20	nXBLS2	U16		Y20	

Notes: **1.** Not connected, leave pin open.
2. For internal use only, must be pulled low always.

Remark: All free pins are available for UDL.

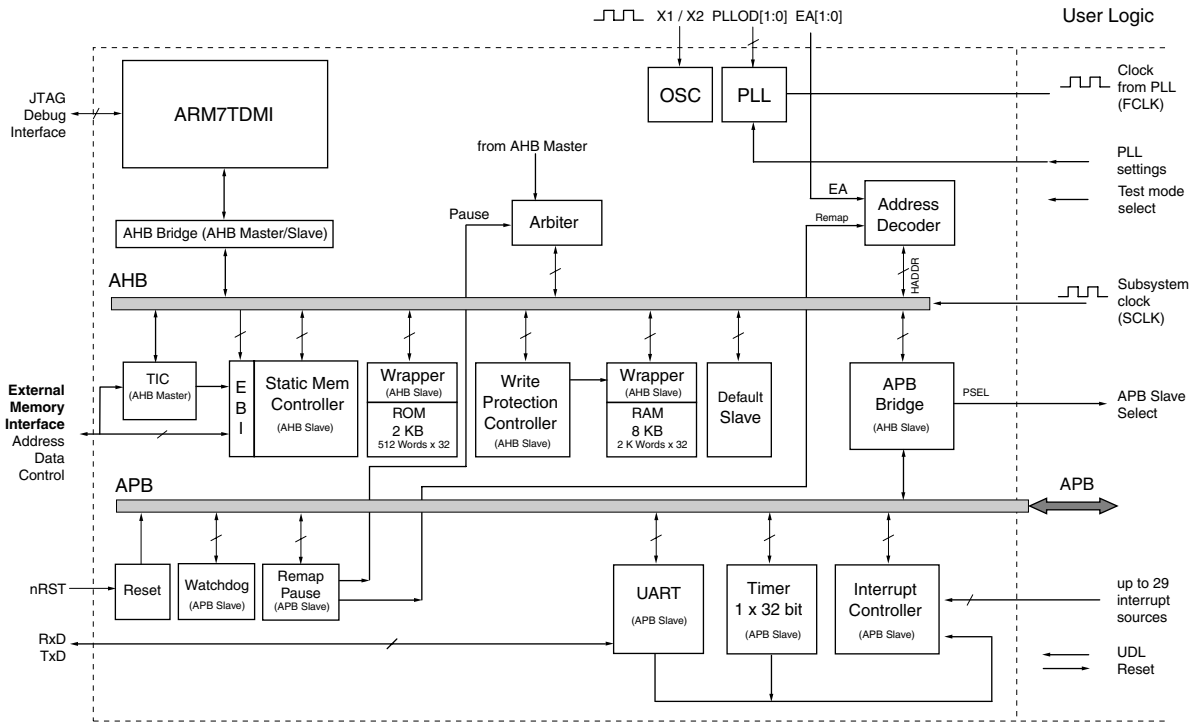
Pin Identification

XADDR0 to XADDR25	: Memory Address Bus	AV_{DD}	: Analog Power Supply for PLL
XDATA0 to XDATA31	: Memory Data Bus	DV_{DD}	: Digital Power Supply for PLL
nXCS0 to nXCS7	: Memory Chip (Bank) Select	AV_{SS}	: Analog Ground for PLL
nXBLS0 to nXBLS3	: Memory Byte Lane Select	DV_{SS}	: Digital Ground for PLL
nXOE	: Memory Output Enable	V_{DD0} to V_{DD11}	: Power Supply
nXWEN	: Memory Read / Write	V_{SS0} to V_{SS12}	: Ground Power Supply
nXWAIT	: Memory Wait	IC1	: Internal connection
RXD	: Receive Data Input	TEST	: Internal connection
TXD	: Transmit Data Output		
X1	: Crystal		
X2	: Crystal		
nRST	: Reset		
EA0	: Enable signal for external or internal boot memory		
EA1	: Enable signal for external or internal boot memory		
DBGEN	: Debug enable		
PLLOD0	: Frequency range of PLL output		
PLLOD1	: Frequency range of PLL output		
JTAG_TDI	: JTAG data in		
JTAG_TDO	: JTAG data out		
JTAG_TCK	: JTAG clock		
JTAG_TMS	: JTAG mode select		
JTAG_TRST	: JTAG reset		
TCLK	: Clock for test purposes		

1.6 Configuration of Function Block

1.6.1 Block Diagram of SoCLite

Figure 1-2: Block Diagram of the SoCLite



1.6.2 On-chip units

(1) CPU (ARM7TDMI)

The ARM7TDMI CPU uses a three-stage pipeline control to enable single-clock execution of address calculations, arithmetic logic operations, data transfers, and almost all other instruction processing.
Other dedicated on-chip hardware, such as the multiplier (32 bits × 32 bits → 32 bits or 32 bits × 32 bits → 64 bits) and the barrel shifter (32 bits), help accelerate processing of complex instructions.
The ARM7TDMI CPU supports little endian format only.

(2) Static Memory Controller (SMC)

SMC starts a required external bus cycle based on the physical address obtained by the CPU. The SMC is used to provide an interface between an AMBA AHB system bus and external (off-chip) memory devices.

(3) Boot ROM

The SoCLite has on-chip boot ROM (2 Kbytes).
During instruction fetch, boot ROM memory can be accessed from the CPU in 1-clock cycles.
If the internal ROM boot mode is set, memory mapping is done from address 00000000H.
Otherwise the boot ROM is located at address 30004000H.

(4) RAM

The SoCLite has on-chip RAM (8 Kbytes).
During instruction fetch, data can be accessed from the CPU in 1-clock cycles.
RAM are mapped from address 30000000H at reset and it can be remapped to address 0H.

(5) TIC

The TIC (Test Interface Controller) is a macro inside the ARM subsystem and needed for test purposes only. As an AHB bus master (only in test mode) the TIC is able to initiate bus cycles on the AHB bus.

(6) Clock Generation (Osc. / PLL)

The SoCLite comprises a PLL to generate an internal system clock (ϕ) to provide them to the UDL in a frequency range of 6,25 MHz to 115 MHz. As the input clock, an external resonator or crystal is connected to pins X1 and X2 or an external clock is input to pin X1.

(7) Interrupt Controller

This controller handles hardware interrupt requests (UART, Timer, 29 UDL interrupts) from on-chip peripheral I/O. Eight levels of interrupt priorities can be specified for these interrupt requests, and multiple-interrupt servicing control can be performed for interrupt sources.

(8) Write Protection Controller

The write protection controller allows to write protect eight 1Kbytes memory areas inside the internal RAM.

(9) Serial Interface

A 1-channel asynchronous serial interface (UART) is provided as serial interface.
The UART transfers data by using the TXD and RXD pins.

(10) Timer

SoCLite utilizes one general purpose 32-bit timer. It is based on a free running 32-bit decrementing counter with a reload register.

(11) Watchdog Timer

SoCLite has a watchdog timer to secure the operation of the device. The watchdog timer is based on a free running 12-bit decrementing counter with a reload register.

(12) Remap / Pause Controller

The remap/pause controller allows to control the boot up memory map and to switch into a low power wait for interrupt state.

[MEMO]

Chapter 2 Pin Functions

The names and functions of the SoCLite pins are listed below. These pins can be divided by function into external pins and internal pins for interconnection between the ARM subsystem and the UDL.

2.1 List of Pin Functions

(1) External Pins

Pin Name	I/O	Function	Alternate
nXOE	O	Data output enable	
nXWEN	O	Write enable signal output	
nXWAIT	I	Control signal input to insert wait in bus cycle	
nXBLS0	O	Byte lane select signal output	
nXBLS1			
nXBLS2			
nXBLS3			
nXCS0	O	Chip select signal output	
nXCS1			
nXCS2			
nXCS3			
nXCS4			
nXCS5			
nXCS6			
nXCS7			
XADDR0 to XADDR25	O	26-bit address bus for external memory	
XDATA0 to XDATA31	I/O	32-bit data bus for external memory	
RXD	I	Serial receive data input	
TXD	O	Serial transmit data output	
X1	I	Crystal resonator connection pin for system clock generation.	
X2	-	Input to X1 pin when providing clocks from outside.	
TCLK**Note1**	I	Clock input for test purpose	
nRST	I	System reset input	
EA0**Note4**	I	Enable external boot memory	
EA1**Note4**	I		
DBGEN**Note4**	I	Debug enable input	
PLLOD0**Note4**	I	Frequency of PLL output clock (FCLK)	
PLLOD1**Note4**	I		
IC1**Note2**	-	Internal connected	
TEST**Note3**	-	Internal connection	
AV_{DD}	I	Positive analog power supply for PLL	
DV_{DD}	I	Positive digital power supply for PLL	
AV_{SS}	I	Analog ground potential for PLL	

Pin Name	I/O	Function	Alternate
DV_{SS}	I	Digital ground potential for PLL	
V_{DD0} to V_{DD11}	I	Positive power supply	
V_{SS0} to V_{SS12}	I	Ground potential	
JTAG_TDI	I	Data in for debugging or boundary scan	
JTAG_TDO	O	Data out for debugging or boundary scan	
JTAG_TCK	I	Clock input for debugging or boundary scan	
JTAG_TMS	I	Mode select input for debugging or boundary scan	
JTAG_TRST	I	Reset input for debugging or boundary scan	

Notes: **1.** Must be kept low in normal operating mode.
2. Not connected, leave pin open.
3. Must be pulled low always.
4. Must be kept unchanged during operation.

(2) Internal pins between ARM subsystem and UDL

Pin Name	Direction	Function	Alternate
PRESETn	to UDL	APB bus reset signal	
PSELUDL	to UDL	APB bus slave select signal	
PENABLE	to UDL	APB bus strobe signal	
PWRITE	to UDL	APB bus write signal	
PADDR0 to PADDR28	to UDL	APB address bus	
PRDATA0 to PRDATA31	from UDL	APB read data bus	
PWDATA0 to PWDATA31	to UDL	APB write data bus	
FCLK	to UDL	PLL output frequency	
SCLK	from UDL	Clock for ARM subsystem	
INT0 to INT28	from UDL	Interrupt lines to interrupt controller	
nRESUDL	from UDL	Reset signal from UDL to reset controller	
nFIRQ	from UDL	FIQ signal directly routed to ARM CPU	
PLL_M0 to PLL_M4	from UDL	M value for M-counter of PLL	
PLL_N0 to PLL_N6	from UDL	N value for N-counter of PLL	
PLL_S0 to PLL_S1	from UDL	VCO frequency range selection	
TARMSS	from UDL	Set ARM subsystem into test mode	
BS_TCK	to UDL	Boundary scan signals for UDL	
BS_TMS			
BS_TRST			

2.2 Description of Pin Functions

2.2.1 External Pins

(1) External memory bus … Input/Output

These signals are the external memory bus interface to connect SRAM, ROM, Flash or peripheral I/O's.

(a) nXOE (Data Output Enable)... Output

This is the output enable pin for external memory banks, active low.

(b) nXWEN (Write ENable)... Output

This is the write enable pin for external memory banks, active low. A high level means a read enable.

(c) nXWAIT (WAIT) … Input

This control signal input pin can insert a data wait to an external memory bus cycle. Sampling is done at the rising edge of the internal HCLK signal in a bus cycle after 2 WST1 or 3 WST2 states. It is an active low input.

(d) nXBLS0 to nXBLS3 (Byte Lane Select) … Output

These pins are the byte lane strobes for external memory banks, active low.

(e) nXCS0 to nXCS7 (Chip Select) … Output

These pins are the chip select signals for external SRAM, external ROM/Flash, or external peripheral I/O's. The signal nXCSa is assigned to memory banks a (a = 0 to 7). This is active low for the period during which a bus cycle that accesses the corresponding memory bank is activated. It is inactive in an idle state.

(f) XADDR0 to XADDR25 (ADDRess) … Output

These pins output the 26-bit address on the address bus during an external access.

(g) XDATA0 to XDATA31 (DATA)... Input/Output

These pins are the 32-bit data on the data bus during an external access.

(2) Miscellaneous signals... Input/Output

These signals control the ARM subsystem macro.

(a) RXD (Receive Data)... Input

This pin inputs serial receive data of the internal UART.

(b) TXD (Transmit Data)... Output

This pin outputs serial transmit data of the internal UART.

(c) X1, X2 (Crystal)

These pins connect a resonator for system clock generation.
They also can input external clocks. For external clock input, connect to the X1 pin and leave the X2 pin open.

(d) nRST (ReSeT) … Input

The nRST input is an asynchronous input. When a signal with a certain low level width is input asynchronously with the operation clock, a system reset that takes precedence over all operations occurs. Besides a normal initialize or start, this signal is also used to release the pause mode.

(e) TCLK (Test ClocK) … Input

This pin is used to provide the system with a test clock during test purposes only. An external clock is connected to this pin for test purposes, otherwise this pin has to be pulled to low level.

(f) EA0, EA1 (ENable boot memory) … Input

These pins decide, if the system starts from internal ROM or external memory with different bit configurations (8-, 16-, 32-bit).

(g) DBGEN (DeBuG ENable) … Input

This pin enables debug capability of the CPU, if it is active high. The boundary scan capability is enabled during low input of this pin.

(h) PLLOD0, PLLOD1 (Frequency range of PLL)... Input

These pins determine the frequency of the internal PLL output clock.

(3) Debug interface … Input/Output

This is a JTAG interface to provide:

- debug capability of the CPU
- boundary scan of the ARM subsystem

(a) JTAG_TDI (Data In)... Input

This is the test data input signal for debugging purposes.

(b) JTAG_TDO (Data Out)... Output

This is the test data output signal for debugging purposes.

(c) JTAG_TCK (Debug Clock) … Input

This is the debug clock input signal for debugging purposes.

(d) JTAG_TMS (Mode Select) … Input

This is the test mode select input signal for debugging purposes.

(e) JTAG_TRST (Debug Reset) … Input

This is the debug reset input signal.

(4) Power pins... Input/Output

(a) AV_{DD} (Power supply for PLL)

This is the positive analog power supply pin for the PLL.

(b) DV_{DD} (Power supply for PLL)

This is the positive digital power supply pin for the PLL.

(c) AV_{SS} (Ground for PLL)

This is the analog ground pin for the PLL.

(d) DV_{SS} (Ground for PLL)

This is the digital ground pin for the PLL.

(e) V_{DD0} to V_{DD11} (Power supply)

These are the positive power supply pins for the SoCLite chip.

(f) V_{SS0} to V_{SS12} (Ground)

These are the ground pins for the SoCLite chip.

2.2.2 Internal Pins between ARM subsystem and UDL

(1) APB bus interface … to UDL/from UDL

(a) PRESETn (Reset)... to UDL
This is the APB bus reset signal.

(b) PSELUDL (Slave Select)... to UDL
This is the APB bus slave select signal for the UDL area.

(c) PENABLE (Strobe) … to UDL
This is the APB bus strobe signal to time all peripheral accesses.

(d) PWRITE (Write) … to UDL
This is the APB bus write signal. During an active high level this signal indicates a write access and during a low level it indicates a read access.

(e) PADDR0 to PADDR28 (Address) … to UDL
These signals output the 29-bit address on the APB address bus during an APB access.

(f) PRDATA0 to PRDATA31 (Read Data) … from UDL
These signals are the 32-bit data on the read data bus during an APB read access.

(g) PWDATA0 to PWDATA31 (Write Data)... to UDL
These signals are the 32-bit data on the write data bus during an APB write access.

(2) Clock signals … to UDL/from UDL

(a) FCLK (PLL Output Clock)... to UDL
This is the PLL output clock which is fed into the UDL.

(b) SCLK (System Clock)... from UDL
This is the clock for the ARM subsystem which comes from the UDL. The internal system clock (HCLK) of the ARM subsystem is diverted from the system clock (SCLK).

(3) Miscellaneous signals... to UDL/from UDL

(a) INT0 to INT28 (Interrupt)... from UDL
These are the interrupt signals from the UDL to the interrupt controller in the ARM subsystem.

(b) nRESUDL (Reset)... from UDL
This is the reset signal from the UDL to the reset controller in the ARM subsystem to provide reset capability from UDL area.

(c) nFIRQ (Fast Interrupt) … from UDL
This is the fast interrupt signal from the UDL directly routed to the ARM subsystem CPU.

(d) PLL_M0 to PLL_M4 (M-Value) … from UDL
These are the M value signals for the M-counter of the PLL in the ARM subsystem.

(e) PLL_N0 to PLL_N6 (N-Value) … from UDL
These are the N value signals for the N-counter of the PLL in the ARM subsystem.

(f) PLL_S0, PLL_S1 (S-Value) … from UDL
These are the S value signals for the VCO frequency range selection.

(g) TARMSS (Test Mode) … from UDL
This signal sets the ARM subsystem into test mode.

2.3 Types of Pin I/O Circuit and Connection of Unused Pin

Pin	I/O Buffer Type[Note]	Recommended Connection
nXOE	BV08	
nXWEN		
nXWAIT	FIV1	
nXBLS0	BV08	
nXBLS1		
nXBLS2	BW03	
nXBLS3		
nXCS0	BV08	
nXCS1		
nXCS2		
nXCS3		
nXCS4		
nXCS5		
nXCS6		
nXCS7		
XADDR0 to XADDR25		
XDATA0 to XDATA31	BW03	
RXD	FIV1	
TXD	BV08	
X1	OSI1 / OSO1	
X2		
TCLK	FIV1	
nRST	FIF1	Schmitt-Trigger input
EA0	FIV1	
EA1		
DBGEN	FDV1	internal pull-down (50K)
PLLOD0	FIV1	
PLLOD1		
JTAG_TDI		
JTAG_TDO	BV08	
JTAG_TCK	FIV1	
JTAG_TMS		
JTAG_TRST		
AV_{DD}	-	Default power supply (analog)
DV_{DD}	-	Default power supply (digital)
AV_{SS}	-	Default ground (analog)
DV_{SS}	-	Default ground (digital)
V_{DD0} to V_{DD11}	-	Default power supply
V_{SS0} to V_{SS12}	-	Default ground

Note: For buffer type information, please refer to document “EA-9HD Family Block Library”.

[MEMO]

Chapter 3 CPU Function

The CPU of the SoCLite is based on a ARM7TDMI processor core with a RISC architecture.

Therefore refer to Document “ARM7TDMI Data Sheet” (ARM DDI 0029E)[Note].

Note: All other product, brand, or trade names used in this publication are the trademarks or registered trademarks of their respective trademark owners.

[MEMO]

Chapter 4 Memory Map

4.1 Operation Modes

4.1.1 Operation Modes

The SoCLite has the following operation modes. Mode specification is carried out by the EA0 and EA1 pins and the RM bit in the Remap/Pause control register (see Chapter 12).

(1) Normal Operation Mode

Access to the internal boot ROM is disabled.
In normal operation mode, after the system reset is cleared, program execution branches to the external device's (memory) reset entry address (00000000H) and instruction processing starts.
The external memory area is mapped from address 00000000H.

(2) Boot Mode

Access to the internal boot ROM is enabled.
The internal boot ROM contains a bootloader routine to allow a download of application software via the internal UART.
This boot mode offers the possibility to download a programming routine for an external programmable memory device (like Flash-ROM) into internal or external RAM for in-system programming.

(3) Remapped Operation Mode

Access to the internal boot ROM is disabled.
The remapped operation mode can be entered from boot mode or normal operation mode only. In this mode the internal RAM is relocated to the bottom address area of the memory map from 00000000H to 00001FFFH.
This mode is used to place the vector table located at the bottom of the memory map into the internal RAM to offer the possibility of modifying the vector table by the application code.

4.1.2 Operation Mode Specification

The operation mode is specified according to the status of pins EA0, EA1 and RM bit of the Remap/ Pause controller. After a reset the system starts either in boot mode or in normal operation mode according to the setting of pins EA0 and EA1. In an application system the input levels of these pins must be fixed and are not allowed to change during operation. Operation is not guaranteed if these pins are changed during operation.

Table 4-1: Operation Modes

RM bit	EA1	EA0	Operation Mode		Remark
L	L	L	Boot Mode	Internal boot ROM is allocated from address 00000000H.	
L	L	H	Normal Operation Mode	External memory is allocated from address 00000000H.	8-bit data bus
L	H	L			16-bit data bus
L	H	H			32-bit data bus
H	X	X	Remapped Operation Mode	Internal RAM is allocated from address 00000000H.	For Internal RAM allocation at other operation modes, please refer to memory map.

Remark: L:Low-level input
H:High-level input

The remapped operation mode can only be entered from boot or normal operation mode. The RM bit in the Remap/Pause control register is default low after a system reset. Therefore after a system reset the system can only be started either in boot mode or in normal operation mode according to EA0 and EA1 pins.

(1) Bootloader Routine

The bootloader routine contains a software, which configures the SoCLite when starting in boot mode to allow in-system programming. The following procedure has to be executed to provide in-system Flash-ROM programming:

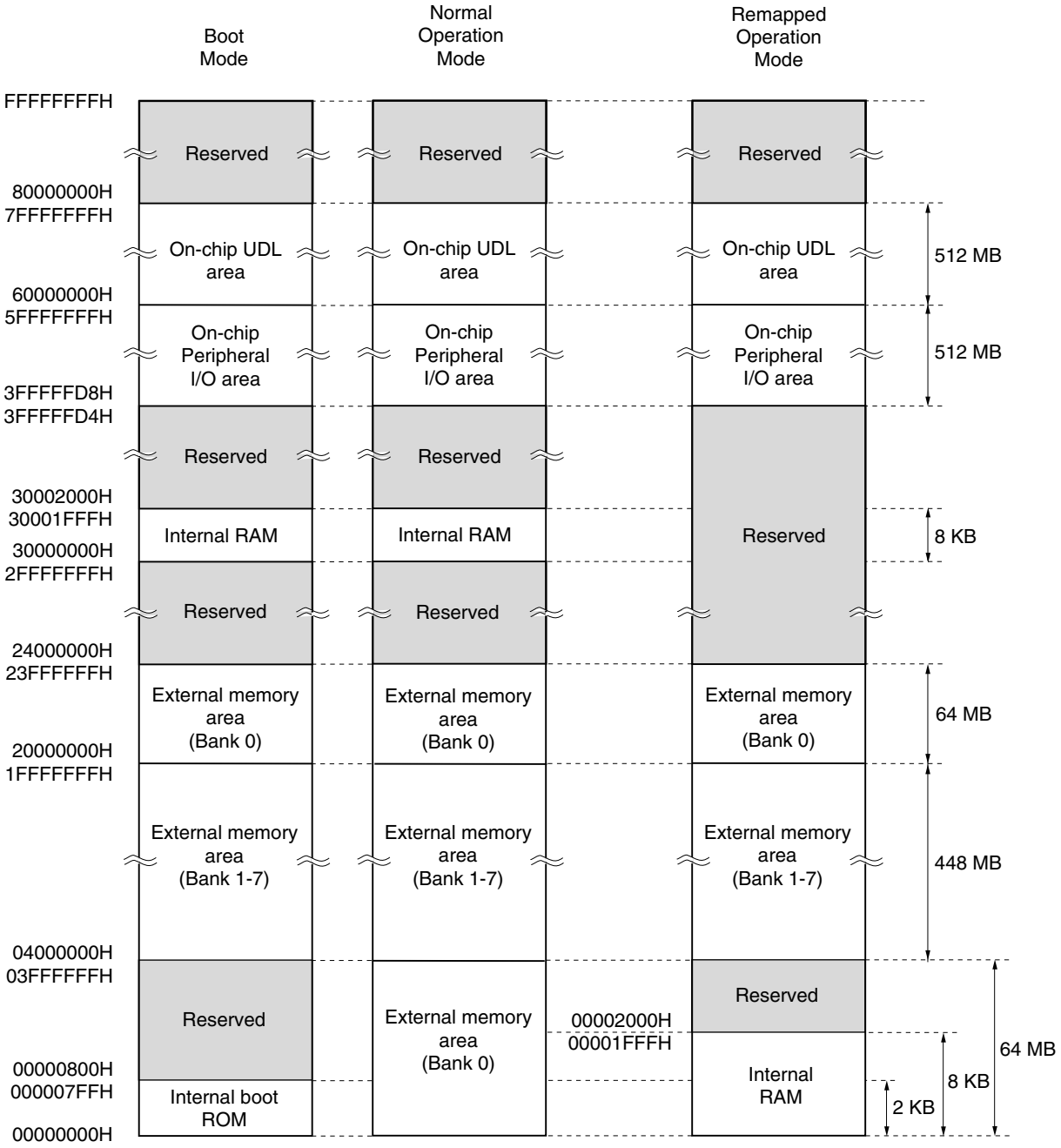
- Set-up the SoCLite for boot mode.
- Connect the SoCLite application board to a personnel computer (PC) by the serial interface (UART).
- Start a terminal program on the PC to control the SoCLite application board.
- Synchronization of the serial interface of SoCLite is done automatically with the serial interface of the PC.
- Download of an application program for in-system Flash programming via the terminal program to the internal or external RAM.
- Download of the customer application program to be flashed.

4.2 Address Space

4.2.1 Memory Map

The SoCLite reserves memory areas as shown in Figure 4-1. The operation modes are specified by the EA0, EA1 pins and RM bit of Remap/Pause control register.

Figure 4-1: Memory Map of SoCLite



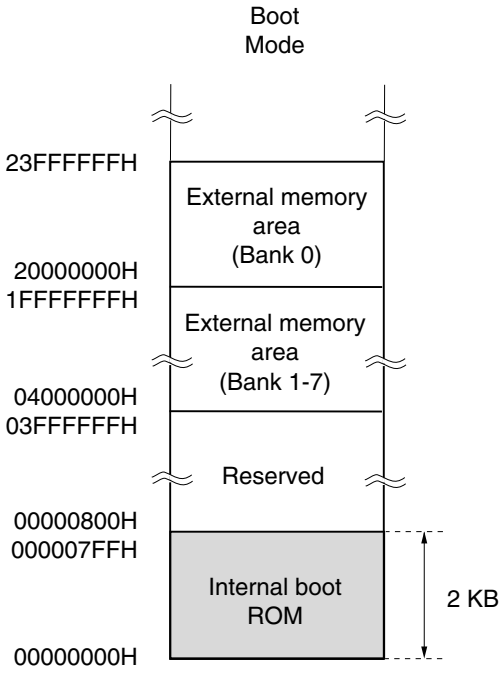
Remark: Access to reserved memory area is forbidden.

4.2.2 Area

(1) Internal ROM Area

2 KB of internal ROM area (00000000H to 000007FFH) are assigned in boot mode only.
The internal ROM area contains a bootloader routine to allow a download of application software for in-system Flash programming via the UART to any RAM area (internal or external). If the boot mode is set (external pins EA[1:0] = 00), the SoCLite starts after reset with the execution of the bootloader routine.

Figure 4-2: Internal ROM area

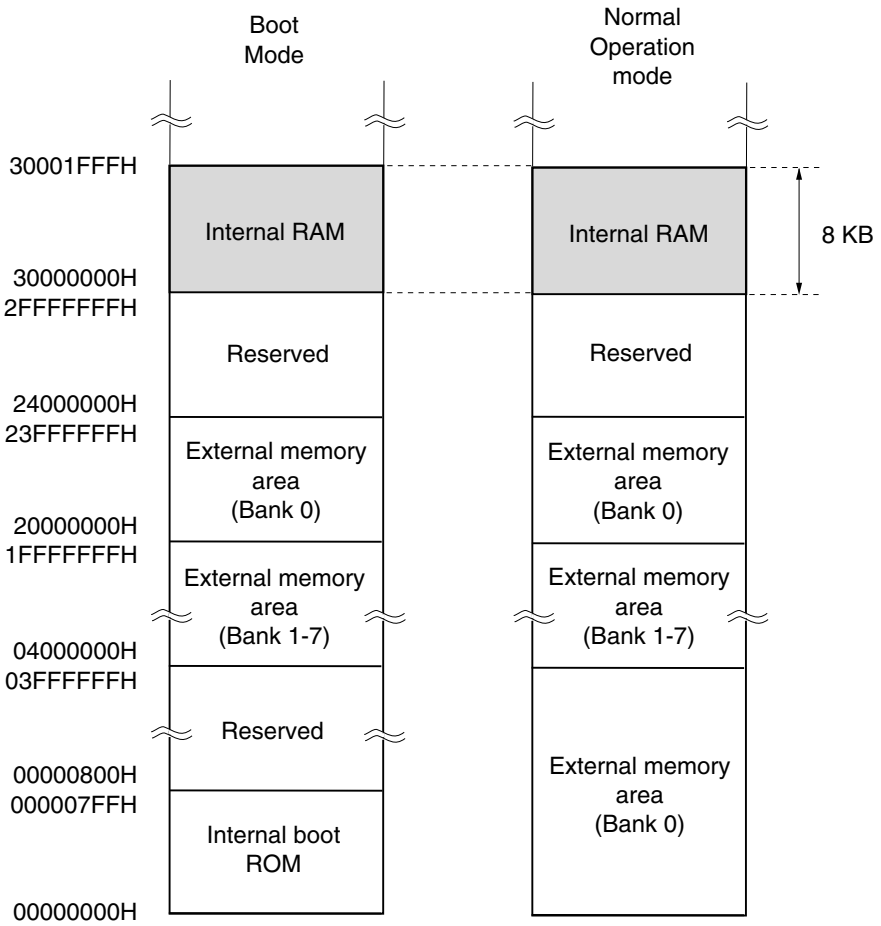


Remark: Access to reserved memory area is forbidden.

(2) Internal RAM Area

8 KB of memory, addresses 30000000H to 30001FFFH, are assigned for the internal RAM area. The internal RAM area can be remapped to addresses 00000000H to 00001FFFH to place the vector table into the internal RAM.

Figure 4-3: Internal RAM area without remapping

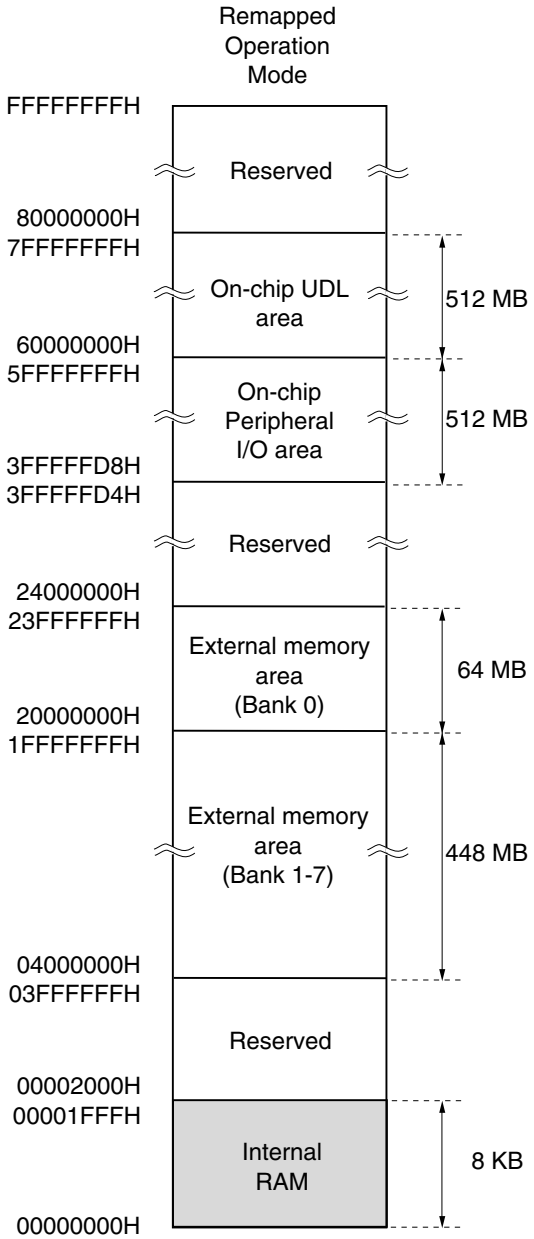


Remark: Access to reserved memory area is forbidden.

(3) Memory Area after Remapping

The internal RAM can be located at address 00000000H, e.g. to place variable interrupt tables in RAM. The remapping of the RAM is controlled by the Remap/Pause controller (see Chapter 12).

Figure 4-4: Memory area after remapping



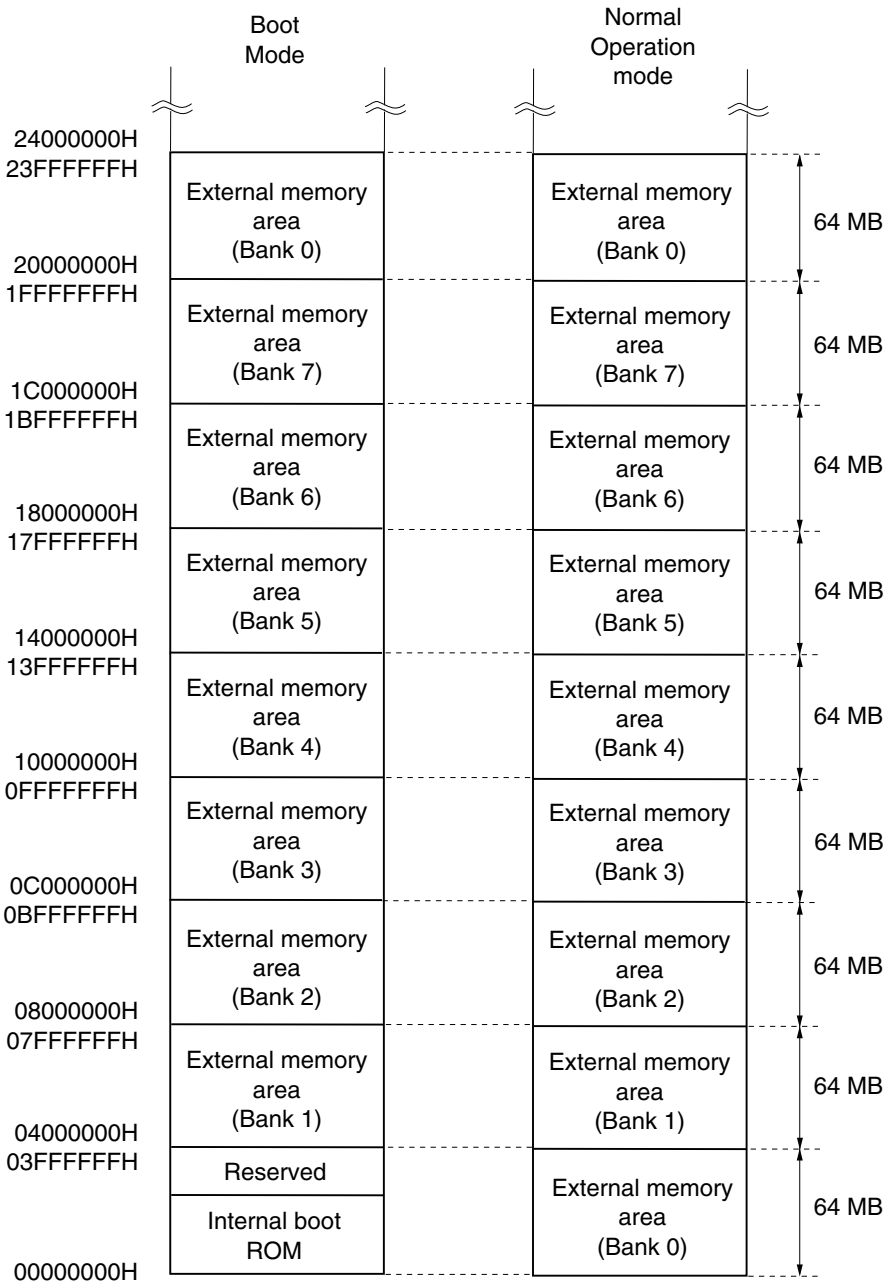
Remark: Access to reserved memory area is forbidden.

(4) External Memory Area

512 MB are available for external memory space, divided into eight banks of 64 MB (bank 0 to 7) to be used for program/data code. The build-in memory controller generates bank related chip select signals.

In normal operation mode a mirror of bank 0 is located at address 00000000H, where the reset vector is located (reset vector at address 00000000H).

Figure 4-5: External memory area

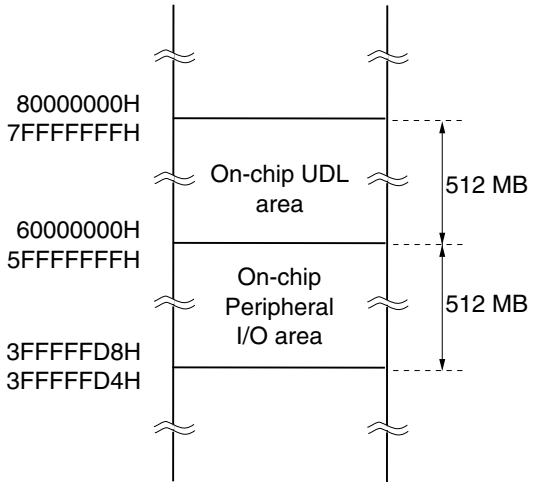


Remark: Access to reserved memory area is forbidden.

(5) On-Chip Peripheral I/O and UDL Area

512 MB of memory is provided as an on-chip peripheral I/O and UDL area each.
The on-chip peripheral I/O area is located from address 3FFFFFD8H to 5FFFFFFFH and the UDL area is located from address 60000000H to 7FFFFFFFH.

Figure 4-6: On-chip peripheral I/O and UDL area



Peripheral I/O registers associated with the operation mode specification and the state monitoring for the on-chip peripherals I/O are all memory mapped to the on-chip peripheral I/O area. Program fetches cannot be executed from this area.

Cautions: 1. In the SoCLite only word accesses are performed to the registers.
2. For registers in which byte access is possible, the higher 24 bits become undefined during the read operation, and the lower 8 bits of data are written to the register during the write operation.
3. For registers in which half-word access is possible, the higher 16 bits become undefined during the read operation, and the lower 16 bits of data are written to the register during the write operation.
4. Addresses that are not defined as registers are reserved for future expansion. If these addresses are accessed, the operation is undefined and not guaranteed.

4.2.3 Interrupt/Exception Table

The SoCLite increases the interrupt response time by assigning handler addresses corresponding to interrupts/exceptions.
The collection of these handler addresses is called an interrupt/exception table, which starts at 0H in different memory locations depending on the operation mode.

Table 4-2: Interrupt/Exception Table in Memory Areas

Operation Mode	Memory Area of Interrupt/Exception Table
Boot Mode	Internal boot ROM
Normal Operation Mode	External memory (bank 0)
Remapped Operation Mode	Internal RAM

When an interrupt/exception request is accepted, program execution jumps to the given handler address, and the interrupt/exception program is executed.
Table 4-3 shows the sources of interrupts/exceptions, and the corresponding addresses.

Table 4-3: Interrupt/Exception Table

Start Address of Interrupt/ Exception Table	Interrupt/Exception Source
00000000H	RESET
00000004H	Undefined Instruction
00000008H	Software Interrupt
0000000CH	Prefetch Abort
00000010H	Data Abort
00000014H	Reserved
00000018H	IRQ (interrupt)
0000001CH	FIRQ (fast interrupt)

4.2.4 Peripheral I/O Registers

Table 4-4: List of Peripheral I/O Registers

Address	Function Register Name	Symbol	Size	R/W	Initial Value
3FFFFFD8H	Write protection control register	WPC	32 bit	R/W	00000000H
3FFFFFDCH	Write protection status register	WPS	32 bit	R/W	00000000H
3FFFFFE0H	SMC bank0 configuration register	SMCBCR0[Note]	32 bit	R/W	2000FBEFH
3FFFFFE4H	SMC bank1 configuration register	SMCBCR1	32 bit	R/W	2000FBEFH
3FFFFFE8H	SMC bank2 configuration register	SMCBCR2	32 bit	R/W	1000FBEFH
3FFFFFECH	SMC bank3 configuration register	SMCBCR3	32 bit	R/W	0000FBEFH
3FFFFFF0H	SMC bank4 configuration register	SMCBCR4	32 bit	R/W	2000FBEFH
3FFFFFF4H	SMC bank5 configuration register	SMCBCR5	32 bit	R/W	2000FBEFH
3FFFFFF8H	SMC bank6 configuration register	SMCBCR6	32 bit	R/W	1000FBEFH
3FFFFFFCH	SMC bank7 configuration register	SMCBCR7	32 bit	R/W	0000FBEFH
40000000H	Interrupt control0 register	INTC0	32 bit	R/W	07070707H
40000004H	Interrupt control1 register	INTC1	32 bit	R/W	07070707H
40000008H	Interrupt control2 register	INTC2	32 bit	R/W	07070707H
4000000CH	Interrupt control3 register	INTC3	32 bit	R/W	07070707H
40000010H	Interrupt control4 register	INTC4	32 bit	R/W	07070707H
40000014H	Interrupt control5 register	INTC5	32 bit	R/W	07070707H
40000018H	Interrupt control6 register	INTC6	32 bit	R/W	07070707H
4000001CH	Interrupt control7 register	INTC7	32 bit	R/W	07070707H
40000020H	Interrupt status register	INTSTA	32 bit	R/W	00000000H
40000024H	Interrupt source register	INTSRC	8 bit	R	00H
40000028H	Interrupt source register for emulation	INTSRCEMU	8 bit	R	00H
4000002CH	Interrupt status register for emulation	INTISSEMU	32 bit	R	00000000H
40000040H	Serial reception data buffer register	SRXB	8 bit	R	00H
40000044H	Serial transmission data buffer register	STXB	8 bit	R/W	00H
40000048H	Serial status register	SSTA	8 bit	R/W	00H
4000004CH	Serial control register	SCTR	32 bit	R/W	00000000H
40000050H	Serial reception data buffer register for emulation	SRXBEMU	8 bit	R	00H
40000060H	Timer value register	TMV	32 bit	R/W	FFFFFFFFH
40000064H	Timer load register	TML	32 bit	R/W	FFFFFFFFH
40000068H	Timer control register	TMC	8 bit	R/W	00H
40000070H	Remap/Pause control register	RMP	8 bit	R/W	00H
40000074H	Watchdog timer control register	WDCTRL	16 bit	R/W	1000H
40000078H	APB bus wait state control register	APBW	8 bit	R/W	00H

Note: EA[1:0] pins overwrite the MW bits in the SMC bank0 configuration register at reset.

4.3 Specific Register

The SoCLite contains one specific register, the watchdog timer control register (WDCTRL), which is protected from being written with illegal data due to erroneous program execution. The write access to this specific register is allowed with a specific bit pattern only. If an abnormal access occurs, no write access to this register is executed.
For details of the WDCTRL register, refer to Chapter 10 “Watchdog Timer”.

The specific bit pattern “10_b” must be placed on bit position bit[15:14] of the WDCTRL register to write a watchdog timer reload value and to enable the watchdog.

[MEMO]

Chapter 5 External Bus Interface

The SoCLite contains an external bus interface to connect external memories such as ROM and RAM, and I/O. The external bus interface is controlled by the on-chip static memory controller (SMC).

5.1 Features

- 32-bit/16-bit/8-bit data bus sizing function
- 8 Chip Selects
  - 8 chip select signals externally available (nXCS0 to nXCS7)
  - Each chip select represents one memory bank of 64 MB
- Wait function
  - Programmable wait function, capable of inserting up to 32 wait states for each memory bank
  - External wait function through nXWAIT pin
- Idle state insertion function
- Write protection for each memory bank (different from write protection of internal RAM)

5.2 Bus Control Pins

The following pins are available for connecting external devices.

Table 5-1: Bus Control Pins

Bus Control Pin	Direction in Normal state	Direction in RESET state
Address bus (XADDR0 to XADDR25)	Output	Output (driven low)
Data bus (XDATA0 to XDATA31)	Input/Output	Input
Chip select (nXCS0 to nXCS7)	Output	Output (driven high)
Byte lane control (nXBLS0 to nXBLS3)	Output	Output (driven high)
Data output enable (nXOE)	Output	Output (driven high)
Write/read control (nXWEN)	Output	Output (driven high)
External wait control (nXWAIT)	Input	Input

Remark: Signal level high means logical “1” and signal level low means logical “0”.

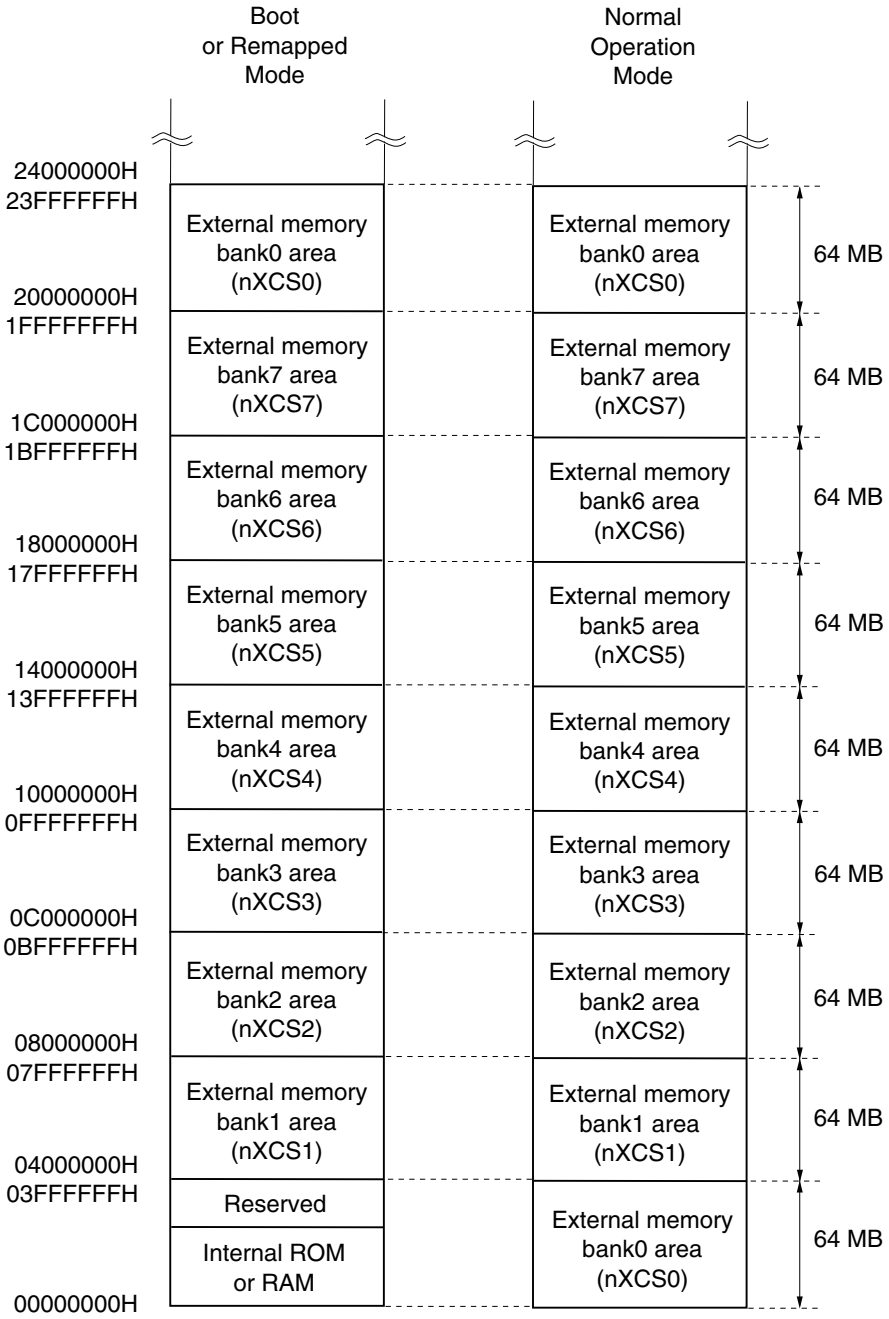
5.2.1 Pin status during internal ROM/RAM and on-chip peripheral I/O access

While accessing internal ROM/RAM and on-chip peripheral I/O the address bus and the data bus retain the value from the last access on the external memory bus. The control signals of the external memory bus are all driven high (inactive, logical “1”).
During the reset phase until the first external memory access the address bus is driven low (logical “0”) and the control signals are all driven high (inactive, logical “1”) and the data bus becomes input.

5.3 Memory Bank Function

The 512 MB external memory space is divided into eight memory banks of 64 MB units each.
Each external memory bank can be configured independently by the SMC bank configuration registers.

Figure 5-1: Memory Bank Function



5.4 Memory Bank Control Function

In the SoCLite, the following external devices can be connected directly to each memory bank.

- external SRAM, external ROM, external I/O
- external Flash
- external page and burst mode ROM (asynchronous access)

Connected external devices are specified by the SMC bank configuration registers 0 to 7 (SMCBCR0 to SMCBCR7). These eight 32-bit read and write registers are used to configure each memory bank independently.

5.4.1 External Memory Bank Configuration

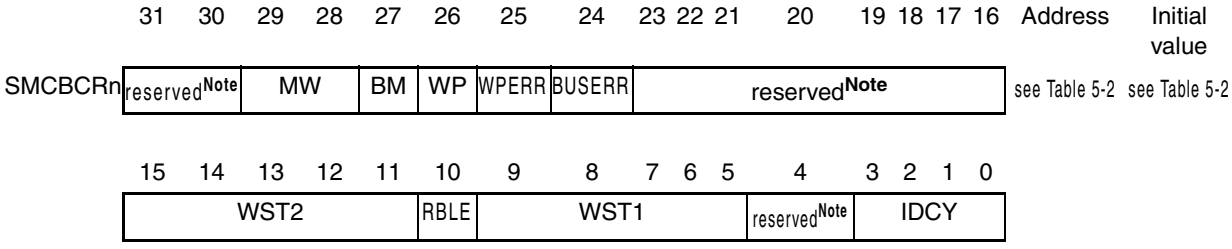
(1) SMC Bank Configuration Registers 0 to 7(SMCBCR0 to SMCBCR7)

These registers can be read/written in 32-bit units

Table 5-2: SMC Bank Configuration Register Assignment

Register Name	Address	Initial Value
SMCBCR0	3FFFFFE0H	2000FBEFH
SMCBCR1	3FFFFFE4H	2000FBEFH
SMCBCR2	3FFFFFE8H	1000FBEFH
SMCBCR3	3FFFFFECH	0000FBEFH
SMCBCR4	3FFFFFF0H	2000FBEFH
SMCBCR5	3FFFFFF4H	2000FBEFH
SMCBCR6	3FFFFFF8H	1000FBEFH
SMCBCR7	3FFFFFFCH	0000FBEFH

Figure 5-2: SMC Bank Configuration Register (SMCBCRn) (n = 0 to 7) (1/3)



Note: Always write 0, unpredictable when read.

Figure 5-2: SMC Bank Configuration Register (SMCBCRn) (2/3)

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>31 to 30</td><td>-</td><td>Reserved</td></tr>
<tr><td>29 to 28</td><td>MW</td><td>Memory Width
Specifies the memory bus width of the external memory bank.
<table><tr><th>MW</th><th>Memory width</th></tr><tr><td>00_b</td><td>8-bit</td></tr><tr><td>01_b</td><td>16-bit</td></tr><tr><td>10_b</td><td>32-bit</td></tr><tr><td>11_b</td><td>reserved</td></tr></table>
Remark: The MW-bits for the external memory bank0 are overwritten by the EA[1:0] bits at reset.</td></tr>
<tr><td>27</td><td>BM</td><td>Burst Mode
Selects the burst mode operation for external ROM devices.
<table><tr><th>BM</th><th>Operation mode</th></tr><tr><td>0</td><td>non-burst devices (default at reset)</td></tr><tr><td>1</td><td>burst ROM (asynchronous)</td></tr></table></td></tr>
<tr><td>26</td><td>WP</td><td>Write Protect
Selects write protection for each external memory bank.
<table><tr><th>WP</th><th>Operation mode</th></tr><tr><td>0</td><td>Memory bank not write protected (default at reset)</td></tr><tr><td>1</td><td>Memory bank write protected</td></tr></table></td></tr>
<tr><td>25</td><td>WPERR</td><td>Write Protection Error Flag
Controls the write protect error status flag.
<table><tr><th>WPERR</th><th>Write protect status</th></tr><tr><td>0</td><td>no write protect error (default at reset)</td></tr><tr><td>1</td><td>Write protect error
Writing a ’1’ to this bit will clear the write protect error status flag</td></tr></table></td></tr>
<tr><td>24</td><td>BUSERR</td><td>Bus Transfer Error Flag
Controls the bus transfer error status flag.
<table><tr><th>BUSERR</th><th>Bus transfer error status</th></tr><tr><td>0</td><td>no bus transfer error (default at reset)</td></tr><tr><td>1</td><td>Bus transfer error
Writing a ’1’ to this bit will clear the bus transfer error status flag</td></tr></table></td></tr>
<tr><td>23 to 16</td><td>-</td><td>Reserved</td></tr>
</table>

Figure 5-2: SMC Bank Configuration Register (SMCBCRn) (3/3)

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>15 to 11</td><td>WST2</td><td>Wait state 2
Specifies the wait states for write access (SRAM) and burst access time (burst ROM).
<table><tr><th>WST2</th><th>Wait state 2</th></tr><tr><td>$XXXXX_b$</td><td>WST2 is the write access time in the case of SRAM and the burst access time in the case of burst ROM.
This wait state time is (WST2 + 1) * f_{CPU} in the case of SRAM or (WST2) * f_{CPU} in the case of burst ROM.
WST2 = 11111_b (default at reset)</td></tr></table></td></tr>
<tr><td>10</td><td>RBLE</td><td>Read Byte Lane Enable
Defines byte lane enable signal for read access.
<table><tr><th>RBLE</th><th>Bus transfer error status</th></tr><tr><td>0</td><td>All byte lane strobes nXBLS[3:0] held HIGH during system reads from external memory (default at reset).</td></tr><tr><td>1</td><td>All byte lane strobes nXBLS[3:0] held LOW during system reads from external memory.</td></tr></table></td></tr>
<tr><td>9 to 5</td><td>WST1</td><td>Wait State 1
Specifies the wait states for read access (SRAM/ROM) and the initial access time (burst ROM).
<table><tr><th>WST1</th><th>Wait state 1</th></tr><tr><td>$XXXXX_b$</td><td>WST1 is the read access time in the case of SRAM and ROM, or the initial access time in the case of burst ROM.
This wait state time is (WST1 + 1) * f_{CPU}.
WST1 = 11111_b (default at reset)</td></tr></table></td></tr>
<tr><td>4</td><td>-</td><td>Reserved</td></tr>
<tr><td>3 to 0</td><td>IDCY</td><td>Idle Cycle Turn Around Time
Specifies the idle cycle turn around time.
<table><tr><th>IDCY</th><th>Idle cycle turn around time</th></tr><tr><td>$XXXX_b$</td><td>IDCY is the idle cycle external memory data bus turn around time.
The turn around time is (IDCY + 1) * f_{CPU}.
IDCY = 1111_b (default at reset)</td></tr></table></td></tr>
</table>

Cautions: 1. The internal ROM area and internal RAM area are not subject to these programmable waits (WST1 and WST2) and ordinarily no wait access is carried out. The on-chip peripheral I/O area is also not subject to these programmable wait states.

2. Write to the SMCBCR 0 to 7 registers after reset, and then do not change the set values. Also, do not access an external memory bank area other than that for this initialization routine until initial setting of the SMCBCR 0 to 7 registers is finished. However, it is possible to access external memory bank areas whose initialization has been finished.

(2) Bus width of external memory banks at reset

Table 5-3 shows the default memory width of the external bus interface at reset.

Table 5-3: External memory bus width at reset

External memory bank	Default memory width
Bank0	Depends on external EA[1:0] pins: EA[1:0] pins: 00 → internal ROM; 01 → 8-bit; 10 → 16-bit; 11 → 32-bit
Bank1	32-bit
Bank2	16-bit
Bank3	8-bit
Bank4	32-bit
Bank5	32-bit
Bank6	16-bit
Bank7	8-bit

The external memory bus width at reset is determined by the external EA[1:0] pins.

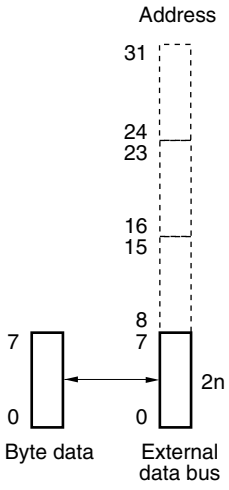
5.5 Bus Width

The SoCLite accesses peripheral I/O and external memory in 8-bit, 16-bit, or 32-bit units. The following shows the operation for each type of access. Access all data in order starting from the lower order side. The SoCLite is in little endian configuration only.
The SoCLite is not provided with an address misalign function, therefore all operations must be aligned to the specific access type (in case of halfword or word accesses).

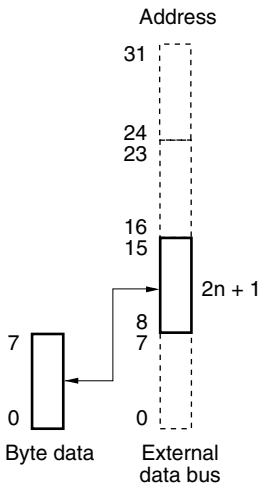
5.5.1 Byte Access (8 bits)

(1) When the bus width is 32 bits

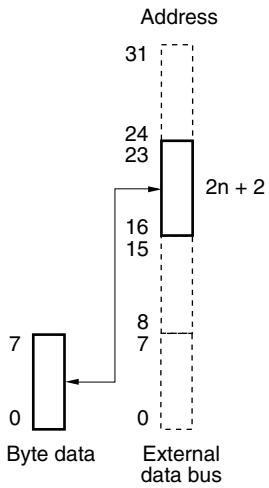
<1> Access to even address (2n)



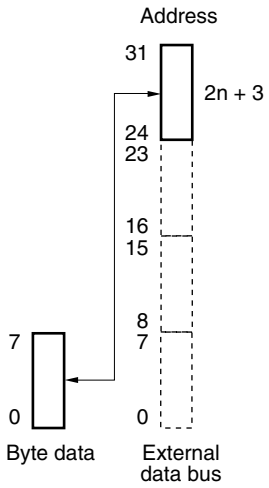
<2> Access to odd address (2n + 1)



<3> Access to even address (2n + 2)

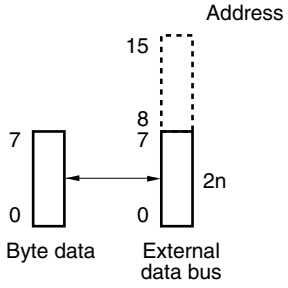


<4> Access to odd address (2n + 3)

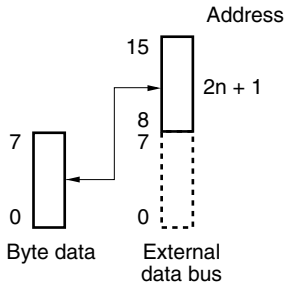


(2) When the bus width is 16 bits

<1> Access to even address (2n)

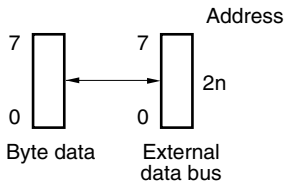


<2> Access to odd address (2n + 1)

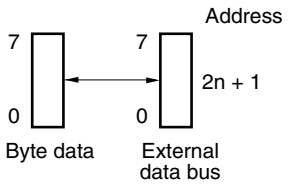


(3) When the bus width is 8 bits

<1> Access to even address (2n)



<2> Access to odd address (2n + 1)

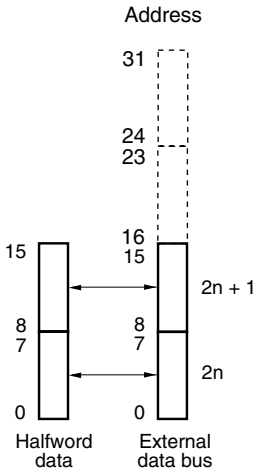


5.5.2 Halfword Access (16 bits)

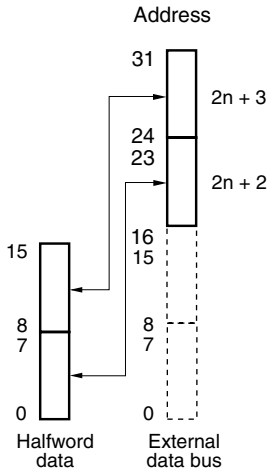
Independent of the external memory bus width, halfword accesses ***must be*** halfword aligned.

(1) When the bus width is 32 bits

<1> Access to even address (2n)

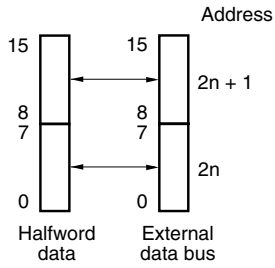


<2> Access to even address (2n + 2)



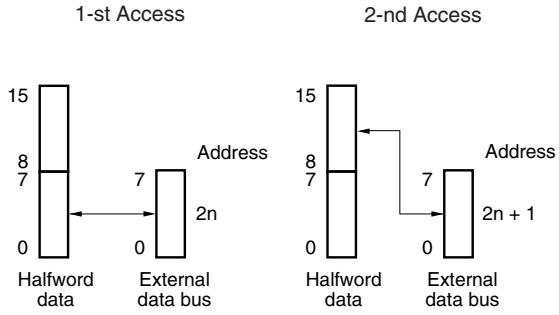
(2) When the bus width is 16 bits

<1> Access to even address (2n)



(3) When the data bus width is 8 bits

<1> Access to even address (2n)

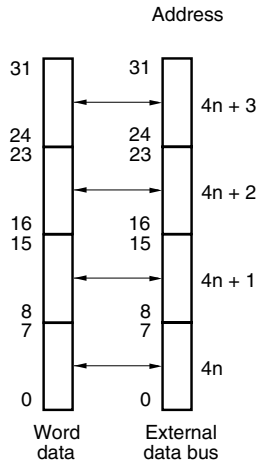


5.5.3 Word Access (32 bits)

Independent of the external memory bus width, word accesses ***must be*** word aligned.

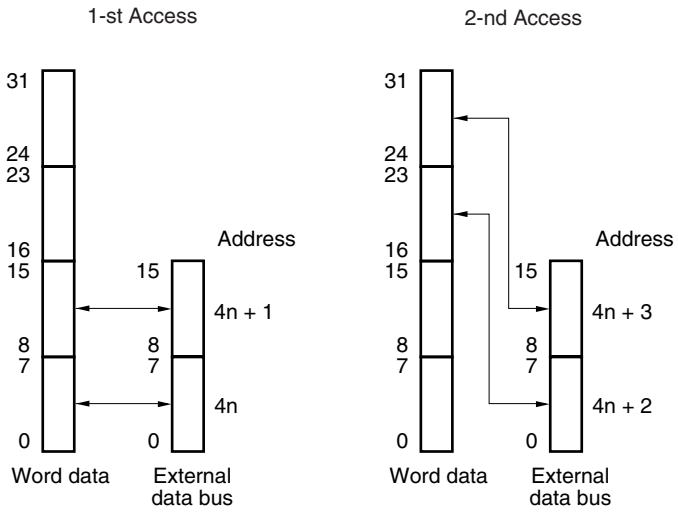
(1) When the bus width is 32 bits

<1> Access to address 4n



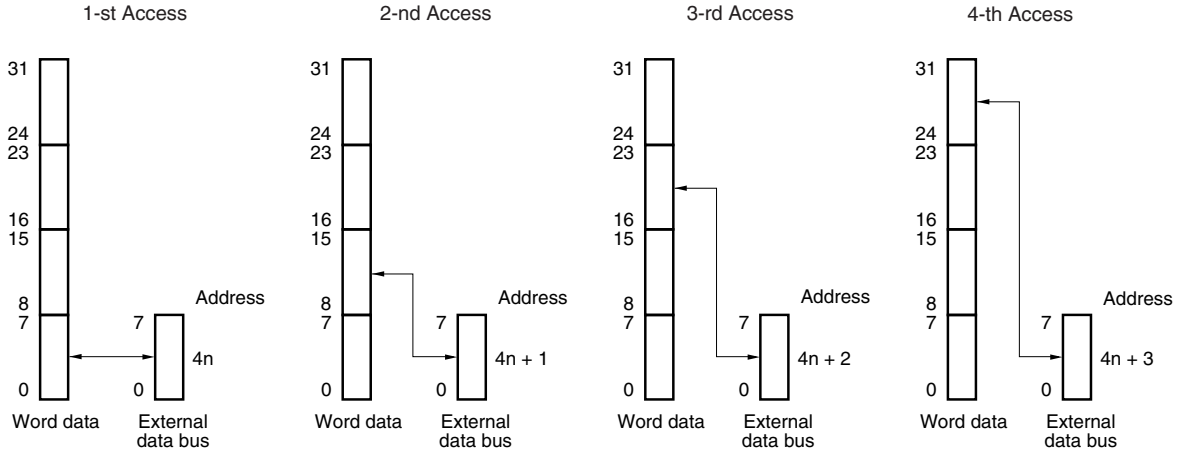
(2) When the bus width is 16 bits

<1> Access to address 4n



(3) When the bus width is 8 bits

<1> Access to address 4n



5.5.4 Access Sequence

The data width of each external memory bank is configured by programming the appropriate SMC bank configuration register (SMCBCR0 to SMCBCR7). The access sequence is dependent on the external memory bus width, e.g. the external memory bus is narrower than the transfer initiated from the internal system bus master, the internal system bus transfer will take several external bus transfers to complete. For example, in the case that bank 0 is configured as 8-bit wide memory and a 32-bit read access is initiated, the internal system bus will stall while the external bus interface reads four consecutive bytes from the memory. During these accesses the data path is controlled to demultiplex the four bytes into one 32-bit word on the internal system bus.

The data-out path allows 32-bit internal system bus writes to be converted into several external memory halfword or byte writes. The separate byte lane control signals during writes ensure that the entire memory bus is driven at all times, independent which width of memory is being accessed.
The data-in path constructs 32-bit internal system bus data words from halfword and byte wide external memory devices.

Only little-endian operation access is supported in the SoCLite.

5.6 Wait Function

5.6.1 Programmable Wait Function

To facilitate interfacing with low-speed memory or with I/Os, it is possible to insert up to 32 data wait states with respect to the starting bus cycle for each external memory bank area.
The number of wait states can be specified by program using the WST1 bits for read accesses and the WST2 bits for write accesses in the SMC bank configuration registers 0 to 7 (SMCBCR0 to SMCBCR7).
Just after system reset, all external memory banks have 32 data wait states inserted.

5.6.2 External Wait Function

When a slow device, I/O, or asynchronous system is connected externally, any number of wait states can be inserted in a bus cycle by the external wait input signal nXWAIT to extend the number of wait states specified by the WST1 and WST2 bits in the SMCBCRn (n = 0 to 7) registers. This signal is used for slow external peripheral devices to stretch read and write transfers on the external bus interface.
There exist only one nXWAIT signal on the external bus interface, which influences the timing of all eight memory banks simultaneously.
Accesses to internal ROM, internal RAM, and internal peripheral I/O areas cannot be controlled by external waits.

A change of the external nXWAIT signal is recognized two clocks later in the system due to the need of two clock cycles for synchronization. In case of nXWAIT is pulled low within the same clock cycle as nXCS, the WST1 and WST2 bits in the SMCBCRn (n = 0 to 7) registers have to be set according to Table 5-4 to guarantee recognition of nXWAIT.

Table 5-4: Internal settings of WST1/2 for nXWAIT recognition

Register bit (SMCBCRn)	Value	Effects
WST1	≥ 2	Read transfer
WST2	≥ 3	Write transfer

Remark: If nXWAIT is not pulled low 2 clock cycles before the WST1/WST2 wait cycles have been elapsed, nXWAIT will not be considered for wait state insertion.

In case the external wait signal nXWAIT is active two clock cycles before a read or write access is executed, it will be recognized regardless of WST1 and WST2 settings.
Internal wait states, which have not been elapsed before nXWAIT is asserted, are added after the external nXWAIT signal is deasserted.

5.7 Idle Cycle Turn Around Insertion Function

To ease interfacing with low-speed memory devices, idle turn around cycles can be inserted between read and write external memory accesses to avoid bus contention on the external memory data bus. The number of idle turn around cycles can be specified by program using the IDCY bit in the SMC bank configuration registers 0 to 7 (SMCBCR0 to SMCBCR7).
It is possible to insert up to 16 idle turn around cycles for each external memory bank area. At least one idle turn around cycles is inserted all the time.

An idle state is inserted after read/write cycles for SRAM, external I/O, or external ROM.
Table 5-5 shows idle turn around cycles that are generated between external bus transfers.

Table 5-5: Idle turn around cycles

External Bus Transfer	Idle Turn Around Cycles
Read to write access to same memory bank	Generated
Read to write access to different memory bank	Generated
Write to write access to same memory bank	Not generated
Write to write access to different memory bank	Generated
Write to read access to same memory bank	Not generated
Write to read access to different memory bank	Generated

Immediately after the system reset, idle turn around cycle insertion is automatically programmed for all memory banks to the maximum value of 16 idle turn around cycles.

5.8 Byte Lane Control

The SoCLite external bus interface provides byte lane control signals nXBLS[3:0] to control accesses to external memory banks constructed from different memory devices.
Each external memory bank can either be 8, 16 or 32 bits wide. The type of memory used to configure a particular memory bank determines how the nXWEN and nXBLS[3:0] signals are connected to provide byte, halfword and word access. For read access, it is necessary to control the nXBLS[3:0] signals by driving them either all high or all low.
This is achieved by programming the RBLE bit within each SMC bank configuration register (SMCBCR).
After reset all byte lane strobes nXBLS[3:0] held high during system reads from external memory banks. Due to the reset value of the RBLE bit it must be considered the access to the memory banks constructed from different memory devices, either 8-bit and non byte partitioned memory devices or 16-bit and 32-bit memory devices.

Table 5-6 shows the byte lane strobes nXBLS[3:0] for write accesses according to different memory configurations.

Table 5-6: Byte lane strobes, write accesses (1/2)

External bus width	Internal transfer width	XADDR[1:0]	nXBLS[3:0]	System data mapping on to external data bus			
				31:24	23:16	15:8	7:0
8-bit	Word (4 transfers)	11_b	1110_b	-	-	-	31:24
		10_b	1110_b	-	-	-	23:16
		01_b	1110_b	-	-	-	15:8
		00_b	1110_b	-	-	-	7:0
	Halfword (2 transfers)	11_b	1110_b	-	-	-	31:24
		10_b	1110_b	-	-	-	23:16
	Halfword (2 transfers)	01_b	1110_b	-	-	-	15:8
		00_b	1110_b	-	-	-	7:0
	Byte	11_b	1110_b	-	-	-	31:24
	Byte	10_b	1110_b	-	-	-	23:16
	Byte	01_b	1110_b	-	-	-	15:8
	Byte	00_b	1110_b	-	-	-	7:0
16-bit	Word (2 transfers)	$1x_b$	1100_b	-	-	31:24	23:16
		$0x_b$	1100_b	-	-	15:8	7:0
	Halfword	$1x_b$	1100_b	-	-	31:24	23:16
	Halfword	$0x_b$	1100_b	-	-	15:8	7:0
	Byte	$1x_b$	1101_b	-	-	31:24	-
	Byte	$1x_b$	1110_b	-	-	-	23:16
	Byte	$0x_b$	1101_b	-	-	15:8	-
	Byte	$0x_b$	1110_b	-	-	-	7:0

Table 5-6: Byte lane strobes, write accesses (2/2)

External bus width	Internal transfer width	XADDR[1:0]	nXBLS[3:0]	System data mapping on to external data bus			
				31:24	23:16	15:8	7:0
32-bit	Word	xx_b	0000_b	31:24	23:16	15:8	7:0
	Halfword	xx_b	0011_b	31:24	23:16	-	-
	Halfword	xx_b	1100_b	-	-	15:8	7:0
	Byte	xx_b	0111_b	31:24	-	-	-
	Byte	xx_b	1011_b	-	23:16	-	-
	Byte	xx_b	1101_b	-	-	15:8	-
	Byte	xx_b	1110_b	-	-	-	7:0

Chapter 6 Memory Access Control Function

6.1 SRAM, External ROM, External I/O Interface

6.1.1 Features

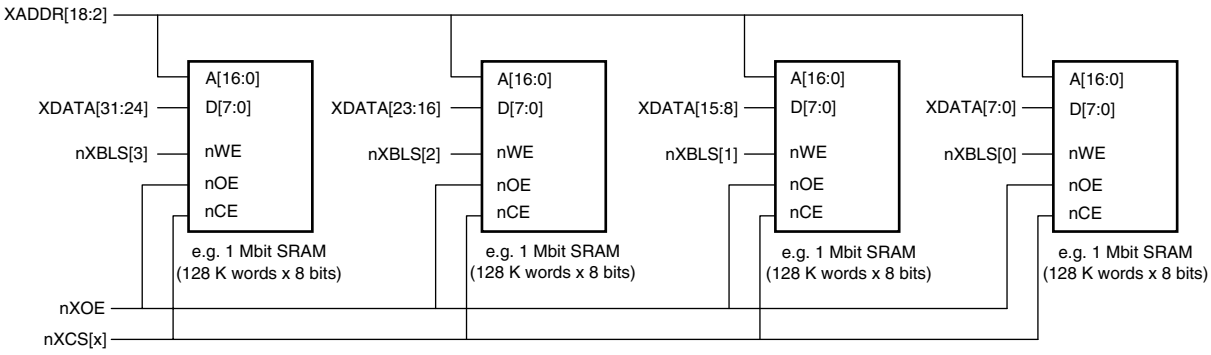
- Access to SRAM is executed in 3 states (minimum).
- Up to 32 states of programmable data waits can be inserted by software (SMCBCR0 to SMCBCR7 registers).
- Data wait can be controlled with external input pin (nXWAIT).
- Up to 16 idle states can be inserted after the read/write cycle by software (SMCBCR0 to SMCBCR7 registers).

6.1.2 External SRAM Connections

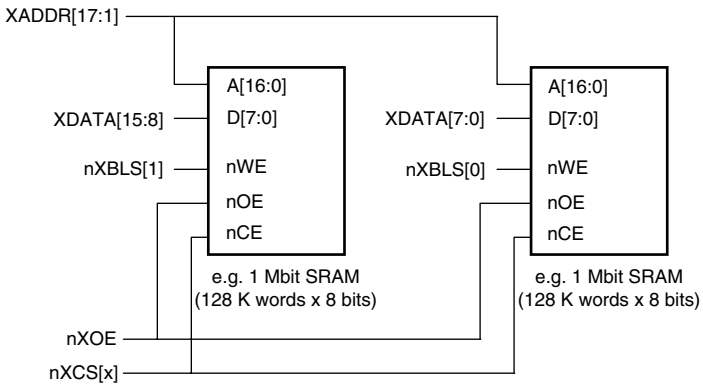
An example of connection to external SRAM is shown below.

Figure 6-1: Example of connection to 8-bit organized memory devices

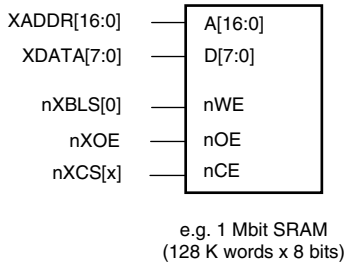
(a) When data bus width is 32 bits



(b) When data bus width is 16 bits



(c) When data bus width is 8 bits

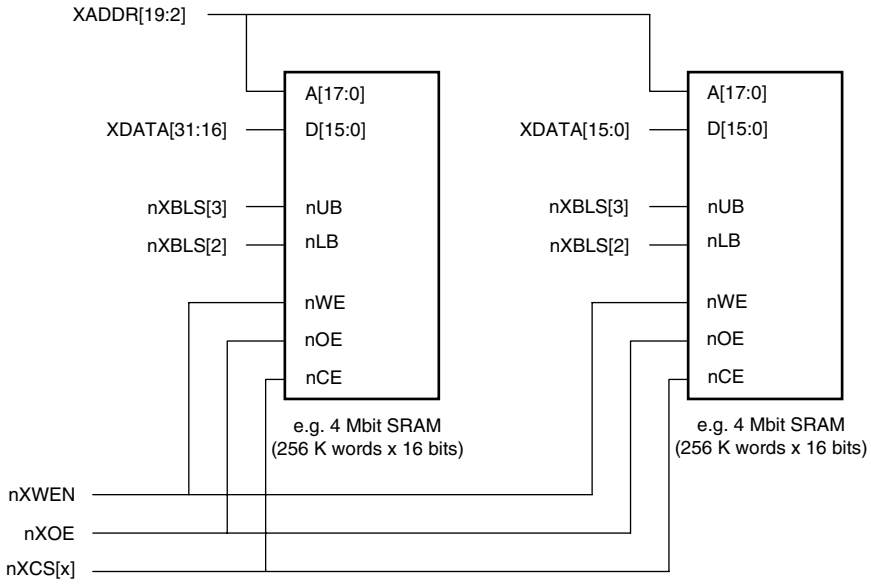


For external memory banks constructed from 8-bit or non byte-partitioned memory devices, it is important that the RBLE bit in the relevant SMCBCR register is cleared to zero. This forces all nXBLS[3:0] lines to high level during a read access to that particular bank.
The above figure shows 8-bit memory being used to configure external memory banks, which are 8-, 16- or 32-bits wide. In each of these configuration the nXBLS[3:0] lines are connected to write enable inputs of each 8-bit memory.

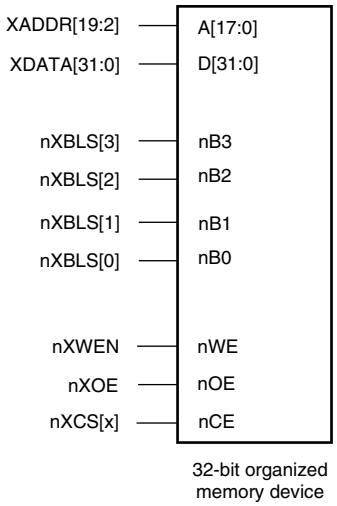
For write transfers only the relevant nXBLS[3:0] lines are asserted low and steer the data to the addressed bytes.
For read transfers all nXBLS[3:0] lines are deasserted high, which allows the external memory bus to be defined for at least the width of the accessed memory.

Figure 6-2: Example of connection to 16/32-bit organized memory devices

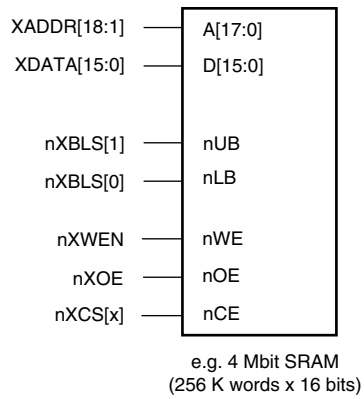
(a) When data bus width is 32 bits (16-bit organized memory)



(b) When data bus width is 32 bits (32-bit organized memory)



(c) When data bus width is 16 bits (16-bit organized memory)



For external memory banks constructed from 16-bit or 32-bit memory devices, it is important that the RBLE bit in the relevant SMCBCR register is set to one. This forces all nXBLS[3:0] lines to low level during a read access to that particular bank.
The above figures show 16-bit/32-bit memory being used to configure external memory banks, which are 16- or 32-bits wide. Each of these memory devices are controlled by external byte select signals (nXBLS 3 to 0) and these signals must be appropriately connected.

6.1.3 External SRAM, External ROM, External I/O Access

Figure 6-3: External SRAM, external ROM, external I/O access timing (1/4)

(a) Read Transfer (with wait insertion)

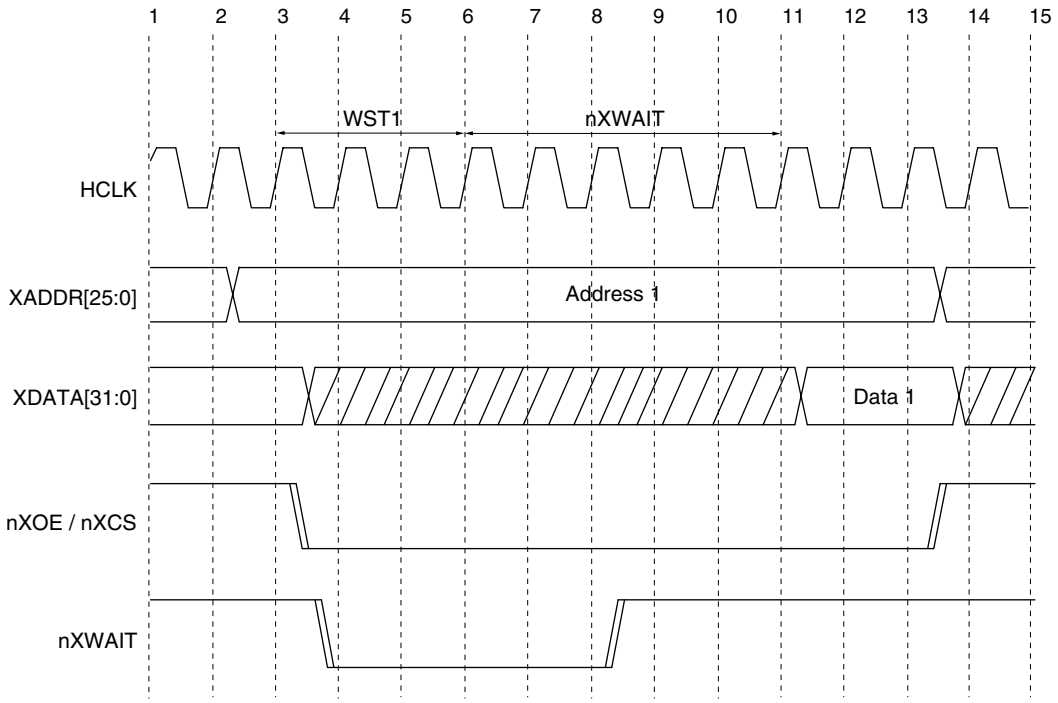


Figure 6-3 (1/4) shows the timing diagram for a single read transfer. The WST1 bits in the corresponding SMCBCR register are set to a value of 2 causing 3 wait cycles. Due to 2 cycles delay for nXWAIT synchronization and the fact that the nXWAIT input must be asserted within an ongoing wait cycle, the nXWAIT must be low at the rising edge of clock cycle (4). If the nXWAIT input is not low at minimum two clock cycles before the WST1 wait cycles are elapsed, the nXWAIT input will not be asserted for inserting additional wait states.
As long as the nXWAIT input is active during the rising clock edges, consecutive wait cycles are inserted into the data cycle. As soon as the nXWAIT input goes inactive the remaining wait cycles from the WST1 bits in the corresponding SMCBCR register will be inserted.
In the example above, all WST1 wait cycles were elapsed before the nXWAIT wait cycles are inserted.

Figure 6-3: External SRAM, external ROM, external I/O access timing (2/4)

(b) Sequential Read Transfer

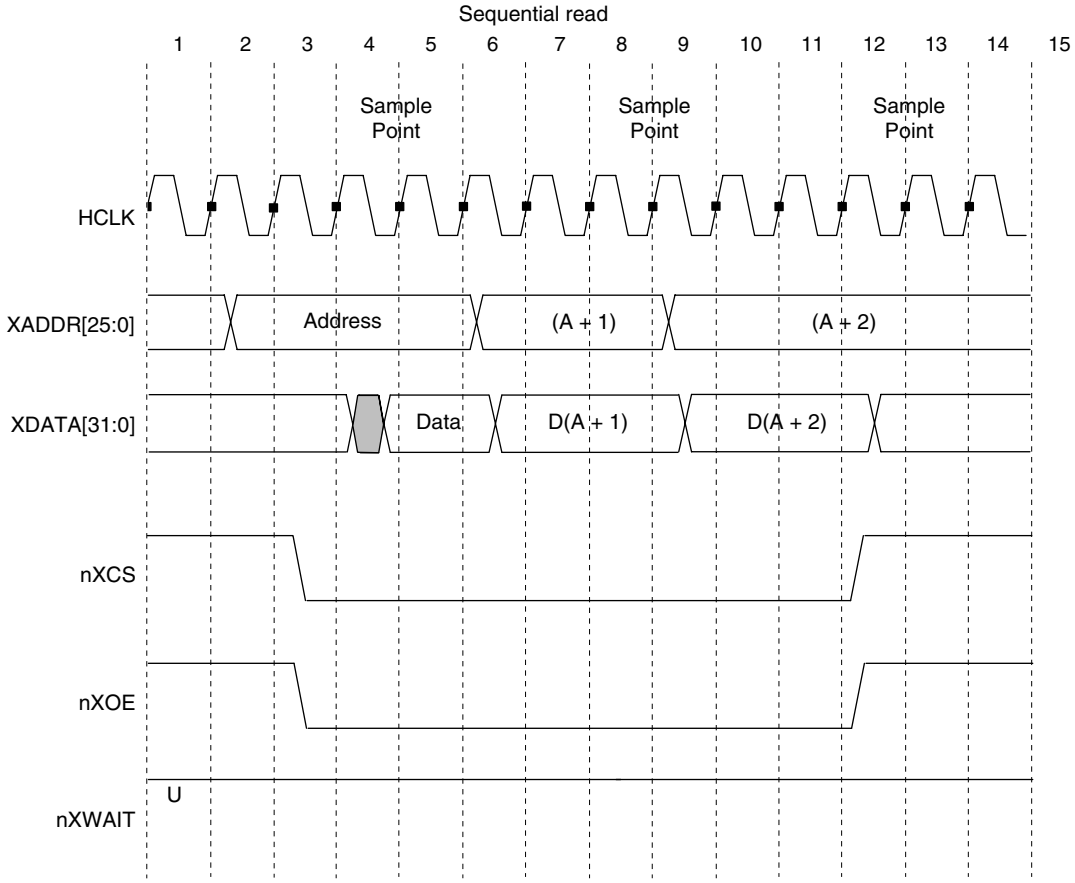


Figure 6-3 (2/4) shows the timing diagram for a sequential read transfer. The WST1 bits in the corresponding SMCBCR register are set to a value of 0 causing 1 wait cycle as default.
The nXWAIT signal is deasserted in this example.
The sequential read transfer needs at least 3 clock cycles for accessing each data.

Figure 6-3: External SRAM, external ROM, external I/O access timing (3/4)

(c) Write Transfer (with wait insertion)

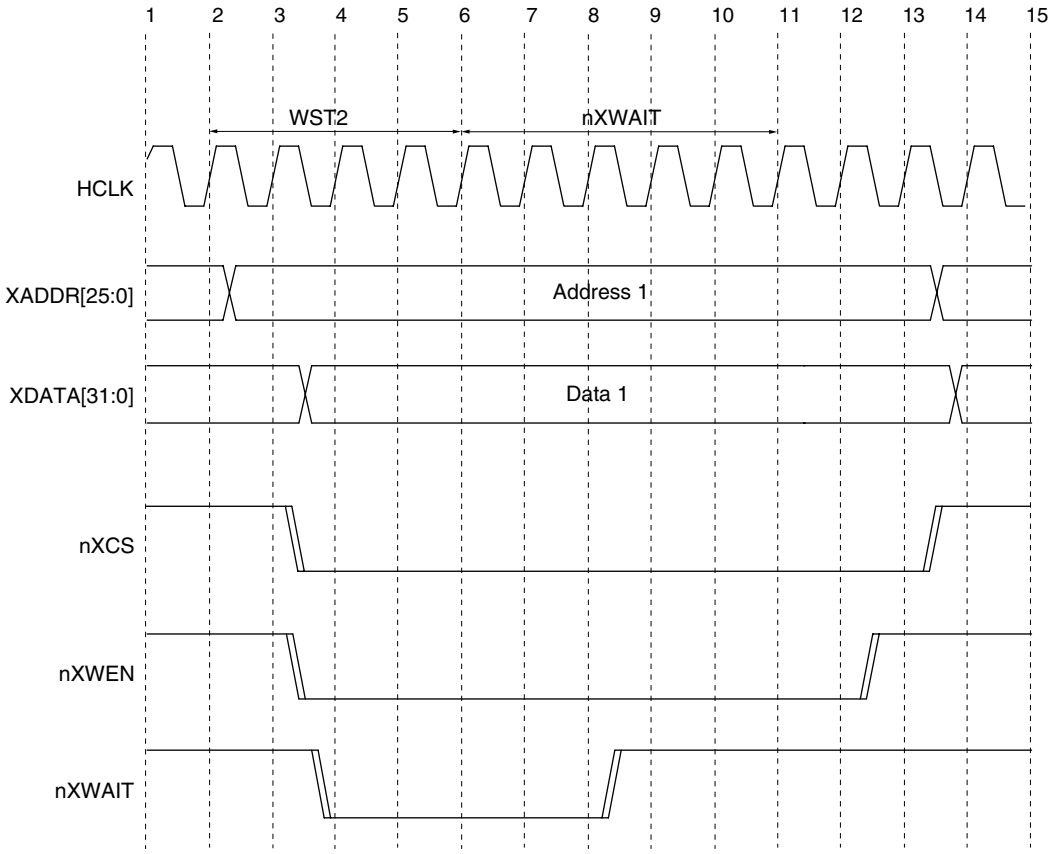


Figure 6-3 (3/4) shows the timing diagram for a single write transfer. The WST2 bits in the corresponding SMCBCR register are set to a value of 3 causing 4 wait cycles. Due to 2 cycles delay for nXWAIT synchronization and the fact that the nXWAIT input must be asserted within an ongoing wait cycle, the nXWAIT must be low at the rising edge of clock cycle (4). If the nXWAIT input is not low at minimum two clock cycles before the WST2 wait cycles are elapsed, the nXWAIT input will not be asserted for inserting additional wait states.
As long as the nXWAIT input is active during the rising clock edges, consecutive wait cycles are inserted into the data cycle. As soon as the nXWAIT input goes inactive the remaining wait cycles from the WST2 bits in the corresponding SMCBCR register will be inserted.
In the example above, all WST2 wait cycles were elapsed before the nXWAIT wait cycles are inserted.

Figure 6-3: External SRAM, external ROM, external I/O access timing (4/4)

(d) Sequential Write Transfer

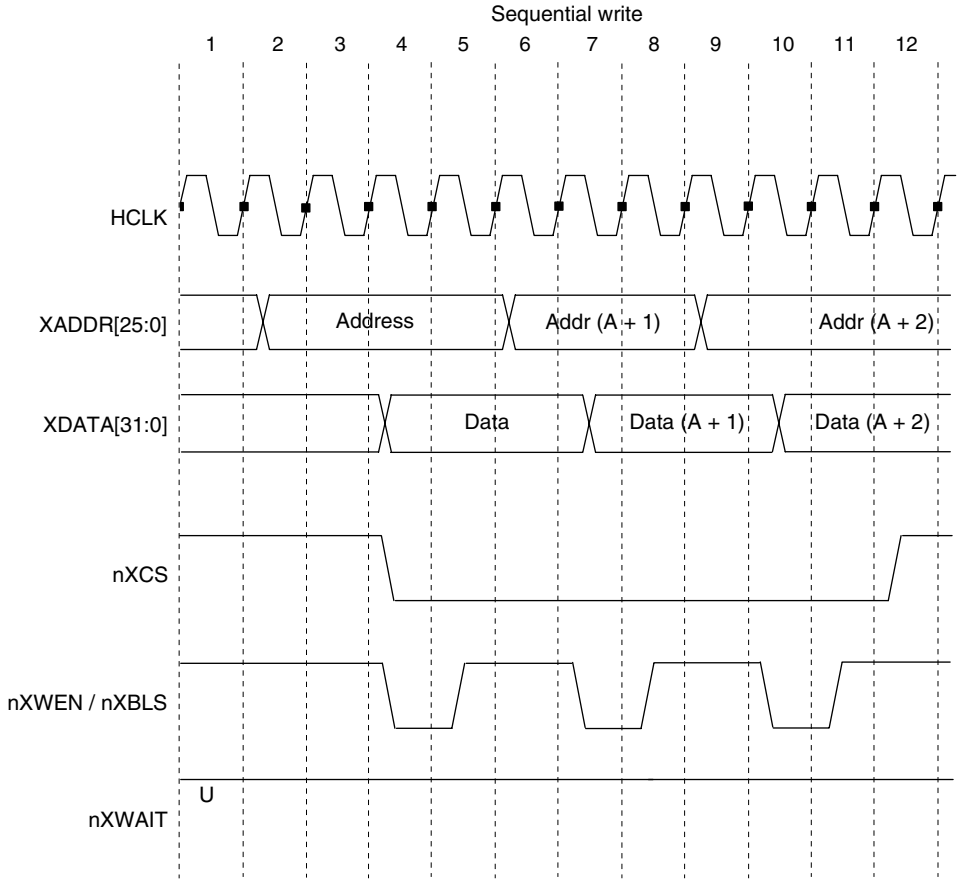


Figure 6-3 (4/4) shows the timing diagram for a sequential write transfer. The WST2 bits in the corresponding SMCBCR register are set to a value of 0 causing 1 wait cycle as default.
The nXWAIT signal is deasserted in this example.
The sequential write transfer needs at least 3 clock cycles for writing each data out.

6.2 Page Mode ROM Interface

The external bus interface provides access to page mode ROM with the page access function.
This external bus interface can handle sequential access burst transfers of up to four consecutive locations in 8-, 16- or 32-bit memory devices.

6.2.1 Features

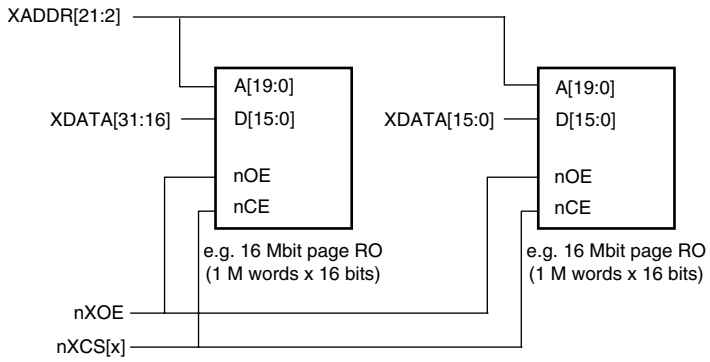
- Direct connection to 8-bit/16-bit page mode ROM supported.
- Up to 32 states of programmable initial access time can be inserted by software (SMCBCR0 to SMCBCR7 registers).
- Up to 32 states of programmable burst access time can be inserted by software (SMCBCR0 to SMCBCR7 registers).
- Data wait can be controlled with external input pin (nXWAIT).

6.2.2 Page ROM Connections

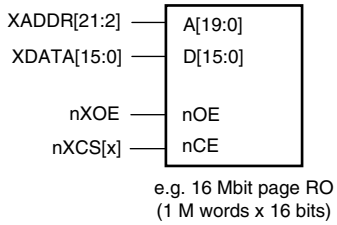
Examples of page ROM connections are shown below.

Figure 6-4: Example of connection to page mode memory device

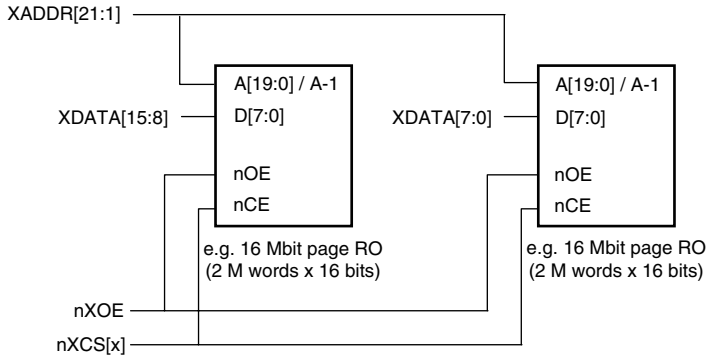
(a) When data bus width is 32 bits (16-bit organized memory)



(b) When data bus width is 16 bits (16-bit organized memory)



(c) When data bus width is 16 bits (8-bit organized memory)



The external bus interface supports sequential access burst reads of up to four consecutive locations in 8-, 16- or 32-bit wide memories in different data bus width configurations.

6.2.3 Page ROM Access

Figure 6-5: Page mode ROM access timing

(a) Burst Read Transfer

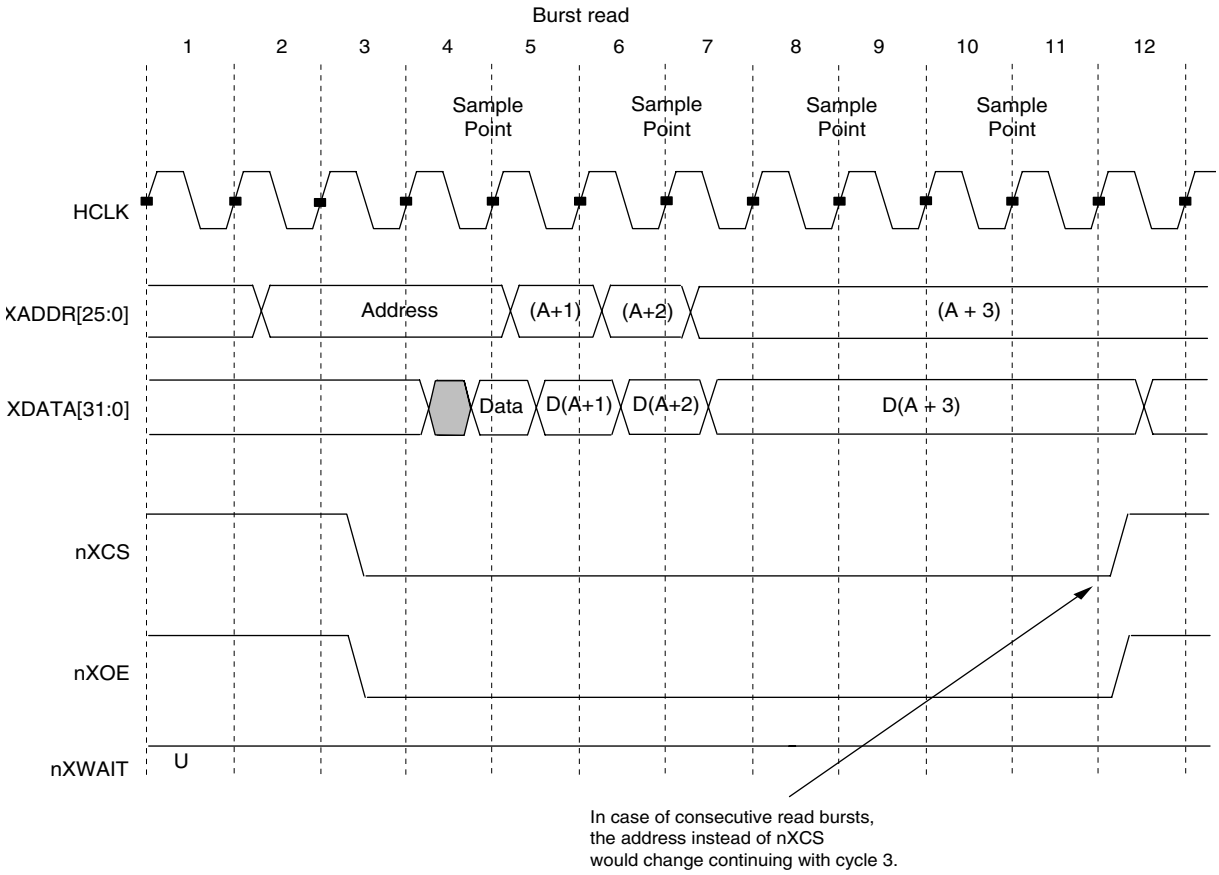


Figure 6-5 shows the timing diagram for a burst read transfer. The burst read transfer is executed with WST1 and WST2 bits are set to a value of 0.
The nXWAIT signal is deasserted in this example.

[MEMO]

Chapter 7 Interrupt Controller Function

The SoCLite is provided with a dedicated interrupt controller (INTC) for interrupt servicing and can process a total of 32 interrupt requests.
An interrupt is an event that occurs independently of program execution, where as an exception is an event dependent on program execution. Generally, an exception takes precedence over an interrupt.
The SoCLite can process interrupt requests from the on-chip peripheral hardware and external UDL sources. Moreover, exception processing can be started by the SWI instruction (software exception) or by generation of an exception event (i.e. fetching of an illegal opcode, data access memory abort).
The exception processing is handled in the ARM7TDMI core, therefore refer to Document “ARM7TDMI Data Sheet” (ARM DDI 0029E).
Eight levels of software-programmable priorities can be specified for each interrupt request.

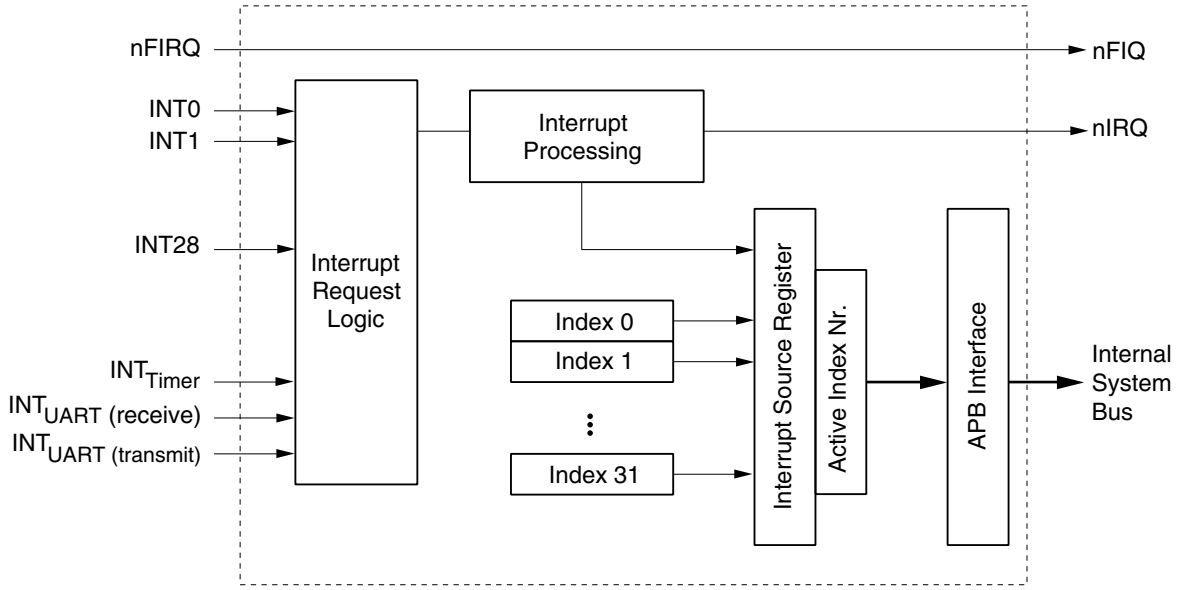
7.1 Features

- 32 interrupt sources (3 internal subsystem + 29 UDL).
- All interrupt sources are routed to the nIRQ.
- 8 levels of programmable priorities.
- Multiple interrupt control according to priority.
- All interrupts are maskable.
- Sensitivity for each interrupt are programmable: falling/rising edge or positive/negative level detection.
- 32-bit wide internal data bus.
- Support of an index number for all interrupts.

7.2 Functional Overview

The interrupt controller (INTC) provides a hardware interface to the ARM7TDMI core interrupt interface. The 32 interrupt sources are routed only to the nIRQ of the ARM7TDMI core through the interrupt controller. Therefore these interrupts can only generate a nIRQ interrupt.
The fast interrupt request (nFIQ) of the ARM7TDMI core is not routed through the interrupt controller, but directly connected to the UDL as nFIRQ signal to retain fast and low latency interrupt handling.
The 32 interrupts provide an index number to be added to an address for an interrupt service routine. Reading from the interrupt source register (INTSRC) provides the interrupt index number of the current requested interrupt. Writing to the corresponding bit position in the interrupt status register (INTSTA) clears the current interrupt pending flag and updates the interrupt priority hardware that masks out the current and any lower priority interrupt requests.
Figure 7-1 shows a block diagram of the interrupt controller.

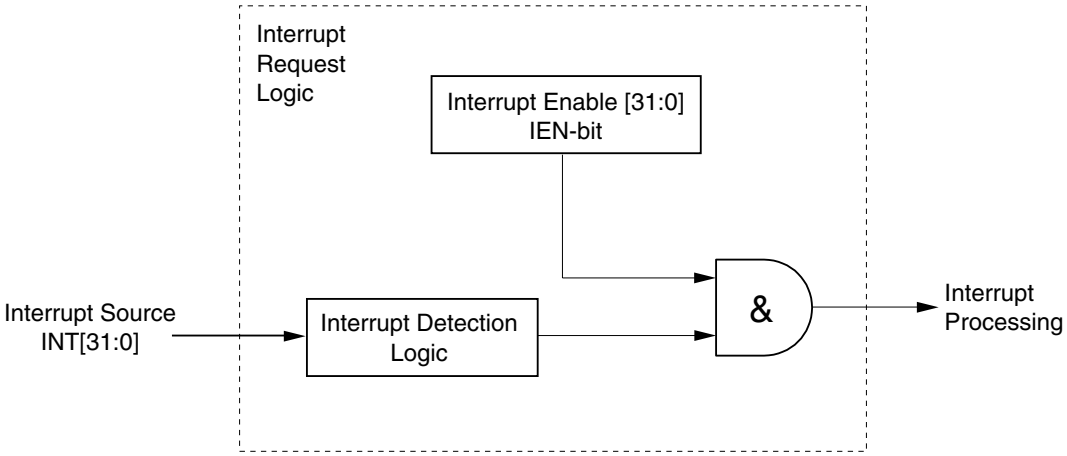
Figure 7-1: Interrupt Controller



7.2.1 Interrupt Request Logic

The interrupt request logic receives the interrupt requests from the UDL area or the on-chip peripherals. The interrupt request logic masks out the interrupt requests which are not enabled (IEN bit in the appropriate interrupt control register 0 to 7) and routes the enabled interrupt requests to the interrupt control registers. The interrupt status bit (IST bit) in the corresponding interrupt control register (INTC 0 to 7) and related bit in the interrupt status register (INTSTA) is not set, when the IEN bit is cleared.

Figure 7-2: Interrupt Request Logic



7.3 Interrupt Controller Operation

An interrupt request is detected by the rising/falling edge or the high/low level of the interrupt request signal. The internal interrupts (Timer and UART) of the on-chip peripherals are all high level sensitive. The sensitivity of the interrupt signals from the UDL are configurable by the interrupt control registers.

The interrupt requests can be masked by the interrupt control registers as well. All interrupt sources of the SoCLite are maskable.
If two or more interrupt requests are generated at the same time, they are acknowledged according to the default priority. In addition to the default priority, eight levels of priorities can be specified by using the interrupt control registers (programmable priority control).
When an interrupt request has been acknowledged, the acceptance of other interrupt requests, with the same or lower priority, are suspended.
Note that only interrupts with a higher priority will have the capability of nesting; interrupts with the same priority level cannot be nested.

If an interrupt occurs, the INTC performs the following processing, and transfers control to the CPU of the SoCLite:

(a) The appropriate interrupt status bit is set in the interrupt status register (INTSTA) and the corresponding interrupt control register (IST bit in INTC0 to 7).
(b) The INTC checks, which status bit represents the interrupt with the highest priority of all set status bits.
(c) The nIRQ output to the CPU is asserted low for the highest priority interrupt and the interrupt index number is available in the interrupt source register (INTSRC).
(d) When the interrupt source register (INTSRC) is read by the application software, an internal source related flag inside the INTC is set. This internal flag indicates that the nIRQ output has been activated for this interrupt and that the interrupt index number has been read. At the same time, the nIRQ output to the CPU is deasserted.
(e) The interrupt source register (INTSCR) contains the interrupt index number of the highest prioritized interrupt, which has been occurred until the interrupt source register is read. Even though the nIRQ output to the CPU has been activated by a low prioritized interrupt and a higher prioritized interrupt occurs before the interrupt source register has been read, the interrupt source register contains the interrupt index number of the highest prioritized interrupt (supports nested interrupts).
(f) After reading the interrupt source register, the interrupt service routine of the application software handles the appropriate interrupt request and has to clear the corresponding interrupt status bit in the interrupt status register at the end of the interrupt service routine. This enables the interrupt controller to process further interrupt requests of any priority.

7.3.1 Interrupt Handling

Figure 7-3 shows a typical timing for interrupt handling. The interrupt source INT4 with priority level 3, configured as positive edge sensitive, is asserted to the interrupt controller. One clock cycle later the interrupt status register (INTSTA) is updated, status bit INTSTA4 is set. After this the nIRQ is pulled low because there's no other interrupt pending and therefore interrupt source INT4 has the highest priority. The interrupt source register (INTSRC) is updated three clocks after activating the nIRQ and read later by the application software causing the nIRQ signal to be suspended. In the meantime, an interrupt of the same priority level and same configuration (INT5) occurs and the appropriate status bit is set in the interrupt status register. After the INT4 interrupt source has been serviced, the application software clears the corresponding status bit in the interrupt status register, which releases the interrupt output nIRQ. Now the INT5 interrupt source has the possibility to be serviced. Therefore the nIRQ output to the CPU is asserted again and the interrupt source register is updated.

A second example is shown in Figure 7-4. A rising edge on the INT4 interrupt source causes the nIRQ output to be pulled low and to set the status bit INTSTA4 in the interrupt status register. A few clock cycles later a higher prioritized interrupt source INT3 appears. The corresponding status bit (INTSTA3) in the interrupt status register is set and the interrupt source register changes its contents three clock cycles later. The nIRQ output remains active. The interrupt service routine was originally called because of the INT4 interrupt source event. Now the service routine reads the index number of interrupt source INT3, which has appeared in the meantime. The software handles the interrupt source INT3 and before exiting the interrupt handler, it acknowledges this interrupt by clearing the appropriate status bit in the interrupt status register. Since the index number of INT4 interrupt source has never been read, this still pending interrupt source causes the nIRQ output to the CPU to become asserted once more.

Figure 7-3: Interrupt Timing

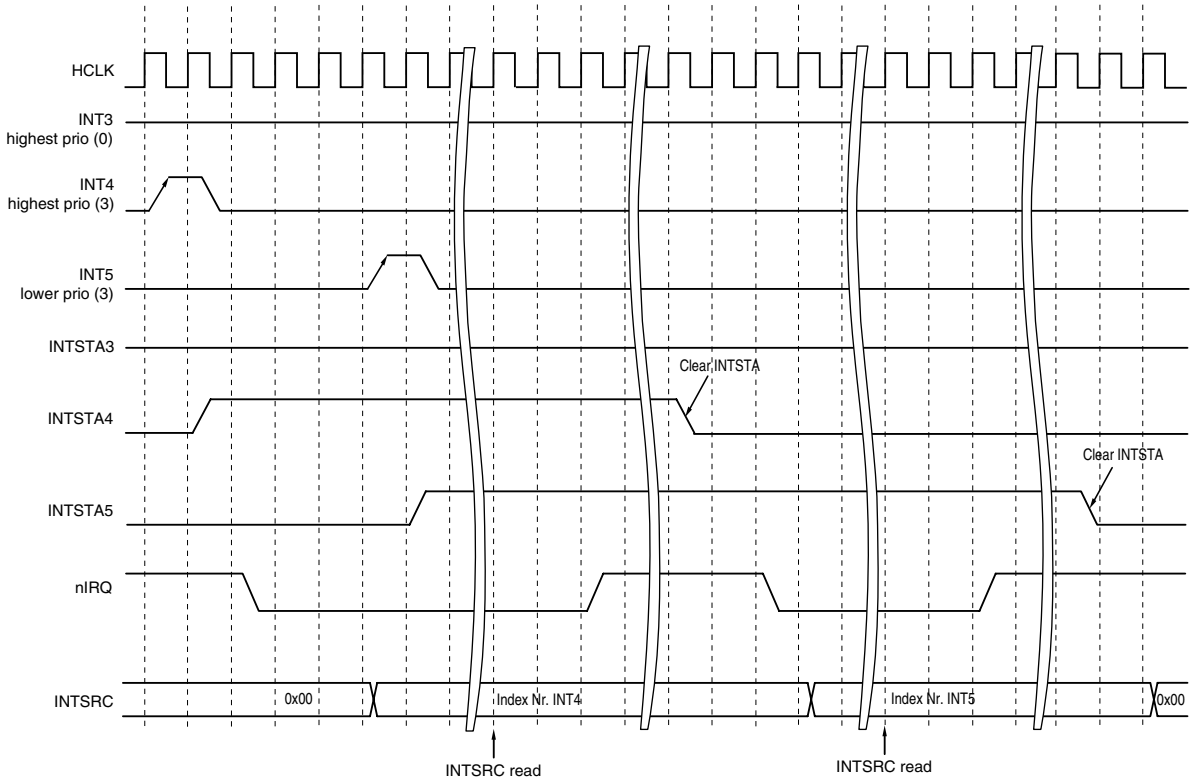
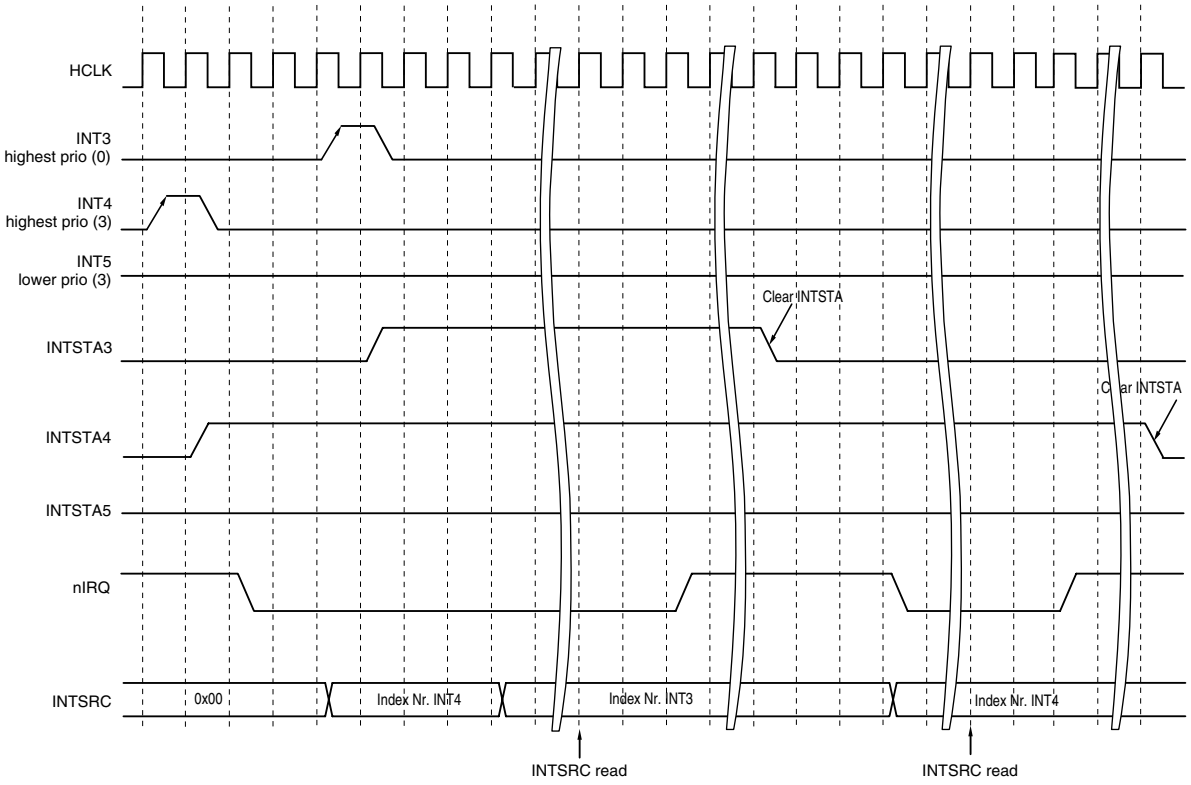


Figure 7-4: Interrupt Timing (nested)



7.3.2 Interrupt Priority

The SoCLite provides multiple interrupt servicing in which an interrupt is acknowledged while another interrupt is being serviced. Multiple interrupts can be controlled by priority levels.
There are two types of priority level control: control based on the default priority levels, and control based on the programmable priority levels that are specified by the interrupt priority level specification bits (PRIO) of the interrupt control registers (INTC0 to INTC7). When two or more interrupts, having the same priority level specified by the PRIO bit, are generated at the same time, interrupts are serviced in order depending on the priority level allocated to each interrupt request type (default priority level) beforehand. For more information, refer to Table 7-1. The programmable priority control customizes interrupt requests into eight levels by setting the priority level specification bits.

Table 7-1: Default Interrupt Priority

Interrupt Source	Priority level (PRIO bits in INTC0 to INTC7 registers)	Internal Priority (default, when interrupts occur simultaneously)
INT0	same priority level	highest priority
INT1		
INT2		
INT3		
INT4		
INT5		
INT6		
INT7		
INT8		
INT9		
INT10		
INT11		
INT12		
INT13		
INT14		
INT15		
INT16		
INT17		
INT18		
INT19		
INT20		
INT21		
INT22		
INT23		
INT24		
INT25		
INT26		
INT27		
INT28		
INT_{Timer}		
$INT_{UART(receive)}$		
$INT_{UART(transmit)}$		lowest priority

7.4 Interrupt Function Registers

7.4.1 Interrupt Control Registers 0 to 7 (INTC0 to INTC7)

Eight 32-bit organized interrupt control registers are assigned to set the control conditions for each interrupt source.
These registers can be read/written.

Table 7-2: Interrupt Control Register Assignment

Register Name	Address	Initial Value
INTC0	40000000H	07070707H
INTC1	40000004H	07070707H
INTC2	40000008H	07070707H
INTC3	4000000CH	07070707H
INTC4	40000010H	07070707H
INTC5	40000014H	07070707H
INTC6	40000018H	07070707H
INTC7	4000001CH	07070707H

During reset all control conditions are set to:

- lowest priority level
- positive level sensitive
- interrupt disabled
- interrupt not pending

Each 32-bit interrupt control register is divided into 4 groups a 8 bits and each group controls one interrupt source.

Table 7-3: Interrupt Control Registers (INTCn) (n = 0 to 7)

Interrupt Control Register	Bit Position	Group No.	Interrupt Source
INTC0	31 - 24	4	INT3
	23 - 16	3	INT2
	15 - 8	2	INT1
	7 - 0	1	INT0
INTC1	31 - 24	4	INT7
	23 - 16	3	INT6
	15 - 8	2	INT5
	7 - 0	1	INT4
INTC2	31 - 24	4	INT11
	23 - 16	3	INT10
	15 - 8	2	INT9
	7 - 0	1	INT8
INTC3	31 - 24	4	INT15
	23 - 16	3	INT14
	15 - 8	2	INT13
	7 - 0	1	INT12
INTC4	31 - 24	4	INT19
	23 - 16	3	INT18
	15 - 8	2	INT17
	7 - 0	1	INT16
INTC5	31 - 24	4	INT23
	23 - 16	3	INT22
	15 - 8	2	INT21
	7 - 0	1	INT20
INTC6	31 - 24	4	INT27
	23 - 16	3	INT26
	15 - 8	2	INT25
	7 - 0	1	INT24
INTC7	31 - 24	4	$INT_{UART(transmit)}$
	23 - 16	3	$INT_{UART(receive)}$
	15 - 8	2	INT_{Timer}
	7 - 0	1	INT28

Figure 7-5: Structure of Interrupt Control Register (INTCn) (n = 0 to 7) (1/2)

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

							Address	Initial value
INTCn	IST	IEN	0	ILC[Note]	IPC[Note]	PRIO2-0	see Table 7-2	see Table 7-2
	R				R/W			

Note: For Timer and UART interrupts the settings are fixed to ILC=IPC=0.

Figure 7-5: Structure of Interrupt Control Register (INTCn) (n = 0 to 7) (2/2)

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>31 / 23 / 15 / 7</td><td>IST</td><td>Interrupt Status (read-only)
Reflects the interrupt status of the corresponding interrupt source.
<table><tr><th>IST</th><th>Operation mode</th></tr><tr><td>0</td><td>interrupt not pending</td></tr><tr><td>1</td><td>interrupt pending</td></tr></table></td></tr>
<tr><td>30 / 22 / 14 / 6</td><td>IEN</td><td>Interrupt Enable
Enables or disables the corresponding interrupt source.
<table><tr><th>IEN</th><th>Operation mode</th></tr><tr><td>0</td><td>interrupt disabled (default)</td></tr><tr><td>1</td><td>interrupt enabled</td></tr></table>
Remark: If the interrupt status bit (IST) is set and the IEN bit for this corresponding interrupt is disabled, the interrupt will be handled as if it is enabled.</td></tr>
<tr><td>29 / 21 / 13 / 5</td><td>-</td><td>Reserved</td></tr>
<tr><td>28 / 20 / 12 / 4</td><td>ILC</td><td>Interrupt Level Control
Selects the control level of the corresponding interrupt source.
<table><tr><th>ILC</th><th>Operation mode</th></tr><tr><td>0</td><td>level sensitive (default)</td></tr><tr><td>1</td><td>edge sensitive</td></tr></table>
Remark: Timer and UART interrupt sources are always positive level sensitive. Values for these interrupt sources can not be changed.</td></tr>
<tr><td>27 / 19 / 11 / 3</td><td>IPC</td><td>Interrupt Polarity Control
Selects the polarity of the control level of the corresponding interrupt source.
<table><tr><th>IPC</th><th>Operation mode</th></tr><tr><td>0</td><td>positive level / edge sensitive (default)</td></tr><tr><td>1</td><td>negative level / edge sensitive</td></tr></table>
Remark: Timer and UART interrupt sources are always positive level sensitive. Values for these interrupt sources can not be changed.</td></tr>
<tr><td>26 - 24 / 18 - 16 / 10 - 8 / 2 - 0</td><td>PRIO2 to PRIO0</td><td>Interrupt Priority
Controls the interrupt priority level of the corresponding interrupt source.
<table><tr><th>PRIO2</th><th>PRIO1</th><th>PRIO0</th><th>Interrupt Priority</th></tr><tr><td>0</td><td>0</td><td>0</td><td>Specifies level 0 (highest)</td></tr><tr><td>0</td><td>0</td><td>1</td><td>Specifies level 1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>Specifies level 2</td></tr><tr><td>0</td><td>1</td><td>1</td><td>Specifies level 3</td></tr><tr><td>1</td><td>0</td><td>0</td><td>Specifies level 4</td></tr><tr><td>1</td><td>0</td><td>1</td><td>Specifies level 5</td></tr><tr><td>1</td><td>1</td><td>0</td><td>Specifies level 6</td></tr><tr><td>1</td><td>1</td><td>1</td><td>Specifies level 7 (lowest) (default)</td></tr></table></td></tr>
</table>

7.4.2 Interrupt Status Register (INTSTA)

This register represents the status of each of the 32 interrupt sources.
When an interrupt request is asserted, the bit of this register corresponding to the priority level of that interrupt request is set to 1 and remains set while the interrupt is serviced.
This register can be read/written.

Figure 7-6: Interrupt Status Register (INTSTA)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	Address	Initial value
INTSTA	INT UART TXD	INT UART RXD	INT Timer	INT 28	INT 27	INT 26	INT 25	INT 24	INT 23	INT 22	INT 21	INT 20	INT 19	INT 18	INT 17	INT 16	40000020H	00000000H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0

Note: The status bits are equal to the IST bits inside the appropriate INTCn (n = 0 to 7) registers.

Bit Position	Bit Name	Function
31- 0	INTSTAn	Interrupt Status Reflects the interrupt status of the corresponding interrupt source.

INTSTA	Operation mode	
	Read of bit	Write to bit
0	interrupt not pending	no effect
1	interrupt pending	clear status bit

7.4.3 Interrupt Source Register (INTSRC)

The interrupt source register provides the interrupt index number of the asserted interrupt. When this register is read, an internal flag for the corresponding interrupt source is set to indicate that the interrupt is in progress to be serviced. The interrupt source register contains the highest prioritized interrupt index number of the appropriate interrupt source.
The interrupt source register is cleared when there is no more interrupt pending.
This register is read-only.

Figure 7-7: Interrupt Source Register (INTSRC) (1/2)

	7	6	5	4	3	2	1	0	Address	Initial value
INTSRC	0	0	0	Interrupt index number					40000024H	00H

Figure 7-7: Interrupt Source Register (INTSRC) (2/2)

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>7 - 5</td><td>-</td><td>Reserved</td></tr>
<tr><td>4-0</td><td>Interrupt Index Number</td><td>Interrupt Index Number
Reflects the interrupt index number of the corresponding interrupt source.
<table>
<tr><th rowspan="2">Interrupt Source</th><th colspan="6">Interrupt Index Number</th></tr>
<tr><th>Dec</th><th>Bit 4</th><th>Bit 3</th><th>Bit 2</th><th>Bit 1</th><th>Bit 0</th></tr>
<tr><td>INT0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>INT1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>INT2</td><td>2</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>INT3</td><td>3</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>INT4</td><td>4</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>INT5</td><td>5</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>INT6</td><td>6</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>INT7</td><td>7</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>INT8</td><td>8</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>INT9</td><td>9</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>INT10</td><td>10</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>INT11</td><td>11</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>INT12</td><td>12</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>INT13</td><td>13</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>INT14</td><td>14</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>INT15</td><td>15</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>INT16</td><td>16</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>INT17</td><td>17</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>INT18</td><td>18</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>INT19</td><td>19</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>INT20</td><td>20</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>INT21</td><td>21</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>INT22</td><td>22</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>INT23</td><td>23</td><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>INT24</td><td>24</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>INT25</td><td>25</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>INT26</td><td>26</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>INT27</td><td>27</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>INT28</td><td>28</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>INT_{Timer}</td><td>29</td><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>$INT_{UART(receive)}$</td><td>30</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>$INT_{UART(transmit)}$</td><td>31</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
</table>

Remark: The above interrupt index number is placed on the internal data bus of the SoCLite and will be read by the CPU.

7.4.4 Interrupt Source Register for Emulation (INTSRCEMU)

This register is identical to the interrupt source register (INTSRC). The main difference is that a read from the interrupt source register for emulation (INTSRCEMU) has no effect on the internal status flags of the interrupt controller (ISS bits in the interrupt in service status register for emulation / INTISSEMU). This register is intended to be used for emulation and debug purposes.
This register is read-only.

Figure 7-8: Interrupt Source Register (INTSRCEMU)

	7	6	5	4	3	2	1	0	Address	Initial value
INTSRCEMU	0	0	0	Interrupt index number					40000028H	00H

Note: The interrupt index number table is the same as in Figure 7-7.

7.4.5 Interrupt In-Service Status Register for Emulation (INTISSEMU)

The interrupt in-service status register for emulation is used for emulation and debug purposes. Each bit of the interrupt in-service status register is assigned to the corresponding interrupt source. When the interrupt source register (INTSRC) is read the appropriate ISS bit of the corresponding interrupt source is set.
This register is read-only.

Figure 7-9: Interrupt In-Service Status Register for Emulation (INTISSEMU)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	Address	Initial value
INTISSEMU	INT UART TXD	INT UART RXD	INT Timer	INT 28	INT 27	INT 26	INT 25	INT 24	INT 23	INT 22	INT 21	INT 20	INT 19	INT 18	INT 17	INT 16	4000002CH	00000000H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT15	INT14	INT 13	INT 12	INT 11	INT 10	INT 9	INT 8	INT 7	INT 6	INT 5	INT 4	INT 3	INT 2	INT 1	INT 0

Bit Position	Bit Name	Function
31- 0	ISSn (n = 0 to 31)	Interrupt In-Service Status Reflects the interrupt in-service status of the corresponding interrupt source.

ISSn	Operation mode
0	INTSRC has not been read for the corresponding interrupt source.
1	INTSRC has been read. The interrupt source corresponding to the index number in INTSRC is now being serviced by the handler software. This bit is cleared, when the appropriate bit in the INTSTA is cleared.

7.5 Interrupt Latency

A high prioritized interrupt request is detected and serviced by the interrupt controller within a maximum latency of 4 clock cycles after the interrupt event has occurred.
Table 7-4 shows the interrupt latency of the interrupt controller depending on the operation mode.
The interrupt latency specifies the number of clock cycles (HCLK), which are elapsed from asserting the interrupt event until the nIRQ signal to the CPU is activated.

Table 7-4: Interrupt Latency

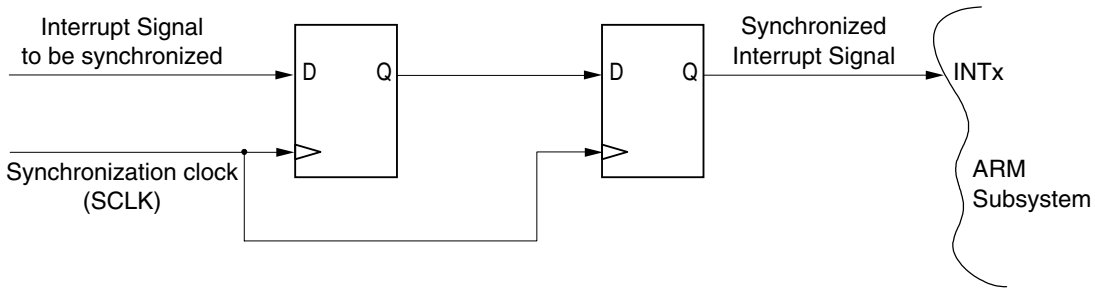
Operation mode	Interrupt latency / cycle
high/low level sensitive	2
positive/negative edge sensitive	
nested interrupt[Note]	4

Note: A previous interrupt is still being serviced and a new interrupt event occurs.

7.6 Synchronization of Interrupt Sources from UDL

The interrupt controller does not include a synchronization circuit for the interrupt signals, coming from the UDL (INT0 to INT28). As a consequence all interrupt signals from the UDL (INT0 to INT28) have to be adjusted to the system clock (SCLK in UDL) for the ARM subsystem.
For asynchronous interrupt signals an appropriate circuit has to be implemented inside the UDL area (see Figure 7-10).

Figure 7-10: Example for Interrupt Synchronization Circuit



[MEMO]

Chapter 8 Clock Generator Function

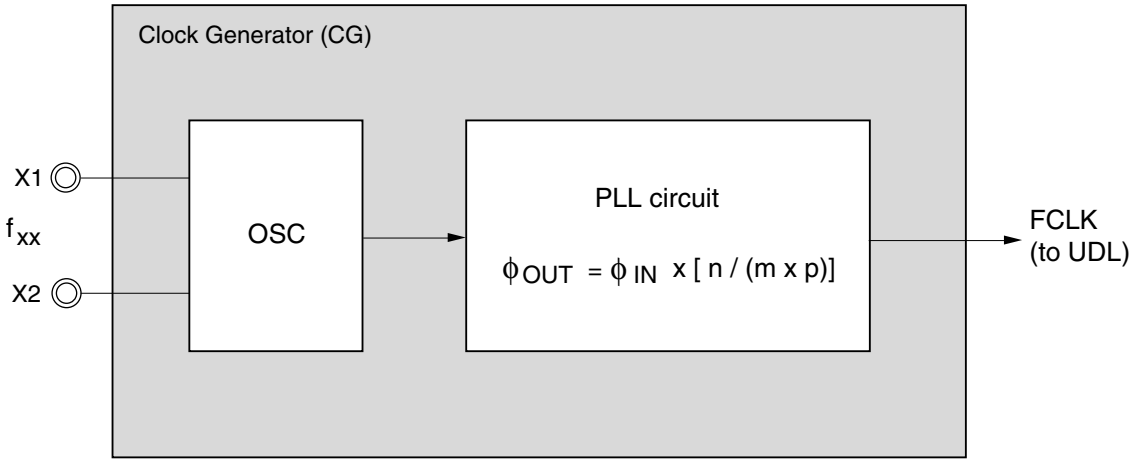
The clock generator (CG) generates and controls the internal system clock (ϕ) that is supplied to each internal unit of the SoCLite, such as the CPU.

8.1 Features

- Multiplication function by PLL synthesizer: max. multiplication rate 32
- Clock sources
  - Oscillation by connecting a resonator or crystal
  - External clock input

8.2 Configuration

Figure 8-1: Block Diagram of the Clock Generator



Remark: f_{XX}: External resonator or external clock frequency.

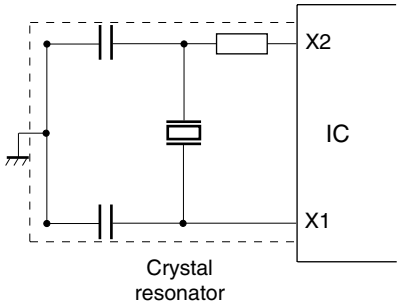
The clock generator consists of an oscillator and a PLL synthesizer.
The external signal supplied by a connected resonator is multiplied by the PLL synthesizer to generate the clock for the UDL area (FCLK), where all other clocks are derived from. The input signal from an external resonator is recommended to range from 2 MHz to 16 MHz to keep the PLL lock time, and the PLL synthesizer multiplies the source clock signal up to 32.
Due to the limitation of the VCO output, the clock to the UDL area (FCLK) is limited to 115 MHz.

8.3 Main system clock oscillator

The SoCLite utilizes an oscillator circuit to allow connection of a resonator or crystal.
The main system clock oscillator oscillates with a crystal resonator connected to the X1 and X2 pins.

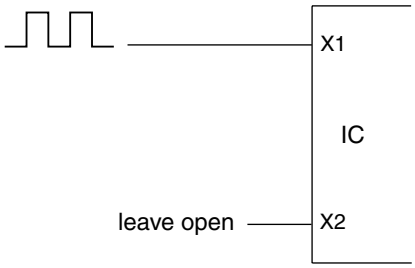
Figure 8-2: Main system clock oscillator

(a) with a crystal resonator



Also, an external clock can be input directly to the oscillator. In this case, the clock signal should be input only to pin X1 (pin X2 should be left open).

(b) without a crystal resonator

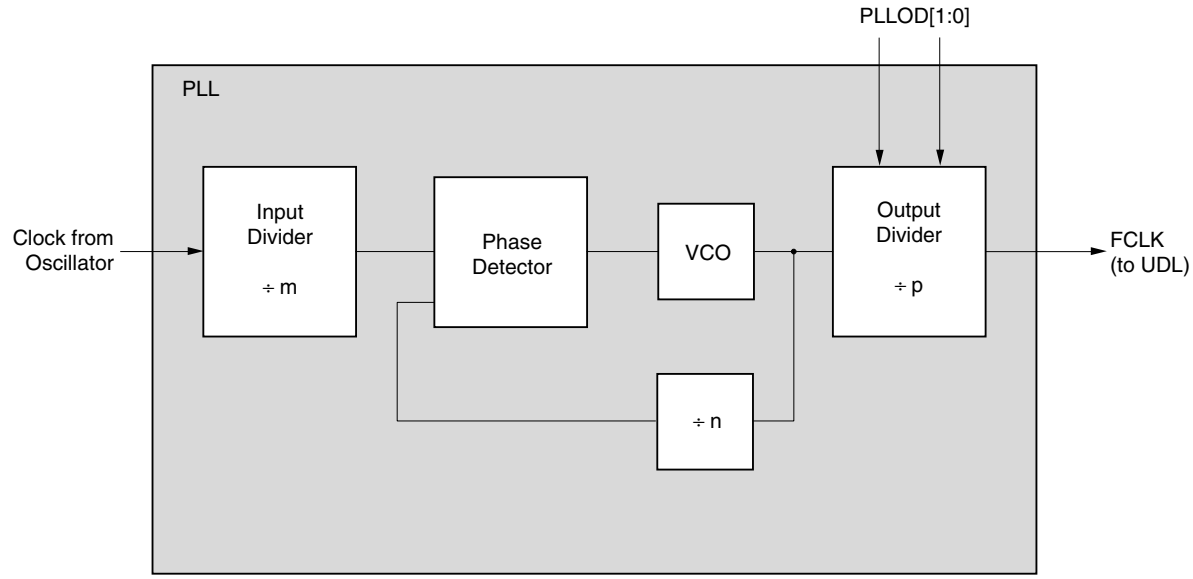


8.4 PLL

8.4.1 Configuration of PLL

The SoCLite contains a PLL to generate the FCLK for the UDL area. The system clock SCLK for the ARM subsystem is derived from the FCLK coming out of the PLL.
The PLL is always active and can not be switched off.
Figure 8-3 shows the structure of the PLL considering the internal dividers.

Figure 8-3: PLL Structure



The multiplication rate of the PLL synthesizer result from the following formula:

$$R = n / (m \times p)$$

Remarks: **1.** R is the multiplication rate
2. n is the **n**-Value
3. m is the **m**-Value of the input divider
4. p is the **p**-Value of the output divider

8.4.2 Clock Selection

The ratios of the dividers inside the PLL can be set by parameter signals. The SoCLite opens these parameter signals to the UDL area except the parameter signals for the output divider. The parameter signals for the output divider can be set by external pins.
The parameter signals for the PLL must be clamped by F091 buffer inside the UDL area.
The parameter signal and external pin levels should be fixed according to the application system. Switching these parameter signals and external pins during operation may cause malfunction.

(1) Ratio of Input Divider (m-Value)

The ratio for the input divider can be set from 2 to 32 according Table 8-1.
The M-Value table below (M-Values of 4 to 29, respectively m-Values of 5 to 30 not shown explicitly) shows the combinations of M-Values which are possible to set a specific **m**-Value. The first logic state where M = 0 is an invalid logic state since it would imply a division by 0.
The M-Value is programmed by clamping the parameter signals PLL_M[4:0] by F091 buffers inside the UDL area.

Table 8-1: Ratio of Input Divider (m-Value)

PLL_M4	PLL_M3	PLL_M2	PLL_M1	PLL_M0	M-Value	m-Value
0	0	0	0	0	Invalid	Invalid
0	0	0	0	1	1	2
0	0	0	1	0	2	3
0	0	0	1	1	3	4
--	--	--	--	--	--	--
1	1	1	1	0	30	31
1	1	1	1	1	31	32

(2) Ratio of N-Counter (n-Value)

The ratio for the N-Counter can be set from 2 to 128 according Table 8-2.
The N-Value table below (N-Values of 4 to 125, respectively n-Values of 5 to 126 not shown explicitly) shows the combinations of N-Values which are possible to set a specific **n**-Value. The first logic state where N = 0 is an invalid logic state since it would imply a division by 0.
The N-Value is programmed by clamping the parameter signals PLL_N[6:0] by F091 buffers inside the UDL area.

Table 8-2: Ratio of N-Counter (n-Value)

PLL_N6	PLL_N5	PLL_N4	PLL_N3	PLL_N2	PLL_N1	PLL_N0	N-Value	n-Value
0	0	0	0	0	0	0	Invalid	Invalid
0	0	0	0	0	0	1	1	2
0	0	0	0	0	1	0	2	3
0	0	0	0	0	1	1	3	4
--	--	--	--	--	--	--	--	--
1	1	1	1	1	1	0	126	127
1	1	1	1	1	1	1	127	128

(3) Ratio of Output Divider (p-Value)

The ratio for the output divider can be set to values of 2, 4 and 8 according Table 8-3.
This is controlled by external pins PLLOD[1:0] to set a specific **p**-Value. The p-Value is used to set the VCO output frequency in order to get a 50% duty cycle.

Table 8-3: Ratio of Output Divider (p-Value)

External Pins		p-Value
PLLOD1	PLLOD0	
0	0	2
0	1	4
1	0	8
1	1	8

(4) VCO Frequency Selection

The parameter signals PLL_S[1:0] are used to set up the VCO to the appropriate operating frequency. Note that this is the frequency range of the VCO, and not the output frequency of the PLL. The improper setting of the parameter signals PLL_S[1:0] will result in an increase in PLL jitter.
The VCO frequency range is programmed by clamping the parameter signals PLL_S[1:0] by F091 buffers inside the UDL area.
The frequency range of the VCO is shown in Table 8-4.

Table 8-4: VCO Frequency Range

PLL_S1	PLL_S0	VCO Frequency Range
0	0	50 - 90 MHz
0	1	90 - 130 MHz
1	0	130 - 170 MHz
1	1	170 - 230 MHz

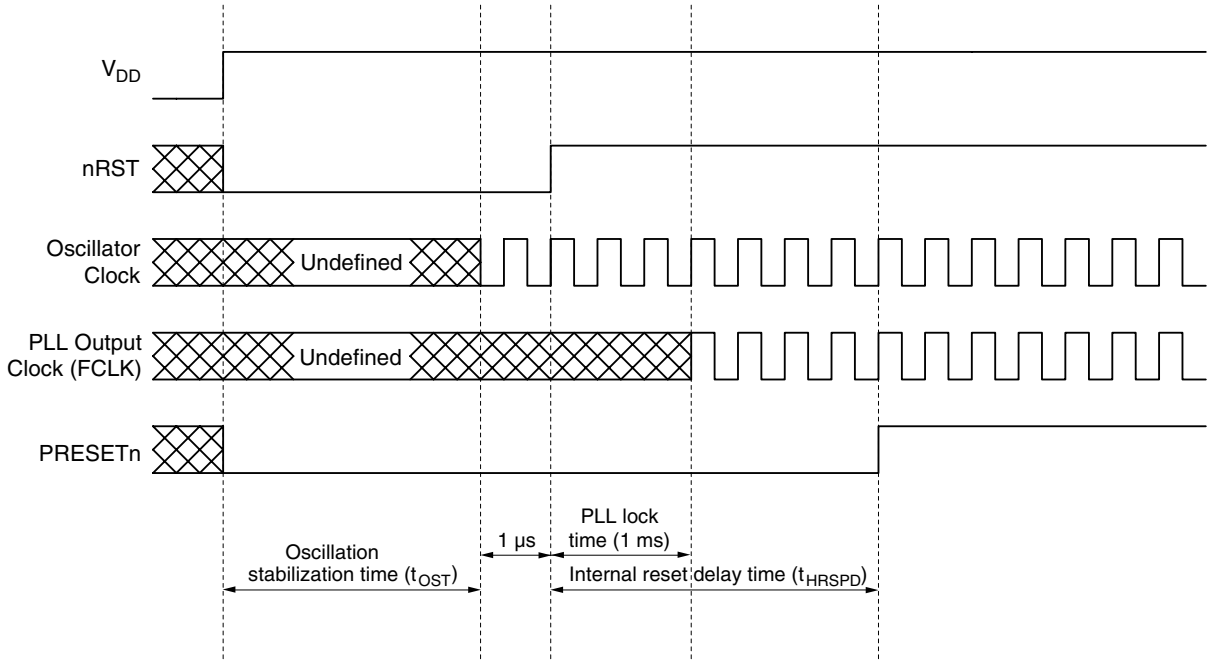
8.5 Securing Oscillation Stabilization Time

The external reset signal nRST must be held active during power-up to secure the oscillator stabilization time.
This time is given by t_{OST}, typical 10 ms. Note that the value for the oscillator stabilization time is just a typical value. It depends very much on the application (e.g. external components, PCB layout, crystal or ceramic oscillator).
After the oscillation stabilization time it needs additional 1 ms for locking the PLL. After this time the PLL outputs a stable FCLK to the UDL area.

After t_{OST} + 1 µs followed by an internal reset delay time the CPU starts program execution from reset handler address.

Figure 8-4 shows the reset and clock signals after power-up.

Figure 8-4: Reset and Clock Signals after power-up



8.6 Clock Distribution

8.6.1 Overview

The main clock HCLK of the ARM subsystem is derived from the system clock SCLK, which is provided by the UDL area. The frequency of the system clock SCLK must be in the frequency range specified for the ARM subsystem (max. 35 MHz).

The PLL output clock FCLK is not directly connected to the system clock SCLK to allow high flexibility in clocking the ARM subsystem. The PLL output clock FCLK can be directly connected to the system clock SCLK inside the UDL area, which is the easiest way to provide the ARM subsystem clock. It is also possible to have a higher output frequency of the PLL and divide this clock inside the UDL area by extra glue logic to get the ARM subsystem clock SCLK.

The clock for the APB bus inside the UDL area must be derived from the system clock SCLK for synchronous APB operation.
The APB bus signals generated by the ARM subsystem are delayed by the HCLK path, therefore the APB bus clock PCLK inside the UDL area must be delayed for the same time to allow synchronous operation of the APB bus. The delay for the APB bus clock PCLK inside the UDL area must be implemented according to the AC specification.

The multiplication rate for the PLL synthesizer is programmed by clamping the parameter signals PLL_M[4:0], PLL_N[6:0] and PLL_S[1:0] inside the UDL area.

Figure 8-5 and Figure 8-6 show the clock distribution in the SoCLite.

Figure 8-5: Clock Distribution (Synchronous Operation)

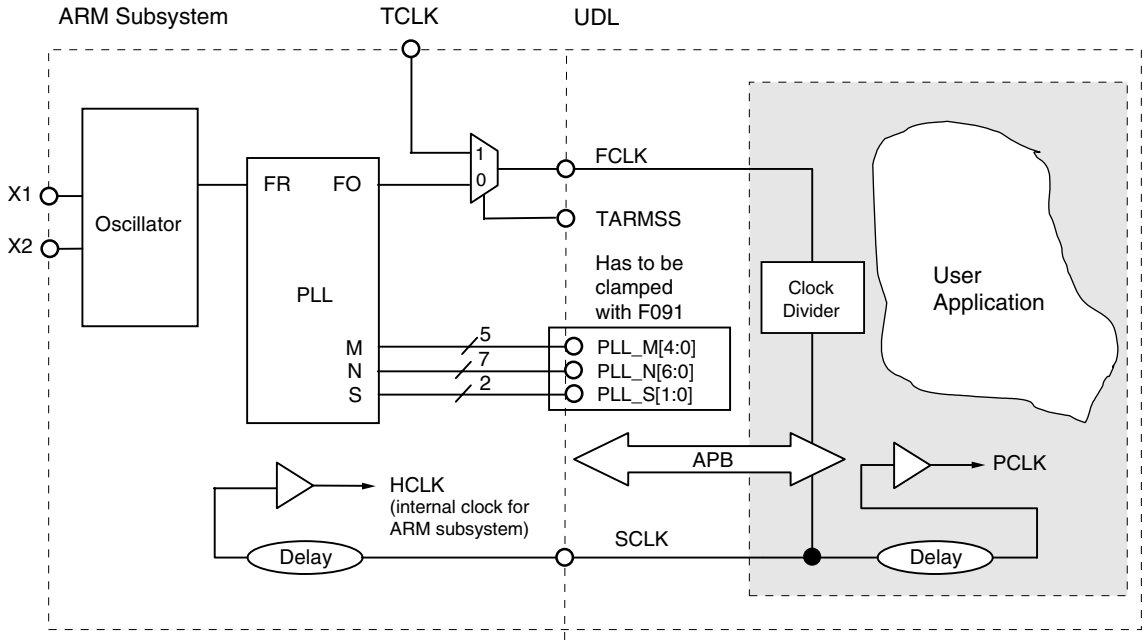
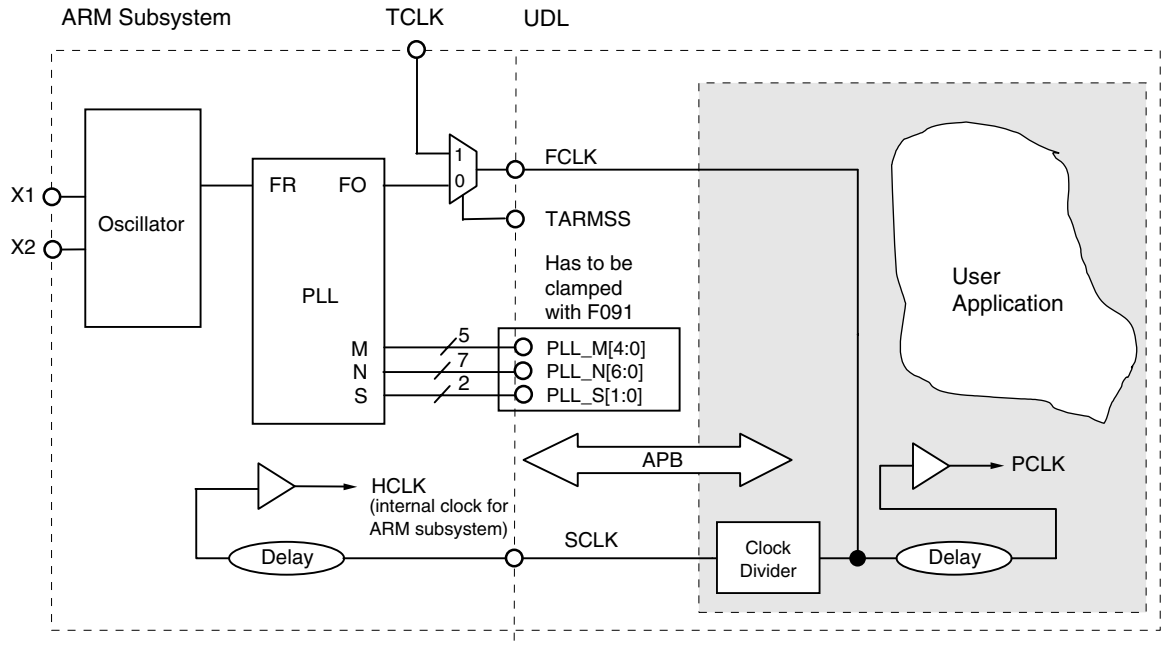


Figure 8-6: Clock Distribution (Asynchronous Operation)



8.6.2 Clock Distribution in Test Mode

In test mode, the system clock SCLK is generated from the external test clock TCLK. The test mode is entered when the UDL signal TARMSS is asserted to high level. The Table 8-5 shows which clock source is used in the different modes.

Table 8-5: Clock Sources for Operation Modes

TARMSS signal	System clock SCLK clocked from	Operation mode
0	PLL output clock	normal operation mode
1	External test clock TCLK	test mode

In test mode, the FCLK output of the ARM subsystem is always clocked from the external test clock TCLK input pin. The PLL output is not used in test mode, only in normal operation mode.

Chapter 9 Timer Function

The SoCLite is provided with one general purpose 32-bit timer. The timer is based on a free running decrementing counter with a reload register. The clock source for the timer is fixed and equal to the internal system clock (HCLK) of the ARM subsystem.

9.1 Features

- Free running 32-bit decrementing counter
- 32-bit reload register
- Interval timer based on system clock (ϕ)
- Interrupt request sources: 1

9.2 Functional Overview

The timer in the SoCLite is based on a free running 32-bit decrementing counter. The timer will be enabled by the TR bit in the timer control register (TMC). By setting this bit to 1 the initial value, containing in the timer load register (TML), is loaded into the counter and the counter starts decrementing every system clock cycle.
The timer value register (TMV) contains the current value of the counter. When this 32-bit counter reaches the value 0x0H an interrupt is generated and the counter is reloaded with the initial value in the timer load register.
After a reset the counter is loaded with the maximum value of FFFFFFFFH.

Figure 9-1 shows the block diagram of the timer.

Figure 9-1: Timer Structure

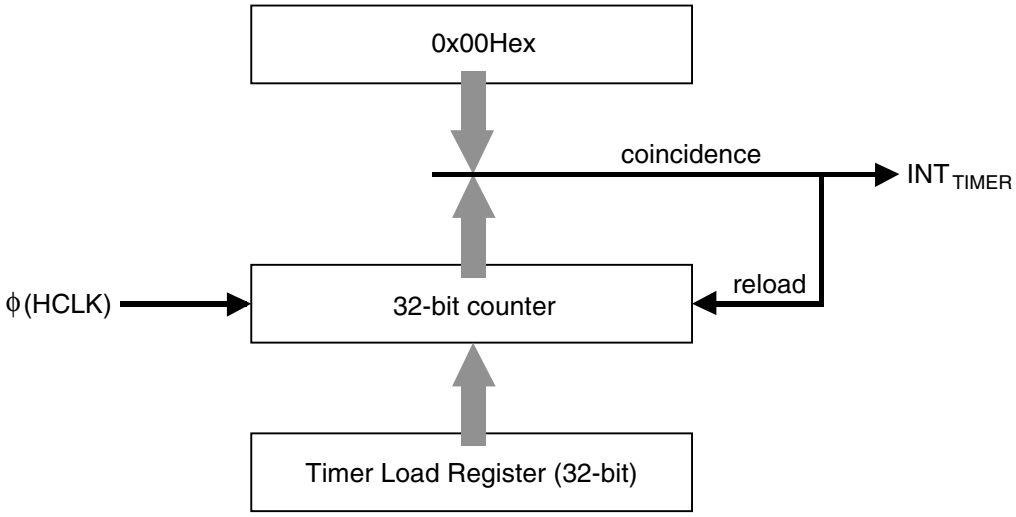
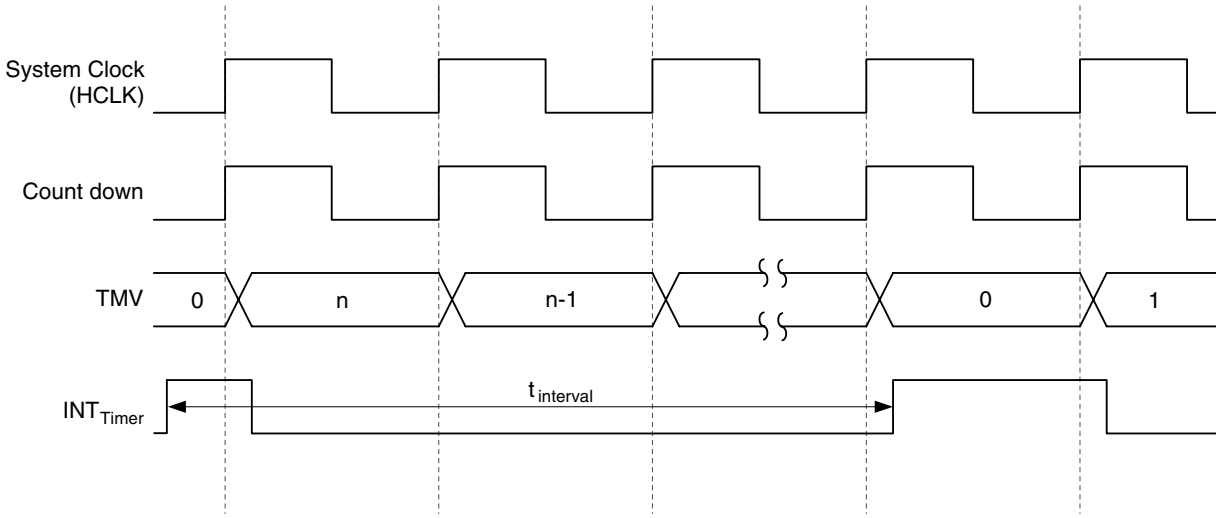


Figure 9-2 shows the operation of the timer.

Figure 9-2: Timer Operation



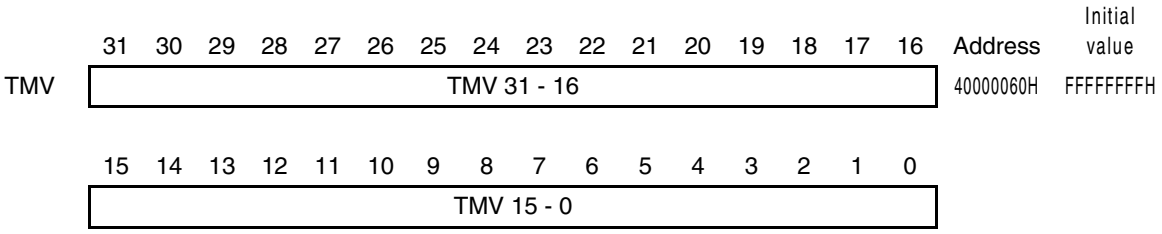
Remarks: **1.** n: Setting value of TML (00000002H to FFFFFFFFH)
2. The interval time between two interrupts is: $t_{interval}$ = (n + 1) x system clock cycle

9.3 Timer Function Registers

9.3.1 Timer Value Register (TMV)

This register contains the current value of the 32-bit counter. When the value reaches 0, this register is reloaded with the value of the timer load register (TML).
This register can be read/written.

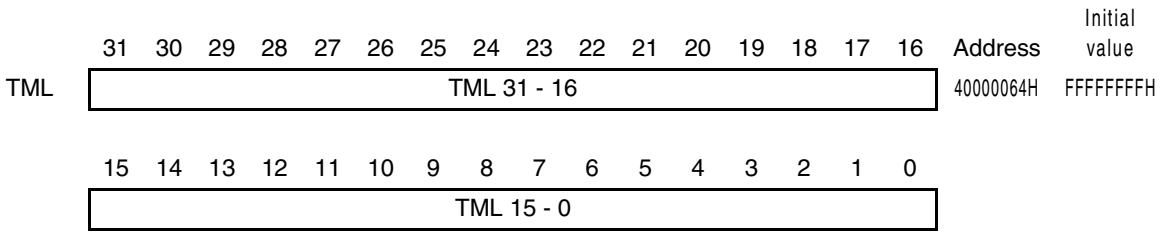
Figure 9-3: Timer Value Register (TMV)



9.3.2 Timer Load Register (TML)

This register contains the reload value to be loaded into the 32-bit counter, when the counter reaches 0H or by program code.
This register can be read/written.

Figure 9-4: Timer Load Register (TML)



Note: Never use a load value < 2_{Dec}, otherwise the timer interrupt will not occur.

9.3.3 Timer Control Register (TMC)

This register controls the operation of the 32-bit counter.
This register can be read/written.

Figure 9-5: Timer Control Register (TMC)

	7	6	5	4	3	2	1	0	Address	Initial value
TMC	0	0	0	0	0	0	0	TR	40000068H	00H

Bit Position	Bit Name	Function
0	TR	Timer run Enables or disables the timer.

TR	Operation mode
0	Timer is disabled, not counting (default).
1	Timer is enabled, timer value register is decremented every clock cycle

[MEMO]

Chapter 10 Watchdog Timer Function

The SoCLite provides a watchdog timer to secure the operation of the device. The watchdog timer has to be retriggered by the user software in certain time intervals. If the watchdog timer is not retriggered, it will generate a system reset. The clock source for the watchdog timer is fixed and equal to the system clock.

10.1 Features

- Generates reset (reset output is connected to reset controller)
- Has to be started once by software control (afterwards protected)
- Can only be stopped by reset
- Watchdog timer based on system clock (ϕ)
- Secure write access to the watchdog timer control register

10.2 Functional Overview

The watchdog timer in the SoCLite is based on a free running 12-bit decrementing counter with a reload register. The clock source of the decrement counter is the internal system clock (HCLK) of the ARM subsystem divided by a fixed prescaler of 2 or 2^{14}. The reload value is loaded into the counter register each time the watchdog timer is retriggered. If the counter register reaches the value 0x0H, the reset output is asserted thus causing a system reset.
The watchdog timer interval is determined by the prescale factor and the value in the reload register. The prescale factor and the reload value can be written only once. To protect unintentional write access, a special bit pattern must be written together with the prescale factor and the reload value. After this access to the watchdog timer control register (WDCTRL), the following steps are executed inside the watchdog timer:

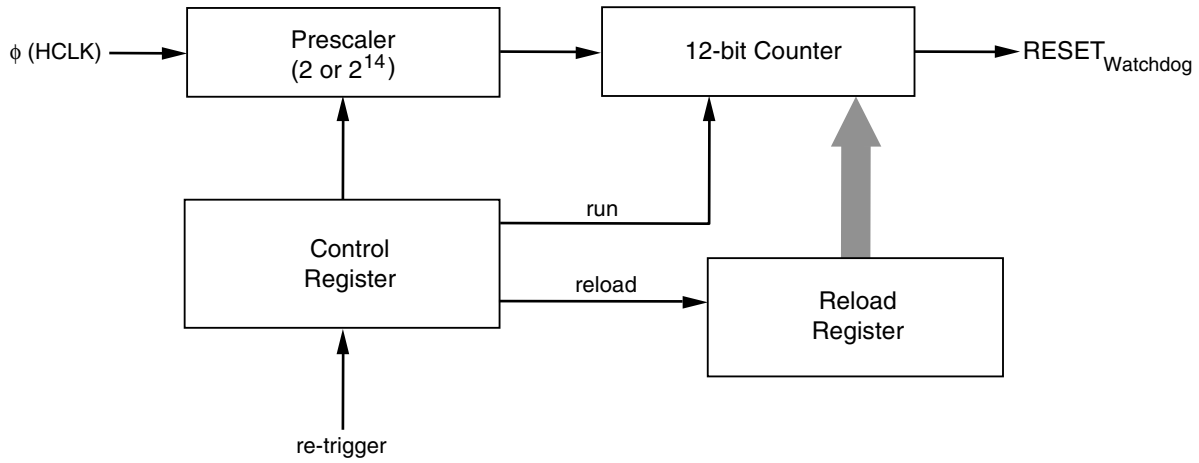
(1) The watchdog counter is loaded with the reload value.
(2) The prescale counter is set to the value configured by PSF bits in the WDCTRL register.
(3) The watchdog timer starts counting.
(4) Any further write accesses to the WDCTRL register have no effect on the prescale factor and the reload value.

To retrigger the watchdog timer, the watchdog timer control register (WDCTRL) must be written. It doesn't care which value is written, because the reload register is blocked after the first write access. The following steps are executed inside the watchdog timer for retriggering:

(1) The watchdog counter is loaded with the reload value.
(2) The prescale counter is set to the value configured by PSF bits in the WDCTRL register at the first write access.
(3) The watchdog timer continues counting.

Figure 10-1 shows the block diagram of the watchdog timer.

Figure 10-1: Watchdog Timer Structure



10.3 Watchdog Timer Function Register

10.3.1 Watchdog Timer Control Register (WDCTRL)

The value written to this register at the first time after reset determines the interval time of the watchdog timer. After the watchdog timer is running, a write of any value to this register retriggers the watchdog timer thus loading the reload value into the counter again. To secure the first write access after reset, a special bit pattern for the upper 3 bits of this register is used to start the watchdog timer.
This register can be read/written.

Figure 10-2: Watchdog Timer Control Register (WDCTRL) (1/2)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Initial value
TMV	WDSI1-0		WDE	PSF	WDRV 11-0												40000074H	1000H

Figure 10-2: Watchdog Timer Control Register (WDCTRL) (2/2)

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>15 - 14</td><td>WDSIn</td><td>Watchdog Timer Secure ID
Controls the write access to WDE, PSF and WDRV bits by a secure ID.
<table><tr><th>WDSI1</th><th>WDSI0</th><th>Operation mode</th></tr><tr><td>1</td><td>0</td><td>Write access to WDE, PSF and WDRV allowed.
The lower 14 bits of WDCTRL contain the value to be written into WDE, PSF and WDRV.</td></tr><tr><td colspan="2">others</td><td>Write access to WDE, PSF and WDRV prohibited.</td></tr></table>
Notes: 1. To write a value into the WDCTRL register, the upper 2 bits WDSI[1:0] must be set to 10_b and the lower 14 bits contain the value for WDE, PSF and WDRV bits. If the upper two bits are different from 10_b, no write access is executed.
2. Reading the WDSI bits always returns 00_b.</td></tr>
<tr><td>13</td><td>WDE</td><td>Watchdog Timer Enable Flag
Enables the watchdog timer.
<table><tr><th>WDE</th><th>Operation mode</th></tr><tr><td>0</td><td>Watchdog timer is disabled (default)</td></tr><tr><td>1</td><td>Watchdog timer is enabled</td></tr></table>
Note: Writing a 0 to this bit has no effect. The watchdog timer can only be started.</td></tr>
<tr><td>12</td><td>PSF</td><td>Prescaler Factor
This bit sets the division factor of the prescaler.
<table><tr><th>PSF</th><th>Operation mode</th></tr><tr><td>0</td><td>Prescaler is set to divide system clock by 2</td></tr><tr><td>1</td><td>Prescaler is set to divide system clock by 2^{14} (default)</td></tr></table></td></tr>
<tr><td>11 - 0</td><td>WDRV</td><td>Watchdog Timer Reload Value
These bits contain the reload value of the watchdog timer.
$$\mathbf{WDRV} = \frac{\phi \times \mathbf{WD_{Interval}}}{\mathbf{PSF}} - \mathbf{1}$$
Remarks: 1. ϕ: System clock frequency in Hz
2. $WD_{Interval}$: Interval time in seconds
3. PSF: Prescaler division factor (2 or 2^{14})</td></tr>
</table>

Chapter 11 Serial Interface Function

The SoCLite utilises one asynchronous serial interface (UART). The UART is clocked by the internal system clock (HCLK) of the ARM subsystem. The baudrate is generated from the internal system clock and is determined by the serial control register (SCTR).
After a byte has been received or transmitted completely, an interrupt is generated. The flags in the serial control register can inhibit transmitting or receiving data.

11.1 Features

- Fixed communication protocol
  - Character length: 8 bits
  - Stop bits: 1 bit
  - Start bits: 1 bit
  - Parity functions: none
- Variable baudrate rate generator, adjustable via register
- Interrupt generation for transmit and receive byte each
  - Reception completion interrupt ($INT_{UART(receive)}$)
  - Transmission completion interrupt ($INT_{UART(transmit)}$)
- Hardware support for baudrate detection (synchronization mode)
- Transfer rate: 300 bps to 2 Mbps
  (using a dedicated baud rate generator and an internal system clock of 35 MHz)
- Full-duplex communication
  - On-chip reception buffer register (RXB)
- Two-pin configuration
  - TXD: Transmit data output pin
  - RXD: Receive data input pin

11.2 Functional Overview

The UART is controlled by the serial control register (SCTR) and the serial status register (SSTA). Receive data is maintained in the serial reception data buffer register (SRXB), and transmit data is written to the serial transmission data buffer register (STXB).
Figure 11-1 shows the configuration of the asynchronous serial interface (UART).

(1) Serial control register (SCTR)

The SCTR register is a 32-bit register for specifying the operation of the asynchronous serial interface.

(2) Serial status register (SSTA)

The SSTA register contains a set of flags that indicate the status of receive and transmit operations.

(3) Serial reception data buffer register (SRXB)

SRXB is an 8-bit buffer register for holding receive data.
During a reception enabled state, receive data is transferred from the receive shift register to the SRXB, synchronized with the end of the shift-in processing of one frame.
Also, the reception completion interrupt request ($INT_{UART(receive)}$) is generated by the transfer of data to the SRXB.

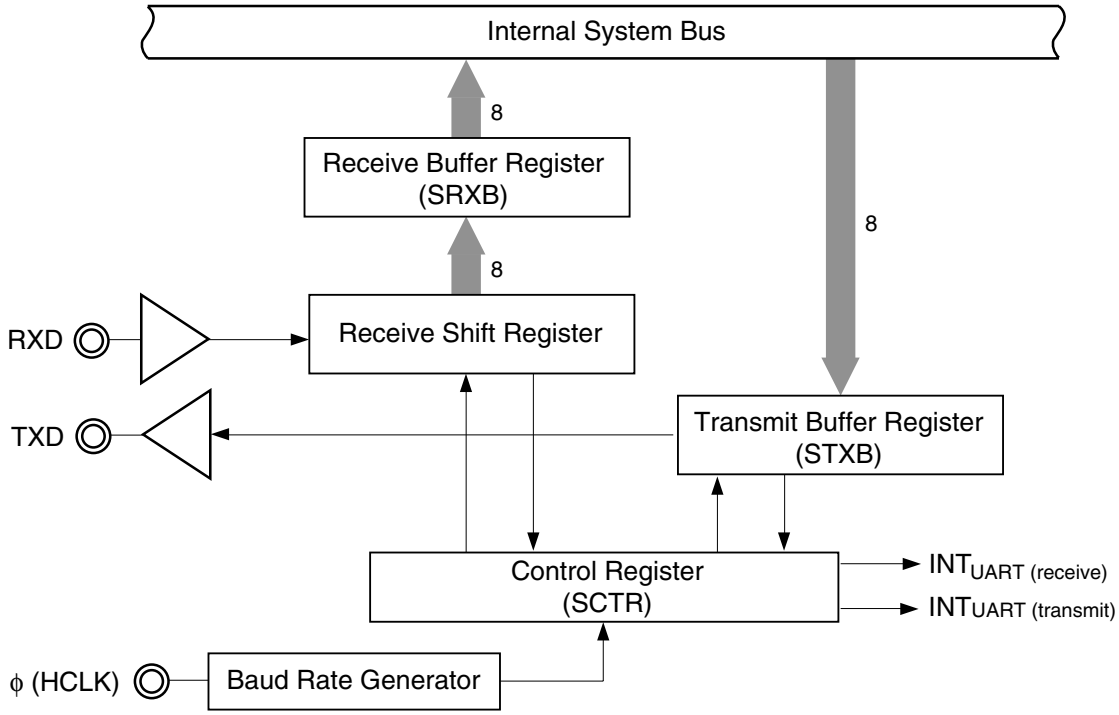
(4) Serial transmission data buffer register (STXB)

STXB is an 8-bit buffer for transmit data. A transmit operation is started by writing transmit data to STXB. When one byte of data is written to the STXB, the data is output from the TXD pin. The transmission completion interrupt request ($INT_{UART(transmit)}$) is generated synchronized with the completion of transmission of one frame.
This register converts the parallel data, received from the internal system bus, to serial data.

(5) Receive shift register

This shift register converts the received serial data to parallel data. After receiving one byte, followed by the stop bit, the receive data is transferred to the serial reception data buffer register (SRXB).
This register cannot be directly manipulated.

Figure 11-1: Asynchronous Serial Interface (UART)



11.3 UART Operation

The UART provides two operation modes:

- Synchronization mode
- Normal operation mode

The synchronization mode is used to adjust the baudrate of the UART to communicate with an external partner device. In the normal operation mode reception and transmission of data will be carried out.

11.3.1 Synchronization Mode

The UART can be switched into synchronization mode to support baudrate recognition. The setting of the SYNC bit in the serial control register (SCTR) activates the synchronization mode.

To adjust the baudrate of the UART to an external partner device (e.g. PC with terminal program), the UART is switched to synchronization mode. In the synchronization mode the baudrate is evaluated by an appropriate software.
The external partner device shall send a character to the UART with a certain number of low bits in this data byte (e.g. character “n”). The internal counter will count during the low bits of the data byte occur on the RXD pin. Therefore the counter value should be polled by the appropriate software until no further change of the counter is recognized. After receiving the whole character (one data byte), the counter contains the value for the formula to calculate the actual baudrate. The calculation has to be done in the appropriate user software. The formula is as follows:

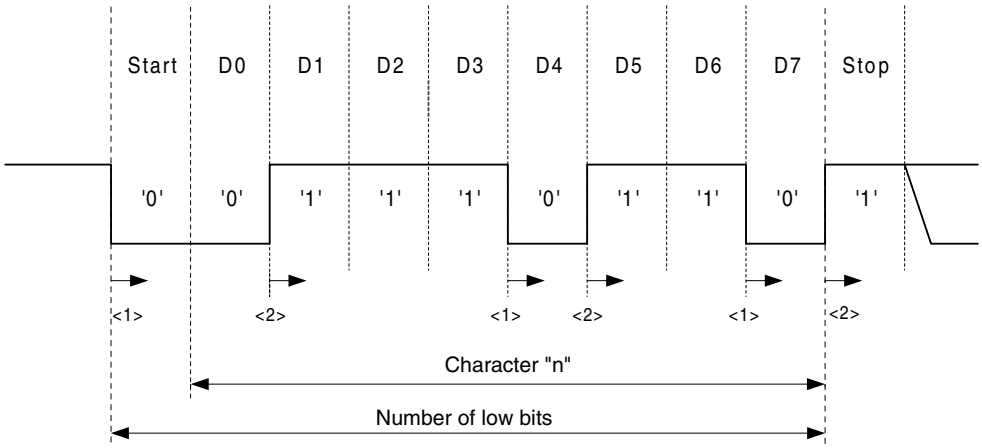
Figure 11-2: Formula to determine BDV value

$$\text{BDV value} = \frac{\text{Counter value}}{\text{Number of low bits x 8}}$$

Remarks: **1.** BDV value: Value for the BDV bits in normal operation mode to calculate the baudrate
2. Counter value: Actual value of BDV bits in synchronization mode
3. Number of low bits: Number of low bits in the character, which has been sent (plus start bit)

A recommended character to determine the baudrate shall have a number of low bits between 3 and 4 to precise the calculation and avoid an overflow of the counter (including the start bit).
Figure 11-3 shows the synchronization with the character “n” (case sensitive).

Figure 11-3: Synchronization Character



Remarks: **1.** <1>: BDV counter counts up.
2. <2>: BDV counter stops.

In the synchronization mode, the following processing has to be performed:

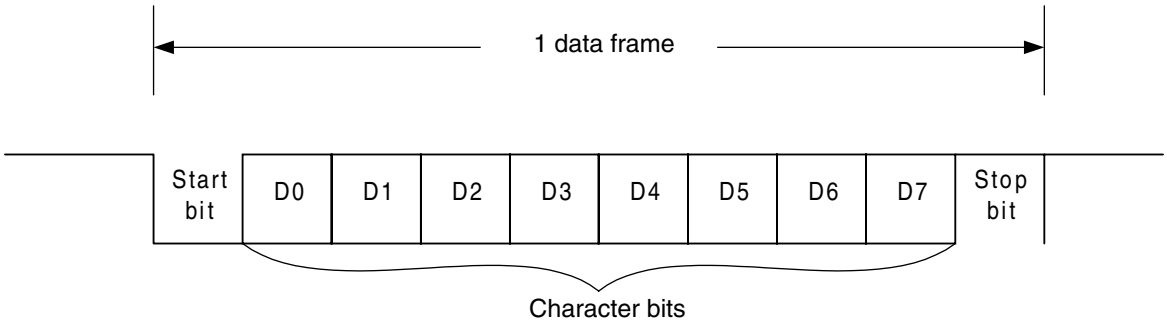
(a) BDV bits in SCTR register are switched to counter mode (up-counter).
(b) BDV counter is enabled whenever the RXD line is'0' (see Figure 11-3).
(c) BDV counter is based on system clock (SCLK).
(d) The external partner device should send a data byte with a certain number of low bits.
(e) As long as the RXD pin is low (='0'), BDV counter counts up.
(f) The software polls the BDV counter value as long as the certain number of low bits have been received.
(g) After receiving the complete data byte with the certain number of low bits, the counter should not change any more. Then the counter represents the value for the formula to calculate the baudrate.
(h) At least two measurements are necessary, because the first one may start just in the middle of the start bit.

11.3.2 Normal Operation Mode

(1) Data Format

Full-duplex serial data transmission and reception can be performed.
The transmit/receive data format consists of one data frame containing a start bit, character bits, no parity bit, and one stop bit as shown in Figure 11-4.
The data frame is fixed and can not be changed.
Also, data is transferred with LSB first.

Figure 11-4: Asynchronous Serial Interface Transmit/Receive Data Format



- Start bit ··· 1 bit
- Character bits ··· 8 bits
- Parity bit ··· no parity
- Stop bits ··· 1 bit

(2) Transmit Operation

When TXE bit is set to 1 in the SCTR register, transmission is enabled, and the transmit operation is started by writing transmit data to the serial transmission data buffer register (STXB).

(a) Transmission enabled state

This state is set by the TXE bit in the SCTR register.

- TXE = 1: Transmission enabled state
- TXE = 0: Transmission disabled state

Since UART does not have a CTS (transmission enabled signal) input pin, it should be taken care whether the destination is in a reception enabled state.

(b) Starting a transmit operation

In transmission enabled state, a transmit operation is started by writing transmit data to the serial transmission data buffer register (STXB). When a transmit operation is started, the serial transmission data buffer register (STXB) outputs data to the TXD pin (the transmit data is transferred sequential starting with the start bit). The start bit and stop bit are added automatically.

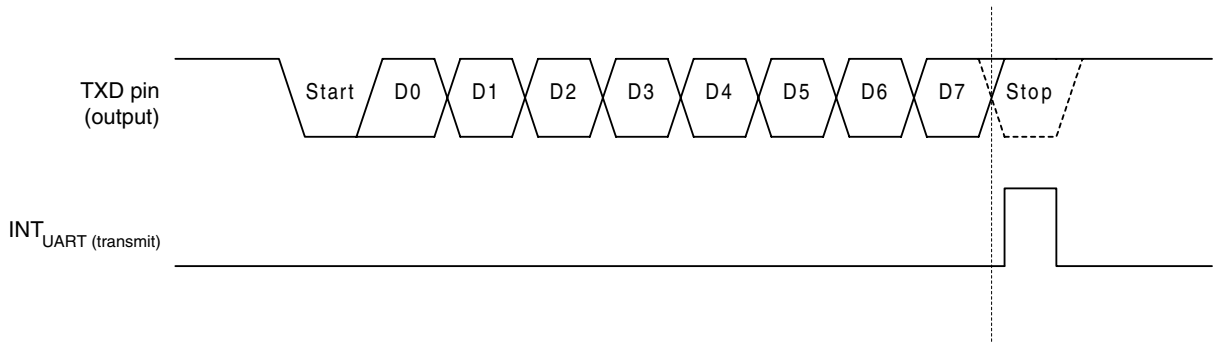
(c) Transmission interrupt request

When the serial transmission data buffer register becomes empty, a transmission completion interrupt request ($INT_{UART(transmit)}$) is generated. The $INT_{UART(transmit)}$ interrupt is generated when the last stop bit is output.

If the next data to be transmitted has not been written to the STXB register, the transmit operation is suspended.

Caution: Normally, when the serial transmission data buffer register becomes empty, a transmission completion interrupt ($INT_{UART(transmit)}$) is generated. However, no transmission completion interrupt ($INT_{UART(transmit)}$) is generated if the serial transmission data buffer register becomes empty due RESET input.

Figure 11-5: Asynchronous Serial Interface Transmission Completion Interrupt Timing



(3) Continuous Transmission Operation

The UART can write the next transmit data to the STXB register when the TDB bit in the serial status register (SSTA) is cleared. This enables an efficient transmission rate to be realized by continuously transmitting data even during the $INT_{UART(transmit)}$ interrupt service after the transmission of one data frame.

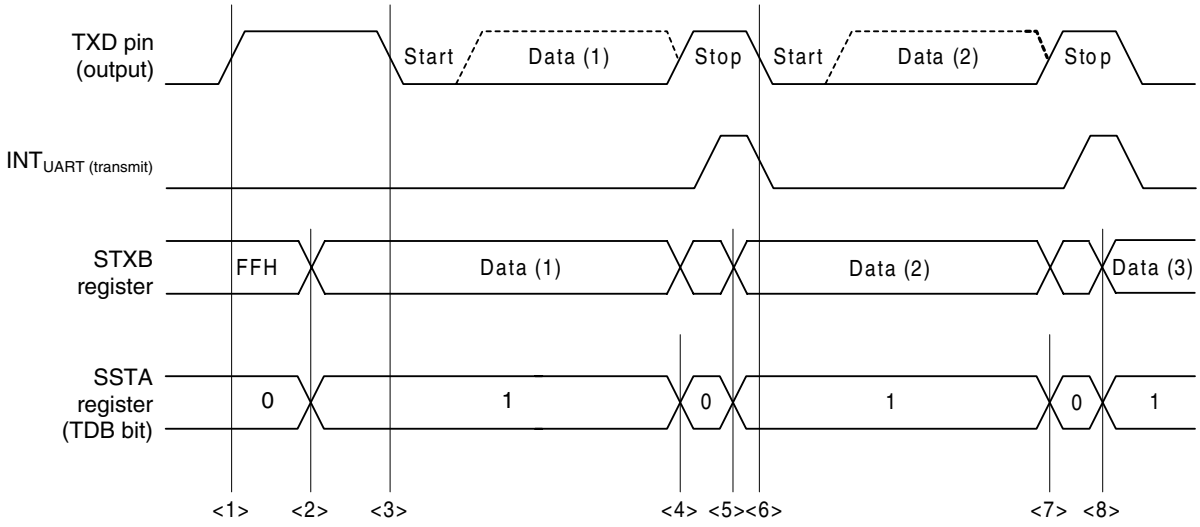
When continuous transmission is performed, data should be written after referencing the serial status register (SSTA) to confirm the transmission status and whether or not data can be written to the STXB register.

Caution: Transmit data should be written when the TDB bit is 0. If these action is performed at other times, the transmit data cannot be guaranteed.

(a) Starting Procedure

Figure 11-6 shows the procedure to start continuous transmission.

Figure 11-6: Continuous Transmission Starting Procedure

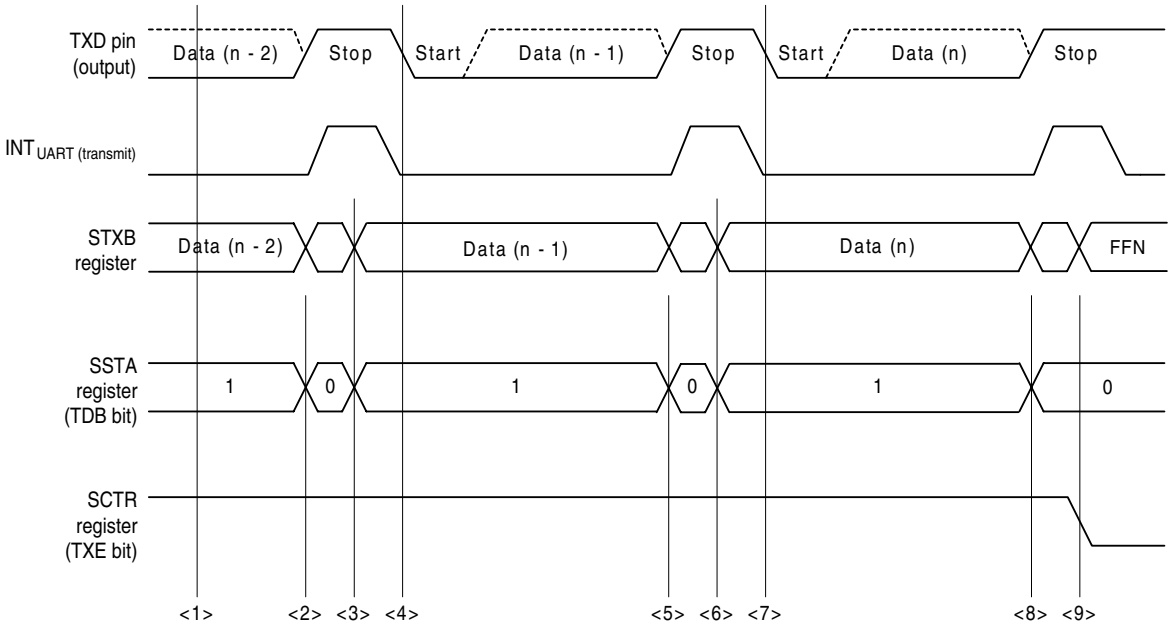


Transmission Start Procedure	Internal Operation	SSTA Register
		TDB bit
<1> Set transmission mode		0
<2> Write data (1) to STXB register		1
	<3> Generate start bit Start data (1) transmission	1
	<<Transmission in progress>>	0
	<4> Generate transmission completion interrupt ($INT_{UART(transmit)}$)	
<5> Read SSTA register (confirm that TDB bit = 0) Write data (2) to STXB register		1
	<6> Generate start bit Start data (2) transmission	
	<<Transmission in progress>>	0
	<7> Generate transmission completion interrupt ($INT_{UART(transmit)}$)	
<8> Read SSTA register (confirm that TDB bit = 0) Write data (3) to STXB register		1

(b) Ending Procedure

Figure 11-7 shows the procedure to end continuous transmission.

Figure 11-7: Continuous Transmission End Procedure



Transmission End Procedure	Internal Operation	SSTA Register
		TDB bit
	<1> Transmission of data (n - 2)is in progress	1
	<2> Generate transmission completion interrupt ($INT_{UART(transmit)}$)	0
<3> Read SSTA register (confirm that TDB bit = 0) Write data (n - 1) to STXB register		1
	<4> Generate start bit Start data (n - 1) transmission	
	<<Transmission in progress>>	0
	<5> Generate transmission completion interrupt ($INT_{UART(transmit)}$)	
<6> Read SSTA register (confirm that TDB bit = 0) Write data (n) to STXB register		1
	<7> Generate start bit Start data (n) transmission	
	<<Transmission in progress>>	0
	<8> Generate transmission completion interrupt ($INT_{UART(transmit)}$)	
<9> Read SSTA register (confirm that TDB bit is still 0)		
There is no write data left in STXB register. Clear (0) the TXE bit of SCTR register	Initialize internal circuits	

(4) Receive Operation

An awaiting reception state is set by setting RXE bit to 1 in the SCTR register. To start a receive operation, a start bit has to be detected first. The start bit is detected by sampling RXD pin (8 times oversampling). When the receive operation begins, serial data is stored sequential in the receive shift register according to the baud rate that was set.
A reception completion interrupt ($INT_{UART(receive)}$) is generated each time the reception of one frame of data is completed. Normally, the receive data is transferred from the receive buffer register (SRXB) to memory by this interrupt servicing.

(a) Reception enabled state

The receive operation is set to reception enabled state by setting the RXE bit in the SCTR register to 1.

- RXE bit = 1: Reception enabled state
- RXE bit = 0: Reception disabled state

In reception disabled state, the reception hardware stands by in the initial state. At this time, the contents of the receive buffer register (SRXB) are retained, and no reception completion interrupt is generated.

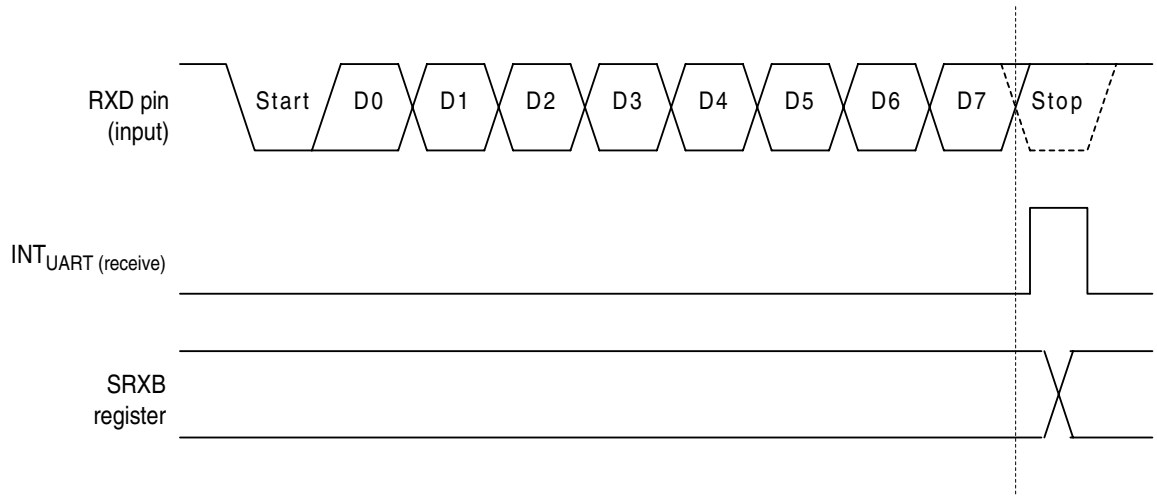
(b) Starting a receive operation

A receive operation is started by the detection of a start bit.
The RXD pin is sampled according to the serial clock baud rate generator.

(c) Reception interrupt request

When RXE bit in the SCTR register is set and the reception of one frame of data is completed (the stop bit is detected), a reception completion interrupt ($INT_{UART(receive)}$) is generated and the receive data within the receive shift register is transferred to the SRXB register at the same time.
If an overrun error (RDO) occurs, the receive data at that time is not transferred to the serial reception data buffer register (SRXB), but no reception completion interrupt ($INT_{UART(receive)}$) is generated.
If the RXE bit is reset (0) during a receive operation, the receive operation is immediately stopped. The contents of the serial reception data buffer register (SRXB) and of the serial status register (SSTA) are reset as well, and no reception completion interrupt ($INT_{UART(receive)}$) is generated.
No reception completion interrupt is generated when RXE bit = 0 (reception is disabled).

Figure 11-8: Asynchronous Serial Interface Reception Completion Interrupt Timing

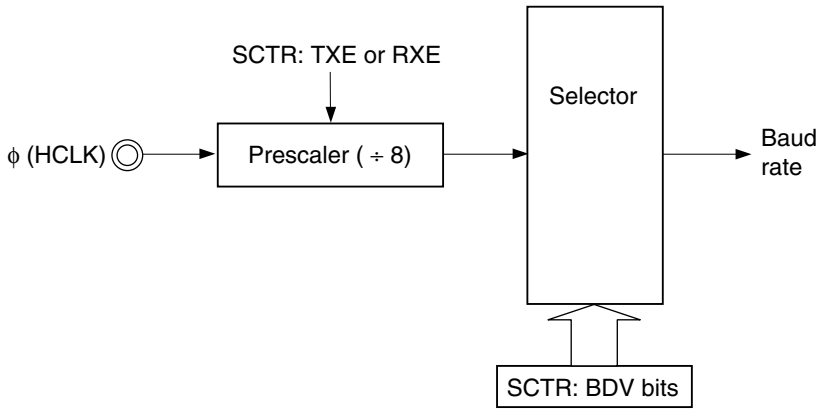


11.4 Baud Rate Generator

A dedicated baud rate generator, which consists of a source clock selector and an 8-bit counter, generates the serial clocks during transmission/reception at the UART. Its output is selected as the serial clock for transmission and reception.

11.4.1 Baud Rate Generator Configuration

Figure 11-9: Baud Rate Generator Configuration



When TXE or RXE bit = 1 in the SCTR register, the clock selected according to the BDV bits of the SCTR register is supplied to the transmission/reception unit. This clock is based on the internal system clock (HCLK) of the ARM subsystem.

11.4.2 Baud Rate

The baud rate is the value obtained according to the following formula.

$$\text{Baud rate} = \frac{\phi\ (\text{SCLK})}{8 \times (\text{BDV value} + 1)}$$

Remarks: **1.** ϕ (SCLK): System clock (SCLK) feed into ARM subsystem (equal to HCLK)
2. BDV value: Value of BDV bits in SCTR register

(1) Baud Rate Error

The baud rate error is obtained according to the following formula.

$$\text{Error} = \left(\frac{\text{Actual baud rate (baud rate with error)}}{\text{Desired baud rate (normal baud rate)}} - 1\right) \times 100 \quad [\%]$$

Cautions: 1. Make sure that the baud rate error during transmission does not exceed the allowable error of the reception destination.
2. Make sure that the baud rate error during reception is within the allowable baud rate range during reception, which is described in (3) error tolerance and sampling errors.

Example: Basic clock frequency = 30 MHz = 30,000,000 Hz
Settings of BDV bits in SCTR register = 00000001100001_b (= 97_{dec})
Target baud rate = 38,400 bps

Baud rate = 30 M / (8 × (97 + 1))
= 30000000 / (8 × 98)
= 38,265 [bps]

Error = (38265 / 38400 - 1) × 100
= -0.35 [%]

(2) Baud Rate Setting Examples

Table 11-1: Baud Rate Generator Setting Data

Baud Rate [bps]	ϕ = 35 MHz		ϕ = 30 MHz		ϕ = 20 MHz		ϕ = 10 MHz	
	BDV	ERR	BDV	ERR	BDV	ERR	BDV	ERR
300	14582	0.002	12499	0	8332	0.004	4165	0.02
600	7290	0.009	6249	0	4165	0.02	2082	0.02
1200	3644	0.02	3125	-0.03	2082	0.02	1040	0.06
2400	1821	0.05	1561	0.03	1040	0.06	519	0.16
4800	910	0.05	780	0.03	519	0.16	259	0.16
9600	454	0.16	389	0.16	259	0.16	129	0.16
19200	227	-0.06	194	0.16	129	0.16	64	0.16
38400	112	-0.06	97	-0.35	64	0.16	32	-1.36
76800	56	-0.06	48	-0.35	32	-1.36	15	1.73
153600	27	1.73	23	1.73	15	1.73	7	1.73

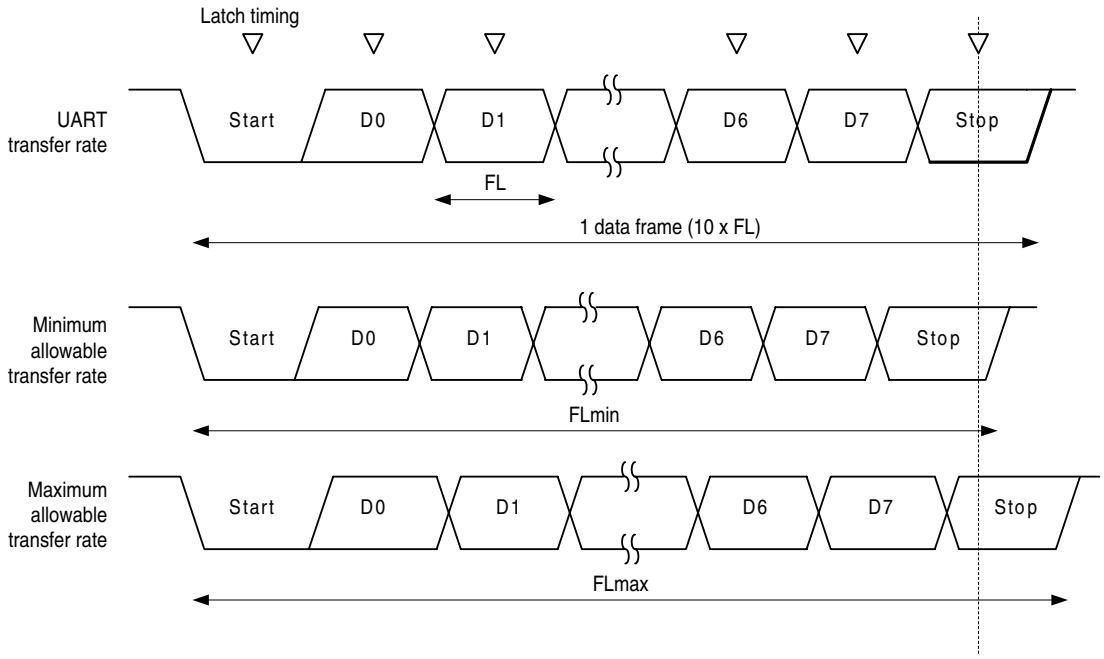
Remark: ϕ: System clock frequency (SCLK)
BDV: Setting value of BDV bits in SCTR register
ERR: Baud rate error [%]

(3) Error Tolerance and Sampling Errors

The degree to which a discrepancy from the transmission destination's baud rate is allowed during reception is shown below.

Caution: The equations described below should be used to set the baud rate error during reception so that it always is within the allowable error range.

Figure 11-10: Allowable Baud Rate Range During Reception



As shown in Figure 11-10, after the start bit is detected, the receive data latch timing is determined according to an 8 times oversampling. If all data up to the final data (stop bit) is in time for this latch timing, the data can be received normally.
Applying this to 10-bit reception is, theoretically, as follows.

$$FL = BR^{-1}$$

BR: UART baud rate
FL: 1-bit data length

When the latch timing margin is made 3 sample clocks to recognize the last bit, the minimum allowable transfer rate (FLmin) is as follows.

$$FLmin = 10 \times FL - \frac{3}{8} \times FL = 9,625 \times FL$$

Therefore, the transfer destination's maximum baud rate (BRmax) that can be received is as follows.

$$BRmax = \left(\frac{FLmin}{10}\right)^{-1} = 1,038 \times BR$$

Similarly, the maximum allowable transfer rate (FLmax) can be obtained as follows.

$$FLmax = 10 \times FL + \frac{3}{8} \times FL = 10,375 \times FL$$

Therefore, the transfer destination's minimum baud rate (BRmin) that can be received is as follows.

$$BRmin = \left(\frac{FLmax}{10}\right)^{-1} = 0,964 \times BR$$

The error tolerance of the UART and the transfer destination can be obtained as follows from the expressions described above for computing the minimum and maximum baud rate values.

- Maximum allowable baud rate error: +3.8%
- Minimum allowable baud rate error: -3.6%

11.5 Serial Interface Function Registers

11.5.1 Serial Reception Data Buffer Register (SRXB)

The SRXB register contains the data byte, which has been received. If the next receive operation is completed before the data is read, an overrun error occurs. The new received data byte is discarded. When the SRXB register is read, the RDV bit inside the serial status register (SSTA) is cleared. This register is read only.

Figure 11-11: Serial Reception Data Buffer Register (SRXB)

	7	6	5	4	3	2	1	0	Address	Initial value
SRXB	RXB 7 - 0								40000040H	00H

Note: The SRXB register is cleared when reception is disabled (RXE bit in SCTR register is switched from “1” to “0”).

11.5.2 Serial Transmission Data Buffer Register (STXB)

The STXB register contains the data byte to be transmitted.
This register can be read/written.

Figure 11-12: Serial Transmission Data Buffer Register (STXB)

	7	6	5	4	3	2	1	0	Address	Initial value
STXB	TXB 7 - 0								40000044H	00H

Note: The STXB register can only be written when transmission is enabled (TXE bit in SCTR register is set) and no transmission is in progress.

11.5.3 Serial Status Register (SSTA)

The SSTA register contains the status information for serial receive and transmit operations. This register can be read/written.

Figure 11-13: Serial Status Register (SSTA)

	7	6	5	4	3	2	1	0	Address	Initial value
SSTA	0	0	0	TDB	0	0	RDO	RDV	40000048H	00H

Bit Position	Bit Name	Function
4	TDB	Transmit Data Busy This bit indicates if transmit data can be written to the STXB register. TDB: 0 — Transmit data can be written into STXB register. TDB: 1 — The current data in STXB is in progress to be transmitted, therefore write access to STXB register has no effect. UART transmit interrupt request is triggered when this bit changes from '1' to '0'.
1	RDO	Receive Data Overflow This bit indicates if an overrun has been occurred. RDO: 0 — No receive data overflow has occurred. RDO: 1 — A receive data overflow has occurred. One data byte got lost, because SRXB register was not read out before the next data byte arrives. **Note:** RDO bit can be cleared by writing '1'. Writing '0' has no effect.
0	RDV	Receive Data Valid This bit indicates that a new data byte has been received. RDV: 0 — No new data byte received. RDV: 1 — Received data byte is available in SRXB register. UART receive interrupt request is triggered on each data byte received. This bit is cleared when: • SRXB register is read • Reception is disabled (RXE bit in SCTR register switched from '1' to '0' • Receive data overflow RDO bit is cleared

11.5.4 Serial Control Register (SCTR)

The SCTR register controls the serial interface operations. The SCTR register determines if the UART is operating in synchronization mode or normal operation mode. It also contains the divider value, which determines the baud rate. In synchronization mode, the BDV bits are representing the counter value. This register can be read/written.

Figure 11-14: Serial Control Register (SCTR) (1/2)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	Address	Initial value
SCTR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SYNC	4000004CH	00000000H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE	RXE	BDV													

Figure 11-14: Serial Control Register (SCTR) (2/2)

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>16</td><td>SYNC</td><td>Synchronization Mode
This bit determines the operation mode of the UART.
<table><tr><th>SYNC</th><th>Operation mode</th></tr><tr><td>0</td><td>UART is in normal operation mode (default).</td></tr><tr><td>1</td><td>UART is in synchronization mode.</td></tr></table></td></tr>
<tr><td>15</td><td>TXE</td><td>Transmit Enable
This bit enables transmit operation of the UART.
<table><tr><th>TXE</th><th>Operation mode</th></tr><tr><td>0</td><td>Transmit unit of the UART is disabled (default).</td></tr><tr><td>1</td><td>Transmit unit of the UART is enabled.</td></tr></table></td></tr>
<tr><td>14</td><td>RXE</td><td>Receive Enable
This bit enables receive operation of the UART.
<table><tr><th>RXE</th><th>Operation mode</th></tr><tr><td>0</td><td>Receive unit of the UART is disabled (default).</td></tr><tr><td>1</td><td>Receive unit of the UART is enabled</td></tr></table></td></tr>
<tr><td>13 - 0</td><td>BDV</td><td>Baud Rate Divider Value
These bits determine the baud rate of the UART or represent the counter value dependent on the SYNC bit.
<table><tr><th>SYNC</th><th>Function of BDV bits</th></tr><tr><td>0</td><td>Normal operation mode:
The BDV bits determine the baud rate of the UART. The BDV value (decimal) is obtained from the following formula:
$$\mathbf{BDV} = \frac{\phi\ \mathbf{(SCLK)}}{\mathbf{8 \times Baudrate}} - \mathbf{1}$$
ϕ (SCLK): Unit in Hz
Baud rate: Unit in 1/s
Note: BDV value should be set to 1 at least.</td></tr><tr><td>1</td><td>Synchronization mode:
The BDV bits represent the counter value. The counter counts up whenever the RXD input line is’0’.</td></tr></table></td></tr>
</table>

11.5.5 Serial Reception Data Buffer Register for Emulation (SRXBEMU)

The SRXBEMU register is identical to the SRXB register. The difference is that the RDV bit inside the serial status register (SSTA) is not affected by a read to the SRXBEMU register.
This register is read only.

Figure 11-15: Serial Reception Data Buffer Register for Emulation (SRXBEMU)

	7	6	5	4	3	2	1	0	Address	Initial value
SRXBEMU	RXB 7 - 0								40000050H	00H

Note: The SRXBEMU register is cleared when reception is disabled (RXE bit in SCTR register is switched from “1” to “0”).

11.6 Interrupt Requests

The following two types of interrupt requests are generated from UART.

- Reception completion interrupt ($INT_{UART(receive)}$)
- Transmission completion interrupt ($INT_{UART(transmit)}$)

The reception completion interrupt has a higher default priority than the transmission completion interrupt.

Table 11-2: Generated Interrupts and Default Priorities

Interrupt	Priority
Reception completion	1
Transmission completion	2

(1) Reception Completion Interrupt ($INT_{UART(receive)}$)

When reception is enabled, a reception completion interrupt is generated when data is shifted into the reception shift register and transferred to the serial reception data buffer register (SRXB). When reception is disabled, no reception completion interrupt is generated.

(2) Transmission Completion Interrupt ($INT_{UART(transmit)}$)

A transmission completion interrupt is generated when one frame of transmit data containing an 8-bit character is shifted out from the serial transmission data buffer register.

Chapter 12 Remap/Pause Controller Function

The SoCLite comprises a remap/pause controller. The remap/pause controller allows to control the boot up memory map and to switch the SoCLite into a low power (pause mode) to wait for interrupt state.

The remap/pause controller consists of two independent parts:

- The remap control unit to map the internal RAM at the bottom of the memory map
- The pause control unit to set the CPU into a power saving mode (pause mode)

12.1 Remap/Pause Controller Operation

The remap/pause controller provides two operation modes:

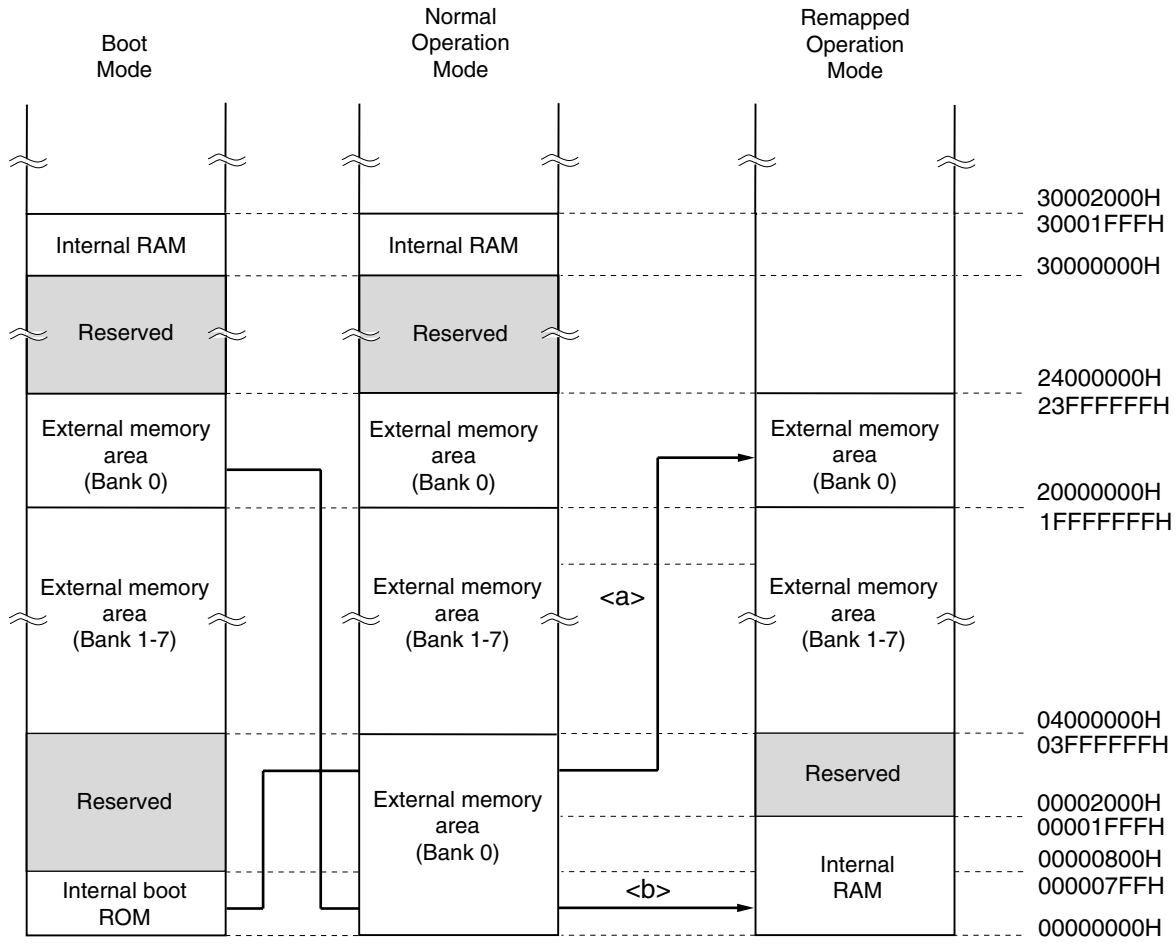
- Remapped operation mode
- Pause mode

Both modes can be set independent from each other. The different modes are controlled by the remap/pause control register (RMP).

12.1.1 Remapped Operation Mode

The internal RAM will be mapped to the bottom of the memory map at address location 00000000H in the remapped operation mode.
This is achieved by setting the RM bit in the remap/pause control register (RMP). After the RM bit is set once, it can not be cleared any more except during a reset. Therefore, if the internal RAM is remapped to the bottom of the memory map, it can never be remapped again except by a reset.
Figure 12-1 shows the remapping in detail.

Figure 12-1: Remapped Memory Map



The SoCLite starts in normal operating mode or in boot mode. From these modes the switch to the remapped operation mode is possible. The following procedures have to be done depending on the starting operation mode, when switching to remapped operation mode:

(1) Switch from normal operation mode to remapped operation mode:

(a) Fetch reset vector from address 00000000H at power on (alias of ROM).
(b) Execute the reset vector to jump to real address of next ROM instruction, mirrored ROM in upper memory location (see Figure 12-1 step <a>). This is done by loading the next instruction into program counter; LDR PC, =0x20000004.
(c) Write to the remap/pause control register (RMP) and set RM bit. This maps the internal RAM to the bottom of the memory map, starting at 00000000H and the SoCLite is running in remapped operation mode.
(d) Copy vector table from ROM to internal RAM (see Figure 12-1 step <b>).

(2) Switch from boot mode to remapped operation mode:

(a) Fetch reset vector from address 00000000H at power on (internal ROM).
(b) Execute bootloader routine from internal ROM to download programming routine into internal or external RAM.
(c) Execute programming routine from internal or external RAM to program external Flash (e.g. at external memory area bank 0).
(d) Jump to real address of next program memory instruction (e.g. Flash at external memory bank 0), (see Figure 12-1 step <a>). This is done by loading the next instruction into program counter; LDR PC, =0x20000004.
(e) Write to the remap/pause control register (RMP) and set RM bit. This maps the internal RAM to the bottom of the memory map, starting at 00000000H and the SoCLite is running in remapped operation mode.
(f) Copy vector table from program memory (e.g. Flash at external memory bank 0) to internal RAM (see Figure 12-1 step <b>).

Note: After executing the command to set the RM bit in the RMP register, 6 internal system clock cycles (HCLK) elapse until the internal RAM is remapped to the bottom of the memory map at address location 00000000H (this is equivalent to 6 NOP commands).

12.1.2 Pause Mode

The pause mode is enabled by setting the PA bit in the remap/pause control register (RMP) to switch the SoCLite in a power saving state. In this mode the SoCLite CPU is prevented from reading or writing data, or fetching opcodes - the SoCLite CPU is paused.
The pause mode is released whenever an interrupt occurs activated by the interrupt controller or the fast interrupt, generated by the UDL area.
If an interrupt occurs, the PA bit inside the remap/pause control register (RMP) is cleared and the pause mode is released.

After executing the command to set the PA bit in the remap/pause control register (RMP), 5 internal system clock cycles (HCLK) elapse until the pause mode is entered and the SoCLite CPU core is stopped. Within this time the SoCLite CPU core still executes program code.

The pause mode is released one internal system clock cycle (HCLK) after an interrupt is detected active. In case of a fast interrupt detection (nFIQ), the SoCLite CPU recognizes the fast interrupt several internal system clock cycles (HCLK) later. Therefore it must be ensured, that the fast interrupt signal is active until the fast interrupt exception handler in the programming code is executed.

Table 12-1 shows the operating status of the single modules in the ARM subsystem during the pause mode.

Table 12-1: Operating Status in Pause Mode

Modules in ARM subsystem	Operation Status
Oscillator	Operates
PLL	Operates
Internal system clocks	Operates
APB clock in UDL (PCLK)	Operates
CPU	Operates, but no code fetch or data transfer is possible
Peripherals in ARM subsystem	Operates
Watchdog timer	Operates

Notes: **1.** All internal data is retained before entering the pause mode, such as CPU registers, status, data and internal RAM.
2. Special caution must be taken if the watchdog timer is running while the pause mode is entered. It must be taken care that the watchdog timer is retriggered frequently to prevent an unintended reset.

12.2 Remap/Pause Controller Function Register

12.2.1 Remap/Pause Control Register (RMP)

The RMP register controls the remapped and pause operation modes.
This register can be read/written.

Figure 12-2: Remap/Pause Control Register (RMP)

	7	6	5	4	3	2	1	0	Address	Initial value
RMP	0	0	0	0	0	0	PA	RM	40000070H	00H

Bit Position	Bit Name	Function
1	PA	Pause Mode This bit enables pause mode operation of the SoCLite. PA: 0 — Pause mode operation disabled. PA: 1 — Pause mode operation enabled.
0	RM	Remapped Mode This bit enables remapped operation mode of the SoCLite. RM: 0 — Remapped operation mode disabled. Internal ROM or rather external ROM/Flash is located at the bottom of the memory map (location 00000000H). RM: 1 — Remapped operation mode enabled. Internal RAM is located at the bottom of the memory map (location 00000000H).

[MEMO]

Chapter 13 Write Protect Controller Function

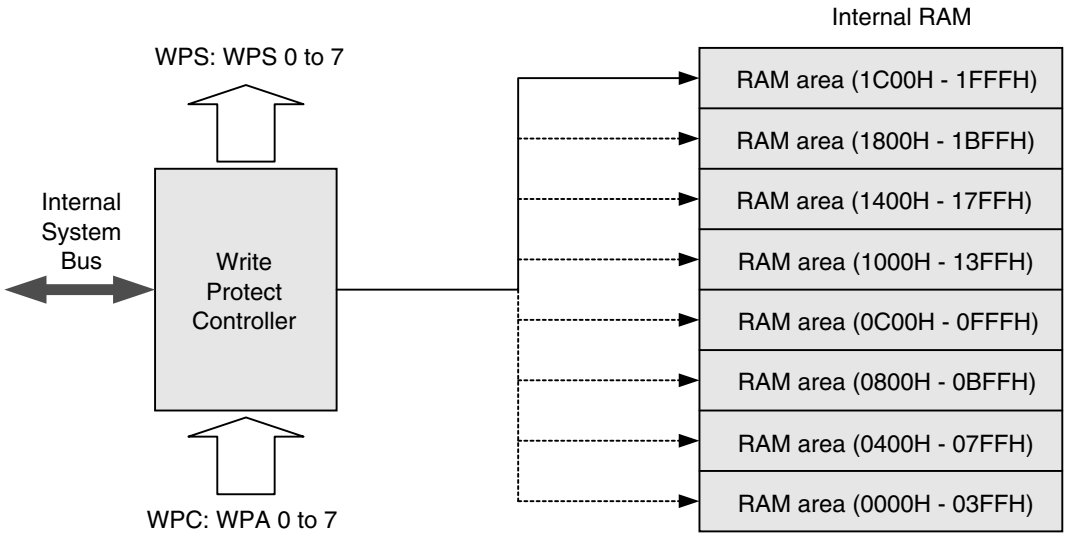
The SoCLite utilizes a write protect controller to allow write memory protection areas inside the internal RAM. It is limited for the internal RAM only.

13.1 Write Protect Controller Operation

Write protection of an internal RAM location is achieved by setting the corresponding WPA bit inside the write protection control register (WPC).
The internal RAM is subdivided into eight areas of the same size (1KByte), which can be independently write protected. If a write access to a write protected area of the internal RAM is performed, an error response is indicated causing an abort exception inside the SoCLite CPU.
At the same time the corresponding WPS status flag inside the write protection status register (WPS) is set to allow the abort exception handler routine to identify the cause for the exception. The corresponding WPS status flag is cleared by writing a “1” to the write protection status register.

Figure 13-1 shows the operation of the write protect controller.

Figure 13-1: Write Protection



The definition of the internal RAM areas is related to the lower address bits (address bits 0 to 15). The absolute 32 bit address is defined by the memory map.

13.2 Write Protect Controller Function Registers

13.2.1 Write Protection Control Register (WPC)

The write protection control register defines the write protected internal RAM areas. At reset all internal RAM areas are not write protected.
This register can be read/written.

Figure 13-2: Write Protection Control Register (WPC)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	Address	Initial value
WPC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3FFFFFD8H	00000000H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	WPA 7 - 0							

Bit Position	Bit Name	Function
7	WPA7	Write Protection Assignment 7 This bit enables write protection for internal RAM area 1C00H to 1FFFH. WPA7 / Operation mode: 0 – Write protection disabled; 1 – Write protection enabled
6	WPA6	Write Protection Assignment 6 This bit enables write protection for internal RAM area 1800H to 1BFFH. WPA6 / Operation mode: 0 – Write protection disabled; 1 – Write protection enabled
5	WPA5	Write Protection Assignment 5 This bit enables write protection for internal RAM area 1400H to 17FFH. WPA5 / Operation mode: 0 – Write protection disabled; 1 – Write protection enabled
4	WPA4	Write Protection Assignment 4 This bit enables write protection for internal RAM area 1000H to 13FFH. WPA4 / Operation mode: 0 – Write protection disabled; 1 – Write protection enabled

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>3</td><td>WPA3</td><td>Write Protection Assignment 3
This bit enables write protection for internal RAM area 0C00H to 0FFFH.
<table><tr><th>WPA3</th><th>Operation mode</th></tr><tr><td>0</td><td>Write protection disabled</td></tr><tr><td>1</td><td>Write protection enabled</td></tr></table></td></tr>
<tr><td>2</td><td>WPA2</td><td>Write Protection Assignment 2
This bit enables write protection for internal RAM area 0800H to 0BFFH.
<table><tr><th>WPA2</th><th>Operation mode</th></tr><tr><td>0</td><td>Write protection disabled</td></tr><tr><td>1</td><td>Write protection enabled</td></tr></table></td></tr>
<tr><td>1</td><td>WPA1</td><td>Write Protection Assignment 1
This bit enables write protection for internal RAM area 0400H to 07FFH.
<table><tr><th>WPA1</th><th>Operation mode</th></tr><tr><td>0</td><td>Write protection disabled</td></tr><tr><td>1</td><td>Write protection enabled</td></tr></table></td></tr>
<tr><td>0</td><td>WPA0</td><td>Write Protection Assignment 0
This bit enables write protection for internal RAM area 0000H to 03FFH.
<table><tr><th>WPA0</th><th>Operation mode</th></tr><tr><td>0</td><td>Write protection disabled</td></tr><tr><td>1</td><td>Write protection enabled</td></tr></table></td></tr>
</table>

13.2.2 Write Protection Status Register (WPS)

The write protection status register reflects the status whether a write access to a corresponding write protected internal RAM area was performed or not. A write access to the corresponding WPS flag in the write protection status register clears the status flag.
This register can be read/written.

Figure 13-3: Write Protection Status Register (WPS)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	Address	Initial value
WPS	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3FFFFFDCH	00000000H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	WPS 7 - 0							

Bit Position	Bit Name	Function
7	WPS7	Write Protection Status 7 This bit reflects the write protection status for internal RAM area 1C00H to 1FFFH.

WPA7	Read operation	Write operation
0	No write to protected area has been occurred	No effect
1	Write to protected area has been occurred	Clear status bit

Bit Position	Bit Name	Function
6	WPS6	Write Protection Assignment 6 This bit reflects the write protection status for internal RAM area 1800H to 1BFFH.

WPA6	Read operation	Write operation
0	No write to protected area has been occurred	No effect
1	Write to protected area has been occurred	Clear status bit

Bit Position	Bit Name	Function
5	WPS5	Write Protection Assignment 5 This bit reflects the write protection status for internal RAM area 1400H to 17FFH.

WPA5	Read operation	Write operation
0	No write to protected area has been occurred	No effect
1	Write to protected area has been occurred	Clear status bit

Bit Position	Bit Name	Function
4	WPS4	Write Protection Assignment 4 This bit reflects the write protection status for internal RAM area 1000H to 13FFH.

WPA4	Read operation	Write operation
0	No write to protected area has been occurred	No effect
1	Write to protected area has been occurred	Clear status bit

Bit Position	Bit Name	Function
3	WPS3	Write Protection Assignment 3 This bit reflects the write protection status for internal RAM area 0C00H to 0FFFH.

WPA3	Read operation	Write operation
0	No write to protected area has been occurred	No effect
1	Write to protected area has been occurred	Clear status bit

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th></tr>
<tr><td>2</td><td>WPS2</td><td>Write Protection Assignment 2
This bit reflects the write protection status for internal RAM area 0800H to 0BFFH.
<table>
<tr><th>WPA2</th><th>Read operation</th><th>Write operation</th></tr>
<tr><td>0</td><td>No write to protected area has been occurred</td><td>No effect</td></tr>
<tr><td>1</td><td>Write to protected area has been occurred</td><td>Clear status bit</td></tr>
</table></td></tr>
<tr><td>1</td><td>WPS1</td><td>Write Protection Assignment 1
This bit reflects the write protection status for internal RAM area 0400H to 07FFH.
<table>
<tr><th>WPA1</th><th>Read operation</th><th>Write operation</th></tr>
<tr><td>0</td><td>No write to protected area has been occurred</td><td>No effect</td></tr>
<tr><td>1</td><td>Write to protected area has been occurred</td><td>Clear status bit</td></tr>
</table></td></tr>
<tr><td>0</td><td>WPS0</td><td>Write Protection Assignment 0
This bit reflects the write protection status for internal RAM area 0000H to 03FFH.
<table>
<tr><th>WPA0</th><th>Read operation</th><th>Write operation</th></tr>
<tr><td>0</td><td>No write to protected area has been occurred</td><td>No effect</td></tr>
<tr><td>1</td><td>Write to protected area has been occurred</td><td>Clear status bit</td></tr>
</table></td></tr>
</table>

[MEMO]

Chapter 14 APB Bridge Control Function

The SoCLite comprises an APB bridge to open the peripheral APB bus to the ARM subsystem and the UDL area. The peripheral APB bus is a simple, not pipelined bus with control signals (select, write and strobe). The APB bridge inserts wait states on the ARM subsystem system bus (AHB) when accesses to the APB bus are performed.

14.1 APB Bridge Operation

The APB bridge in the SoCLite supports the insertion of wait states on the APB bus. The APB bus wait state control register (APBW) determines the number of wait cycles to be inserted on read and write transfers on the APB bus, when accessing the UDL area.
Wait states are only inserted on the APB bus, when accessing the on-chip UDL memory area.
All internal peripherals of the SoCLite, which are connected to the APB bus, are working with no wait state.

The APB bridge provides the following signals to the UDL area, which are shown in Table 14-1.

Table 14-1: APB Bus to UDL Area Signals

Pin Name	Direction	Function
PRESETn	to UDL	APB bus reset signal
PSELUDL	to UDL	APB bus slave select signal
PENABLE	to UDL	APB bus strobe signal
PWRITE	to UDL	APB bus write signal
PADDR0 to PADDR28	to UDL	APB address bus
PRDATA0 to PRDATA31	from UDL	APB read data bus
PWDATA0 to PWDATA31	to UDL	APB write data bus

The clock for the UDL area (PCLK) must be generated from the system clock SCLK by a delay unit.

Figure 14-1: PCLK Generation

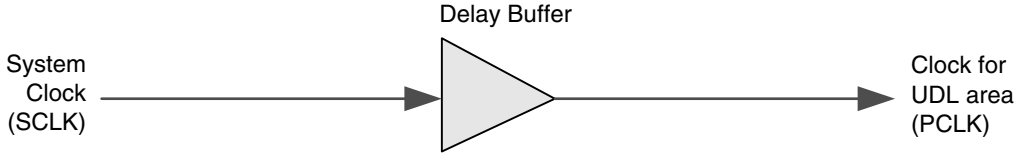
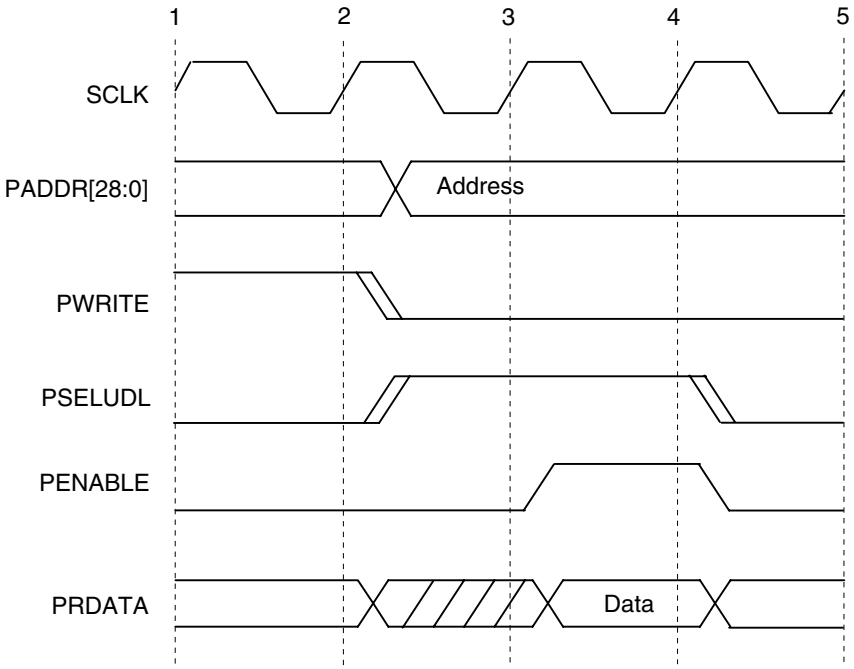


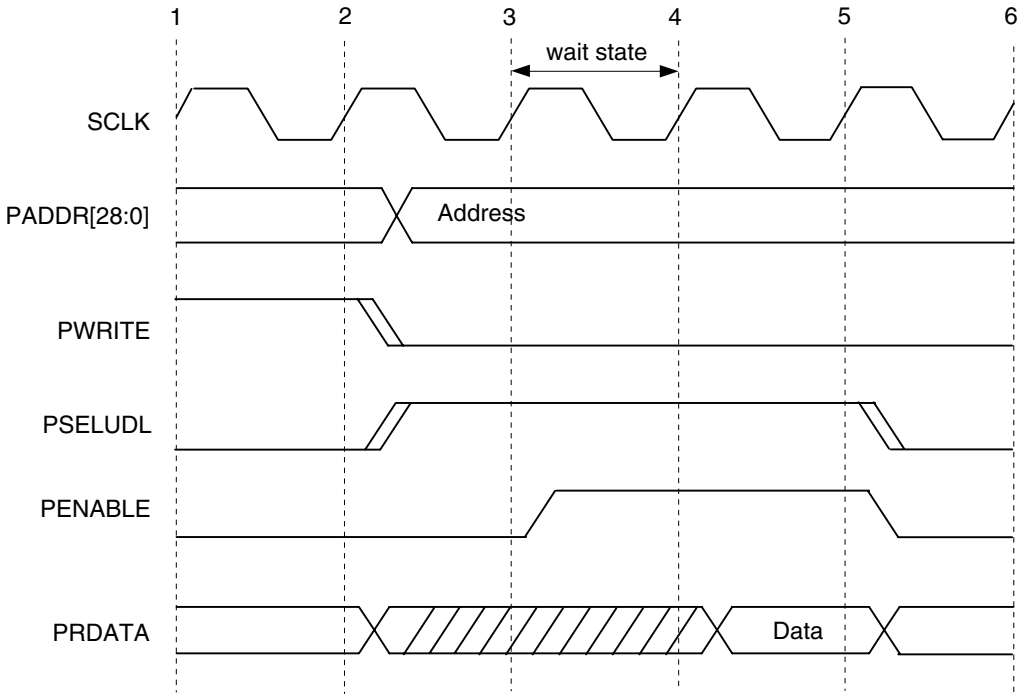
Figure 14-2 and Figure 14-3 show a read operation on the peripheral APB bus without and with one wait state.

Figure 14-2: Read Access on APB bus (no wait state)



No wait state is inserted, the WSTP bits in the APBW register are both cleared.
Two APB clock cycles are elapsed for a read access on the peripheral APB bus.

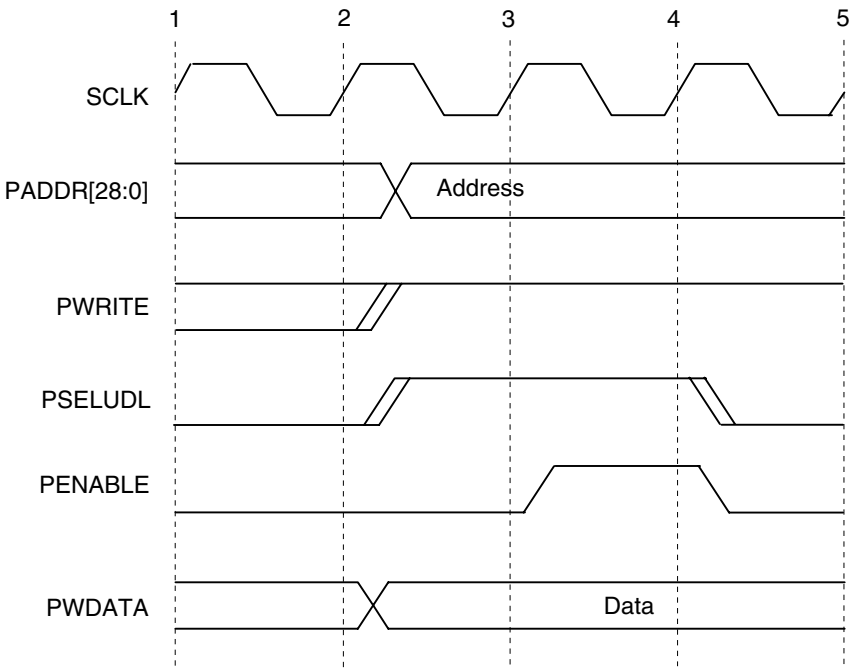
Figure 14-3: Read Access on APB bus (one wait state)



One wait state is inserted, the WSTP bits in the APBW register are set to WSTP[1:0] = 01_b.
Three APB clock cycles are elapsed for a read access on the peripheral APB bus, including one wait state.

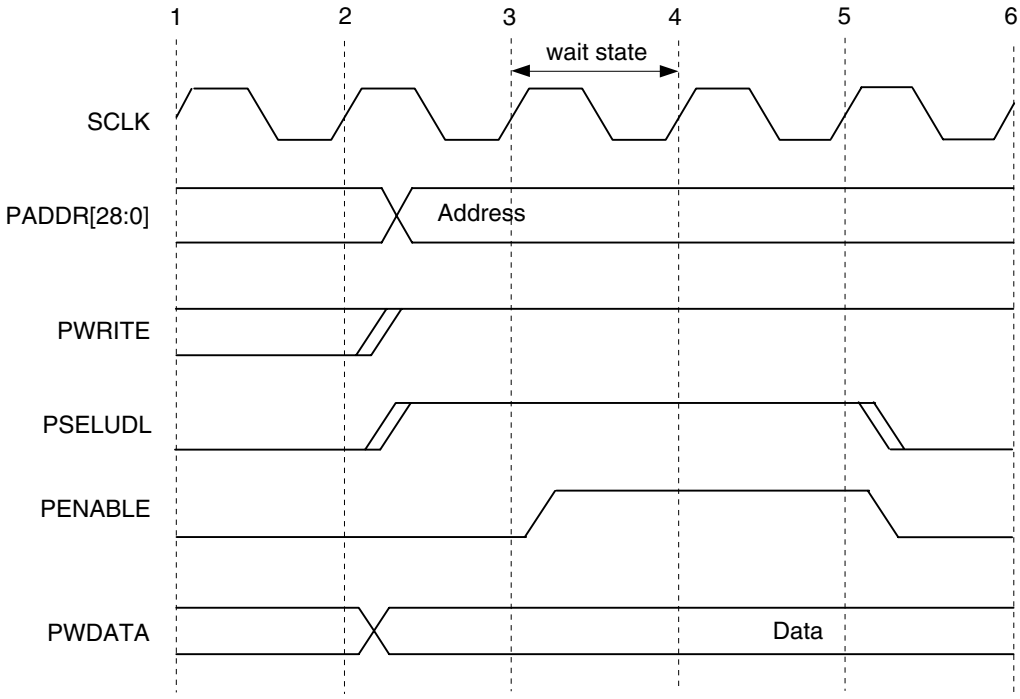
Figure 14-4 and Figure 14-5 show a write operation on the peripheral APB bus without and with one wait state.

Figure 14-4: Write Access on APB bus (no wait state)



No wait state is inserted, the WSTP bits in the APBW register are both cleared.
Two APB clock cycles are elapsed for a write access on the peripheral APB bus.

Figure 14-5: Write Access on APB bus (one wait state)



One wait state is inserted, the WSTP bits in the APBW register are set to WSTP[1:0] = 01_b. Three APB clock cycles are elapsed for a write access on the peripheral APB bus, including one wait state.

14.2 APB Bridge Function Register

14.2.1 APB Bus Wait State Control Register (APBW)

The APB bus wait state control register determines the number of wait states to be inserted during an APB transfer to the UDL area. At reset no wait states are inserted.
This register can be read/written.

Figure 14-6: APB Bus Wait State Control Register (APBW)

	7	6	5	4	3	2	1	0	Address	Initial value
APBW	0	0	0	0	0	0	WSTP1-0		40000078H	00H

Bit Position	Bit Name	Function
1 - 0	WSTP1 to WSTP0	Wait States These bits determine the number of inserted wait states.

WSTP1	WSTP0	Operation mode
0	0	No wait inserted (default)
0	1	1 wait state inserted
1	0	3 wait state inserted
1	1	7 wait state inserted

Note: Insertion of wait states on the APB bus to the UDL area causes additional wait states on the internal system bus (AHB) to the CPU of the ARM subsystem.

Chapter 15 RESET Function

A system reset to initialize the SoCLite is generated by:

- a low level signal is input on external nRST pin
- an internal watchdog overflow
- an UDL reset

When the external nRST pin changes from low level to high level or the watchdog reset respectively the UDL reset is released, reset status is released and the CPU starts program execution. The user has to set the contents of various registers as needed within his user program.

15.1 Feature

- Noise elimination of external reset pin (nRST) using a delay.

15.2 Pin Functions

During a system reset period, all output pins have a defined signal level except the JTAG_TDO (which is high impedance).

Table 15-1 shows the operation status of each pin during a reset period.

Table 15-1: Operation Status of Each Pin During Reset Period

Group	Pin Name	Pin Status
External Memory Interface	nXOE, nXWEN, nXBLS[3:0], nXCS[7:0]	High level
	XADDR[25:0]	Low level
	XDATA[31:0]	Input
On-Chip Peripherals	TXD	High level
	RXD	Input
Misc. Signals	X2	Toggle
	nRST, EA[1:0], DBGEN, PLLOD[1:0], TCLK, X1	Input
JTAG Interface	JTAG_TDI, JTAG_TCK, JTAG_TMS, JTAG_TRST	Input
	JTAG_TDO	High Impedance

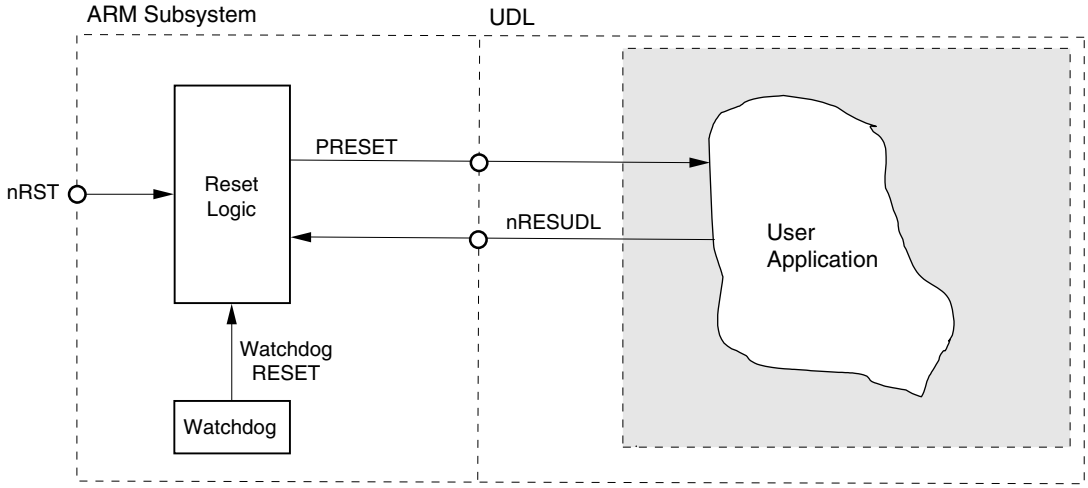
15.3 Reset Operation

The SoCLite provides three opportunities to perform a system reset.

- Reset by external reset pin nRST.
- Reset by watchdog timer.
- Reset by UDL area reset signal nRESUDL.

Figure 15-1 shows the reset structure of the SoCLite.

Figure 15-1: Reset Structure



15.3.1 Reset by External Reset Pin

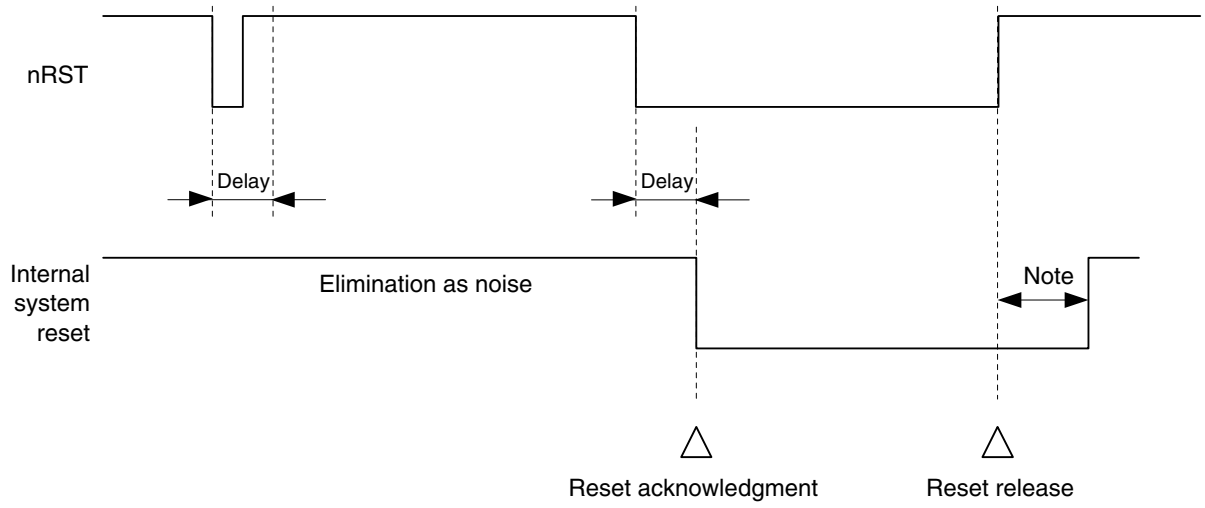
If a low level signal is input to the external pin nRST, a system reset is performed and the system is initialized. When the level changes from low to high at the nRST pin, the internal reset state is not released immediately. Due to the power up requirements of the PLL the internal reset state is released after the PLL lock time.
This ensures a stable output clock (FCLK) from the PLL (see Figure 8-4 in Chapter 8).

The reset pin incorporates a noise filter and a synchronization stage to prevent malfunction due to noise. The following conditions must be kept to ensure a stable operation at power up.

- The oscillator must run before a reset at the external reset pin nRST is deasserted to ensure the setup time for the synchronization stage.
- Stabilization of the oscillator must be ensured by an appropriate duration of the low level phase of the external reset signal.

(1) Reset Signal Acknowledgment

Figure 15-2: Reset Signal Acknowledgment

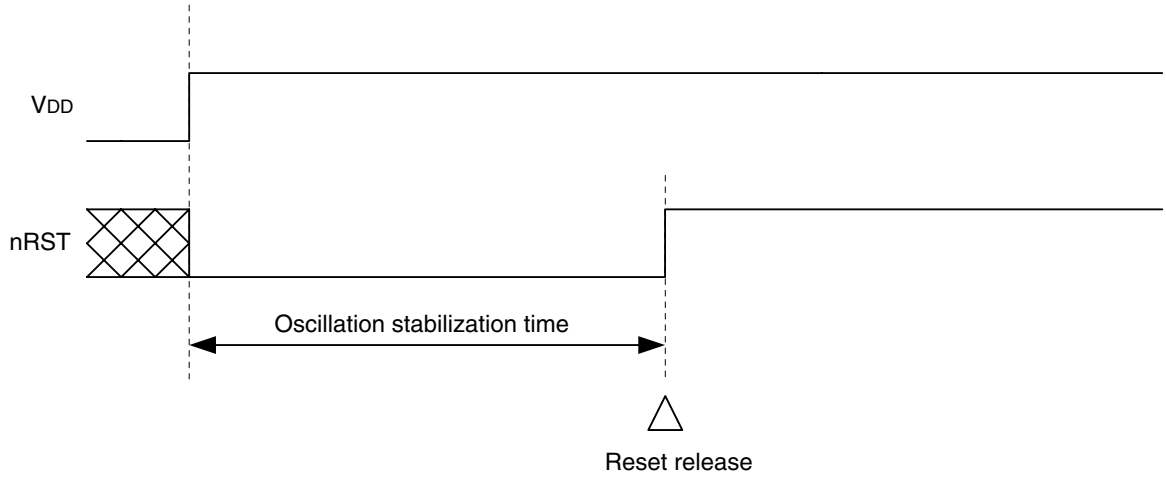


Note: The internal system reset signal continues in active status for a period of 2^{14} oscillator clocks after releasing the nRST signal.

(2) Power-on Reset

A power-on reset must guarantee oscillation stabilization time from power-on until reset acknowledgment due to the low level width of the nRST signal.

Figure 15-3: Reset at Power-On



15.3.2 Reset by Watchdog Timer

If the watchdog timer inside the SoCLite was started and has not been retriggered in time, a watchdog reset is generated.
The oscillation stabilization time must not be considered for this reset.

15.3.3 Reset by UDL Area

The logic inside the UDL area is given the opportunity to cause a reset. The reset signal nRESUDL from the UDL area can be asynchronous and is synchronized inside the ARM subsystem.
The nRESUDL signal must be held at low level for a minimum of two system clock cycles (SCLK) to initiate a reset. After the nRESUDL reset signal level changes from low to high, the system reset state is released seven system clock cycles (SCLK) later. Therefore the system clock (SCLK) must be active to allow synchronization of the UDL reset signal.
The oscillation stabilization time must not be considered for this reset.

15.4 Registers at Reset

The contents of each register has to be initialized as needed within the user program.
Table 15-2 shows the initial values of the on-chip peripherals and internal RAM after reset.

Table 15-2: Initial Values of On-Chip Peripherals and Internal RAM After Reset

On-Chip Hardware	Register Name	Initial Value after Reset
Internal RAM		Undefined
Write protect controller function	Write protection control register (WPC)	00000000H
	Write protection status register (WPS)	00000000H
External bus interface	SMC bank0 configuration register (SMCBCR0)	2000FBEFH
	SMC bank1 configuration register (SMCBCR1)	2000FBEFH
	SMC bank2 configuration register (SMCBCR2)	1000FBEFH
	SMC bank3 configuration register (SMCBCR3)	0000FBEFH
	SMC bank4 configuration register (SMCBCR4)	2000FBEFH
	SMC bank5 configuration register (SMCBCR5)	2000FBEFH
	SMC bank6 configuration register (SMCBCR6)	1000FBEFH
	SMC bank7 configuration register (SMCBCR7)	0000FBEFH
Interrupt controller function	Interrupt control0 register (INTC0)	07070707H
	Interrupt control1 register (INTC1)	07070707H
	Interrupt control2 register (INTC2)	07070707H
	Interrupt control3 register (INTC3)	07070707H
	Interrupt control4 register (INTC4)	07070707H
	Interrupt control5 register (INTC5)	07070707H
	Interrupt control6 register (INTC6)	07070707H
	Interrupt control7 register (INTC7)	07070707H
	Interrupt status register (INTSTA)	00000000H
	Interrupt source register (INTSRC)	00H
	Interrupt source register for emulation (INTSRCEMU)	00H
	Interrupt status register for emulation (INTISSEMU)	00000000H
Serial interface function	Serial reception data buffer register (SRXB)	00H
	Serial transmission data buffer register (STXB)	00H
	Serial status register (SSTA)	00H
	Serial control register (SCTR)	00000000H
	Serial reception data buffer register for emulation (SRXBEMU)	00H
Timer function	Timer value register (TMV)	FFFFFFFFH
	Timer load register (TML)	FFFFFFFFH
	Timer control register (TMC)	00H
Remap/Pause controller function	Remap/Pause control register (RMP)	00H
Watchdog timer function	Watchdog timer control register (WDCTRL)	1000H
APB bridge control function	APB bus wait state control register (APBW)	00H

Caution: In the table above, “Undefined” means either undefined at the time of a power-on reset or undefined due to data destruction when a system reset occurs.

[MEMO]

Chapter 16 TEST Function

The SoCLite can operate in several test modes. The test modes are entered with a static combination of input levels at specific pins of the SoCLite.

16.1 Features

- Test by test interface controller (TIC) for functional test.
- Functional test with program code via external memory.
- Boundary scan to test interconnections on the printed circuit board (PCB) level.
- Debug mode for ARM CPU debugging.

16.2 Functional Overview

The SoCLite can be switched into four test modes to perform testing of the device.
Table 16-1 shows the different test modes.

Table 16-1: Test Modes

Test Mode	Test Function
TESTFUNC	***Functional Test Mode*** The functionality of the SoCLite is tested.
TESTTIC	***TIC Test Mode*** Using the test interface controller to perform functional test of SoCLite.
TESTBSCAN	***Boundary SCAN*** Test of interconnections on PCB level is performed.
DEBUG	***Debug Mode*** Debugging of ARM CPU is performed.

The main signal to set the SoCLite into test mode is the TARMSS signal coming from the UDL area. The TARMSS signal must be set to high level to switch the SoCLite into test mode. Depending on the test sub-mode signals TSM[1:0] the required test mode is selected.
The TSM[1:0] signals are no dedicated pins but shared with the PLLOD[1:0] pins.

The UDL area signal TARMSS determines whether normal operation mode or test mode is selected. The test mode is entered as soon as the TARMSS signal is set to high level.
The dedicated sub-mode is selected according to the signal levels at the TMS[1:0] pins. The sub-modes are entered according to Table 16-3.

Table 16-2 shows the pin assignment, which is different between normal operation mode and test mode.

Table 16-2: Pin Assignment in Test Mode

Pin Assignment in Normal Operation Mode	Pin Assignment in Test Mode
PLLOD1	TSM1
PLLOD0	TSM0
nXBLS3	TESTREQB
nXBLS2	TESTREQA
nXBLS1	TESTACK
XDATA[31:0]	TICBUS[31:0]

Notes:
1. The TMS[1:0] signals are used for selection of test sub-mode. These signals are shared with PLLOD[1:0] pins and are only valid during test mode.
2. The signals TESTREQB, TESTREQA and TESTACK are only available during TESTTIC test mode and needed for the serial communication with the test interface controller (TIC). These signals are shared with nXBLS[3:1] pins.
3. The signals TICBUS[31:0] are only available during TESTTIC test mode and needed for the communication with the test interface controller (TIC). These signals are shared with XDATA[31:0] pins.

The test sub-mode settings are shown in Table 16-3.

Table 16-3: Test Sub-Mode Settings

Signal Name				Test Mode	Test Function
DBGEN	TARMSS	TSM1	TSM0		
1	0	X	X	DEBUG	Normal operation mode (Debug mode continuously enabled)
0	0	X	X	TESTBSCAN	Boundary scan test
X	1	0	0	TESTFUNC	Functional test mode
	1	1	0	TESTTIC	TIC test mode
all other settings				Not allowed	Settings prohibited

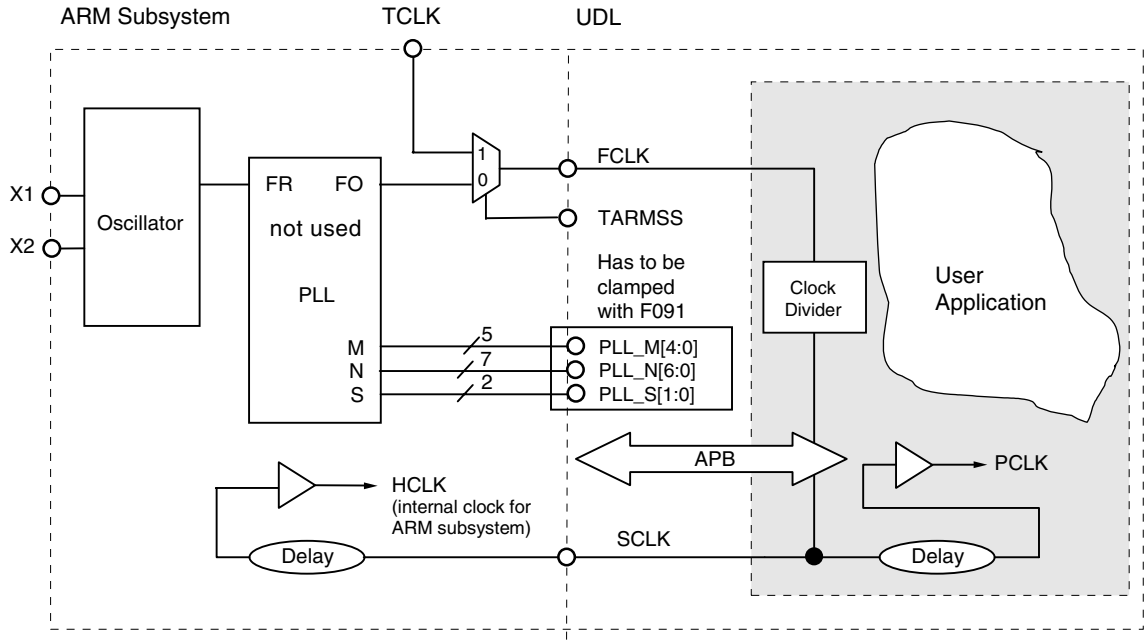
Note: The boundary scan test mode and debug mode are controlled by the DBGEN pin.

16.2.1 Clock Distribution in Test Mode

In the TESTFUNC and TESTTIC test modes, the output clock of the ARM subsystem (FCLK) is always clocked from the test clock input pin (TCLK). The PLL output clock is not used in these test modes, it is only used in normal operation mode including debug mode.

Figure 16-1 shows the clock distribution in the SoCLite during the above mentioned test modes.

Figure 16-1: Clock Distribution during Test Mode



The PLL is not used during test mode, therefore the PLL lock time must not be considered after a system reset. The internal reset delay time (t_{HRSPD}) is reduced to 2 system clock cycles (SCLK).

16.3 Test Mode Operation

The SoCLite provides four opportunities to perform a device test.

- Functional test mode
- TIC test mode
- Boundary scan test mode
- Debug mode

16.3.1 Functional Test Mode

In the functional test mode (TESTFUNC) the interconnection of all macros inside the SoCLite is checked by functional patterns written in program code. These patterns shall be as simple as possible, e.g. write and read to/from a register inside a macro to verify the connection to the internal system bus. The CPU instructions should be placed in external memory on the external memory bus for execution.

The functionality of all external pins and internal signals remain the same as in normal operation mode, except the two pins needed to control the test sub-mode TSM[1:0].

During this mode, the system is clocked from TCLK.

16.3.2 TIC Test Mode

The test interface controller (TIC) is a macro inside the ARM subsystem needed for test purposes (TESTTIC). As an internal system bus master the TIC is able to initiate bus cycles on the internal system bus. The internal system bus transfers are controlled by test vectors, which are fed into the TIC by an external 3 pin serial interface plus a 32-bit test bus for data.
Therefore the external pins nXBLS[3:1] and XDATA[31:0] are switched to TESTREQB, TESTREQA, TESTACK and TICBUS[31:0] in test mode.

If the test interface would be used then NEC should be contacted for assistance.

During this mode, the system is clocked from TCLK.

16.3.3 Boundary Scan Test Mode

The SoCLite supports boundary scan to give the possibility of testing interconnections on PCB level. The boundary scan implementation is compliant to the IEEE1149.1 specification.

The boundary scan test mode (TESTBSCAN) is enabled by the DBGEN pin. Table 16-4 shows the settings by the DBGEN pin.

Table 16-4: DBGEN pin Settings

DBGEN Pin	Operation Function
0	Boundary scan test mode
1	Debug mode

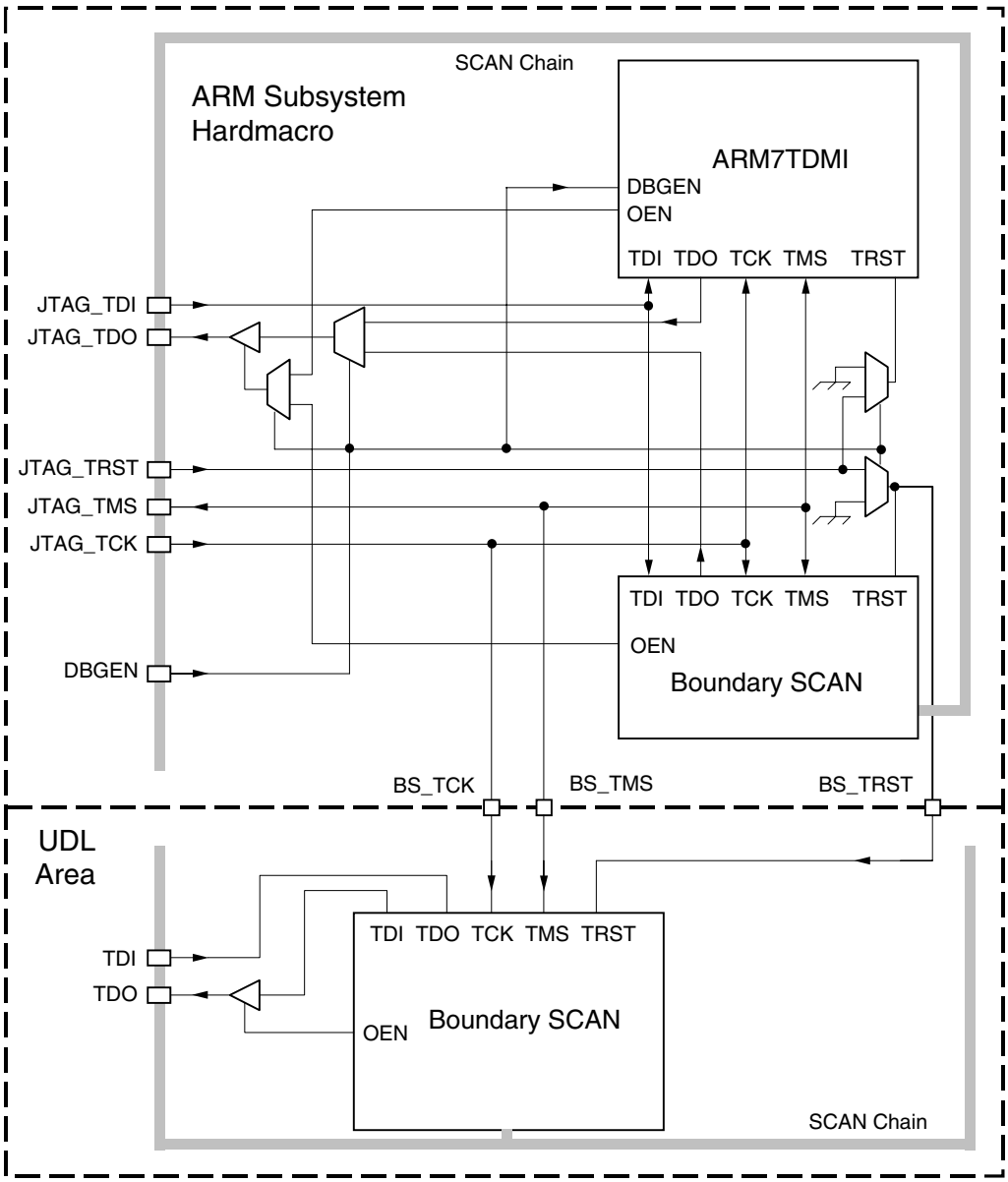
Because of the two parts in the SoCLite, the ARM subsystem and the UDL area, two boundary scan chains can be included in the SoCLite.

- One boundary scan chain is placed in the ARM subsystem. The boundary scan is inserted in the way that the ARM subsystem as hardmacro is seen as a complete device with independent boundary scan and JTAG interface. The boundary scan for the ARM subsystem hardmacro is fixed implemented. To save pins the JTAG interface for boundary scan is multiplexed with the ARM CPU JTAG interface, which is used for ARM CPU debugging. The DBGEN pin determines whether boundary scan or ARM CPU debugging is available through the JTAG interface.
- One boundary scan chain can be implemented in the UDL area for the customized user logic. The boundary scan implementation in the UDL area is also independent with its own test access port (TAP) controller and JTAG signals. To save pins on the UDL area side, three signals of the ARM subsystem JTAG interface are made available to the UDL area. These signals are BS_TCK, BS_TMS and BS_TRST. The TDI and TDO pins for boundary scan must be inserted separately.

During this mode, the system is clocked from FCLK.

Figure 16-2 shows the connections of the JTAG interface including ARM CPU debugging and boundary scan connections.

Figure 16-2: JTAG Interface



(1) JTAG Controller and Registers

The ARM subsystem contains the following registers and JTAG controller:

- Instruction register
- Boundary scan register
- Bypass register
- Test access port (TAP) controller

Table 16-5 lists the registers accessible through the JTAG interface. It depends on the DBGEN pin whether the TAP controller of the ARM CPU for debugging or the TAP controller for boundary scan is controlled by the JTAG interface of the SoCLite.
For more information about the ARM CPU JTAG interface, please refer to the document “ARM7TDMI data sheet” (ARM DDI 0029E).

Table 16-5: JTAG accessible Registers

DBGEN Pin	Register Name	Width (Bits)
0	JTAG Instruction register	3
	JTAG Bypass register	1
	JTAG Boundary Scan register	101
1	Please refer to “ARM7TDMI Data Sheet”	

(2) JTAG Instruction Register

The JTAG instruction register includes three shift register organization cells. This register is used to select the test to be performed and the test data register to be accessed.
The JTAG instruction register can only be accessed when DBGEN is at low level.
Table 16-6 lists the test and data registers of the JTAG instruction register to be performed.

Table 16-6: JTAG Instruction Register

Binary Code	Instruction	Data Register	Function
000	EXTEST	Boundary Scan	SoCLite is placed into an external boundary scan test mode and selects the boundary scan register to be connected between JTAG_TDI and JTAG_TDO.
010	SAMPLE/ PRELOAD		Allows the SoCLite to remain in normal operation mode and selects the boundary scan register. During this instruction the boundary scan register can be accessed via a data scan operation to take a sample of the functional data. This instruction is also used to preload test data into the boundary scan register before loading an EXTEST instruction.
101	CLAMP	Bypass	Sets the external output pins to logic levels determined by the contents of the boundary scan register. Before loading this instruction, the contents of the boundary scan register can be preset with the SAMPLE/PRELOAD instruction without affecting the condition of the output pins.
110	HIGHZ		Sets all external output pins to a high impedance state.
111	BYPASS		Allows the SoCLite to remain in normal operation mode and selects the bypass register. The bypass instruction allows serial data to be transmitted through the SoCLite from JTAG_TDI to JTAG_TDO without affecting the operation of the SoCLite.
Others			Settings prohibited

16.3.4 Debug Mode

When the DBGEN pin is at high level, the SoCLite runs in normal operation mode including the debug mode. In debug mode the ARM CPU debug interface (JTAG interface) is available for debugging purposes of the SoCLite.
For more information about the debug interface, please refer to the document “ARM7TDMI data sheet” (ARM DDI 0029E).

During this mode, the system is clocked from FCLK.

Three emulation registers are provided to support debugging of the SoCLite as shown in Table 16-7. For the function of these registers, please refer to the corresponding chapters.

Table 16-7: Emulation Registers

Register Name	Emulation Register	Refer to the following chapter
INTSRCEMU	Interrupt source register for emulation	Chapter 7
INTISSEMU	Interrupt status register for emulation	
SRXBEMU	Serial reception data buffer register for emulation	Chapter 11

[MEMO]

Appendix A Index

NEC

Facsimile Message

From:

Name

Company

Tel. FAX

Address

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

Thank you for your kind support.

North America
NEC Electronics Inc.
Corporate Communications Dept.
Fax: 1-800-729-9288
1-408-588-6130

Europe
NEC Electronics (Europe) GmbH
Technical Documentation Dept.
Fax: +49-211-6503-274

South America
NEC do Brasil S.A.
Fax: +55-11-6465-6829

Hong Kong, Philippines, Oceania
NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

Korea
NEC Electronics Hong Kong Ltd.
Seoul Branch
Fax: 02-528-4411

Taiwan
NEC Electronics Taiwan Ltd.
Fax: 02-2719-5951

Asian Nations except Philippines
NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Japan
NEC Semiconductor Technical Hotline
Fax: 044-548-7900

I would like to report the following error/make the following suggestion:

Document title: _____

Document number: _____ Page number: _____

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	☐	☐	☐	☐
Technical Accuracy	☐	☐	☐	☐
Organization	☐	☐	☐	☐