

# ARM PrimeCell™

Smart Card Interface (PL131)

## **Technical Reference Manual**

**ARM**

# ARM PrimeCell™ Smart Card Interface (PL131)

## Technical Reference Manual

Copyright © 2001 ARM Limited. All rights reserved.

### Release information

#### Change history

Description	Issue	Change
November 2001	A	First release.

### Proprietary notice

Words and logos marked with ® or ™ are registered trademarks or trademarks owned by ARM Limited, except as otherwise stated below in this proprietary notice. Other brands and names mentioned herein may be the trademarks of their respective owners.

All other products or services mentioned herein may be trademarks of their respective owners.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by ARM Limited in good faith. However, all warranties implied or expressed, including but not limited to implied warranties or merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. ARM Limited shall not be liable for any loss or damage arising from the use of any information in this document, or any error or omission in such information, or any incorrect use of the product.

### Document confidentiality status

This document is Open Access. This document has no restriction on distribution.

### Product status

The information in this document is Final (information on a developed product).

### Web address

<http://www.arm.com>

# Contents

## ARM PrimeCell Smart Card Interface (PL131)

### Technical Reference Manual

	<b>Preface</b>	
	About this document .....	vi
	Further reading.....	ix
	Feedback .....	x
<b>Chapter 1</b>	<b>Introduction</b>	
	1.1 About the ARM PrimeCell Smart Card Interface (PL131) .....	1-2
<b>Chapter 2</b>	<b>Functional Overview</b>	
	2.1 ARM PrimeCell Smart Card Interface (PL131) overview .....	2-2
	2.2 PrimeCell SCI functional description.....	2-3
	2.3 PrimeCell SCI operation.....	2-8
	2.4 PrimeCell SCI DMA interface.....	2-25
	2.5 SCI clock stop mode .....	2-28
	2.6 PrimeCell SCI clock and data driver configurations .....	2-29
<b>Chapter 3</b>	<b>Programmer's Model</b>	
	3.1 About the programmer's model.....	3-2
	3.2 Summary of PrimeCell SCI registers .....	3-3
	3.3 Register descriptions .....	3-7
	3.4 Interrupts .....	3-40

<b>Chapter 4</b>	<b>Programmer's Model for Test</b>	
4.1	PrimeCell SCI test harness overview .....	4-2
4.2	Scan testing.....	4-3
4.3	Test registers.....	4-4
4.4	Integration testing of block inputs .....	4-10
4.5	Integration testing of block outputs.....	4-12
4.6	Integration test summary .....	4-17
<b>Appendix A</b>	<b>ARM PrimeCell Smart Card Interface (PL131) Signal Descriptions</b>	
A.1	AMBA APB signals.....	A-2
A.2	On-chip signals.....	A-3
A.3	Signals to pads .....	A-5

# Preface

This preface introduces the ARM PrimeCell Smart Card Interface (PL131) and its reference documentation. It contains the following sections:

- *About this document* on page vi
- *Further reading* on page ix
- *Feedback* on page x.

## About this document

This document is the technical reference manual for the ARM PrimeCell *Smart Card Interface* (SCI).

## Intended audience

This document has been written for *System-on-Chip* (SoC) designers and system architects, and provides a description of components within the SCI architecture.

## Using this manual

This document is organized into the following chapters:

### **Chapter 1**    *Introduction*

Read this chapter for an introduction on the ARM PrimeCell SCI and its features.

### **Chapter 2**    *Functional Overview*

Read this chapter for a description of the block diagram and functionality of the PrimeCell SCI.

### **Chapter 3**    *Programmer's Model*

Read this chapter for a description of the PrimeCell SCI registers and programming details.

### **Chapter 4**    *Programmer's Model for Test*

Read this chapter for an description of the logic in the PrimeCell SCI for integration testing.

### **Appendix A**    *ARM PrimeCell SCI (PL131) Signal Descriptions*

Read this chapter for a description of the AMBA APB signals, on-chip signals and signals to pads.

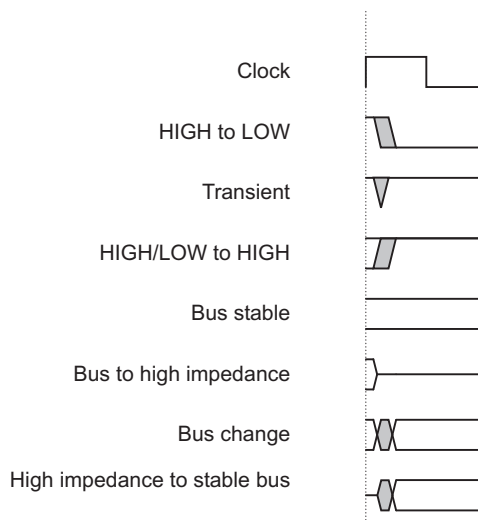
## Typographical conventions

The following typographical conventions are used in this document:

<b>bold</b>	Highlights ARM processor signal names, and interface elements such as menu names. Also used for terms in descriptive lists, where appropriate.
<i>italic</i>	Highlights special terminology, cross-references, and citations.
<code>monospace</code>	Denotes text that can be entered at the keyboard, such as commands, file names and program names, and source code.
<u>monospace</u>	Denotes a permitted abbreviation for a command or option. The underlined text can be entered instead of the full command or option name.
<code>monospace italic</code>	Denotes arguments to commands or functions where the argument is to be replaced by a specific value.
<code>monospace bold</code>	Denotes language keywords when used outside example code.

## Timing diagram conventions

This manual contains one or more timing diagrams. The following key explains the components used in these diagrams. Any variations are clearly labeled when they occur. Therefore, no additional meaning should be attached unless specifically stated.



### Key to timing diagram conventions

Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.



## Further reading

This section lists publications by ARM Limited, and by third parties.

ARM periodically provides updates and corrections to its documentation. See <http://www.arm.com> for current errata sheets and addenda.

See also the ARM Frequently Asked Questions list at:  
<http://www.arm.com/DevSupp/Sales+Support/faq.html>

## ARM publications

This document contains information that is specific to the SCI. Refer to the following documents for other relevant information:

- *AMBA Specification (Rev 2.0)* (ARM IHI 0011)
- *ARM PrimeCell Smart Card (PL131) Integration Manual* (PL131 INTM 0000)
- *ARM PrimeCell Smart Card (PL131) Design Manual* (PL131 DDES 0000).

## Other publications

This section lists relevant documents published by third parties:

- *EMV '96 Integrated Circuit Card, Specifications for Payment Systems* (Version 3.1.1 May 31, 1998) hereby referred to as *EMV Standard*.  
Part I: Electromechanical characteristics, logical interface and transmission protocols.  
Published by Europay International S.A., Mastercard International Inc, and Visa International Association.
- *ISO/IEC 7816-3: Information technology - Identification cards - Integrated circuit(s) card(s) with contacts (Second Edition 1997-12-15)*  
Part 3: Electronic signals and transmission protocols.
- *ISO/IEC 7816-10: Identification cards - Integrated circuit(s) card(s) with contacts (First Edition 1999-11-01)*  
Part 10: Electronic signals and answer to reset for synchronous cards.

## Feedback

ARM Limited welcomes feedback both on the SCI, and on the documentation.

### Feedback on the SCI

If you have any comments or suggestions about this product, contact your supplier giving:

- the product name
- a concise explanation of your comments.

### Feedback on this document

If you have any comments on about this document, send an email to [errata@arm.com](mailto:errata@arm.com) giving:

- the document title
- the document number
- the page number(s) to which your comments refer
- a concise explanation of your comments.

General suggestions for additions and improvements are also welcome.

# Chapter 1

## Introduction

This chapter introduces the ARM PrimeCell Smart Card Interface (PL131). It contains the following sections:

- *About the ARM PrimeCell Smart Card Interface (PL131)* on page 1-2.

## 1.1 About the ARM PrimeCell Smart Card Interface (PL131)

The PrimeCell *Smart Card Interface* (SCI) is an *Advanced Microcontroller Bus Architecture* (AMBA) compliant *System-on-a-Chip* (SoC) peripheral that is developed, tested and licensed by ARM. Refer to Figure 1-1 on page 1-3 for a block diagram of the SCI connections.

The PrimeCell SCI is an AMBA slave module that connects to the AMBA *Advanced Peripheral Bus* (APB), and interfaces to an external Smart Card reader. The SCI can autonomously control data transfer to and from the smart card. Transmit and receive data FIFOs are provided to reduce the required interaction between the CPU core or an AMBA *Advanced High-performance Bus* (AHB) master and the peripheral.

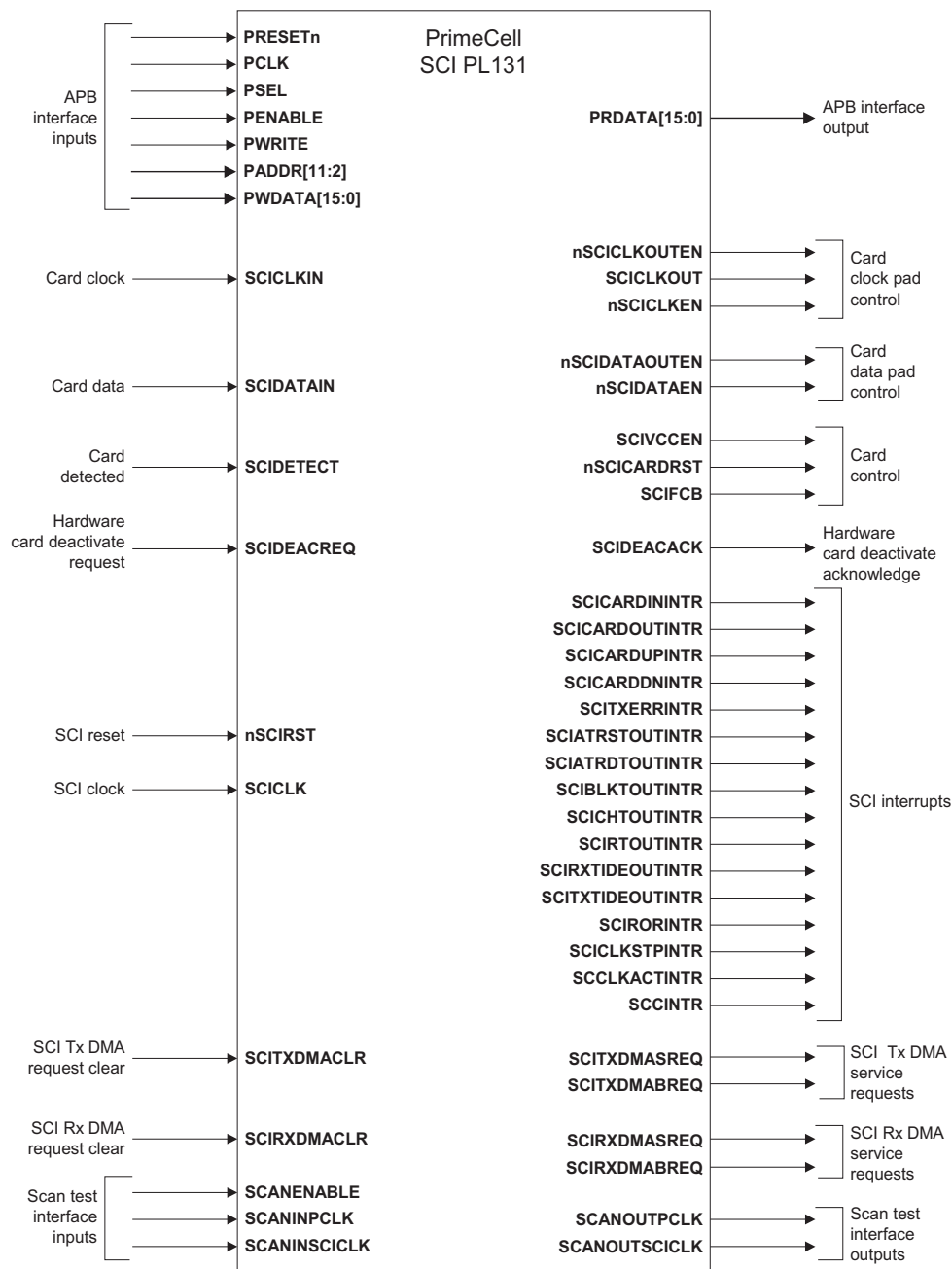


Figure 1-1 PrimeCell SCI connections

The features of the PrimeCell SCI are covered under the following headings:

- *Features of the PrimeCell SCI* on page 1-4
- *Programmable parameters* on page 1-5.

### 1.1.1 Features of the PrimeCell SCI

The following features are provided by the PrimeCell SCI:

- Compliance to the *AMBA Specification (Rev 2.0)* onwards for easy integration into *System-on-a-Chip* (SoC) implementation.
- Supports asynchronous T0 and T1 transmission protocols.
- Supports clock rate conversion factor  $F = 372$  or  $512$ , with bit rate adjustment factors  $D = 1, 2, 4, 8,$  and  $16$  supported.
- Eight character deep buffered TX and RX paths.
- Direct interrupts for TX and RX FIFO level monitoring.
- Independent masking of all interrupts.
- Support for *Direct Memory Access* (DMA).
- Interrupt status register.
- Hardware initiated card deactivation sequence on detection of card removal.
- Software initiated card deactivation sequence on transaction complete.
- Limited support for synchronous smart cards through registered input/output.
- Identification registers that uniquely identify the PrimeCell SCI. These can be used by an operating system to automatically configure itself.

### 1.1.2 Programmable parameters

The following key parameters are programmable:

- Smart Card clock frequency
- communication baud rate
- protocol convention
- card activation time
- card deactivation time
- check for maximum time for first character of *Answer To Reset* (ATR) reception
- check for maximum duration of ATR character stream
- check for maximum time for receipt of first character of data stream
- check for maximum time allowed between characters
- character guard time
- block guard time
- transmit character retry
- receive character retry
- transmit FIFO tide level
- receive FIFO tide level
- clock stop time
- clock start time
- clock inactive level.

Additional test registers and modes are implemented to provide efficient testing.





# Chapter 2

## Functional Overview

This chapter describes the major functional blocks of the ARM PrimeCell Smart Card Interface (PL131). It contains the following sections:

- *ARM PrimeCell Smart Card Interface (PL131) overview* on page 2-2
- *PrimeCell SCI functional description* on page 2-3
- *PrimeCell SCI operation* on page 2-8
- *PrimeCell SCI DMA interface* on page 2-25
- *SCI clock stop mode* on page 2-28
- *PrimeCell SCI clock and data driver configurations* on page 2-29.

## 2.1 ARM PrimeCell Smart Card Interface (PL131) overview

The PrimeCell *Smart Card Interface* (SCI) conforms to Part 1 of the *Integrated Circuit Specification for Payment Systems Electromechanical Characteristics, Logical Interface, and Transmission Protocols* (Version 3.1.1 May 31, 1998). This standard is published jointly by Europay International S.A., Mastercard International Incorporated, and Visa International Service Association and is subsequently referred to as the *EMV Standard*. This standard refers to, and is based on, the *ISO 7816* series of standards. The user is expected to be familiar with both the *EMV Standard* and *ISO 7816-3*.

The PrimeCell SCI performs:

- serial to parallel conversion on data received
- parallel to serial conversion on data transmitted to an external smart card.

The host CPU reads and writes data and control information through the AMBA APB interface. The transmit and receive paths are buffered with internal FIFO memories allowing up to 8 bytes to be stored independently in both transmit and receive modes. Data can also be transferred through the DMA interface.

The PrimeCell SCI includes a programmable baud rate generator and, in conjunction with a secondary value counter, provides programmable *elementary time units* (etus).

The PrimeCell SCI has been designed to enable close monitoring of all stages of a typical card session using mask enabled interrupts. The interrupt architecture allows a choice of:

- a polled approach by examining the interrupt status register on assertion of a single common interrupt
- the interrupt sources can be fed directly to the interrupt controller for immediate identification.

The transmit and receive FIFO interrupts are asserted and deasserted automatically depending on their programmed trigger threshold levels.

Parity errors are automatically checked by hardware on received data.

Interpretation of the received data stream is always performed by the user's application software.

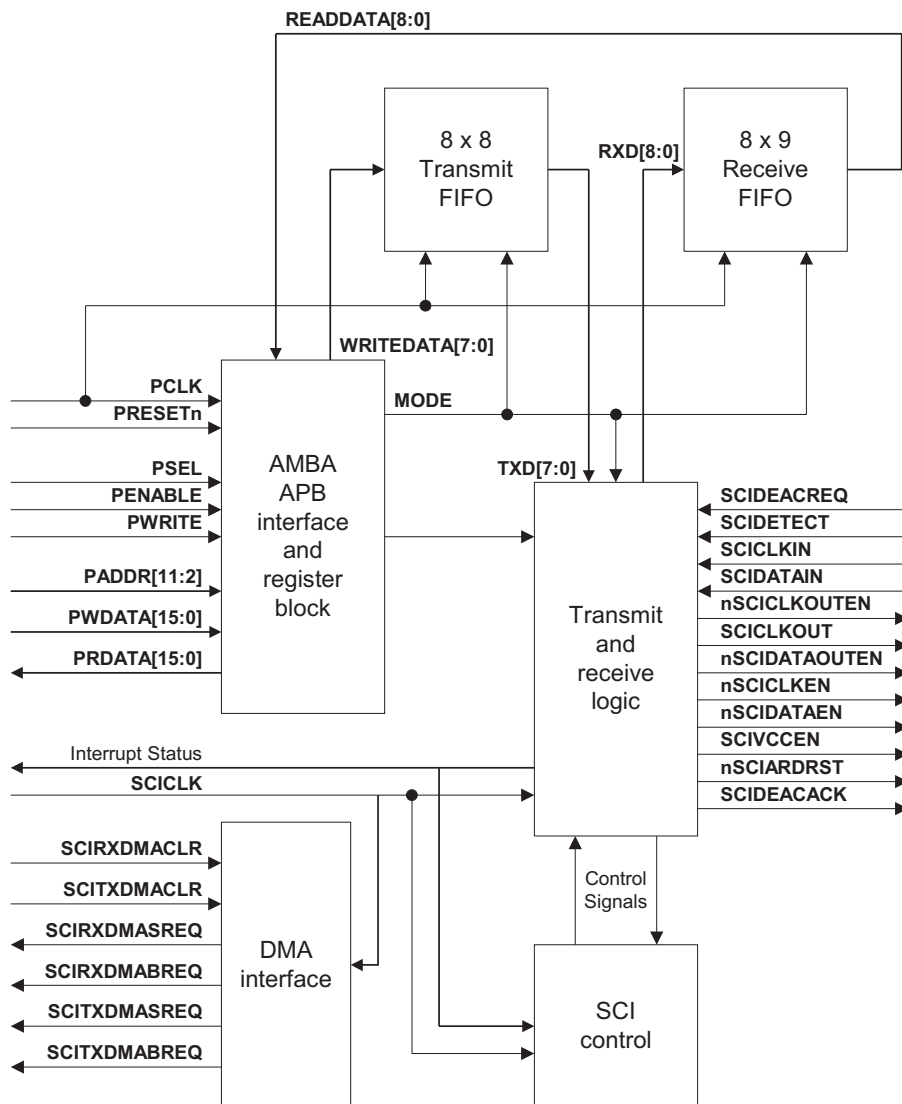
Card deactivation is initiated automatically through hardware on card removal, but it is also possible to deactivate the card through software by writing to the respective control register. A second deactivation request input is available for direct control through an alternative hardware source.

## 2.2 PrimeCell SCI functional description

A diagrammatic view of the PrimeCell SCI is shown in Figure 2-1.

### - Note

For clarity test logic is not shown.



### Figure 2-1 PrimeCell SCI block diagram

The PrimeCell SCI is described in the following sections:

- *AMBA APB interface* on page 2-4
- *Register block* on page 2-4
- *Transmit and receive logic* on page 2-4
- *SCI control logic* on page 2-5
- *Transmit FIFO* on page 2-5
- *Receive FIFO* on page 2-6
- *Interrupt generation logic* on page 2-6
- *DMA interface* on page 2-6
- *Synchronizing registers and logic* on page 2-7
- *Test registers and logic* on page 2-7.

### 2.2.1 AMBA APB interface

The AMBA APB interface generates read and write decodes for accesses to the status/control registers and transmit/receive FIFO memories.

The AMBA APB is a local secondary bus which provides a low-power extension to the higher bandwidth AMBA *Advanced High-performance Bus* (AHB), within the AMBA system hierarchy. The AMBA APB groups narrow-bus peripherals to avoid loading the system bus, and provides an interface using memory-mapped registers which are accessed under programmed control.

### 2.2.2 Register block

The register block stores data written or to be read across the AMBA APB interface.

### 2.2.3 Transmit and receive logic

This block implements the main functionality of the PrimeCell SCI with the control information being fed from the PrimeCell SCI control block. It contains all the counters used for timing the various stages of transactions during a card session. Also, it drives all the card control signals with the exception of data.

In this implementation some counters are multi-purpose, that is, they are loaded with values required by a particular transaction stage, thereby minimizing the logic required to perform the overall function.

The following timers are included:

- The SCIBAUDCNT and SCIVALECNT counters provide the means of performing the bit rate conversion and bit rate adjustment. Together they define the *elementary time unit* (etu) period.
- The card clock, **SCICLKOUT**, is derived from the output of the SCICLKICCCNT divide counter.

The **SCIACTTIME** counter is multi-purpose and is used for six different operations:

- debounce timing
- activation timing
- *answer to reset* (ATR) start time
- ATR duration time
- deactivation time
- warm reset timing.

The **SCIDATETIME** counter is a multi-purpose counter that is used to measure:

- block guard time
- character guard time
- time between characters
- time between characters and blocks.

The **SCIACTTIME** and **SCIDATETIME** counters are controlled by the SCI Control block.

The transmit and receive block controls the sequencing of the power, clock, reset and data line during activation and deactivation processes.

With the exception of the transmit and receive FIFO interrupts, this block generates all the remaining interrupts that provide a means of monitoring the individual stages of a card session.

## 2.2.4 SCI control logic

This block controls the PrimeCell SCI using the timeouts and other control signals received from the transmit and receive logic block.

## 2.2.5 Transmit FIFO

The transmit FIFO is an 8-deep 8-bit wide circular buffer. Reads and writes to the transmit FIFO result in an access to buffer locations pointed to by a read and write pointer respectively. The transmit FIFO is used to buffer data that is subsequently sent to the card.

The Transmit Tide Mark Interrupt **SCITXTIDEINTR** is generated from within this block and is asserted when the level falls below the programmed value.

If there is an unsuccessful transmission when using the T0 protocol, the PrimeCell SCI asserts the SCI Transmit Error Interrupt (**SCITXERRINTR**) and sets bit [4], the SCI Transmit Error Interrupt Status (**SCITXERRIS**) bit, of the **SCIIR** register. The PrimeCell SCI stops transmitting and, before any further characters can be transmitted, the error condition must be cleared by flushing the transmit FIFO.

### 2.2.6 Receive FIFO

The receive FIFO is an 8-deep 9-bit wide circular buffer. Reads and writes to the receive FIFO result in an access to buffer locations pointed to by a read and write pointer respectively. Data is stored in the least significant eight bits, and the ninth bit is set when a parity error is detected on the received data.

The **SCIRXTIDEINTR** interrupt is generated from within this block and is asserted when the level rises above the programmed value.

An SCI Receive Timeout Interrupt (**SCIRTOUTINTR**) is asserted and bit [9], the SCI Receive Timeout Status (**SCIRTOUTIS**), of the **SCIIR** register is set under the following condition:

The receive FIFO contains at least one character, and no characters have been read for a time corresponding to the value programmed in the **SCIRXTIME** register.

If the receive FIFO is full and another character is received, the SCI Receive OverRun Interrupt (**SCIROINTR**) is asserted, and bit [10] (**RORRIS**) is set in the **SCIRIS** register.

### 2.2.7 Interrupt generation logic

Individual maskable active HIGH interrupts are generated by the PrimeCell SCI. A combined interrupt output is also generated as an OR function of the individual interrupt requests.

You can use the single combined interrupt with a system interrupt controller that provides another level of masking on a per-peripheral basis. This enables you to use modular device drivers that always know where to find the interrupt source control register bits.

You can also use the individual interrupt requests with a system interrupt controller that provides masking for the outputs of each peripheral. In this way, a global interrupt service routine can read the entire set of sources from one wide register in the system interrupt controller. This is useful where the time to read from the peripheral registers is significant compared to the CPU clock speed in a real-time system.

The peripheral supports both the above methods.

### 2.2.8 DMA interface

The PrimeCell SCI provides an interface to connect to the DMA controller. See *PrimeCell SCI DMA interface* on page 2-25 for more information.

## 2.2.9 Synchronizing registers and logic

The PrimeCell SCI supports both asynchronous and synchronous operation of the clocks, **PCLK** and **SCICLK**. Synchronization registers and handshaking logic have been implemented, and are active at all times. This has a minimal impact on performance or area. Synchronization of control signals is performed on both directions of data flow, that is from the **PCLK** to the **SCICLK** domain, and from the **SCICLK** to the **PCLK** domain.

## 2.2.10 Test registers and logic

There are registers and logic for functional block verification, and integration testing using TicTalk or code based vectors.

Test registers must not be read or written to during normal use.

The integration testing verifies that the SCI has been wired into a system correctly. It enables each input and output to be both written to and read.

## 2.3 PrimeCell SCI operation

The PrimeCell SCI provides a communication medium for the data transactions between a Smart Card and hardware/software that effectively form a Smart Card Reader.

The operation of the PrimeCell SCI is described in the following sections:

- *Interface reset*
- *Clock signals*
- *Response to an ideal card session* on page 2-9
- *Warm reset sequence* on page 2-15
- *Response to a non-ideal card session* on page 2-16
- *Data transfer* on page 2-18
- *Character framing* on page 2-20
- *EMV character timing for T=0 (character protocol)* on page 2-21
- *EMV character timing for T=1 (block protocol)* on page 2-22
- *Transmit* on page 2-22
- *Receive* on page 2-23
- *Block time and time between characters* on page 2-23
- *Parity error* on page 2-24
- *RXREAD interrupt* on page 2-24.

### 2.3.1 Interface reset

The PrimeCell SCI is reset by the global reset signal **PRESETn** and a block-specific reset signal **nSCIRST**. An external reset controller must use **PRESETn** to assert **nSCIRST** asynchronously and negate it synchronously to **SCICLK**. **PRESETn** should be asserted LOW for a period long enough to reset the slowest block in the on-chip system, and then taken HIGH again. The PrimeCell SCI requires **PRESETn** to be asserted for at least one period of **PCLK** and **nSCIRST** to be asserted for at least one period of **SCICLK**.

The values of the registers after reset are detailed in Chapter 3 *Programmer's Model*.

### 2.3.2 Clock signals

The PrimeCell SCI has two input clock signals, **PCLK** and **SCICLK**.

The **SCICLK** frequency value should be selected in accordance with the chosen method of operation. For further details refer to *Data transfer* on page 2-18.



**SCICLK** can be driven using the **PCLK** signal when the frequency value is suitable for the chosen method of operation, but separate clock signals might be required because of other system constraints. The recommended constraint on the ratio of frequencies for **PCLK** and **SCICLK** is:

$$F_{\text{SCICLK}} \leq 2 \times F_{\text{PCLK}}$$

The frequency of **SCICLK** must be less than twice the frequency of **PCLK**. If **SCICLK** is more than twice the frequency of **PCLK**, transmission rates might not be fast enough to comply with the specification.

There is no restraint if the frequency of **SCICLK** is less than the frequency of **PCLK**.

### 2.3.3 Response to an ideal card session

This section gives a brief overview of an ideal transaction sequence, and how it is monitored. Following this is a section that describes the typical errors that can occur during a non-ideal transaction and the ensuing actions.

Notification of stage completion, timeouts, and errors is provided to the host by the PrimeCell SCI through a choice of:

- fifteen direct interrupts
- a single ORed version, the SCI Interrupt signal **SCIINTR**, coupled with subsequent reading of the SCI Raw Interrupt status register **SCIRIS**.

#### ———— Note ————

In the following descriptions, setting of any of the fifteen individual inputs implies that the **SCIINTR** signal is also consequently set.

### Stages of a card session

A card session comprises the following stages:

- *PrimeCell SCI reset and initial configuration* on page 2-10
- *Smart card insertion and detection* on page 2-11
- *Contact activation and cold reset sequence* on page 2-12
- *Answer To Reset sequence* on page 2-13
- *Execution of a transaction* on page 2-13
- *Contact deactivation sequence and card removal* on page 2-13.

PrimeCell SCI reset and initial configuration

The PrimeCell SCI is reset by the asynchronous LOW application of **PRESETn** and **nSCIRST**. These signals are then synchronously removed with reference to their respective clocks. The PrimeCell SCI is then configured by writing the initial values listed in Table 2-1 in preparation for a card being inserted into a Smart Card Reader and subsequent communication.

———— **Note** —————

The SCIDTIME register controls the timing of the card deactivation sequence and must be initialized before the activation sequence takes place. This is mandatory. Failure to carry out this action can result in damage to the card if it is removed prematurely.

Table 2-1 Initial configuration values

Register	Value	Comments
SCICR0	0x0	The control register 0 is set to direct convention, even parity, receive/transmit handshaking disabled.
SCICR1	0x0	Receive mode, timeouts initially disabled.
SCIIER	0x7FFF	Enable all interrupts.
SCISTABLE	0x96	For a 100MHz (10ns) reference clock, the stable (debounce) time is in terms of multiples of 0.66ms (0xFFFF x 10ns). An initial value of 100ms is proposed.
SCIATIME	0xAFC8	The SCIATIME register must be programmed to between 40000 and 45000 (0xAFC8) smart card clock cycles to satisfy the minimum cold and warm reset <b>nSCICARDRST</b> LOW time requirements.
SCIDTIME	0x00C8	The SCIDTIME is in terms of smart card clock cycles. It times the three stages of the deactivation sequence. The deactivation sequence must complete within 100ms, and it is recommended that power is disabled within 1ms. Because of this, for a smart card clock period of 1µs, an initial value of 0x00C8 should be programmed.
SCIATRSTIME	0x9C40	The SCIATRSTIME is in terms of smart card clock cycles. After deassertion of the reset <b>nSCICARDRST</b> signal, the start of the ATR sequence must occur within 40000 (0x9C40) smart card cycles.
SCIATRDTIME	0x4B00	The SCIATRDTIME is in terms of <i>elementary time units</i> (etus). The complete ATR character sequence must be received within 19200 (0x4B00) etus.
SCICHTIME	0x2580	The SCICHTIME is in terms of etus and is the maximum interval between the leading edges of two consecutive characters. In the case of the ATR sequence this is 9600 (0x2580) etus. It is also applicable to characters in the transaction data stream and is T = 0, or T = 1 mode dependent.

Table 2-1 Initial configuration values (continued)

Register	Value	Comments
SCITIDE	0x10	This is the receive and transmit FIFO tide level. Program the value to 0x10 so that the <b>SCIRXTIDEINTR</b> interrupt is generated when the initial TS character is loaded into the receive FIFO.
SCICKICC	0x31	The SCICKICC value is used to divide down the reference clock to provide the smart card clock. The final smart card frequency should be within the range 1-5MHz. For a 100MHz reference clock, SCICKICC should be programmed with a value of 49 (0x31) to provide an initial 1MHz Smart Card clock frequency.
SCIBAUD	0xE91	For a 100MHz reference clock, 1MHz smart card clock, and an SCIVALUE of 10 (0xA), SCIBAUD should be programmed to 3729 (0xE91).
SCIVALUE	0xA	An SCIVALUE value of 10 (0xA) is proposed as the sample value.

### Smart card insertion and detection

A Smart Card is inserted into the Smart Card Reader.

The Smart Card Reader signals to the PrimeCell SCI that a card has been detected by setting the **SCIDTECT** signal HIGH. In response to this, the PrimeCell SCI starts its *debounce* timer.

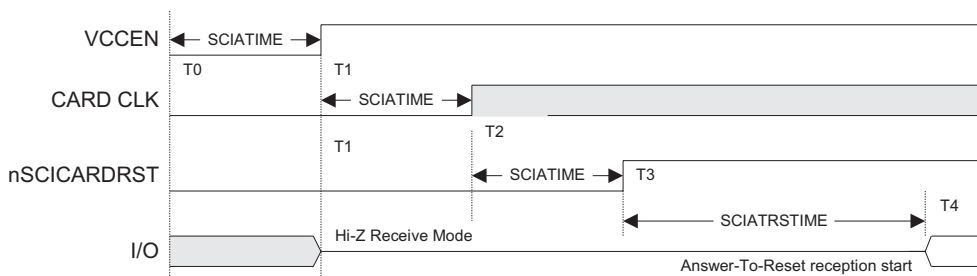
The Smart Card must remain in the interface for the debounce period, which is initially defined by the value written to the SCISTABLE register.

On expiry of the debounce period, the PrimeCell SCI notifies the host that a card has been successfully inserted by asserting the SCI Card In Interrupt signal **SCICARDININTR** and setting bit [0], the CARDINRIS bit, of the SCIRIS status register.

At this point there are no clocks or power applied to the Smart Card and the input/output signals are held LOW by the interface. These signals are then applied in a controlled way by the activation sequence.

## Contact activation and cold reset sequence

The PrimeCell SCI can now activate the card, that is, power up the inserted card in an ordered way. The PrimeCell SCI activation sequence has been divided into three equal phases, timed by the value programmed in the SCI activation time register **SCIATIME**. Figure 2-2 shows the combined cold reset and activation sequence.



**Figure 2-2 Asynchronous card cold reset and start of ATR reception**

The PrimeCell SCI activation sequence includes the *cold* reset sequence which must be in effect for between 40000 and 45000 Smart Card clock cycles. This is programmed using the **SCIATIME** register.

The activation sequence is initiated by writing a 1 to bit [0], **STARTUP**, of the SCI control register 2 (**SCICR2**).

The PrimeCell SCI performs the following activation sequence:

1. Assert **nSCICARDRST** LOW.
2. Wait for **SCIATIME** Smart Card clock cycles.
3. Enable VCC, configure the smart card input/output signal as high impedance.
4. Wait for **SCIATIME** Smart Card clock cycles.
5. Enable **SCICLKOUT** clock.
6. Wait for **SCIATIME** Smart Card clock cycles.
7. Deassert **nSCICARDRST** HIGH.

The PrimeCell SCI notifies the host that the activation sequence is complete by asserting the SCI card up interrupt signal **SCICARDUPINTR** and setting bit [2], **CARDUPRIS**, within the **SCIRIS** register.

The ATR on the input/output line from the Smart Card begins between 400 and 40000 cycles from reset deassertion.

## Answer To Reset sequence

The ATR sequence contains information about the card requirements for subsequent data transactions. The first character within the ATR stream is called the TS character and contains the convention information (direct or inverse format) on how the remaining ATR and future data transaction characters are interpreted.

On reception of the first character, the SCI RXTIDE interrupt **SCIRXTIDEINTR** signal is asserted and bit [10], the SCI RXTIDE interrupt status bit RXTIDERIS is set within the SCIRIS status register.

The PrimeCell SCI will read this character, establish the respective required convention and, if necessary, reprogram the SCI control register 0 (SCICR0). It is also recommended that the RXFIFO RXTIDE level be programmed to a higher value.

These processes should be completed prior to the start of reception of the next character.

In brief, the ATR sequence includes configuration values for:

- the clock frequency
- baud rate
- guard times
- protocol type.

The remainder of the ATR sequence is received, read using the RXFIFO in the selected convention, interpreted by the host software and the PrimeCell SCI programmed accordingly with the extracted values.

## Execution of a transaction

After the interface has been configured by extracting the parameters from the ATR stream, host to card communication, and vice versa, can now proceed.

The direction of flow is controlled through the value written to bit [2], the MODE bit, in the SCI control register 1 (SCICR1).

The host is always in control of how many characters it sends to the card and how many characters it expects to be returned by the card.

The character streams must meet the timing requirements of either the T0 or T1 protocol. These are covered in *EMV character timing for T=0 (character protocol)* on page 2-21 and *EMV character timing for T=1 (block protocol)* on page 2-22.

## Contact deactivation sequence and card removal

The final step in a typical card session is contact deactivation where the signals and power are removed in a defined sequence. Contact deactivation takes precedence over all other operations so that the card is not electrically damaged.

The deactivation sequence can be initiated by software by writing a 1 to bit [1], the FINISH bit, of the SCICR2 register. Alternatively, two hardware signals can be used:

- On detection of a card's removal at any time in a session, signified by the **SCIDTECT** signal being LOW, contact deactivation is initiated. The **SCIDTECT** signal usually originates from a Smart Card reader.
- In addition, it is possible to initiate contact deactivation by asserting the signal **SCIDEACREQ** which can be fed from an alternative source other than a Smart Card reader.

The PrimeCell SCI deactivation sequence has been divided into three equal phases, timed by the value programmed in the SCI Deactivation Time (SCIDTIME) register and is in terms of reference clock **SCICLK** periods.

The PrimeCell SCI deactivation sequence must complete within 1 millisecond. This means that the SCIDTIME register has to be programmed with a third of the total time. For example, for an **SCICLK** frequency of 100MHz (period 10ns) an SCIDTIME value of 20000 (0x4E20) would equate to a total deactivation time of around 0.6ms.

The PrimeCell SCI performs the following deactivation sequence:

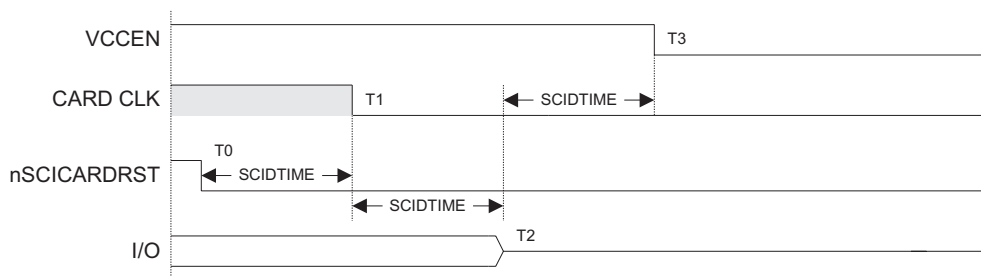
1. Assert **nSCICARDRST** LOW.
2. Wait for SCIDTIME reference clock (**SCICLK**) cycles.
3. Drive **SCICLKOUT** LOW.
4. Wait for SCIDTIME reference clock (**SCICLK**) cycles.
5. Drive **nSCIDATAEN** HIGH.
6. Wait for SCIDTIME reference clock (**SCICLK**) cycles.
7. Drive VCC LOW.

On completion of the deactivation sequence, the PrimeCell SCI asserts the SCI Card Down Interrupt signal **SCICARDDNINTR** and sets bit [3], the SCI Card Down Interrupt Status bit CARDDNRIS, of the SCIRIS status register.

The card can then be safely removed if required, but it can remain and be reactivated for another transaction if required by the host.

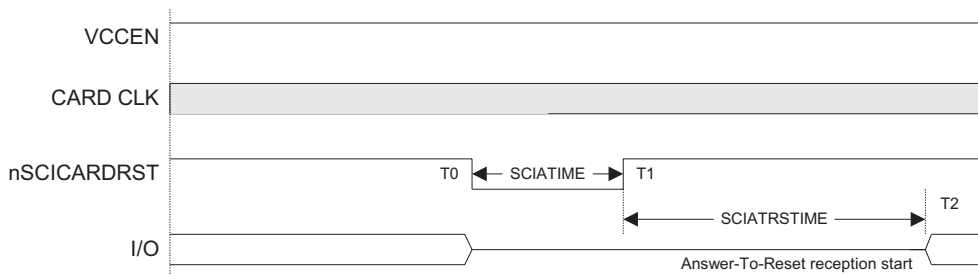
On recognition of the card being removed, that is, **SCIDTECT** transitioning from HIGH to LOW, the PrimeCell SCI asserts the SCI Card Out Interrupt **SCICARDOUTINTR** and sets bit [1], the SCI Card Out Interrupt Status bit (CARDOUTRIS) of the SCIRIS register.

2-15



### 2.3.4 Warm reset sequence

1. Assert **nSCICARDRST** LOW.
2. Maintain VCC and clock stable.
3. Put the PrimeCell SCI into reception mode.
4. Wait for SCIATIME Smart Card clock cycles.
5. Deassert **nSCICARDRST** HIGH.



The warm reset sequence is initiated by writing a 1 to bit [2], WRESET, of the SCICR2 control register.

The ATR on the input/output line from the Smart Card begins between 400 and 40000 cycles from reset deassertion.

If the start bit of the ATR stream is not received within this time, the PrimeCell SCI automatically initiates the deactivation sequence without the requirement for software intervention.

### 2.3.5 Response to a non-ideal card session

The PrimeCell SCI has to resolve non-ideal transactions such as:

- removal of the card before a transaction has completed
- timing/parity errors that can occur during the data flow.

The PrimeCell SCI monitors each transaction stage through interrupt and status generation, enabling software to respond accordingly. For more detailed information on the individual and final shared interrupts, see *Programmer's Model* on page 3-1.

Notification of stage completion, timeouts, and errors is provided to the host by the PrimeCell SCI through a choice of either fifteen direct interrupts or their single ORed version, the SCI Interrupt Signal **SCIINTR**, coupled with subsequent reading of the SCI Masked Interrupt Status Register SCIMIS.

#### ———— Note ————

In the following descriptions, setting of any of the fifteen individual inputs implies that the **SCIINTR** signal is also consequently set.

### Card removed at any time between activation and deactivation

The PrimeCell SCI must ensure that no electrical damage is caused to the card if it is removed while still powered up, that is, it must immediately be deactivated in a defined sequence.

Card deactivation takes precedence over any other operation and can be initiated by software or hardware. The card must be powered down in less than 100ms, which takes into account the decay time of the power supply. It is recommended that the SCI is programmed so that the VCCEN is deasserted within 1ms, enabling a further 99ms for the power supply to decay. See *Contact deactivation sequence and card removal* on page 2-13 for details of this sequence.

On recognition of the card being removed, that is, **SCIDTECT** transitioning from HIGH to LOW, the PrimeCell SCI asserts the SCI Card Out Interrupt **SCICARDOUTINTR** and sets bit [1], the SCI Card Out Interrupt Status bit (CARDOUTRIS) of the SCIRIS register.



### Card inserted, debounce time not met, then card removed

This is very similar to the above, though not as critical, as power has not been applied to the card. The card is only activated on successful completion of the debounce period. As above the card must immediately be deactivated using the controlled deactivation sequence.

On recognition of the card being removed, that is **SCIDTECT** transitioning from HIGH to LOW, the PrimeCell SCI asserts the SCI Card Out Interrupt **SCICARDOUTINTR** and sets bit [1], the SCI Card Out Interrupt Status bit (CARDOUTIS) of the SCIRIS register.

### Card inserted, debounce time met, no ATR start bit received within the specified time

The SCIATRSTIME register value is programmed with the value 40000. This represents the maximum number of Smart Card clock cycles during which the start bit of the card's ATR sequence should be received.

If the ATR start bit is not received before this maximum number of Smart Card clock cycles has expired, the PrimeCell SCI asserts the ATR Start Timeout Interrupt **SCIATRSTOUTINTR** and sets bit [5] of the SCIRIS register, the SCI Start Time Out Interrupt Status bit (ATRSTOUTRIS).

The PrimeCell SCI automatically initiates the deactivation sequence without the requirement for software intervention.

### ATR start bit received, but time between two successive characters exceeds the specified time

If the ATR start bit is received within the specified time, the time between leading edges of the ATR characters is checked to be less than the specified maximum limit.

This value is programmed into the SCI Character Time register (SCICHTIME).

During the ATR, the delay between the leading edges of any two consecutive characters from the card must be a minimum of 12, but not more than 9600 etus. See Table 2-1 on page 2-10 for more details of the definition of the initial etu values for ATR reception.

If the SCICHTIME value is exceeded at any time during reception of the ATR data stream, the SCI Character TimeOut Interrupt signal (**SCICHTOUTINTR**) is asserted and bit [8], the SCI Character TimeOut Interrupt status bit (CHTOUTRIS), of the SCIRIS register is set.

If the card has an external reset, a warm reset sequence is initiated by writing a 1 to the WRESET bit of the SCICR2 register.

If the card has an internal reset, the host deactivates the card by writing a 1 to bit [2], FINISH, of the SCICR2 register.

### **ATR start received, but the duration of the total ATR data stream exceeded the specified time**

The card must transmit all the characters to be returned during an ATR within 19200 etus. This time is measured between the leading edge of the start bit of the first character (TS) and 12 etus after the leading edge of the start bit of the last character.

The maximum value, which is currently fixed at 19200 by the *EMV Specification*, is programmed into the SCI ATR Duration Time register (SCIATRDTIME). If the SCIATRDTIME time is not met, the SCI ATR Duration TimeOut Interrupt signal **SCIATRDTOUITNTR** is asserted and bit [6], the SCI ATR TimeOut Status bit (ATRDTOUITRIS) of the SCIRIS register is set.

In response to the interrupt, a warm reset sequence is initiated by writing a 1 to the WRESET bit of the SCICR2 register.

If the card has an internal reset, the host deactivates the card by writing a 1 to bit [2], FINISH, of the SCICR2 register.

### **ATR received, but parity errors are found within the received data.**

If, during the reception of the ATR stream, an error such as parity failure is recognized by the PrimeCell SCI or the associated software, the PrimeCell SCI initiates a warm reset by writing a 1 to the WRESET bit of the SCICR2 control register.

### **Data transaction in progress, time for block arrival exceeded**

If the maximum delay from start leading edges of the last character from the PrimeCell SCI, that gave the right to send to the card, and the first character sent by the card exceeds a specified time, the PrimeCell SCI asserts the SCI Block TimeOut Interrupt (**SCIBLKTOUITNTR**) and sets bit [7], the SCI Block TimeOut Interrupt Status (BLKTOUITRIS) bit, of the SCIRIS register.

The block timeout value is programmed by writing to the SCIBLKTIME register.

## **2.3.6 Data transfer**

### **Data rates**

The duration of a bit within a character is termed the *elementary time unit* (etu). The etu is set by programming the SCIBAUD and SCIVALUE registers.

## Value X BAUD rate clock

The value in the SCIBAUD register is used to define a clock which is a multiple of the baud rate. This is known as the Value X BAUD rate clock. The Value X BAUD rate clock is generated by dividing the reference clock by  $1 + \text{SCIBAUD}$ . The SCIVALUE register defines the number of Value X BAUD rate clock periods which make up an etu. The etu is programmable and has different values depending on the stage of card processing.

During the ATR, the bit duration is known as the initial etu and is given by the following equation:

$$\text{initial etu} = \frac{372}{f} \text{ seconds}$$

where  $f$  is the Smart Card clock frequency in Hertz.

Following the ATR (and establishment of the global parameters  $F$  and  $D$ ), the bit duration is known as the current etu, and is given by the following equation:

$$\text{current etu} = \frac{F}{f} \times \frac{1}{D} \text{ seconds}$$

where  $F$  and  $D$  are the clock rate conversion and bit rate adjustment parameters returned by the card, and  $f$  is the clock frequency applied to the Smart Card.

The etu is set by programming the SCIBAUD and SCIVALUE registers.

The SCIVALUE defines the number of baud rate clock periods that define the etu.

Therefore:

$$1 \text{ etu} = \frac{(1 + \text{SCIBAUD})}{(\text{Reference Clock})} \times \text{SCIVALUE}$$

Thus the following equation must always be satisfied:

$$\frac{(1 + \text{SCIBAUD})}{(\text{Reference Clock})} \times \text{SCIVALUE} = \frac{F}{f} \times \frac{1}{D}$$

See ISO 7816-3 for the possible values of  $F$  and  $D$  that can be returned by the card.

### ————— **Note** —————

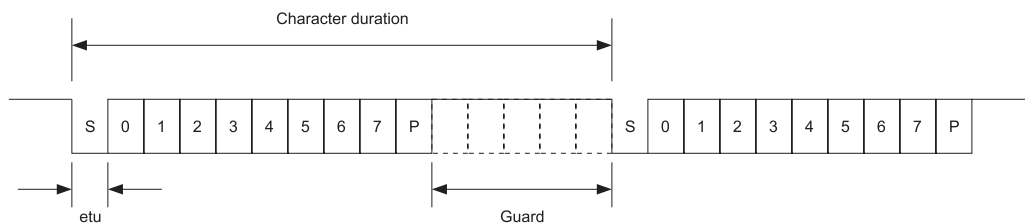
The EMV standard specifies that  $f$  must be in the range 1–5 MHz. ISO 7816-3 merely specifies a lower bound of 1 MHz.

### 2.3.7 Character framing

Figure 2-5 shows the structure of a character. Each character consists of:

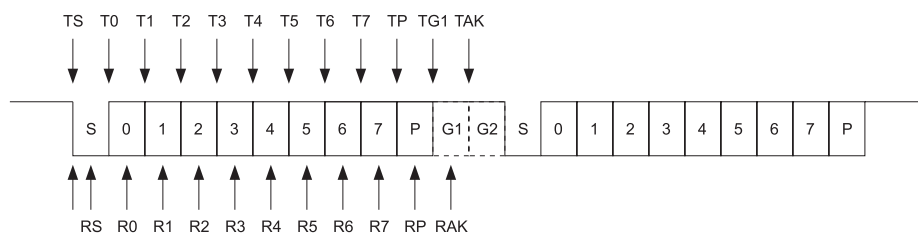
- a start bit
- eight data bits
- a parity bit
- a number of bits making up the *guard time* before the start bit of the next character.

The *guard time* is the delay between the trailing edge of the parity bit of a character, and the leading edge of the start bit of the next character. The guard time for characters transmitted by the interface is controlled by the SCICHGUARD register.



**Figure 2-5 Character structure**

Figure 2-6 shows how the interface interprets characters transmitted from the card. At time TS, the card pulls the bidirectional input/output line LOW to begin the start bit.

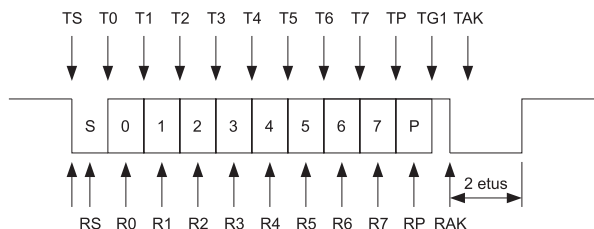


**Figure 2-6 Character timings**

The interface detects the leading edge of the start bit after four reference clock cycles. The input/output line is sampled at RS, approximately half an etu after the leading edge of the start bit. If the input/output line is not LOW, the start bit is deemed to be invalid and is ignored by the interface.

The first data bit is sampled at 1 etu after RS, and subsequent data bits (including the parity bit) are sampled at 1 etu intervals. The T0 protocol defines a mechanism whereby the receiver can request retransmission of a character by pulling the input/output line LOW during the guard time following the character. When a retransmission is requested, the interface starts to pull the input/output line LOW at RAK.

The transmitter (the card in this case) samples the input/output line at TAK. The interface holds the input/output line for a total of 2 etus. This is shown in Figure 2-7.



**Figure 2-7 T0 retransmit request**

### 2.3.8 EMV character timing for T=0 (character protocol)

The minimum interval between the leading edges of the start bits of two consecutive characters sent by the interface to the Smart Card is between 12 and 266 etus as indicated by the value of TC1 returned within the ATR stream.

The minimum interval between the leading edges of the start bits of two consecutive characters sent by the Smart Card to the interface is 12 etus.

The maximum interval between the start leading edge of any character sent by the Smart Card and the start leading edge of the previous character sent either by the Smart Card or the interface (the *work waiting time*) must not exceed  $960 \times D \times WI = 9600$  etus. (The bit rate conversion factor, D, has a default value of 1. WI does not have a default value of 10 because TC2, which contains the value, is not returned in the ATR.)

The minimum interval between the leading edges of the start bits of two consecutive characters sent in opposite directions is 16 etus.

#### **Note**

The minimum interval between the leading edges of the start bits of two characters sent by interface to the Smart Card is always governed by the value of TC1. It can be less than minimum interval of 16 etus allowed between two characters sent in opposite directions.

### 2.3.9 EMV character timing for T=1 (block protocol)

The minimum interval between the leading edges of the start bits of two consecutive characters, sent by the interface to the Smart Card must be between 11 and 266 etus as indicated by the value of TC1 returned within the ATR stream.

The minimum interval between the leading edges of the start bits of two consecutive characters sent by the Smart Card to the interface is 11 etus.

The maximum interval between the leading edges of the start bits of two consecutive characters in the same block the *Character Waiting Time* (CWT) must not exceed  $(2 \times \text{CWI} + 11)$  etus. The *Character Waiting Integer* (CWI) must have a value of 0 to 5, resulting in the CWT having the range 12 to 43 etus.

The maximum interval between the leading edge of the start bit of the last character that gave the right to send to the Smart Card and the leading edge of the first character sent by the Smart Card the *Block Waiting Time* (BWT) must not exceed  $((2 \times \text{BWI} \times 960) + 11)$  etus. The *Block Waiting Integer* (BWI) has a value in the range 0 to 4, resulting in the BWT having the range 971 to 15371 etus.

The minimum interval between the leading edges of the start bits of two consecutive characters sent in opposite directions the *Block Guard Time* (BGT) is 22 etus.

### 2.3.10 Transmit

Characters that are to be sent to the card are first written into the SCIDATA FIFO and then automatically transmitted to the card at timed intervals. Direction of communication is controlled by the MODE bit of the SCICR1 register. Changing from transmit to receive does not take place until the last character stored in the transmit FIFO has been sent. This enables the MODE bit to be written immediately after writing the last character in a block. This is necessary because the card can respond to the transmission almost immediately (minimum turnaround time measured from start bit to start bit is 16 etus for T0 and 22 etus for T1).

If character-transmit handshaking is enabled (mandatory for T0), the input/output line is sampled at 1 etu after the parity bit. If the card indicates that it did not receive the character correctly, the character is retransmitted a maximum of TXRETRY times (set using the SCIRETRY register) before the transmission is aborted and a **SCITXERRINTR** interrupt generated. The interface waits for four etus after an error is detected before the character is retransmitted. If a character fails to be transmitted and a **SCITXERRINTR** interrupt is generated, the transmit/receive interlock mechanism must be reset by flushing the transmit FIFO before any subsequent transmit or receive operation.

The interval between successive characters sent by the interface is governed by the SCICHGUARD register, which defines the character guard time. SCICHGUARD is only used to control the transmission of characters and is not used by the receive hardware.

The minimum interval between the last character sent by the card and the next character sent by the interface is governed by the SCIBLKGUARD register, which defines the block guard time.

When the number of characters held in the transmit FIFO falls below the level defined in the SCITXTIDE register, a **SCITXTIDEINTR** interrupt is generated. The number of characters held in the transmit FIFO can be determined by reading the SCITXCOUNT register. Writing to the SCITXCOUNT register flushes the transmit FIFO.

### 2.3.11 Receive

Characters are read from the interface by reading the SCIDATA register. When read, SCIDATA has nine bits:

- eight data bits
- one parity status bit.

See *Data register, SCIDATA* on page 3-8 for more information.

Before characters can be read from the interface, the MODE bit of the SCICR1 register must be set to 0. An interlock mechanism ensures that if the MODE bit is set to receive and there are still characters remaining in the transmit FIFO, these are transmitted before any characters are read from the card.

To switch from receive mode to transmit mode, write a 1 to the MODE bit within the SCICR1 control register during or after reception of the final byte within the incoming datastream.

### 2.3.12 Block time and time between characters

Two registers define an upper limit on the time waited for a character to be transmitted by the card:

SCIBLKTIME	defines the timeout limit for the first character in a block
SCICHTIME	defines the maximum allowed time between characters (excluding the first character in a block).

The SCICHTIME counter is linked to the SCIBLKTIME counter and does not start until the SCIBLKTIME counter has stopped (That is, the first character in the block has arrived, or if the block timer has been disabled). The transmit/receive interlock mechanism prevents the SCICHTIME and SCIBLKTIME counters from running while characters are still present in the transmit FIFO.

### 2.3.13 Parity error

If character-transmit handshaking is enabled ( $RXNAK = 1$ ) and the interface detects a parity error, it signals this to the card by pulling the input/output line down for 2 etus at 10.5 etus after the leading edge of the start bit.

The maximum number of times the interface attempts to receive a character is governed by the `SCIRETRY` register. If, after `RXRETRY` further attempts, the character has not been successfully received, a parity error for the character is flagged. Bit [8] of the `SCIDATA` register is set to 1 to indicate that a parity error has occurred.

### 2.3.14 RXREAD interrupt

An `RXREAD` interrupt is caused by a read access timeout (defined by `SCIRXTIME`). This occurs if the receive FIFO contains at least one character, and no characters have been read for a time corresponding to `SCIRXTIME` Smart Card clock cycles.

The read access timeout limit can be reprogrammed dynamically, which provides a way of enabling the receive timeout mechanism only at the end of blocks. This is achieved by initially setting the receive timeout threshold to a high enough value to guarantee that an **`RXREAD`** interrupt can only have been caused by a tide mark condition (excluding an error condition where a card stops transmitting part way through a block). When the end of the block is reached, `RXTIME` is reprogrammed with the correct value to process the trailing characters in the block.



## 2.4 PrimeCell SCI DMA interface

The PrimeCell SCI provides an interface to connect to the DMA controller. The DMA operation of the SCI is controlled through the SCI DMA control register, SCIDMACR. The DMA interface includes the following signals:

For receive:

### **SCIRXDMASREQ**

Single character DMA transfer request, asserted by the SCI. For receive, one character consists of up to nine bits. This signal is asserted when the receive FIFO contains at least one character.

### **SCIRXDMABREQ**

Burst DMA transfer request, asserted by the SCI. This signal is asserted when the receive FIFO contains more than or an equal number of characters defined by the programmed watermark level. You can program the watermark level for each FIFO through the SCITIDE register.

### **SCIRXDMACLR**

DMA request clear, asserted by the DMA controller to clear the receive request signals. If DMA burst transfer is requested, the clear signal is asserted during the transfer of the last data in the burst.

For transmit:

### **SCITXDMASREQ**

Single character DMA transfer request, asserted by the SCI. For transmit one character consists of up to eight bits. This signal is asserted when there is at least one empty location in the transmit FIFO.

### **SCITXDMABREQ**

Burst DMA transfer request, asserted by the SCI. This signal is asserted when the transmit FIFO contains less than or an equal number of characters defined by the programmed watermark level. You can program the watermark level for each FIFO through the SCITIDE register.

### **SCITXDMACLR**

DMA request clear, asserted by the DMA controller to clear the transmit request signals. If DMA burst transfer is requested, the clear signal is asserted during the transfer of the last data in the burst.

The burst transfer and single transfer request signals are not mutually exclusive, they can both be asserted at the same time. For example, when there is more data than the watermark level in the receive FIFO, the burst transfer request and the single transfer request are asserted. When the amount of data left in the receive FIFO is less than the watermark level, the single request only is asserted. This is useful for situations where the number of characters left to be received in the stream is less than a burst.

For example, if 19 characters have to be received and the watermark level is programmed to be four. The DMA controller then transfers four bursts of four characters and three single transfers to complete the stream.

———— **Note** —————

For the remaining three characters the SCI cannot assert the burst request.

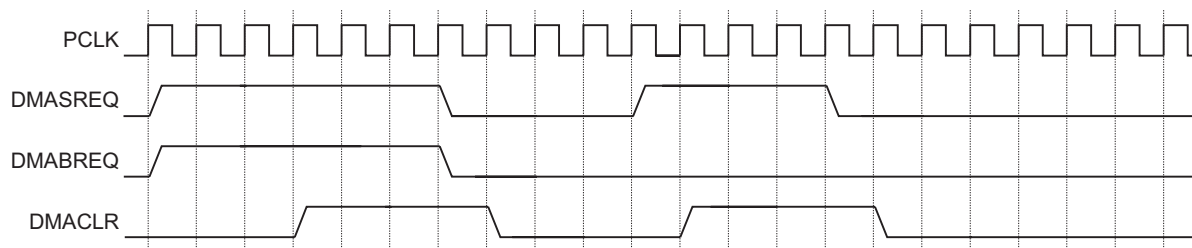
Each request signal remains asserted until the relevant **DMACLR** signal is asserted. After the request clear signal is deasserted, a request signal can become active again, depending on the conditions described above. All request signals are deasserted if the SCI is disabled or the DMA enable signal is cleared.

Data transfers can be made by either single or burst transfers depending on the programmed watermark level and the amount of data in the FIFO. Table 2-2 shows the trigger points for **DMABREQ** depending on the watermark level, for both the transmit and receive FIFOs.

**Table 2-2 DMA trigger points for the transmit and receive FIFOs**

Watermark level	Burst length	
	Transmit (number of empty locations)	Receive (number of filled locations)
1	7	1
2	6	2
3	5	3
4	4	4
5	3	5
6	2	6
7	1	7
8	0	8

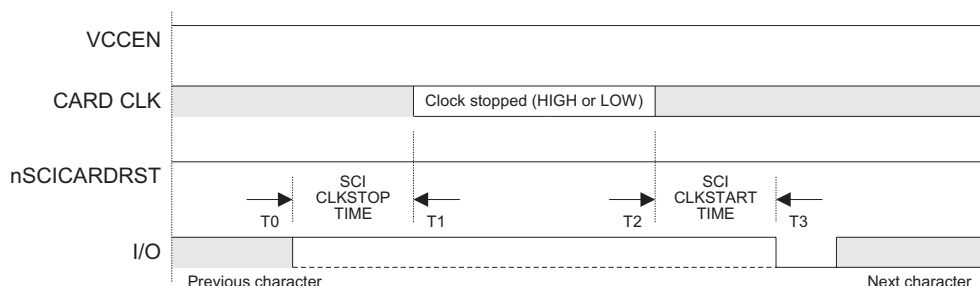
Figure 2-8 shows the timing diagram for both a single transfer request and a burst transfer request with the appropriate DMA clear signal. The signals are all synchronous to **PCLK**. For the sake of clarity it is assumed that there is no synchronization of the request signals in the DMA controller.



**Figure 2-8 DMA transfer waveforms**

## 2.5 SCI clock stop mode

ISO 7816-3 states that for cards supporting the clock stop mode, when the interface expects no transmission from the card and when the I/O has remained in a high impedance state for at least 1860 card clock cycles, the interface can stop the card clock. Figure 2-9 shows the clock stop mode.



**Figure 2-9 Clock stop mode**

This procedure is initiated by writing a 1 to bit [6], CLKDIS, in the SCICR0 register.

The stop time duration is in terms of card clock cycles and, in the case of the SCI, it is programmable through the 12-bit wide SCISTOPTIME register. When this duration has elapsed, the card clock stops and the clock stopped interrupt, SCICLKSTPINTR, is asserted to inform the system that the card clock is inactive.

The inactive state of clock is controlled through bit [7], CLKVAL, in the SCICR0 register. When this bit is programmed as 0, the clock is held LOW when inactive. When high, the clock is held HIGH when inactive. The clock can be restarted by writing a 0 to the CLKDIS bit.

ISO 7816-3 states that information exchange on the I/O can only continue after at least 700 card cycles have elapsed. In the case of the SCI, this duration is programmable through the 12-bit wide SCISTARTTIME register.

The clock becomes active on detection of the CLKDIS value being 0. When the SCISTARTTIME duration has elapsed, the clock active interrupt, SCICLKACTINTR, is asserted to inform the system that information exchange can continue.

Figure 2-9 shows the sequence of events for stopping and restarting of the clock, where at time:

- T0, the card clock stop is initiated
- T1, the card clock becomes inactive (LOW or HIGH)
- T2, the card clock start is initiated
- T3, information exchange can continue.

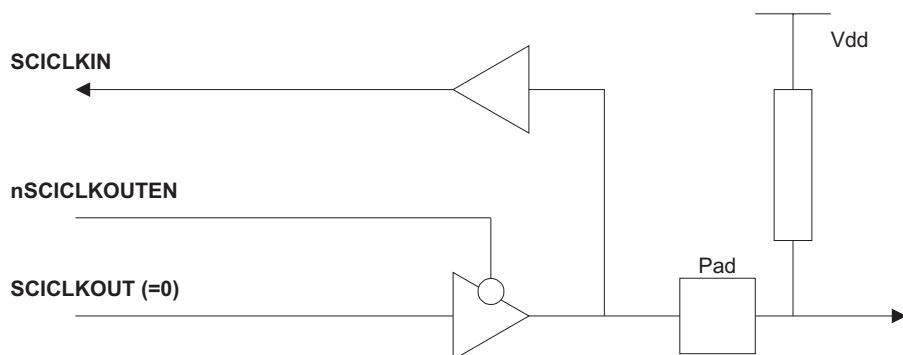
## 2.6 PrimeCell SCI clock and data driver configurations

The PrimeCell SCI has been designed so that the clock and data pads can be configured to drive on-chip open drain pads or external off-chip buffers. Four configurations are available:

- *On-chip open drain CLOCK configuration (nSCICLKOUTEN used to enable pulldown) on page 2-29*
- *Off-chip buffer driven CLOCK configuration (SCICLKOUT clock, nSCICLKEN tristate control) on page 2-30*
- *On-chip open drain DATA configuration (nSCIDATAOUTEN used to enable pulldown) on page 2-31*
- *Off-chip buffer driven DATA configuration (nSCIDATAOUTEN data, nSCIDATAEN tristate control) on page 2-32*
- *Instantiating two data out pads on page 2-33.*

### 2.6.1 On-chip open drain CLOCK configuration (nSCICLKOUTEN used to enable pulldown)

When configured as shown in Figure 2-10, CLKZ1 (bit 3) in the SCI control register SCICR1 must be programmed to a 1, which causes a 0 to be permanently driven onto the SCICLKOUT signal pad input. The SCICLKOUT pad can either drive a zero when the active LOW enable signal **nSCICLKOUTEN** is a 0 or be pulled HIGH when it is a 1. The required clock signal sequence is provided by the PrimeCell SCI state machine.



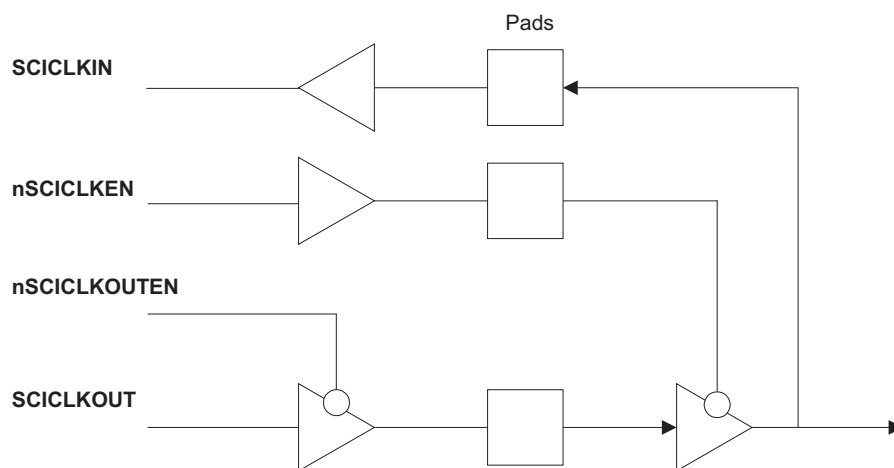
**Figure 2-10 On-chip open drain CLOCK configuration**

In the configuration shown in Figure 2-10:

- **SCICLKOUT** is forced LOW by the chip
- the open drain output is a full p/n channel driver, with the p channel permanently turned off in this configuration.

## 2.6.2 Off-chip buffer driven CLOCK configuration (SCICLKOUT clock, nSCICLKEN tristate control)

When configured as shown in Figure 2-11, CLKZ1 (bit 3) in the SCI control register SCICR1 must be programmed to a 0, which causes a 0 to be permanently driven onto the active LOW **nSCICLKOUTEN** enable input. The **SCICLKOUT** pad is permanently enabled and feeds the input of an off-chip non-inverting buffer, controlled by the active low enable signal **nSCICLKEN**. The required clock signal sequence is provided by the PrimeCell SCI state machine.



**Figure 2-11 Off-chip buffer driven CLOCK configuration**

In the configuration shown in Figure 2-11:

- **nSCICLKOUTEN** is forced low by the chip, which permanently enables the **SCICLKOUT** pad
- the off-chip buffer tristate control is provided by **nSCICLKEN**.

### 2.6.3 On-chip open drain DATA configuration (nSCIDATAOUTEN used to enable pulldown)

When configured as shown in Figure 2-12, **nSCIDATAEN** is not used and the SCI DATA pad can either drive a zero when the active low enable signal **nSCIDATAOUTEN** is a 0, or be pulled high when it is a 1. There is no requirement to program CLKZ1 (bit 3) in the SCICR1 register. The required data signal sequence is provided by the PrimeCell SCI state machine.

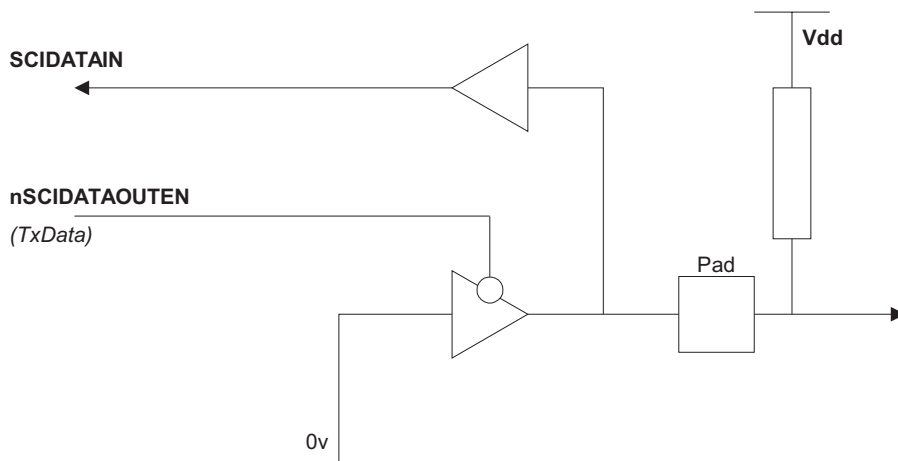


Figure 2-12 On-chip open drain DATA configuration

### 2.6.4 Off-chip buffer driven DATA configuration (nSCIDATAOUTEN data, nSCIDATAEN tristate control)

When configured as shown in Figure 2-13, the external buffer data input is fed by **nSCIDATAOUTEN** and controlled by the active low **nSCIDATAEN** signal. There is no requirement to program CLKZ1 (bit 3) in the SCICR1 register. The required data signal sequence is provided by the PrimeCell SCI state machine.

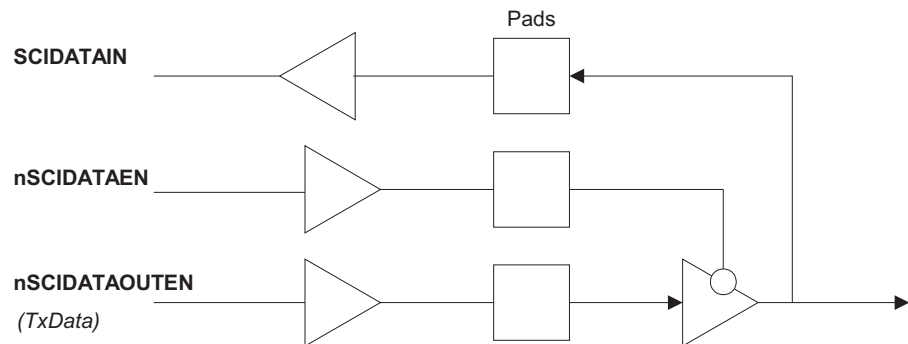


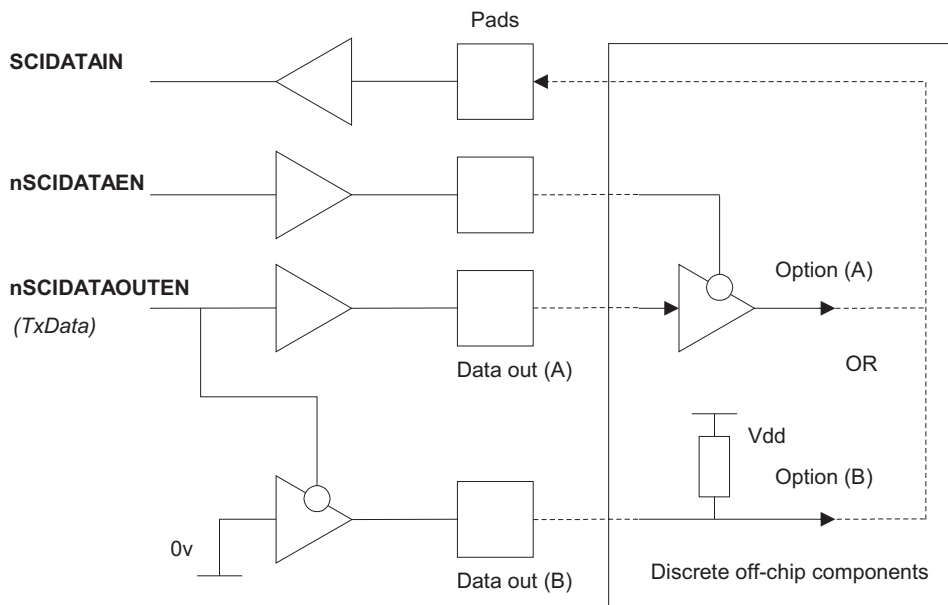
Figure 2-13 Off-chip buffer driven DATA configuration



### 2.6.5 Instantiating two data out pads

To provide the option of being able to drive the external data out line with either an open drain or buffered output, two data out buffers can be instantiated. Figure 2-14 shows how these two data pads are connected.

The connection back from either of the data out driver options (A) or (B) is then made external to the SCI interface.



**Figure 2-14 Instantiation of two data out pads to provide data drive options**



# Chapter 3

## Programmer's Model

This chapter describes the ARM PrimeCell Smart Card Interface (PL131) registers and provides details required when programming the microcontroller. It contains the following sections:

- *About the programmer's model* on page 3-2
- *Summary of PrimeCell SCI registers* on page 3-3
- *Register descriptions* on page 3-7
- *Interrupts* on page 3-40.

### **3.1 About the programmer's model**

The base address of the PrimeCell SCI is not fixed, and can be different for any particular system implementation. The offset of any particular register from the base address, however, is fixed.

The following locations are reserved, and must not be used during normal operation:

- locations at offset 0x88 to 0xEFC are reserved and must not be used
- locations at offsets 0xF00 to 0xF10 are reserved for test purposes
- locations at offsets 0xF14 to 0xFCC are reserved and must not be used
- locations at offsets 0xFD0 to 0xFDC are reserved for possible future extensions.

## 3.2 Summary of PrimeCell SCI registers

The PrimeCell SCI registers are shown in Table 3-1.

**Table 3-1 PrimeCell SCI register summary**

Address	Type	Width	Reset value	Name	Description
SCI Base + 0x00	Read/ write	9 8	0x--	SCIDATA	Data register. See Table 3-2 on page 3-8.
SCI Base + 0x04	Read/ write	8 8	0x00	SCICR0	Control register 0. See Table 3-3 on page 3-9.
SCI Base + 0x08	Read/ write	7 7	0x00	SCICR1	Control register 1. See Table 3-4 on page 3-11.
SCI Base + 0x0C	Write	3	0x0	SCICR2	Control register 2. See Table 3-5 on page 3-12.
SCI Base + 0x10	Read/ write	8 8	0x00	SCICLKICC	External Smart Card clock frequency. See Table 3-6 on page 3-13.
SCI Base + 0x14	Read/ write	8 8	0x00	SCIVALUE	Number of SCIBAUD cycles per etu. See Table 3-7 on page 3-13.
SCI Base + 0x18	Read/ write	16 16	0x0000	SCIBAUD	Baud rate clock divisor value. See Table 3-8 on page 3-13.
SCI Base + 0x1C	Read/ write	8 8	0x00	SCITIDE	Transmit and receive FIFO water level marks. See Table 3-9 on page 3-14.
SCI Base + 0x20	Read/ write	2 2	0x0	SCIDMACR	Direct Memory Access control register. See Table 3-10 on page 3-15.
SCI Base + 0x24	Read/ write	8 8	0x00	SCISTABLE	Card stable after insertion debounce duration. See Table 3-11 on page 3-15.
SCI Base + 0x28	Read/ write	16 16	0x0000	SCIATIME	Card activation event time. See Table 3-12 on page 3-16.
SCI Base + 0x2C	Read/ write	16 16	0x0000	SCIDTIME	Card deactivation event time. See Table 3-13 on page 3-16.
SCI Base + 0x30	Read/ write	16 16	0x0000	SCIATRSTIME	Time to start of the ATR reception. See Table 3-14 on page 3-17.

**Table 3-1 PrimeCell SCI register summary (continued)**

Address	Type	Width	Reset value	Name	Description
SCI Base + 0x34	Read/ write	16 16	0x0000	SCIATRDTIME	Maximum allowed duration of the ATR character stream. See Table 3-15 on page 3-17.
SCI Base + 0x38	Read/ write	12 12	0x000	SCISTOPTIME	Duration before the card clock can be stopped. See Table 3-16 on page 3-18.
SCI Base + 0x3C	Read/ write	12 12	0x000	SCISTARTTIME	Duration before transactions can commence after clock is restarted. See Table 3-17 on page 3-18.
SCI Base + 0x40	Read/ write	6 6	0x00	SCIRETRY	Retry limit register. See Table 3-18 on page 3-19.
SCI Base + 0x44	Read/ write	16 16	0x0000	SCICHTIMELS	Character to character timeout (least significant 16 bits). See Table 3-19 on page 3-20.
SCI Base + 0x48	Read/ write	16 16	0x0000	SCICHTIMEMS	Character to character timeout (most significant 16 bits). See Table 3-20 on page 3-21.
SCI Base + 0x4C	Read/ write	16 16	0x0000	SCIBLKTIMELS	Receive timeout between blocks (least significant 16 bits). See Table 3-21 on page 3-22.
SCI Base + 0x50	Read/ write	16 16	0x0000	SCIBLKTIMEMS	Receive timeout between blocks (most significant 16 bits). See Table 3-22 on page 3-22.
SCI Base + 0x54	Read/ write	8 8	0x00	SCICHGUARD	Character to character extra guard time. See Table 3-23 on page 3-23.
SCI Base + 0x58	Read/ write	8 8	0x00	SCIBLKGUARD	Block guard time. See Table 3-25 on page 3-24.
SCI Base + 0x5C	Read/ write	16 16	0x0000	SCIRXTIME	Receive read time-out register. See Table 3-26 on page 3-24.
SCI Base + 0x60	Read	4	0xa	SCIFIFOSTATUS	Transmit and receive FIFO status. See Table 3-28 on page 3-25.
SCI Base + 0x64	Read/ write	4 4	0x0	SCITXCOUNT	Transmit FIFO fill level. See Table 3-29 on page 3-26.

**Table 3-1 PrimeCell SCI register summary (continued)**

Address	Type	Width	Reset value	Name	Description
SCI Base + 0x68	Read/ write	4 4	0x0	SCIRXCOUNT	Receive FIFO fill level. See Table 3-30 on page 3-26.
SCI Base + 0x6C	Read/ write	15 15	0x0000	SCIIMSC	Interrupt mask set or clear mask. See Table 3-31 on page 3-27.
SCI Base + 0x70	Read/ write	15 15	0x400A	SCIRIS	Raw interrupt status register. See Table 3-32 on page 3-29.
SCI Base + 0x74	Read/ write	15 15	0x0000	SCIMIS	Masked interrupt status register. See Table 3-33 on page 3-30.
SCI Base + 0x78	Write	13	0x0000	SCIICR	Interrupt clear register. See Table 3-34 on page 3-32.
SCI Base + 0x7C	Read/ write	11 4	0x000	SCISYNCACT	Synchronous mode activation register. See Table 3-35 on page 3-33.
SCI Base + 0x80	Read/ write	6 6	0x00	SCISYNCTX	Synchronous mode transmit clock and data stream register. See Table 3-36 on page 3-34.
SCI Base + 0x84	Read/ write	2 2	0x0	SCISYNCRX	Synchronous mode receive clock and data stream register. See Table 3-37 on page 3-35.
SCI Base + 0x88 to 0xEFC	Read/ write	-	-	-	Reserved.
SCI Base + 0xF00 to 0xF10	Read/ write	-	-	-	Reserved for test purposes.
SCI Base + 0xF14 to 0xFCC	Read/ write	-	-	-	Reserved.
SCI Base + 0xFD0 to 0xFDC	Read/ write	-	-	-	Reserved for future expansion.
SCI Base + 0xFE0	Read	8	0x31	SCIPeriphID0	Peripheral identification register bits 7:0. See Table 3-38 on page 3-36.
SCI Base + 0xFE4	Read	8	0x11	SCIPeriphID1	Peripheral identification register bits 15:8. See Table 3-39 on page 3-37.
SCI Base + 0xFE8	Read	8	0x*4 <sup>a</sup>	SCIPeriphID2	Peripheral identification register bits 23:16. See Table 3-40 on page 3-37.

Table 3-1 PrimeCell SCI register summary (continued)

Address	Type	Width	Reset value	Name	Description
SCI Base + 0xFEC	Read	8	0x00	SCIPeriphID3	Peripheral identification register bits 31:24. See Table 3-41 on page 3-37.
SCI Base + 0xFF0	Read	8	0x0D	SCIPCellID0	PrimeCell identification register bits 7:0. See Table 3-42 on page 3-38.
SCI Base + 0xFF4	Read	8	0xF0	SCIPCellID1	PrimeCell identification register bits 15:8. See Table 3-43 on page 3-39.
SCI Base + 0xFF8	Read	8	0x05	SCIPCellID2	PrimeCell identification register bits 23:16. See Table 3-44 on page 3-39.
SCI Base + 0xFFC	Read	8	0xB1	SCIPCellID3	PrimeCell identification register bits 31:24. See Table 3-45 on page 3-39.

a.\* indicates the revision number (see *Peripheral identification register 2, SCIPeriphID2* on page 3-37).



### 3.3 Register descriptions

The following registers are described in this section:

- *Data register, SCIDATA on page 3-8*
- *Control register 0, SCICR0 on page 3-8*
- *Control register 1, SCICR1 on page 3-11*
- *Control register 2, SCICR2 on page 3-12*
- *Clock frequency divider register, SCICLKICC on page 3-12*
- *Value register, SCIVALUE on page 3-13*
- *Baud rate clock register, SCIBAUD on page 3-13*
- *Transmit and receive tide register, SCITIDE on page 3-14*
- *DMA control register, SCIDMACR on page 3-15*
- *Stable (debounce) register, SCISTABLE on page 3-15*
- *Activation event time register, SCIATIME on page 3-16*
- *Deactivation event time register, SCIDTIME on page 3-16*
- *ATR start time register, SCIATRSTIME on page 3-17*
- *ATR duration time register, SCIATRDTIME on page 3-17*
- *Clock stop time register, SCISTOPTIME on page 3-18*
- *Clock start time register, SCISTARTTIME on page 3-18*
- *Transmit and receive retry register, SCIRETRY on page 3-19*
- *Character timeout registers, SCICHTIMELS and SCICHTIMEMS on page 3-20*
- *Block timeout registers, SCIBLKTIMELS and SCIBLKTIMEMS on page 3-21*
- *Character guard time register, SCICHGUARD on page 3-23*
- *Block guard time register, SCIBLKGUARD on page 3-24*
- *Receive read timeout register, SCIRXTIME on page 3-24*
- *FIFO status register, SCIFIFOSTATUS on page 3-25*
- *Transmit FIFO count register, SCITXCOUNT on page 3-26*
- *Receive FIFO count register, SCIRXCOUNT on page 3-26*
- *Interrupt mask set or clear register, SCIMSC on page 3-27*
- *Raw interrupt status register, SCIRIS on page 3-29*
- *Masked interrupt status register, SCIMIS on page 3-30*
- *Interrupt clear register, SCICR on page 3-32*
- *Synchronous card activation control register, SCISYNCACT on page 3-33*
- *Synchronous transmit clock and data register, SCISYNCTX on page 3-34*
- *Synchronous receive clock and data register, SCISYNCRX on page 3-35*
- *Peripheral identification registers on page 3-35*
- *PrimeCell identification registers on page 3-38.*

3.3.1 Data register, SCIDATA

SCIDATA is used for both transmitting and receiving characters. The write data is transmitted through nSCIDATAOUTEN. The read data is received through SCIDATAIN. Table 3-2 shows the bit assignment of the SCIDATA register.

————— **Note** —————

Software should write to this register only after setting the MODE bit to 1. Writes to this register with the MODE bit set to 0 are ignored by the hardware.

**Table 3-2 SCIDATA register bits**

Bits	Name	Type	Function
15:9	-	-	Reserved, do not modify, read as zero.
8	PARITY	Read	Parity error flag. Set to 1 if a parity error was detected when receiving the character corresponding to bits 7 to 0.
7:0	DATA	Read/Write	Eight data bits. These correspond to the character being read or written.

3.3.2 Control register 0, SCICR0

SCICR0 configures the convention for interpreting characters, controls parity convention and enables the handshake mechanism to indicate that a parity error has occurred. The initial character returned by the card in the ATR sequence determines the convention. This register also controls the clock stop mode of operation.

There are two conventions:

- Inverse**

A LOW state on the input/output line is interpreted as logic one and the *Most Significant Bit* (MSB) of the data byte is the first bit after the start bit.
- Direct**

A LOW state on the input/output line is interpreted as logic zero and the *Least Significant Bit* (LSB) of the data byte is the first bit after the start bit.

Separate bits are used to control the logic sense and the bit ordering. This enables nonstandard conventions to be configured.

The register bits (0 to 1) should be set to 00 before reading the initial (TS) character from within the *Answer To Reset* (ATR) stream. The TS character determines the convention that the remainder of the ATR stream has been encoded with.

Inverse convention is configured by writing 11 to SCICR0[1:0] after reading the TS character and before reading any subsequent characters in the ATR.

The register bits 2 and 4 control the parity convention used (odd or even parity) and bits 3 and 5 are used to enable the handshaking mechanism. The handshaking mechanism is initiated by the receiver pulling down the input/output line whenever a parity error has occurred, and is ended by a character retry. Separate controls exist for the transmit and receive paths. The maximum number of attempts made to either transmit or receive a character is specified in the SCIRETRY register.

Because character retry is not applied during ATR reception, this handshake should be programmed initially with the value 0x0.

Bits 2 to 5 should be set to 0xA if the T=0 protocol is requested by the contents of the initial (TS) character of the ATR stream. Table 3-3 shows the bit assignment of the SCICR0 register.

**Table 3-3 SCICR0 register bits**

Bits	Name	Type	Function
15:8	-	-	Reserved, do not modify, read as zero.
7	CLKVAL	Read/write	Defines the inactive state of the card clock when clock stop mode is supported: 0 = Clock held LOW when inactive 1 = Clock held HIGH when inactive.
6	CLKDIS	Read/write	If the card supports clock stop mode, this bit can be used to stop and start the clock: 0 = Clock start initiated 1 = Clock stop initiated.
5	RXNAK	Read/write	Enables character receipt handshaking: If RXNAK = 1, the SCI pulls the input/output line LOW if it detects a parity error. If RXNAK = 0, the SCI does not pull the input/output line LOW if it detects a parity error.
4	RXPARTY	Read/write	Receive parity setting: 0 = Even parity. 1 = Odd parity.

**Table 3-3 SCICR0 register bits (continued)**

Bits	Name	Type	Function
3	TXNAK	Read/write	Enables character transmission handshaking: If TXNAK = 0, the SCI does not check to see if the receiver has pulled the input/output line LOW to indicate a parity error. If TXNAK = 1, the SCI checks, after each character has been transmitted, to see if the receiver has pulled the input/output line LOW to indicate a parity error.
2	TXPARITY	Read/write	Transmit parity setting: 0 = Even parity. 1 = Odd parity.
1	ORDER	Read/write	Specifies ordering of the data bits: 0 = LOW interpreted as logic 0, lsb is the first bit after the start bit (direct convention). 1 = LOW interpreted as logic 1, msb is the first bit after the start bit (inverse convention).
0	SENSE	Read/write	Inverts sense of input/output line for data and parity bits: 0 = Direct convention. 1 = Inverse convention.

If supported by the card, the clock stop mode of operation can be controlled through bits [7:6] of SCICR0.

The stopping of the clock can be initiated by writing a 1 to bit [6], CLKDIS. The clock then stops in its programmed inactive state, defined by the value of bit [7], CLKVAL, after a programmed number of card clock cycles has elapsed. This duration is defined by the value programmed in the SCISTOPTIME register.

The clock can be restarted by writing a 0 to the CLKDIS bit. You should not start further transactions with the card until a number of smart card clock cycles has elapsed. This is defined by the value programmed in the SCISTARTTIME register. The SCICLKACTINTR is asserted when this duration has elapsed.

### 3.3.3 Control register 1, SCICR1

SCICR1 is control register 1. Table 3-4 shows the bit assignment of the SCICR1 register.

**Table 3-4 SCICR1 register bits**

Bits	Name	Type	Function
15:7	-	-	Reserved, do not modify, read as zero.
6	SYNCCARD	Read/write	Asynchronous or synchronous card mode of operation selection: 0 = Asynchronous mode 1 = Synchronous mode.
5	EXDBNCE	Read/write	External debounce. Used to bypass the non programmable portion of the debounce timer, allowing a zero-debounce time by setting the programmable portion of the timer to 0: 0 = Use the whole of the internal debounce timer. 1 = Bypass the non programmable section of the internal debounce timer.
4	BGTEN	Read/write	Block guard timer enable: 0 = Disable. 1 = Enable.
3	CLKZ1	Read/write	<b>SCICLK</b> output configuration: 0 = <b>SCICLK</b> configured as buffer output. 1 = <b>SCICLK</b> configured as pulled down (open drain). If an external pull-up resistor is connected to the Smart Card clock signal, as is the case for synchronous card systems (where both the terminal and the card can pull the clock line LOW), the <b>SCICLK</b> output should be configured as pull-down only. See <i>PrimeCell SCI clock and data driver configurations</i> on page 2-29 for more details.
2	MODE	Read/write	Interface direction of communication control: 0 = Receive. 1 = Transmit.
1	BLKEN	Read/write	Block timeout enable: 0 = Disable. 1 = Enable.
0	ATRDEN	Read/write	ATR duration timeout enable: 0 = Disable. 1 = Enable.

3.3.4 Control register 2, SCICR2

SCICR2 is a write-only register used to initiate activation, deactivation and *warm* reset events. If a write occurs during the deactivation sequence, it is ignored. At any other time, a 1 written to the FINISH bit immediately starts the deactivation sequence. Table 3-5 shows the bit assignment of the SCICR2 register.

————— **Note** —————

Writes to this register should be performed only during the appropriate phase of the card session. Writes performed in a card phase are not internally latched and stored for use in subsequent phases.

The software can write to the STARTUP bit only after a valid card has been found to be present. Writes when the card is not present are ignored by the hardware.

The software can write to the WRESET bit only after the activation sequence is over.

The software can write to the FINISH bit only after a valid card has been found to be present in the system.

**Table 3-5 SCICR2 register bits**

Bits	Name	Type	Function
15:3	-	-	Reserved, do not modify, read as zero.
2	WRESET	Write	Writing a 1 to this bit initiates a warm reset.
1	FINISH	Write	Writing a 1 to this bit deactivates the card.
0	STARTUP	Write	Writing a 1 to this bit starts the activation of the card.

3.3.5 Clock frequency divider register, SCICKICC

SCICKICC is the external Smart Card clock frequency register and contains the divisor used to generate the Smart Card clock frequency.

The Smart Card clock frequency (F) is generated by dividing the reference clock by (SCICKICC + 1) and then dividing again by 2.

$$F = \frac{(\text{Refclock})}{(\text{SCICKICC} + 1) \times 2}$$

If SCICKICC is set to 0,  $F = \text{Refclock}/2$ .

If SCICKICC is set to 1,  $F = \text{Refclock}/4$ .

SCICKICC is an 8-bit register that must be programmed with a value between 0 and 255 before **SCICLK** is enabled. Table 3-6 shows the bit assignment of the SCICKICC register.

**Table 3-6 SCICKICC register bits**

Bits	Name	Type	Function
15:8	-	-	Reserved, do not modify, read as zero.
7:0	CLKICC	Read/write	Defines the Smart Card clock frequency.

### 3.3.6 Value register, SCIVALUE

SCIVALUE is the baud cycles register and defines the number of SCIBAUD cycles per etu. This register is 8-bits wide and can be programmed with any value between 5 and 255. Table 3-7 shows the bit assignment of the SCIVALUE register.

**Table 3-7 SCIVALUE register bits**

Bits	Name	Type	Function
15:8	-	-	Reserved, do not modify, read as zero.
7:0	Value	Read/write	Defines the number of SCIBAUD cycles per etu.

### 3.3.7 Baud rate clock register, SCIBAUD

SCIBAUD defines the divide value used to generate a Value X BAUD rate clock from the reference clock, where Value is set in the SCIVALUE register. SCIBAUD is a 16-bit register that must be programmed with a value between 0x1 and 0xFFFF.

The frequency of the Value X BAUD rate clock is equal to the frequency of the reference clock divided by (SCIBAUD + 1). Table 3-8 shows the bit assignment of the SCIBAUD register.

**Table 3-8 SCIBAUD register bits**

Bits	Name	Type	Function
15:0	BAUD	Read/write	The divide value used to define the baud rate clock frequency.

3.3.8 Transmit and receive tide register, SCITIDE

SCITIDE is the FIFO tide mark register and is used to set the trigger points for the **TXTIDE** and **RXTIDE** interrupts.

The **RXTIDE** field of this register [7:4] contains a trigger point for the receive FIFO. When the number of characters in the receive FIFO equals or exceeds the **RXTIDE** value, an **RXTIDE** interrupt is generated. Setting **RXTIDE** to 1 causes an interrupt as soon as the receive FIFO becomes non-empty. An **RXTIDE** interrupt can only occur when the **MODE** bit of the **SCICR1** register is set to 0 (receive).

The **TXTIDE** field of this register [3:0] contains the trigger point for the **TXTIDE** interrupt. When the number of characters in the transmit FIFO equals or falls below this threshold, a **TXTIDE** interrupt is generated. For both **RXTIDE** and **TXTIDE**, only values between 0 and 8 (inclusive) are valid.

————— **Note** —————

Writes to the **TXFIFO** register are allowed only if the **MODE** bit is set for transmission. The **TXTIDE** interrupt, however, is not qualified with the **MODE** bit. This allows the software to be notified of the current fill level of the transmit FIFO even after the data direction has shifted from transmit to receive. However, the actual act of filling the transmit FIFO should be done based on higher level software considerations and not purely on the fact that the transmit FIFO has room for more data. By not qualifying the **TXTIDE** interrupt with the **MODE** bit, system performance can be increased because the interrupt is being raised as early as possible.

A character is not removed from the transmit FIFO until it has been successfully transmitted. Table 3-9 shows the bit assignment of the **SCITIDE** register.

**Table 3-9 SCITIDE register bits**

Bits	Name	Type	Function
15:8	-	-	Reserved, do not modify, read as zero.
7:4	RXTIDE	Read/write	Trigger point for <b>SCIRXTIDEINTR</b> .
3:0	TXTIDE	Read/write	Trigger point for <b>SCITXTIDEINTR</b> .



### 3.3.9 DMA control register, SCIDMACR

SCIDMACR is a read/write register. All the bits are cleared to 0 on reset. Table 3-10 shows the bit assignment of the SCIDMACR register.

**Table 3-10 SCIDMACR register bits**

Bits	Name	Type	Function
15:2	-	-	Reserved, do not modify, read as zero.
1	Transmit DMA Enable (TXDMAE)	Read/write	If this bit is set to 1, DMA for the transmit FIFO is enabled
0	Receive DMA Enable (RXDMAE)	Read/write	If this bit is set to 1, DMA for the receive FIFO is enabled

### 3.3.10 Stable (debounce) register, SCISTABLE

SCISTABLE determines how long the **SCIDTECT** signal must hold a stable HIGH value, before the interface registers the insertion of a card. This is notified by setting the **CARDININTR** interrupt. Table 3-11 shows the bit assignment of the SCISTABLE register.

**Table 3-11 SCISTABLE register bits**

Bits	Name	Type	Function
15:8	-	-	Reserved, do not modify, read as zero.
7:0	STABLE	Read/write	Stores the debounce time.

The debounce timer is constructed as a 16-bit counter feeding a programmable 8-bit counter, the former being loaded with 0xFFFF and the latter with the value contained in the SCISTABLE register. When enabled, the 16-bit counter decrements the 8-bit counter value until it reaches zero.

The logic is configured to provide a debounce time of (SCISTABLE + 1) multiples of the 16-bit full count.

For a 100MHz reference clock, this gives a programmable debounce time in the range 0.66ms to 168ms in 0.66ms steps.

The 16-bit counter can be bypassed by setting the EXDBNCE bit HIGH. The reference clock then feeds the 8-bit SCISTABLE counter, with the interrupt status and signals being set when this counter reaches zero.

If the debounce period is satisfied, an **SCICARDININTR** interrupt is set.

**Note**

A falling edge on **SCIDTECT** resets **SCICARDININTR**.

**3.3.11 Activation event time register, SCIATIME**

SCIATIME defines the duration, in Smart Card clock cycles, of the time taken for each of the three stages of the card activation process. The programmed value must satisfy the minimum **nSCICARDRST** LOW time of 40000 cycles, and should be sufficient time for the interface power to stabilize.

Table 3-12 shows the bit assignment of the SCIATIME register.

**Table 3-12 SCIATIME register bits**

Bits	Name	Type	Function
15:0	ATIME	Read/write	Stores the time for each of the three stages of the card activation process time.

**3.3.12 Deactivation event time register, SCIDTIME**

SCIDTIME defines the duration, in reference clock cycles, of the time taken for each of the three stages of the card deactivation process. The deactivation sequence can be initiated:

- by software, by writing a 1 to the FINISH bit of the SCICR2 register
- by hardware, on detection of card removal, by the **SCIDTECT** signal going LOW
- on assertion of **SCIDEACREQ** from the PMU.

Table 3-13 shows the bit assignment of the SCIDTIME register.

**Table 3-13 SCIDTIME register bits**

Bits	Name	Type	Function
15:0	DTIME	Read/write	Stores the time for each of the three stages of the card deactivation process time.

### 3.3.13 ATR start time register, **SCIATRSTIME**

SCIATRSTIME defines the receive timeout threshold from the assertion or deassertion of **SCICARDRST** to the start of the first character in the ATR. This is specified in Smart Card clock cycles and its initial value is 40000 Smart Card clock cycles.

The time in seconds, from the assertion or deassertion of **SCICARDRST** to the start of the first character in the ATR, varies with Smart Card clock frequency.

On timeout, this timer sets an interrupt, **SCIATRSTOUTINTR**. Writing a value to this register while the timeout is in progress causes the internal timer to be loaded with the new value, and then count down from this value. Table 3-14 shows the bit assignment of the SCIATRSTIME register.

**Table 3-14 SCIATRSTIME register bits**

Bits	Name	Type	Function
15:0	ATRSTIME	Read/write	ATR reception start timeout threshold from the assertion or deassertion of nSCICARDRST.

### 3.3.14 ATR duration time register, **SCIATRDTIME**

SCIATRDTIME is the maximum duration of the ATR character stream register. It defines the receive timeout threshold for the duration of the ATR sequence, from the start bit of the first ATR character until the end of the ATR block. This is specified in *elementary time units* (etus) and should be programmed to 19200 etus, as per the current specifications. On timeout, this timer sets a **SCIATRDOUTINTR** interrupt.

In normal operation this interrupt is disabled, after the full ATR block has been received, by setting the ATRDEN bit in SCICR1 to 0 using a software routine. Writing a value to this register while the timeout is in progress causes the internal timer to be loaded with the new value, and then count down from this value. Table 3-15 shows the bit assignment of the SCIATRDTIME register.

**Table 3-15 SCIATRDTIME register bits**

Bits	Name	Type	Function
15:0	ATRDTIME	Read/write	ATR reception duration timeout threshold from the start bit of the first ATR character.

3.3.15 Clock stop time register, SCISTOPTIME

The SCISTOPTIME register defines the duration, in card clock cycles, from the initiation of stopping the clock to when the card clock becomes inactive. ISO 7816-3 currently states that the minimum value for this duration must be 1860 card clock cycles.

On timeout, this timer sets the clock stopped interrupt, **SCICLKSTPINTR**. Writing a value to this register while the timeout is in progress causes the internal timer to be loaded with the new value, and then count down from this value. Table 3-16 shows the bit assignment of the SCISTOPTIME register.

Table 3-16 SCISTOPTIME register bits

Bits	Name	Type	Function
15:12	-	-	Reserved, do not modify, read as zero.
11:0	STOPTIME	Read/write	Duration from card clock stop initiation to the card clock becoming inactive, and the clock stopped interrupt <b>SCICLKSTPINTR</b> becoming asserted.

3.3.16 Clock start time register, SCISTARTTIME

The SCISTARTTIME register defines the duration, in card clock cycles, from the initiation of starting the clock to when the clock active interrupt, **SCICLKACTINTR**, is asserted. ISO 7816-3 currently states that the minimum value for this duration must be 700 card clock cycles.

On timeout, this timer sets the clock active interrupt, **SCICLKACTINTR**. Writing a value to this register while the timeout is in progress causes the internal timer to be loaded with the new value, and then count down from this value. Table 3-17 shows the bit assignment of the SCISTARTTIME register.

Table 3-17 SCISTARTTIME register bits

Bits	Name	Type	Function
15:12	-	-	Reserved, do not modify, read as zero.
11:0	STARTTIME	Read/write	Duration from the card clock start initiation to the card clock active interrupt <b>SCICLKACTINTR</b> becoming asserted.

### 3.3.17 Transmit and receive retry register, SCIRETRY

SCIRETRY is used to configure the number of transmit and receive retries that are allowed.

The TXRETRY field of this register contains a 3-bit field [2:0]. You can enable the character transmit handshaking (TXNAK = 1) to specify the maximum number of attempts that can be made to retransmit a character that has been incorrectly received by the card, before aborting the transmission and generating a TXERR interrupt.

For normal T0 operation, the TXRETRY field should be set to 011 for a maximum of three retries. A value of 0x0 (no retries) causes a TXERR interrupt to occur as soon as an error is detected.

For T1 operation, character-based handshaking should be turned off (TXNAK = 0), in which case TXRETRY is not used.

The RXRETRY field of this register contains a 3-bit field [5:3]. This specifies the maximum number of times the interface requests retransmission of a character after detection of a parity error.

For normal T0 operation, the RXRETRY field should be set to 011 for a maximum of three retries. A value of 000 (no retries) writes the received character to the receive FIFO with no request for retransmission in the event of a parity error. This has the same effect as setting RXNAK to 0.

If the character has not been successfully received after RXRETRY attempts, the parity error flag for that character is set. This appears as bit 8 of the SCIDATA register when it is read. Table 3-18 shows the bit assignment of the SCIRETRY register.

**Table 3-18 SCIRETRY register bits**

Bits	Name	Type	Function
15:6	-	-	Reserved, do not modify, read as zero.
5:3	RXRETRY	Read/write	Specifies the maximum number of retries to receive when a parity error has occurred in reception.
2:0	TXRETRY	Read/write	Specifies the maximum number of times that a character is retransmitted following the detection of a parity error by the card.

3.3.18 Character timeout registers, SCICHTIMELS and SCICHTIMEMS

Internally, SCICHTIME defines the maximum time in etus between the leading edge of two consecutive characters for both T0 (character) and T1 (block) protocols. It is also in force during reception of the ATR.

For T0, the time between characters is termed the Work Waiting Time.

For T1, the time between characters is termed the Character Waiting Time (CWT), and the respective characters must reside in the same block.

————— **Note** —————

For T0, the SCICHTIME is effectively offset by 12 etus internally, so the value to be programmed into the SCICHTIME register must be required timeout period in etus minus 12.

For T1, the SCICHTIME is effectively offset by 11 etus internally, so the value to be programmed into the SCICHTIME register must be required timeout period in etus minus 11.

Failure to meet the SCICHTIME results in setting the interrupt SCICHTOUTINTR.

Within the SCI, the SCICHTIME is a 32-bit value. To maintain a 16-bit data width, this parameter requires two 16-bit values to be programmed:

- the upper 16 bits are programmed through the SCICHTIMEMS register
- the lower 16 bits are programmed through the SCICHTIMELS register.

Writing to the SCICHTIMELS register simultaneously updates the internal 32-bit SCICHTIME register with the contents of the SCICHTIMEMS and SCICHTIMELS register values. Because of this, the order of writing these registers is important. The SCICHTIMEMS must be written first, followed by the SCICHTIMELS register. Writing a value to this register while the timeout is in progress causes the internal timer to be loaded with the new value, and then count down from this value.

Table 3-19 shows the bit assignment of the SCICHTIMELS register.

**Table 3-19 SCICHTIMELS register bits**

Bits	Name	Type	Function
15:0	CHTIMELS	Read/write	Defines the least significant 16 bits of the internal 32-bit SCICHTIME register.

Table 3-20 shows the bit assignment of the SCICHTIMEMS register.

**Table 3-20 SCICHTIMEMS register bits**

Bits	Name	Type	Function
15:0	CHTIMEMS	Read/write	Defines the most significant 16 bits of the internal 32-bit SCICHTIME register.

### 3.3.19 Block timeout registers, SCIBLKTIMEELS and SCIBLKTIMEMS

Internally, SCIBLKTIME is the receive timeout between block register. It is used to configure the maximum delay from the leading edge of the last character that gave the right to send to the card, and the first character to be sent by the card. SCIBLKTIME applies to both T0 and T1 protocols:

- For T0, the SCIBLKTIME is effectively offset by 12 etus internally, so the value to be programmed into the SCIBLKTIME register should be required timeout period in etus minus 12.
- For T1, the SCIBLKTIME is effectively offset by 11 etus internally, so the value to be programmed into the SCIBLKTIME register should be required timeout period in etus minus 11.

Failure to meet the SCIBLKTIME value results in the setting of the internal BLKTOUTRIS interrupt bit in the SCIRIS register, and the external **SCIBLKTOUITNTR** interrupt signal. Writing a value to this register while the timeout is in progress causes the internal timer to be loaded with the new value, and then count down from this value.

#### ———— Note —————

The SCIBLKTIME parameter is not applicable to the ATR reception. the reception of the first character of the ATR stream must not exceed the value programmed in the SCIATRSTIME register, which is in terms of Smart Card clock cycles. The SCIATRSTIME value is defined as 40000 Smart Card clock cycles.

Within the SCI, the SCIBLKTIME is a 32-bit value. To maintain a 16-bit data width, this parameter requires two 16-bit values to be programmed:

- the upper 16 bits are programmed through the SCIBLKTIMEMS register
- the lower 16 bits are programmed through the SCIBLKTIMEELS register.

Writing to the SCIBLKTIMELS register simultaneously updates the internal 32-bit SCIBLKTIME register with the contents of the SCIBLKTIMEMS and SCIBLKTIMELS register values. Because of this, the order of writing these registers is important. The SCIBLKTIMEMS must be written first, followed by the SCIBLKTIMELS register.

Table 3-21 shows the bit assignment of the SCIBLKTIMELS register.

Table 3-21 SCIBLKTIMELS register bits

Bits	Name	Type	Function
15:0	BLKTIMELS	Read/write	Defines the least significant 16 bits of the internal 32-bit SCIBLKTIME register.

Table 3-22 shows the bit assignment of the SCICHTIMEMS register.

Table 3-22 SCIBLKTIMEMS register bits

Bits	Name	Type	Function
15:0	BLKTIMEMS	Read/write	Defines the most significant 16 bits of the internal 32-bit SCIBLKTIME register.



### 3.3.20 Character guard time register, SCICHGUARD

SCICHGUARD defines the extra guard time that is added to the minimum duration between leading edges of the start bits of two consecutive characters, for subsequent communication from the interface to the Smart Card.

The SCICHGUARD value is derived from the TC1 value that is extracted from the ATR character stream. Writing a value to this register while the timeout is in progress causes the internal timer to be loaded with the new value, and then count down from this value. Table 3-23 shows the bit assignment of the SCICHGUARD register.

**Table 3-23 SCICHGUARD register bits**

Bits	Name	Type	Function
15:8	-	-	Reserved, do not modify, read as zero.
7:0	SCICHGUARD	Read/write	Defines the minimum duration between the leading edges of the start bits of two consecutive characters for subsequent communication from the interface to the Smart Card.

The TC1 value can be between 0 and 255. The software must read the TC1 value and program the SCICHGUARD register as described in Table 3-24 to provide the resultant guard time in etus.

**Table 3-24 TC1 value settings**

ATR TC1 value	SCICHGUARD value		Resultant guard time (etus)	
	T0	T1	T0	T1
0 =< TC1 < 255	TC1	TC1 + 1	TC1 + 12	TC1 + 11
255	0	0	12	11

A TC1 value of 255 indicates that the minimum delay between the start leading edges of two consecutive characters is 12 etus if T=0, or 11 etus if T=1 is used.

#### ———— Note —————

The TXNAK bit value in the SCICR0 register is used by the interface hardware to determine whether T0 or T1 protocol is in operation.

3.3.21 Block guard time register, SCIBLKGUARD

SCIBLKGUARD defines the minimum time in etus between the leading edges of two consecutive characters sent in opposite directions:

- For T0, the SCIBLKGUARD minimum time is 16 work etus.
- For T1, the SCIBLKGUARD minimum time is 22 work etus.

———— Note —————

For T0, SCIBLKGUARD is effectively offset by 12 etus internally, so the value to be programmed into the SCIBLKGUARD register should be required time period in etus minus 12.

For T1, the SCIBLKGUARD is effectively offset by 11 etus internally, so the value to be programmed into the SCIBLKGUARD register should be required time period in etus minus 11.

Writing a value to this register while the timeout is in progress causes the internal timer to be loaded with the new value, and then count down from this value. Table 3-25 shows the bit assignment of the SCIBLKGUARD register.

Table 3-25 SCIBLKGUARD register bits

Bits	Name	Type	Function
15:8	-	-	Reserved, do not modify, read as zero.
7:0	BLOCKGUARD	Read/write	Defines the minimum time in etus between the leading edges of two consecutive characters sent in opposite directions.

3.3.22 Receive read timeout register, SCIRXTIME

SCIRXTIME stores the receive read timeout value. An **SCIRTOUT** interrupt is triggered if the receive FIFO contains at least one character, and no characters have been read for a time corresponding to SCIRXTIME Smart Card clock cycles.

Writing a value of zero is illegal. Writing a value to this register while the timeout is in progress causes the internal timer to be loaded with the new value, and then count down from this value. Table 3-26 shows the bit assignment of the SCIRXTIME register.

Table 3-26 SCIRXTIME register bits

Bits	Name	Type	Function
15:0	RXTIME	Read/write	Receive read timeout value.

Table 3-27 lists the range and resolution of the timeout value for various Smart Card clock frequencies.

**Table 3-27 RXTIME ranges and resolutions**

Frequency	Range (approx)	Resolution
10 MHz	0-6 ms	0.1 $\mu$ s
5 MHz	0-13 ms	0.2 $\mu$ s
1 MHz	0-65 ms	1 $\mu$ s
500 kHz	0-131 ms	2 $\mu$ s

### 3.3.23 FIFO status register, SCIFIFOSTATUS

SCIFIFOSTATUS is the flag register. It is a read-only register which contains four bits that indicate the status of the transmit and receive FIFOs. Table 3-28 shows the bit assignment of the SCIFIFOSTATUS register.

**Table 3-28 SCIFIFOSTATUS register bits**

Bits	Name	Type	Function
15:4	-	-	Reserved, do not modify, read as zero.
3	RXFE	Read	RXFIFO empty status.
2	RXFF	Read	RXFIFO full status.
1	TXFE	Read	TXFIFO empty status.
0	TXFF	Read	TXFIFO full status.

3.3.24 Transmit FIFO count register, SCITXCOUNT

SCITXCOUNT returns the number of characters (including any character currently being transmitted) in the transmit FIFO when read, and flushes the transmit FIFO when written (with any value).

If there is an unsuccessful transmission when using the T0 protocol, the interface generates a **TXERR** interrupt and stops transmitting. Before any further characters can be transmitted or received, the error condition must be cleared by flushing the transmit FIFO. Table 3-29 shows the bit assignment of the SCITXCOUNT register.

Table 3-29 SCITXCOUNT register bits

Bits	Name	Type	Function
15:4	-	-	Reserved, do not modify, read as zero.
3:0	TXCOUNT	Read/write	A read transmits the FIFO count. A write flushes the transmit FIFO.

3.3.25 Receive FIFO count register, SCIRXCOUNT

SCIRXCOUNT returns the number of characters in the receive FIFO when read, and flushes the receive FIFO when written (with any value). Table 3-30 shows the bit assignment of the SCIRXCOUNT register.

Table 3-30 SCIRXCOUNT register bits

Bits	Name	Type	Function
15:4	-	-	Reserved, do not modify, read as zero.
3:0	RXCOUNT	Read/write	A read receives the FIFO count. A write flushes the receive FIFO.

### 3.3.26 Interrupt mask set or clear register, SCIIMSC

SCIIMSC is a read/write register. On a read, this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask. All the bits are cleared to 0 when reset. Table 3-31 shows the bit assignment of the SCIIMSC register.

**Table 3-31 SCIIMSC register bits**

Bits	Name	Type	Function
15	-	-	Reserved, do not modify, read as zero.
14	TXTIDEIM	Read/write	Transmit tide interrupt mask: 0 = Transmit FIFO watermark level is less than the programmed threshold level interrupt is masked. 1 = Transmit FIFO watermark level is less than the programmed threshold level interrupt is not masked.
13	RXTIDEIM	Read/write	Receive tide interrupt mask: 0 = Receive FIFO watermark level exceeds the programmed threshold level interrupt is masked. 1 = Receive FIFO watermark level exceeds the programmed threshold level interrupt is not masked.
12	CLKACTIM	Read/write	Card clock active interrupt mask: 0 = Card clock active interrupt is masked. 1 = Card clock active interrupt is not masked.
11	CLKSTPIM	Read/write	Card clock stopped interrupt mask: 0 = Card clock stopped interrupt is masked. 1 = Card clock stopped interrupt is not masked.
10	RORIM	Read/write	Receive overrun interrupt mask: 0 = Receive overrun interrupt is masked. 1 = Receive overrun interrupt is not masked.
9	RTOUTIM	Read/write	Read time out interrupt mask: 0 = Read character from receive FIFO time out interrupt is masked. 1 = Read character from receive FIFO time out interrupt is not masked.

**Table 3-31 SCIIMSC register bits (continued)**

<b>Bits</b>	<b>Name</b>	<b>Type</b>	<b>Function</b>
8	CHTOUTIM	Read/write	Character time out interrupt mask: 0 = Character to character time out interrupt is masked. 1 = Character to character time out interrupt is not masked.
7	BLKTOUTIM	Read/write	Block time out interrupt mask: 0 = Data stream character to start of block time out interrupt is masked. 1 = Data stream character to start of block time out interrupt is not masked.
6	ATRDTOUITM	Read/write	Answer To Reset duration time out mask: 0 = Answer To Reset data stream duration time out interrupt is masked. 1 = Answer To time data stream duration time out interrupt is not masked.
5	ATRSTOUITM	Read/write	Answer To Reset start time out interrupt mask: 0 = Answer To Reset receipt of start bit time out interrupt is masked. 1 = Answer To Reset receipt of start bit is not masked.
4	TXERRIM	Read/write	Transmit Error interrupt mask: 0 = Transmit character bits parity error interrupt is masked. 1 = Transmit character bits parity error interrupt is not masked.
3	CARDDNIM	Read/write	Card down interrupt mask: 0 = Card powered down notification interrupt is masked. 1 = Card powered down notification interrupt is not masked.
2	CARDUPIM	Read/write	Card up interrupt mask: 0 = Card powered up notification interrupt is masked. 1 = Card powered up notification interrupt is not masked.

**Table 3-31 SCIMSC register bits (continued)**

Bits	Name	Type	Function
1	CARDOUTIM	Read/write	Card out interrupt mask: 0 = Card removed from reader notification interrupt is masked. 1 = Card removed from reader notification interrupt is not masked.
0	CARDINIM	Read/write	Card inserted interrupt mask: 0 = Card inserted into reader notification interrupt is masked. 1 = Card inserted into reader notification interrupt is not masked.

### 3.3.27 Raw interrupt status register, SCIRIS

SCIRIS is a read-only register. On a read, this register gives the raw status value of the relevant interrupt prior to masking. A write has no effect. Table 3-32 shows the bit assignment of the SCIRIS register.

**Table 3-32 SCIRIS register bits**

Bits	Name	Type	Function
15	-	-	Reserved, do not modify, read as zero.
14	TXTIDERIS	Read	Gives the raw interrupt state (prior to masking) of the SCITXTIDEINTR interrupt.
13	RXTIDERIS	Read	Gives the raw interrupt state (prior to masking) of the SCIRXTIDEINTR interrupt.
12	CLKACTRIS	Read	Gives the raw interrupt state (prior to masking) of the SCICLKACTINTR interrupt.
11	CLKSTPRIS	Read	Gives the raw interrupt state (prior to masking) of the SCICLKSTPINTR interrupt.
10	RORRIS	Read	Gives the raw interrupt state (prior to masking) of the SCIRORINTR interrupt.
9	RTOUTRIS	Read	Gives the raw interrupt state (prior to masking) of the SCIRTOUITINTR interrupt.
8	CHTOUTRIS	Read	Gives the raw interrupt state (prior to masking) of the SCICHTOUTINTR interrupt.

**Table 3-32 SCIRIS register bits (continued)**

Bits	Name	Type	Function
7	BLKTOUTRIS	Read	Gives the raw interrupt state (prior to masking) of the SCIBLKTOUTINTR interrupt.
6	ATRDOUTRIS	Read	Gives the raw interrupt state (prior to masking) of the SCIATRDOUTINTR interrupt.
5	ATRSTOUTRIS	Read	Gives the raw interrupt state (prior to masking) of the SCIATRSTOUTINTR interrupt.
4	TXERRRIS	Read	Gives the raw interrupt state (prior to masking) of the SCITXERRINTR interrupt.
3	CARDDNRIS	Read	Gives the raw interrupt state (prior to masking) of the SCICARDDNINTR interrupt.
2	CARDUPRIS	Read	Gives the raw interrupt state (prior to masking) of the SCICARDUPINTR interrupt.
1	CARDOUTRIS	Read	Gives the raw interrupt state (prior to masking) of the SCICARDOUTINTR interrupt.
0	CARDINRIS	Read	Gives the raw interrupt state (prior to masking) of the SCICARDININTR interrupt.

### 3.3.28 Masked interrupt status register, SCIMIS

SCIMIS is a read only register. On a read, this register gives the current status value of the relevant interrupt after masking. A write has no effect. Table 3-33 shows the bit assignment of the SCIMIS register...

**Table 3-33 SCIMIS register bits**

Bits	Name	Type	Function
15	-	-	Reserved, do not modify, read as zero.
14	TXTIDEMIS	Read	Gives the interrupt state (after masking) of the SCITXTIDEINTR interrupt.
13	RXTIDEMIS	Read	Gives the interrupt state (after masking) of the SCIRXTIDEINTR interrupt.
12	CLKACTMIS	Read	Gives the interrupt state (after masking) of the SCICLKACTINTR interrupt.



**Table 3-33 SCIMIS register bits (continued)**

<b>Bits</b>	<b>Name</b>	<b>Type</b>	<b>Function</b>
11	CLKSTPMIS	Read	Gives the interrupt state (after masking) of the SCICLKSTPINTR interrupt.
10	RORMIS	Read	Gives the interrupt state (after masking) of the SCIRORINTR interrupt.
9	RTOUTMIS	Read	Gives the interrupt state (after masking) of the SCIRTOUTINTR interrupt.
8	CHTOUTMIS	Read	Gives the interrupt state (after masking) of the SCICHTOUTINTR interrupt.
7	BLKTOUTMIS	Read	Gives the interrupt state (after masking) of the SCIBLKOUTINTR interrupt.
6	ATRDOUTMIS	Read	Gives the interrupt state (after masking) of the SCIATRDOUTINTR interrupt.
5	ATRSTOUTMIS	Read	Gives the interrupt state (after masking) of the SCIATRSTOUTINTR interrupt.
4	TXERRMIS	Read	Gives the interrupt state (after masking) of the SCITXERRINTR interrupt.
3	CARDDNMIS	Read	Gives the interrupt state (after masking) of the SCICARDDNINTR interrupt.
2	CARDUPMIS	Read	Gives the interrupt state (after masking) of the SCICARDUPINTR interrupt.
1	CARDOUTMIS	Read	Gives the interrupt state (after masking) of the SCICARDOUTINTR interrupt.
0	CARDINMIS	Read	Gives the interrupt state (after masking) of the SCICARDININTR interrupt.

### 3.3.29 Interrupt clear register, **SCIICR**

SCIICR is a write-only register. On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect. Table 3-34 shows the bit assignment of the SCIICR register.

**Table 3-34 SCIICR register bits**

Bits	Name	Type	Function
15:13	-	-	Reserved, do not modify, read as zero.
12	CLKACTIC	Write	Clears the SCICLKACTINTR interrupt.
11	CLKSTPIC	Write	Clears the SCICLKSTPINTR interrupt.
10	RORIC	Write	Clears the SCIRORINTR interrupt.
9	RTOUTIC	Write	Clears the SCIRTOUTINTR interrupt.
8	CHTOUTIC	Write	Clears the SCICHTOUTINTR interrupt.
7	BLKTOUTIC	Write	Clears the SCIBLKTOUTINTR interrupt.
6	ATRDOUTIC	Write	Clears the SCIATRDOUTINTR interrupt.
5	ATRSTOUTIC	Write	Clears the SCIATRSTOUTINTR interrupt.
4	TXERRIC	Write	Clears the SCITXERRINTR interrupt.
3	CARDDNIC	Write	Clears the SCICARDDNINTR interrupt.
2	CARDUPIC	Write	Clears the SCICARDUPINTR interrupt.
1	CARDOUTIC	Write	Clears the SCICARDOUTINTR interrupt.
0	CARDINIC	Write	Clears the SCICARDININTR interrupt.

### 3.3.30 Synchronous card activation control register, SCISYNCACT

SCISYNCACT provides direct access to Smart Card signals. It is only required if a non-EMV-compliant configuration is used. The status register is updated automatically during activation, deactivation and warm reset events. Table 3-35 shows the bit assignment of the SCISYNCACT register.

#### Note

The SCI does not have a separate bit to distinguish between EMV and non-EMV compliant cards. It is the responsibility of the software to follow certain sequences in either case to ensure correct and consistent behavior. The software should not write to the STARTUP bit when it is dealing with a non-EMV compliant card. It should write a 1 to the STARTUP bit in the case of EMV compliant cards.

If the STARTUP bit has been written to (indicating an EMV compliant card) the software should not write to the Smart Card Status register.

If the STARTUP bit has not been written to (indicating a non-EMV compliant card) the activation sequence should be performed by explicit writes to the relevant bits in this register.

Deactivation is done **ONLY** through internal hardware in both EMV and non-EMV configurations.

Software should not write to this register during card validation using the hardware debounce mechanism.

Writes to this register are ignored during deactivation.

**Table 3-35 SCISYNCACT register bits**

Bits	Name	Type	Function
15:11	-	-	Reserved, do not modify, read as zero.
10	CARDPRESENT	Read	1 if Smart Card is present.
9	nSCIDATAEN	Read	Tristate control for external off-chip buffer for data.
8	nSCIDATAOUTEN	Read	Tristate output buffer control for data.
7	SCICLKOUT	Read	Smart Card clock output.
6	nSCICLKEN	Read	Tristate control for external off-chip buffer for clock.

Table 3-35 SCISYNCACT register bits (continued)

Bits	Name	Type	Function
5	nSCICLKOUTEN	Read	Tristate output buffer control for clock.
4	FCB	Read/write	Function code bit. Used with CRESET to indicate the type of command to be executed.
3	DATAEN	Read/write	Enable Smart Card data. 0 forces the Smart Card data LOW.
2	CLKEN	Read/write	Enable Smart Card clock. 0 forces the Smart Card clock LOW.
1	CRESET	Read/write	Controls Smart Card reset signal.
0	POWER	Read/write	Controls Smart Card VCC.

### 3.3.31 Synchronous transmit clock and data register, SCISYNCTX

SCISYNCTX contains the source of alternate values to be used to drive the Smart Card input/output and clock signal. Table 3-36 shows the bit assignment of the SCISYNCTX register.

Table 3-36 SCISYNCTX register bits

Bits	Name	Type	Function
15:6	-	-	Reserved, do not modify, read as zero.
5	WFCB	Read/write	If the control bit SYNCCARD in SCICR1 is set, the <b>SCIFCB</b> signal is driven with WFCB.
4	WRST	Read/write	If the control bit SYNCCARD in SCICR1 is set, the <b>nSCICARDRST</b> signal is driven with WRST.
3	WCLKEN	Read/write	If the control bit SYNCCARD in SCICR1 is set, and <b>WCLKEN</b> = 0, the <b>SCICLKEN</b> line is forced LOW.
2	WDATAEN	Read/write	If the control bit SYNCCARD in SCICR1 is set, and <b>WDATAEN</b> = 0, the <b>SCIDATAEN</b> line is forced LOW.
1	WCLK	Read/write	If the control bit SYNCCARD in SCICR1 is set,, the Smart Card clock is driven with <b>WCLK</b> .
0	WDATA	Read/write	If the control bit SYNCCARD in SCICR1 is set, and <b>WDATA</b> = 0, the input/output line is forced LOW.

3.3.32 Synchronous receive clock and data register, SCISYNCRX

SCISYNCRX provides read access to the raw status of the Smart Card input/output and clock signals. Table 3-37 shows the bit assignment of the SCISYNCRX register.

Table 3-37 SCISYNCRX register bits

Bits	Name	Type	Function
15:2	-	-	Reserved, do not modify, read as zero.
1	RCLK	Read	Raw value of the clock.
0	RDATA	Read	Raw value of the input/output line.

————— **Note** —————

In non-EMV mode of operation, the incoming bit stream from the card should be read from the SCISYNCRX register. The received data is not available in the receive FIFO.

3.3.33 Peripheral identification registers

The SCIPeriphID0-3 registers are four 8-bit registers, that span address locations 0xFE0 to 0xFEC. The registers can conceptually be treated as a single 32-bit register. The read-only registers provide the following options for the peripheral:

**PartNumber[11:0]** This is used to identify the peripheral. The three digit product code 0x131 is used.

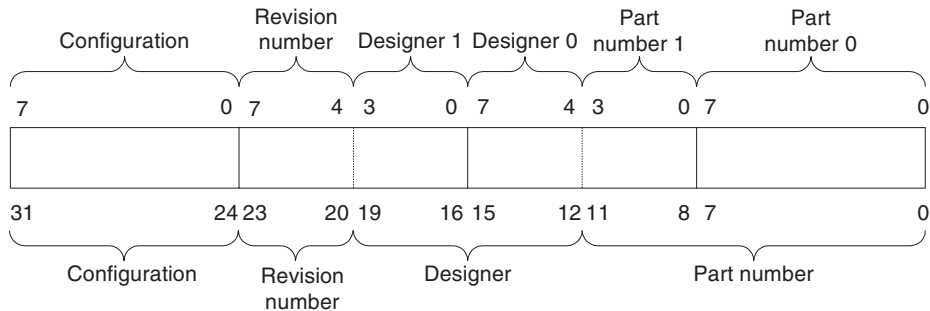
**Designer ID[19:12]** This is the identification of the designer. ARM Limited is 0x41 (ASCII A).

**Revision[23:20]** This is the revision number of the peripheral. The revision number starts from 0.

**Configuration[31:24]**  
This is the configuration option of the peripheral. The configuration value is 0.

Figure 3-1 shows the bit assignment for the SCIPeriphID0-3 registers.

Actual register bit assignment



Conceptual register bit assignment

Figure 3-1 Peripheral identification register bit assignment

———— **Note** ————

When you design a systems memory map you must remember that the register has a 4KB-memory footprint. All memory accesses to the peripheral identification registers must be 32-bit, using the LDR and STR instructions.

The four 8-bit peripheral identification registers are described in the following subsections:

- *Peripheral identification register 0, SCIPeriphID0* on page 3-36
- *Peripheral identification register 1, SCIPeriphID1* on page 3-37
- *Peripheral identification register 2, SCIPeriphID2* on page 3-37
- *Peripheral identification register 3, SCIPeriphID3* on page 3-37.

**Peripheral identification register 0, SCIPeriphID0**

SCIPeriphID0 is hard-coded and the fields within the register determine the reset value. Table 3-38 shows the bit assignment of the SCIPeriphID0 register.

Table 3-38 SCIPeriphID0 register bits

Bits	Name	Description
15:8	-	Reserved, read undefined, must read as zeros
7:0	PartNumber0	These bits read back as 0x31

### Peripheral identification register 1, SCIPeriphID1

SCIPeriphID1 is hard-coded and the fields within the register determine the reset value. Table 3-39 shows the bit assignment of the SCIPeriphID1 register.

Table 3-39 SCIPeriphID1 register bits

Bits	Name	Description
15:8	-	Reserved, read undefined, must read as zeros
7:4	Designer0	These bits read back as 0x1
3:0	PartNumber1	These bits read back as 0x1.

### Peripheral identification register 2, SCIPeriphID2

SCIPeriphID2 is hard-coded and the fields within the register determine the reset value. Table 3-40 shows the bit assignment of the SCIPeriphID2 register.

Table 3-40 SCIPeriphID2 register bits

Bits	Name	Description
15:8	-	Reserved, read undefined, must read as zeros
7:4	Revision	These bits read back as the revision number, which can be between 0 and 15.
3:0	Designer1	These bits read back as 0x4

### Peripheral identification register 3, SCIPeriphID3

SCIPeriphID3 is hard-coded and the fields within the register determine the reset value. Table 3-41 shows the bit assignment of the SCIPeriphID3 register.

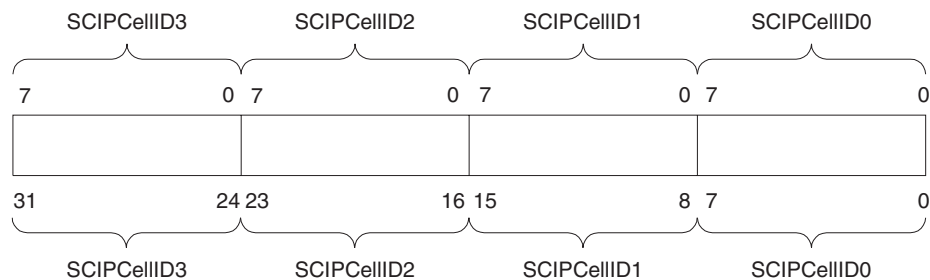
Table 3-41 SCIPeriphID3 register bits

Bits	Name	Description
15:8	-	Reserved, read undefined, must read as zeros
7:0	Configuration	These bits read back as 0x00

3.3.34 PrimeCell identification registers

The SCIPCellID0-3 registers are four 8-bit wide registers, that span address locations 0xFF0 to 0xFFC. The registers can conceptually be treated as a 32-bit register. The register is used as a standard cross-peripheral identification system. The SCIPCellID register is set to 0xB105F00D. Figure 3-2 shows the bit assignment for the SCIPCellID0-3 registers.

Actual register bit assignment



Conceptual register bit assignment

Figure 3-2 PrimeCell identification register bit assignment

The four, 8-bit PrimeCell identification registers are described in the following subsections:

- PrimeCell identification register 0, *SCIPCellID0* on page 3-38
- PrimeCell identification register 1, *SCIPCellID1* on page 3-39
- PrimeCell identification register 2, *SCIPCellID2* on page 3-39
- PrimeCell identification register 3, *SCIPCellID3* on page 3-39.

PrimeCell identification register 0, *SCIPCellID0*

*SCIPCellID0* is hard-coded and the fields within the register determine the reset value. Table 3-42 shows the bit assignment of the *SCIPCellID0* register.

Table 3-42 *SCIPCellID0* register bits

Bits	Name	Description
15:8	-	Reserved, read undefined, must read as zeros
7:0	<i>SCIPCellID0</i>	These bits read back as 0x0D



### PrimeCell identification register 1, SCIPCellID1

SCIPCellID1 is hard-coded and the fields within the register determine the reset value. Table 3-43 shows the bit assignment of the SCIPCellID1 register.

**Table 3-43 SCIPCellID1 register bits**

Bits	Name	Description
15:8	-	Reserved, read undefined, must read as zeros
7:0	SCIPCellID1	These bits read back as 0xF0

### PrimeCell identification register 2, SCIPCellID2

SCIPCellID2 is hard-coded and the fields within the register determine the reset value. Table 3-44 shows the bit assignment of the SCIPCellID2 register.

**Table 3-44 SCIPCellID2 register bits**

Bits	Name	Description
15:8	-	Reserved, read undefined, must read as zeros
7:0	SCIPCellID2	These bits read back as 0x05

### PrimeCell identification register 3, SCIPCellID3

SCIPCellID3 is hard-coded and the fields within the register determine the reset value. Table 3-45 shows the bit assignment of the SCIPCellID3 register.

**Table 3-45 SCIPCellID3 register bits**

Bits	Name	Description
15:8	-	Reserved, read undefined, must read as zeros
7:0	SCIPCellID3	These bits read back as 0xB1

## 3.4 Interrupts

There are fifteen interrupts generated within the PrimeCell SCI. The interrupt mask set or clear register, `SCIIMSC`, provides a way of masking each of these individual interrupts. Setting the appropriate mask bit HIGH enables the interrupt, all of which are active HIGH.

Individual interrupts:

- `SCICARDININTR`: SCI Card In interrupt
- `SCICARDOUTINTR`: SCI Card Out interrupt
- `SCICARDUPINTR`: SCI Card powered Up interrupt
- `SCICARDDNINTR`: SCI Card powered Down interrupt
- `SCITXERRINTR`: SCI Transmit Error interrupt
- `SCIATRSTOUTINTR`: SCI Answer-To-Reset Start Time Out interrupt
- `SCIATRDOUTINTR`: SCI Answer-To-Reset data stream Duration Time Out
- `SCIBLKOUTINTR`: SCI Block Time Out interrupt
- `SCICHTOUTINTR`: SCI Character Time Out interrupt
- `SCIRTOINTR`: SCI Read Time Out interrupt
- `SCIRORINTR`: SCI Overrun interrupt
- `SCICLKSTPINTR`: SCI Clock Stopped interrupt
- `SCICLKACTINTR`: SCI Clock Active interrupt
- `SCIRXTIDEINTR`: SCI Receive FIFO Tide level interrupt
- `SCITXTIDEINTR`: SCI Transmit FIFO Tide level interrupt.

There is also a single interrupt that is the combinatorial OR of the fifteen individual interrupts:

- `SCIINTR`: Combinatorial OR of the individual interrupt

The status of the individual interrupt sources can be read from either the `SCIRIS` register, for raw status, or from the `SCIMIS` register, for the masked status.

The following descriptions assume that the respective mask bit is set HIGH, enabling the interrupt:

### **SCICARDININTR**

The SCI Card In interrupt is asserted when a card that has been inserted into the reader interface has been stable for the programmed debounce period, defined by the `SCISTABLE` register value.

The `SCICARDININTR` interrupt can be cleared by writing a 1 to the card in interrupt clear bit, `CARDINIC`, within the `SCIICR` interrupt clear register.

**SCICARDOUTINTR**

The SCI Card Out interrupt is asserted when the interface recognizes that the input signal SCIDETECT has transitioned from HIGH to LOW, signifying card removal.

The SCICARDOUTINTR interrupt can be cleared by writing a 1 to the card out interrupt clear bit, CARDOUTIC, within the SCIICR interrupt clear register.

**SCICARDUPINTR**

The SCI Card Up interrupt is asserted on completion of the card activation sequence. The activation sequence is a timed sequence of applying power, clock, reset and data signals to the card. The subsequent Answer-To-Reset data stream from the card does not rely on this interrupt being asserted; the interrupt is only used to inform the system that the activation sequence has completed.

The SCICARDUPINTR interrupt can be cleared by writing a 1 to the card up interrupt clear bit, CARDUPIC, within the SCIICR interrupt clear register.

**SCICARDDNINTR**

The SCI Card Down interrupt is asserted on completion of the card deactivation sequence. The deactivation sequence is a timed sequence of removing data, reset, clocks and power from the card. The card can remain in the interface until another session is initiated.

The SCICARDDNINTR interrupt can be cleared by writing a 1 to the card down interrupt clear bit, CARDDNIC, within the SCIICR interrupt clear register.

**SCITXERRINTR**

The SCI Transmit Error interrupt is asserted when the data word transmitted by the interface has been rejected by the card after a programmable amount of retry attempts, defined within the SCIRETRY register. If an transmit error interrupt occurs, the interface initiates the deactivation sequence.

The SCITXERRINTR interrupt can be cleared by writing a 1 to the transmit error interrupt clear bit, TXERRIC, within the SCIICR interrupt clear register.

**SCIATRSTOUTINTR**

The SCI Answer-To-Reset Start Time Out interrupt is asserted when the time, defined by the SCIATRSTIME register value, to the receipt of the leading edge of the first character within the ATR sequence is exceeded.

The SCIATRSTOUTINTR interrupt can be cleared by writing a 1 to the start of ATR interrupt clear bit, ATRSTOUTIC, within the SCIICR interrupt clear register.

### **SCIATRDOUTINTR**

The SCI Answer-To-Reset Duration Time Out interrupt is asserted when the time, defined by the SCIATRDTIME register value, taken to receive the whole of the ATR character data stream exceeds the programmed maximum value.

The SCIATRDOUTINTR interrupt can be cleared by writing a 1 to the ATR duration interrupt clear bit, ATRDOUTIC, within the SCIICR interrupt clear register.

### **SCIBLKOUTINTR**

The SCI Block Time Out interrupt is asserted when the time, defined by the SCIBLKTIME register value, between the leading edges of the start bits of the last character that gave the right to send to the card, and that of the first character of the block received by the interface exceeds the programmed maximum value.

The SCIBLKOUTINTR interrupt can be cleared by writing a 1 to the block time out interrupt clear bit, BLKOUTIC, within the SCIICR interrupt clear register.

### **SCICHTOUTINTR**

The SCI Character Time Out interrupt is asserted when the time, defined by the SCICHTIME register value, between the leading edges of two consecutive characters in the same block exceeds the programmed maximum value.

The SCICHTOUTINTR interrupt can be cleared by writing a 1 to the block time out interrupt clear bit, CHTOUTIC, within the SCIICR interrupt clear register.

### **SCIRTOUTINTR**

The SCI Read Time Out interrupt is asserted when the receive FIFO contains at least one character and no characters have been read within the time defined by the SCIRXTIME register value.

The SCIRTOUTINTR interrupt can be cleared by writing a 1 to the read time out interrupt clear bit, RTOUTIC, within the SCIICR interrupt clear register.

### **SCIROPINTR**

The SCI Receive OverRun interrupt is asserted when the receive FIFO is full and the next character is completely received into the input shift register.

The SCIROPINTR interrupt can be cleared by writing a 1 to the overrun interrupt clear bit, RORIC, within the SCIICR interrupt clear register.

**SCICLKSTPINTR**

The SCICLKSTPINTR is asserted when a card clock stop has been requested, performed through writing a 1 to the CLKDIS bit within the SCICR0 register, and the number of card clock cycles defined by the value contained within the SCICLKSTOPTIME register has elapsed. This interrupt signifies that the clock to the card is inactive.

**SCICLKACTINTR**

The SCICLKACTINTR is asserted when a clock start has been requested, performed through writing a 0 to the CLKDIS bit within the SCICR0 register. The clock is enabled but the interrupt is not asserted until the number of card clock cycles defined by the value contained within the SCICLKSTARTTIME register has elapsed. The interrupt assertion signifies that information exchange can continue between the interface and the card.

**SCIRXTIDEINTR**

The SCI Receive Tide interrupt is asserted when the number of filled locations within the receive FIFO exceeds that of the programmed tide level mark, defined within the SCITIDE register.

This is a dynamic interrupt and is self-clearing by the action of reading characters from the receive FIFO until the level falls below that of the programmed tide level.

**SCITXTIDEINTR**

The SCI Transmit Tide interrupt is asserted when the number of filled locations within the transmit FIFO is below that of the programmed tide transmit level mark, defined within the SCITIDE register.

This is a dynamic interrupt and is self-clearing by the action of writing characters to the transmit FIFO until the level is above that of the programmed transmit tide level.



# Chapter 4

## Programmer's Model for Test

This chapter describes the additional logic for integration testing. It contains the following sections:

- *PrimeCell SCI test harness overview* on page 4-2
- *Scan testing* on page 4-3
- *Test registers* on page 4-4
- *Integration testing of block inputs* on page 4-10
- *Integration testing of block outputs* on page 4-12
- *Integration test summary* on page 4-17.

## 4.1 PrimeCell SCI test harness overview

The additional logic for integration vectors allows:

- capture of input signals to the block
- stimulation of the output signals.

The integration vectors provide a way of verifying that the PrimeCell SCI is correctly wired into a system. This is done by separately testing three groups of signals:

**AMBA signals**      These are tested by checking the connections of all the address and data bits.

### **Primary input/output signals**

These are tested using a simple trickbox that can demonstrate the correct connection of the input/output signals to external pads.

### **Intra-chip signals (such as interrupt sources)**

The tests for these signals are system-specific, and enable you to write the necessary tests. Additional logic is implemented allowing you to read and write to each intra-chip input/output signal.

These test features are controlled by test registers. This allows you to test the PrimeCell SCI in isolation from the rest of the system using only transfers from the AMBA APB.

Off-chip test vectors are supplied using a 32-bit parallel *External Bus Interface* (EBI) and converted to internal AMBA bus transfers. The application of test vectors is controlled through the *Test Interface Controller* (TIC) AMBA bus master module.



## 4.2 Scan testing

The PrimeCell SCI has been designed to simplify:

- insertion of scan test cells
- use of *Automatic Test Pattern Generation (ATPG)*.

This is the recommended method of manufacturing test.

4.3 Test registers

The PrimeCell SCI test registers are memory-mapped as shown in Table 4-1.

Table 4-1 Test registers memory map

Address	Type	Width	Reset value	Name	Description
SCI Base + 0xF00	Read/write	2 2	0x0	SCITCR	Test control register. See Table 4-2 on page 4-4.
SCI Base + 0xF04	Read/write	6 6	0x00	SCIITIP	Integration test input register. See Table 4-3 on page 4-5.
SCI Base + 0xF08	Read/write	16 16	0x0000	SCIITOP1	Integration test output register 1. See Table 4-4 on page 4-6.
SCI Base + 0xF0C	Read/write	13 13	0x0000	SCIITOP2	Integration test output register 2. See Table 4-5 on page 4-8.
SCI Base + 0xF10	Read/write	16 16	0x0000	SCITDR	Test data register. See Table 4-6 on page 4-9.

4.3.1 Test control register, SCITCR

SCITCR controls the operation of the PrimeCell SCI under test conditions. Table 4-2 shows the bit assignment of the SCITCR register.

Table 4-2 SCITCR register bits

Bits	Name	Description
15:2	-	Reserved, unpredictable when read.
1	Test fifo enable (TESTFIFO)	When this bit is 1, a write to the <b>SCITDR</b> writes data into the receive FIFO, and reads from the <b>SCITDR</b> reads data out of the transmit FIFO. When this bit is 0, data cannot be read directly from the transmit FIFO or written directly to the receive FIFO (normal operation). The reset value is 0.
0	ITEN	Integration test enable. When this bit is 1, the PrimeCell SCI is placed in integration test mode, otherwise it is in normal mode.

### 4.3.2 Test input register, SCIITIP

SCIITIP is a read/write register. In integration test mode it allows inputs to be both written to and read from. Table 4-3 shows the bit assignment of the SCIITIP register.

**Table 4-3 SCIITIP register bits**

Bits	Name	Description
15:6	-	Reserved, unpredictable when read.
5	SCITXDMACLR	Writes to this bit specify the value to be driven on the intra-chip input, <b>SCITXDMACLR</b> , in the integration test mode. Reads return the value of <b>SCITXDMACLR</b> at the output of the test multiplexor.
4	SCIRXDMACLR	Writes to this bit specify the value to be driven on the intra-chip input, <b>SCIRXDMACLR</b> , in the integration test mode. Reads return the value of <b>SCIRXDMACLR</b> at the output of the test multiplexor.
3	SCICLKIN	Reads return the value of the <b>SCICLKIN</b> primary input.
2	SCIDATAIN	Reads return the value of the <b>SCIDATAIN</b> primary input.
1	SCIDTECT	Reads return the value of the <b>SCIDTECT</b> primary input.
0	SCIDEACREQ	Reads return the value of the <b>SCIDEACREQ</b> primary input.

### 4.3.3 Test output register 1, SCIITOP1

SCIITOP1 is the integration test output register 1. In integration test mode, the primary and intra-chip outputs can be controlled by writes to the SCIITOP1 register. Reads of the intra-chip bits return the value present at the output of the test multiplexor. Reads of the primary output bits return the value written into the respective SCIITOP1 register bit. Table 4-4 shows the bit assignment of the SCIITOP1 register.

**Table 4-4 SCIITOP1 register bits**

Bits	Name	Description
15	SCITXDMASREQ	Intra-chip output. Writes specify the value to be driven on the <b>SCITXDMASREQ</b> line in the integration test mode. Reads return the value of <b>SCITXDMASREQ</b> at the output of the test multiplexor.
14	SCITXDMABREQ	Intra-chip output. Writes specify the value to be driven on the <b>SCITXDMABREQ</b> line in the integration test mode. Reads return the value of <b>SCITXDMABREQ</b> at the output of the test multiplexor.
13	SCIRXDMASREQ	Intra-chip output. Writes specify the value to be driven on the <b>SCIRXDMASREQ</b> line in the integration test mode. Reads return the value of <b>SCIRXDMASREQ</b> at the output of the test multiplexor.
12	SCIRXDMABREQ	Intra-chip output. Writes specify the value to be driven on the <b>SCIRXDMABREQ</b> line in the integration test mode. Reads return the value of <b>SCIRXDMABREQ</b> at the output of the test multiplexor.
11	SCITXTIDEINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCITXTIDEINTR</b> line in the integration test mode. Reads return the value of <b>SCITXTIDEINTR</b> at the output of the test multiplexor.
10	SCIRXTIDEINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCIRXTIDEINTR</b> line in the integration test mode. Reads return the value of <b>SCIRXTIDEINTR</b> at the output of the test multiplexor.
9	SCIRTOUTINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCIRTOUTINTR</b> line in the integration test mode. Reads return the value of <b>SCIRTOUTINTR</b> at the output of the test multiplexor.

**Table 4-4 SCIITOP1 register bits (continued)**

<b>Bits</b>	<b>Name</b>	<b>Description</b>
8	SCICHTOUTINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCICHTOUTINTR</b> line in the integration test mode. Reads return the value of <b>SCICHTOUTINTR</b> at the output of the test multiplexor.
7	SCIBLKOUTINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCIBLKOUTINTR</b> line in the integration test mode. Reads return the value of <b>SCIBLKOUTINTR</b> at the output of the test multiplexor.
6	SCIATRDOUTINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCIATRDOUTINTR</b> line in the integration test mode. Reads return the value of <b>SCIATRDOUTINTR</b> at the output of the test multiplexor.
5	SCIATRSOUTINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCIATRSOUTINTR</b> line in the integration test mode. Reads return the value of <b>SCIATRSOUTINTR</b> at the output of the test multiplexor.
4	SCITXERRINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCITXERRINTR</b> line in the integration test mode. Reads return the value of <b>SCITXERRINTR</b> at the output of the test multiplexor.
3	SCICARDDNINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCICARDDNINTR</b> line in the integration test mode. Reads return the value of <b>SCICARDDNINTR</b> at the output of the test multiplexor.
2	SCICARDUPINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCICARDUPINTR</b> line in the integration test mode. Reads return the value of <b>SCICARDUPINTR</b> at the output of the test multiplexor.
1	SCICARDOUTINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCICARDOUTINTR</b> line in the integration test mode. Reads return the value of <b>SCICARDOUTINTR</b> at the output of the test multiplexor.
0	SCICARDININTR	Intra-chip output. Writes specify the value to be driven on the <b>SCICARDININTR</b> line in the integration test mode. Reads return the value of <b>SCICARDININTR</b> at the output of the test multiplexor.

#### 4.3.4 Test output register 2, SCIITOP2

SCIITOP2 is the integration test output register 2. In integration test mode, the primary and intra-chip outputs can be controlled by writes to the SCIITOP2 register. Reads of the intra-chip bits return the value present at the output of the test multiplexor. Reads of the primary output bits return the value written into the respective SCIITOP2 register bit. Table 4-5 shows the bit assignment of the SCIITOP2 register.

**Table 4-5 SCIITOP2 register bits**

Bits	Name	Description
15:13	-	Reserved, unpredictable when read.
12	SCIINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCIINTR</b> line in the integration test mode. Reads return the value of <b>SCIINTR</b> at the output of the test multiplexor.
11	SCICLKACTINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCICLKACTINTR</b> line in the integration test mode. Reads return the value of <b>SCICLKACTINTR</b> at the output of the test multiplexor.
10	SCICLKSTPINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCICLKSTPINTR</b> line in the integration test mode. Reads return the value of <b>SCICLKSTPINTR</b> at the output of the test multiplexor.
9	SCIRORINTR	Intra-chip output. Writes specify the value to be driven on the <b>SCIRORINTR</b> line in the integration test mode. Reads return the value of <b>SCIRORINTR</b> at the output of the test multiplexor.
8	SCICLKOUT	Primary output. Writes specify the value to be driven on the <b>SCICLKOUT</b> line in the integration test mode. Reads return the value written into the SCIITOP2 <b>SCICLKOUT</b> register bit.
7	nSCICLKOUTEN	Primary output. Writes specify the value to be driven on the <b>nSCICLKOUTEN</b> line in the integration test mode. Reads return the value written into the SCIITOP2 <b>nSCICLKOUTEN</b> register bit.
6	nSCICLKEN	Primary output. Writes specify the value to be driven on the <b>nSCICLKEN</b> line in the integration test mode. Reads return the value written into the SCIITOP2 <b>nSCICLKEN</b> register bit.

**Table 4-5 SCIITOP2 register bits (continued)**

Bits	Name	Description
5	nSCIDATAOUTEN	Primary output. Writes specify the value to be driven on the <b>nSCIDATAOUTEN</b> line in the integration test mode. Reads return the value written into the SCIITOP2 <b>nSCIDATAOUTEN</b> register bit.
4	nSCIDATAEN	Primary output. Writes specify the value to be driven on the <b>nSCIDATAEN</b> line in the integration test mode. Reads return the value written into the SCIITOP2 <b>nSCIDATAEN</b> register bit.
3	SCIVCCEN	Primary output. Writes specify the value to be driven on the <b>SCIVCCEN</b> line in the integration test mode. Reads return the value written into the SCIITOP2 <b>SCIVCCEN</b> register bit.
2	SCIFCB	Primary output. Writes specify the value to be driven on the <b>SCIFCB</b> line in the integration test mode. Reads return the value written into the SCIITOP2 <b>SCIFCB</b> register bit.
1	nSCICARDRST	Primary output. Writes specify the value to be driven on the <b>nSCICARDRST</b> line in the integration test mode. Reads return the value written into the SCIITOP2 <b>nSCICARDRST</b> register bit.
0	SCIDEACACK	Primary output. Writes specify the value to be driven on the <b>SCIDEACACK</b> line in the integration test mode. Reads return the value written into the SCIITOP2 <b>SCIDEACACK</b> register bit.

#### 4.3.5 Test data register, SCITDR

SCITDR enables data to be written into the receive FIFO and read out from the transmit FIFO for test purposes. This test function is enabled by the **TESTFIFO** signal, bit 1 of the test control register (SCITCR). Table 4-6 shows the bit assignment of the SCITDR register.

**Table 4-6 SCITDR register bits**

Bits	Name	Description
15:0	DATA	When the <b>TESTFIFO</b> signal is asserted, data can be written into the receive FIFO and read out of the transmit FIFO.

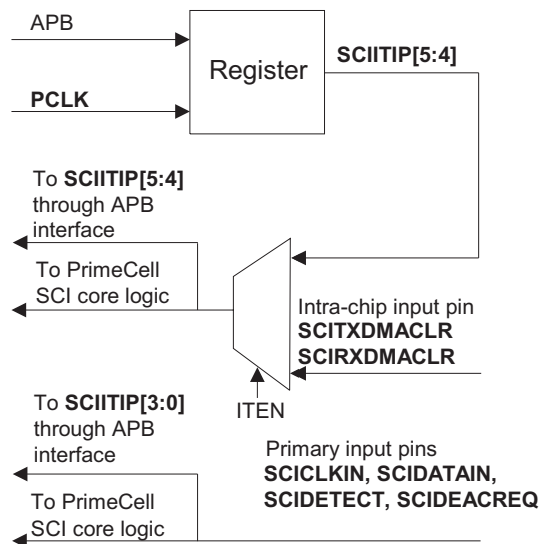
## 4.4 Integration testing of block inputs

The following sections describe the integration testing for the block inputs:

- *Intra-chip inputs* on page 4-10
- *Primary inputs* on page 4-11.

### 4.4.1 Intra-chip inputs

Figure 4-1 explains the implementation details of the input integration test harness. The ITEN bit is used as the control bit for the multiplexor, which is used in the read path of the **SCITXDMACLR** and **SCIRXDMACLR** intra-chip inputs. If the ITEN control bit is deasserted, the **SCITXDMACLR** and **SCIRXDMACLR** intra-chip inputs are routed as the internal **SCITXDMACLR** and **SCIRXDMACLR** inputs respectively, otherwise the stored register values are driven on the internal line. All other bits in the SCIITIP register are connected directly to the primary input pins.



**Figure 4-1 Input integration test harness**

When you run integration tests with the PrimeCell SCI in a standalone test setup:

- Write a 1 to the ITEN bit in the control register. This selects the test path from the SCIITIP[5:4] register bits to the **SCIRXDMACLR** and **SCITXDMACLR** signals.
- Write a 1 and then a 0 to each of the SCIITIP[5:4] register bits, and read the same register bits to ensure that the value written is read out.



When you run integration tests with the PrimeCell SCI as part of an integrated system:

- Write a 0 to the ITEN bit in the control register. This selects the normal path from the external **SCIRXDMACLR** pin to the internal **SCIRXDMACLR** signal, and the path from the external **SCITXDMACLR** pin to the internal **SCITXDMACLR** pin.
- Write a 1 and then a 0 to the internal test registers of the DMA controller to toggle the **SCIRXDMACLR** signal connection between the DMA controller and the PrimeCell SCI. Read from the SCIITIP[4] register bit to verify that the value written into the DMA controller, is read out through the PrimeCell SCI. Similarly, write a 1 and then a 0 to the internal registers of the DMA controller to toggle the **SCITXDMACLR** signal connection between the DMA controller and the PrimeCell SCI. Read from the SCIITIP[5] register bit to verify that the value written into the DMA controller, is read out through the PrimeCell SCI.

#### 4.4.2 Primary inputs

The following primary inputs are tested using the integration vector trickbox:

- **SCICLKIN**
- **SCIDATAIN**
- **SCIDEACREQ**
- **SCIDETECT**.

## 4.5 Integration testing of block outputs

The following sections describe the integration testing for the block outputs:

- *Intra-chip outputs* on page 4-12
- *Primary outputs* on page 4-14.

### 4.5.1 Intra-chip outputs

Use this test for the following outputs:

- **SCIINTR**
- **SCICLKACTINTR**
- **SCICLKSTPINTR**
- **SCIRORINTR**
- **SCITXDMASREQ**
- **SCITXDMABREQ**
- **SCIRXDMASREQ**
- **SCIRXDMABREQ**
- **SCITXTIDEINTR**
- **SCIRXTIDEINTR**
- **SCIRTOUTINTR**
- **SCICHTOUTINTR**
- **SCIBLKTOUTINTR**
- **SCIATRDOUITINTR**
- **SCIATRSOUTINTR**
- **SCITXERRINTR**
- **SCICARDDNINTR**
- **SCICARDUPINTR**
- **SCICARDOUTINTR**
- **SCICARDININTR.**

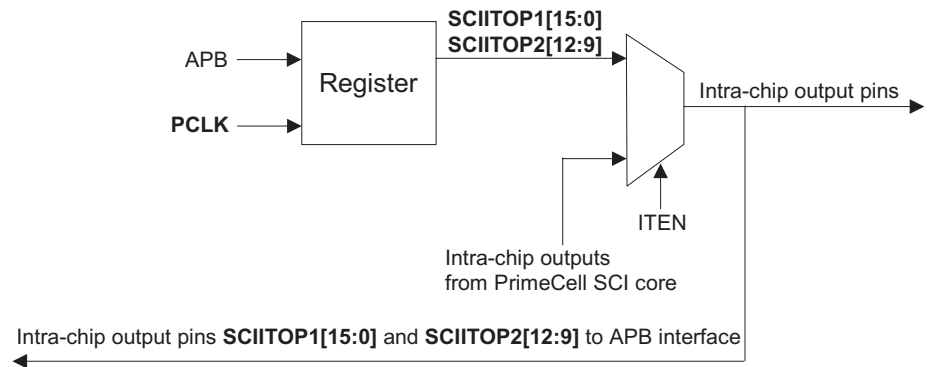
When you run integration tests with the PrimeCell SCI in a standalone test setup:

- Write a 1 to the ITEN bit in the control register. This selects the test path from the SCIITOP1[15:0] and SCIITOP2[12:9] register bits to the intra-chip output signals.
- Write a 1 and then a 0 to the SCIITOP1[15:0] and SCIITOP2[12:9] register bits, and read the same register bits to verify that the value written is read out.

When you run integration tests with the PrimeCell SCI as part of an integrated system:

- Write a 1 to the ITEN bit in the control register. This selects the test path from the SCIITOP1[15:0] and SCIITOP2[12:9] register bits to the intra-chip output signals.
- Write a 1 and then a 0 to the SCIITOP1[15:0] and SCIITOP2[12:9] register bits to toggle the signal connections between the DMA controller/interrupt controller and the PrimeCell SCI. Read from the internal test registers of the DMA controller/interrupt controller to verify that the value written into the SCIITOP1[15:0] and SCIITOP2[12:9] register bits is read out through the PrimeCell SCI.

Figure 4-2 explains the implementation details of the output integration test harness for intra-chip outputs.



**Figure 4-2 Output integration test harness, intra-chip outputs**

## 4.5.2 Primary outputs

Integration testing of primary outputs and primary inputs is carried out using the integration vector trickbox. Use this test for the following outputs:

- **nSCICLKEN**
- **nSCICLKOUTEN**
- **SCICLKOUT**
- **nSCIDATAEN**
- **nSCIDATAOUTEN**
- **SCIDEACACK**
- **SCIVCCEN**
- **nSCICARDRST**
- **SCIFCB.**

---

### Note

Only the **SCICLKOUT**, **nSCIDATAOUTEN**, **SCIDEACACK**, **SCIVCCEN**, **SCIFCB**, and **nSCICARDRST** signals are available at the output pads of a typical configuration. The **nSCICLKEN**, **nSCICLKOUTEN**, and **nSCIDATAEN** signals are internal connections to the pad.

---

Verify the primary input and output pin connections as follows:

- The primary outputs, **SCIDEACACK**, **nSCIDATAOUTEN**, and **SCICLKOUT** are connected to the **SCIDEACREQ**, **SCIDATAIN**, and **SCICLKIN** inputs respectively by the integration trickbox shown in Figure 4-3 on page 4-15. To test the **nSCICLKEN** and **nSCIDATAEN** connections, you can apply a weak pull down/pull to the tristate pins through the trickbox. It is not possible to test the **nSCICLKOUTEN** connection in this configuration.
- The **SCIVCCEN**, **nSCICARDRST**, and **SCIFCB** signals are exclusive ORed within the integration trickbox. The output is connected to the **SCIDTECT** input.
- All the primary outputs can be accessed through the **SCIITOP2[8:0]** register bits. Different data patterns are written to the output pins using the **SCIITOP2[8:0]** register bits.
- The looped-back data is read back through the **SCIITIP[3:0]** register bits.

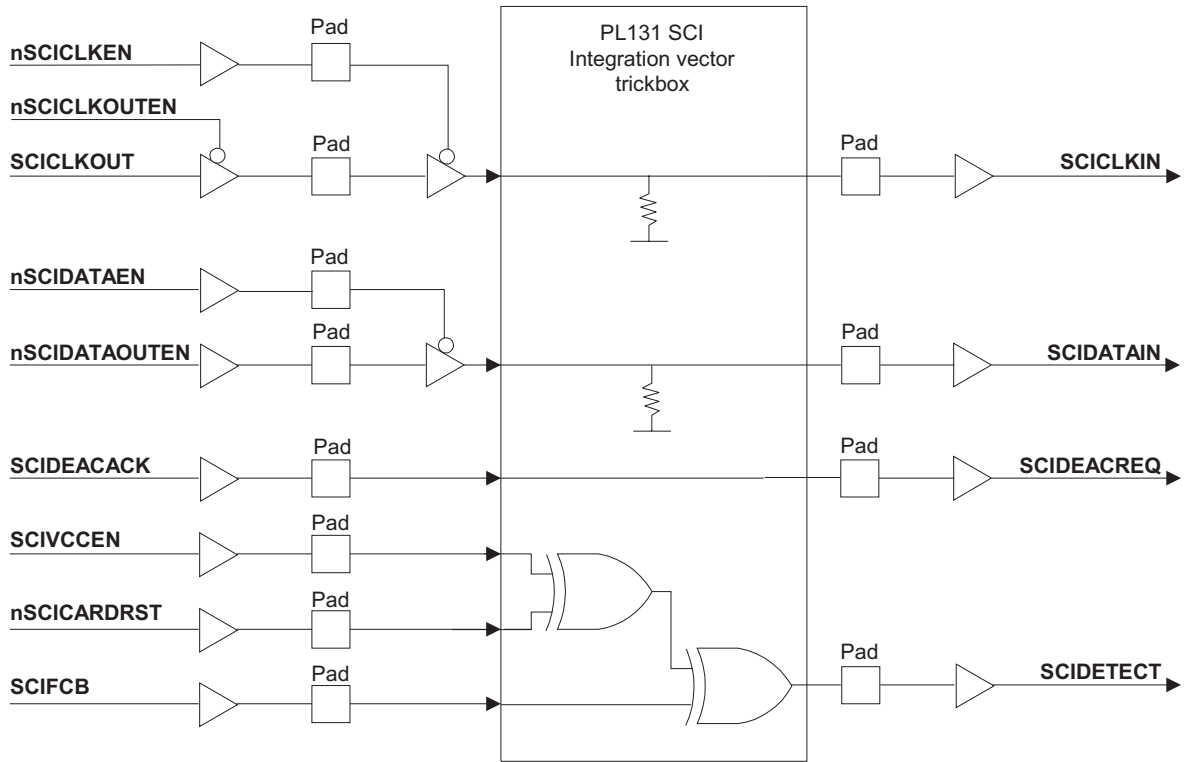
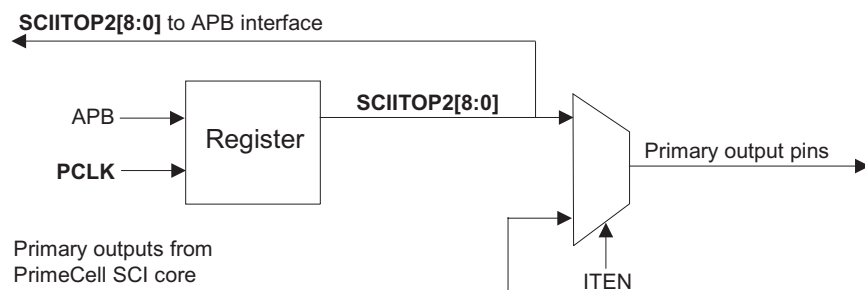


Figure 4-3 Primary outputs routed to primary inputs

Figure 4-4 shows implementation details of the output integration test harness in the case of primary outputs.



**Figure 4-4 Output integration test harness, primary outputs**

## 4.6 Integration test summary

Table 4-7 summarizes the integration test strategy for all PrimeCell SCI pins.

**Table 4-7 PrimeCell SCI integration test strategy**

Name	Type	Source/ destination	Test strategy
<b>PRESETn</b>	Input	Reset controller	Not tested using integration test vectors
<b>PADDR [11:2]</b>	Input	APB	Register read/write
<b>PCLK</b>	Input	APB	Register read/write
<b>PENABLE</b>	Input	APB	Register read/write
<b>PRDATA [15:0]</b>	Output	APB	Register read/write
<b>PSEL</b>	Input	APB	Register read/write
<b>PWDATA [15:0]</b>	Input	APB	Register read/write
<b>PWRITE</b>	Input	APB	Register read/write
<b>SCICLK</b>	Input	Clock generator	Not tested using integration test vectors
<b>nSCIRST</b>	Input	Reset controller	Not tested using integration test vectors
<b>SCITXTIDEINTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCIRXTIDEINTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCICLKACTINTR</b>	Output	Interrupt controller	Using SCIITOP2 register
<b>SCICLKSTPINTR</b>	Output	Interrupt controller	Using SCIITOP2 register
<b>SCIRORINTR</b>	Output	Interrupt controller	Using SCIITOP2 register
<b>SCIRTOUTINTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCICHTOUTINTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCIBLKTOUTINTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCIATRDOUTINTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCIATRSOUTINTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCITXERRINTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCICARDDNINTR</b>	Output	Interrupt controller	Using SCIITOP1 register

**Table 4-7 PrimeCell SCI integration test strategy (continued)**

<b>Name</b>	<b>Type</b>	<b>Source/ destination</b>	<b>Test strategy</b>
<b>SCICARDUPINTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCICARDOUTINTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCICARDININTR</b>	Output	Interrupt controller	Using SCIITOP1 register
<b>SCIINTR</b>	Output	Interrupt controller	Using SCIITOP2 register
<b>SCITXDMASREQ</b>	Output	DMA controller	Using SCIITOP1 register
<b>SCIRXDMASREQ</b>	Output	DMA controller	Using SCIITOP1 register
<b>SCITXDMABREQ</b>	Output	DMA controller	Using SCIITOP1 register
<b>SCIRXDMABREQ</b>	Output	DMA controller	Using SCIITOP1 register
<b>SCITXDMACLR</b>	Input	DMA controller	Using SCIITIP register
<b>SCIRXDMACLR</b>	Input	DMA controller	Using SCIITIP register
<b>SCICLKIN</b>	Input	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>SCIDATAIN</b>	Input	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>SCIDEACREQ</b>	Input	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>SCIDETECT</b>	Input	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>nSCICLKIN</b>	Output	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>nSCICLKOUTEN</b>	Output	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>SCICLKOUT</b>	Output	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>nSCIDATAEN</b>	Output	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>nSCIDATAOUTEN</b>	Output	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers



**Table 4-7 PrimeCell SCI integration test strategy (continued)**

<b>Name</b>	<b>Type</b>	<b>Source/ destination</b>	<b>Test strategy</b>
<b>SCIDEACACK</b>	Output	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>SCIVCCEN</b>	Output	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>nSCICARDRST</b>	Output	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>SCIFCB</b>	Output	Pad	Using integration vector trickbox and SCIITIP and SCIITOP2 registers
<b>SCANENABLE</b>	Input	Test controller	Not tested using integration test vectors
<b>SCANINPCLK</b>	Input	Test controller	Not tested using integration test vectors
<b>SCANINSCICLK</b>	Input	Test controller	Not tested using integration test vectors
<b>SCANOUTPCLK</b>	Output	Test controller	Not tested using integration test vectors
<b>SCANOUTSCICLK</b>	Output	Test controller	Not tested using integration test vectors



# Appendix A

## **ARM PrimeCell Smart Card Interface (PL131)**

### **Signal Descriptions**

This appendix describes the signals that interface with the ARM PrimeCell Smart Card Interface (PL131). It contains the following:

- *AMBA APB signals* on page A-2
- *On-chip signals* on page A-3
- *Signals to pads* on page A-5.

## A.1 AMBA APB signals

The PrimeCell SCI is connected to the AMBA APB bus as a bus slave. With the exception of the **PRESETn** signal, the APB signals have a **P** prefix and are active HIGH. Active LOW signals contain a lower case **n**. The AMBA APB signals are described in Table A-1.

**Table A-1 AMBA APB signal descriptions**

Name	Type	Source/ destination	Description
<b>PRESETn</b>	Input	Reset controller	Bus reset signal, active LOW.
<b>PADDR[11:2]</b>	Input	APB bridge	Subset of AMBA APB address bus.
<b>PCLK</b>	Input	Clock generator	AMBA APB clock, used to time all bus transfers.
<b>PENABLE</b>	Input	APB bridge	AMBA APB enable signal. <b>PENABLE</b> is asserted HIGH for one cycle of <b>PCLK</b> to enable a bus transfer.
<b>PRDATA [15:0]</b>	Output	APB bridge	Subset of unidirectional AMBA APB read data bus.
<b>PSEL</b>	Input	APB bridge	PrimeCell SCI select signal from decoder. When HIGH this signal indicates the slave device is selected by the APB bridge, and that a data transfer is required.
<b>PWDATA [15:0]</b>	Input	APB bridge	Subset of unidirectional AMBA APB write data bus.
<b>PWRITE</b>	Input	APB bridge	AMBA APB transfer direction signal, indicates a write access when HIGH, read access when LOW.

## A.2 On-chip signals

Table A-2 shows the non-AMBA on-chip signals from the block.

**Table A-2 On-chip signals**

Name	Type	Source/ destination	Description
<b>SCICLK</b>	Input	Clock generator	PrimeCell SCI reference clock
<b>nSCIRST</b>	Input	Reset controller	PrimeCell SCI reset signal to <b>SCICLK</b> clock domain, active LOW. The reset controller must use <b>PRESETn</b> to assert <b>nSCIRST</b> asynchronously but negate it synchronously with <b>SCICLK</b> .
<b>SCICARDININTR</b>	Output	Interrupt controller	PrimeCell SCI card in interrupt (active HIGH).
<b>SCICARDOUTINTR</b>	Output	Interrupt controller	PrimeCell SCI card out interrupt (active HIGH).
<b>SCICARDUPINTR</b>	Output	Interrupt controller	PrimeCell SCI card powered up interrupt (active HIGH).
<b>SCICARDDNINTR</b>	Output	Interrupt controller	PrimeCell SCI card powered down interrupt (active HIGH).
<b>SCITXERRINTR</b>	Output	Interrupt controller	PrimeCell SCI character transmission error interrupt (active HIGH).
<b>SCIATRSTOUTINTR</b>	Output	Interrupt controller	PrimeCell SCI ATR start timeout interrupt (active HIGH).
<b>SCIATRSDOUTINTR</b>	Output	Interrupt controller	PrimeCell SCI ATR duration timeout interrupt (active HIGH).
<b>SCIBLKTOUTINTR</b>	Output	Interrupt controller	PrimeCell SCI block timeout interrupt between blocks (active HIGH).
<b>SCICHTOUTINTR</b>	Output	Interrupt controller	PrimeCell SCI character timeout interrupt between characters (active HIGH).
<b>SCIROUTINTR</b>	Output	Interrupt controller	PrimeCell SCI receive FIFO read timeout interrupt (active HIGH)
<b>SCIRXTIDEINTR</b>	Output	Interrupt controller	PrimeCell SCI receive FIFO tide mark reached interrupt (active HIGH).
<b>SCITXTIDEINTR</b>	Output	Interrupt controller	PrimeCell SCI transmit FIFO tide mark reached interrupt (active HIGH).
<b>SCIROUTINTR</b>	Output	Interrupt controller	PrimeCell SCI receive overrun interrupt (active HIGH).
<b>SCICLKSTPINTR</b>	Output	Interrupt controller	PrimeCell SCI clock stop interrupt (active HIGH).

**Table A-2 On-chip signals (continued)**

<b>Name</b>	<b>Type</b>	<b>Source/ destination</b>	<b>Description</b>
<b>SCICLKACTINTR</b>	Output	Interrupt controller	PrimeCell SCI clock active interrupt (active HIGH).
<b>SCIINTR</b>	Output	Interrupt controller	PrimeCell SCI interrupt (active HIGH). A single combined interrupt generated as an OR function of the fifteen individually maskable interrupts above.
<b>SCITXDMASREQ</b>	Output	DMA controller	PrimeCell SCI transmit DMA single request (active HIGH).
<b>SCIRXDMASREQ</b>	Output	DMA controller	PrimeCell SCI receive DMA single request (active HIGH).
<b>SCITXDMABREQ</b>	Output	DMA controller	PrimeCell SCI transmit DMA burst request (active HIGH).
<b>SCIRXDMABREQ</b>	Output	DMA controller	PrimeCell SCI receive DMA burst request (active HIGH).
<b>SCITXDMACLR</b>	Input	DMA controller	DMA request clear, asserted by the DMA controller to clear the transmit request signals. If DMA burst transfer is requested, the clear signal is asserted during the transfer of the last data in the burst.
<b>SCIRXDMACLR</b>	Input	DMA controller	DMA request clear, asserted by the DMA controller to clear the receive request signals. If DMA burst transfer is requested, the clear signal is asserted during the transfer of the last data in the burst.
<b>SCANENABLE</b>	Input	Test controller	PrimeCell SCI scan enable signal for both clock domains.
<b>SCANINPCLK</b>	Input	Test controller	PrimeCell SCI input scan signal for the PCLK domain.
<b>SCANINSCICLK</b>	Input	Test controller	PrimeCell SCI input scan signal for the SCICLK domain.
<b>SCANOUTPCLK</b>	Output	Test controller	PrimeCell SCI output scan signal for the PCLK domain.
<b>SCANOUTSCICLK</b>	Output	Test controller	PrimeCell SCI output scan signal for the SCICLK domain.

## A.3 Signals to pads

Table A-3 describes the signals from the PrimeCell SCI to input/output pads of the chip. It is the responsibility of the user to make proper use of the peripheral pins to meet the exact interface requirements.

**Table A-3 Signals to pads**

Name	Type	Pad type	Description
<b>SCICKIN</b>	Input	Pad	PrimeCell SCI clock input.
<b>SCIDATAIN</b>	Input	Pad	PrimeCell SCI serial data input.
<b>nSCICLKOUTEN</b>	Output	Pad	Tristate output buffer control (active LOW).
<b>SCICLKOUT</b>	Output	Pad	Clock output.
<b>nSCIDATAOUTEN</b>	Output	Pad	Data output enable (typically drives an open-drain configuration, active LOW).
<b>nSCICLKEN</b>	Output	Pad	Tristate control for external off-chip buffer (active LOW).
<b>nSCIDATAEN</b>	Output	Pad	Tristate control for external off-chip buffer (active LOW).
<b>SCIVCCEN</b>	Output	Pad	Supply voltage controls (active HIGH).
<b>nSCICARDRST</b>	Output	Pad	Reset to card (active LOW).
<b>SCIFCB</b>	Output	Pad	Function code bit, used in conjunction with <b>nSCICARDRST</b> .
<b>SCIDTECT</b>	Input	Pad	Card detects signal from card interface device (active HIGH).
<b>SCIDEACREQ</b>	Input	Pmu	Card deactivation request signal from PMU (active HIGH).
<b>SCIDEACACK</b>	Output	Pmu	Card deactivation acknowledgement to PMU (active HIGH).





# Index

The items in this index are listed in alphabetic order. The references given are to page numbers.

## A

Address, base 2  
AMBA APB interface 4  
APB bus 2  
APB signals 2

## B

Base address 2  
Block time and time between  
characters 23

## C

Character framing 20

## D

Data transfer 18  
data rates 18  
value X BAUD rate clock 19

## E

EMV character timing for T=0 21  
EMV character timing for T=1 22

## I

Interrupt generation logic 6  
Interrupts 40

## L

Logic  
interrupt generation 6  
synchronizing 7  
test 7

## M

Maskable interrupts 6

## O

On-chip signals 3

## P

- PADDR 2
- Parity error 24
- PCLK 7, 2
- PENABLE 2
- PRDATA 2
- PRESETn 2
- PrimeCell SCI 2
  - control logic 5
  - features 4
  - functional description 3
  - operation 8
  - overview 2
  - programmable parameters 5
- Programmer's model 1
  - for test 1
- PSEL 2
- PWDATA 2
- PWRITE 2

## R

- Receive 23
- Receive FIFO 6
- Register
  - descriptions 7
  - SCIATIME 16
  - SCIATRDTIME 17
  - SCIATRSTIME 17
  - SCIBAUD 13
  - SCIBLKGUARD 24
  - SCIBLKTIME 21
  - SCICHGUARD 23
  - SCICHTIME 20
  - SCICKICC 12
  - SCICR0 8
  - SCICR1 11
  - SCICR2 12
  - SCIDATA 8
  - SCIDTIME 16
  - SCIFR 25
  - SCIHER 27
  - SCIISTAT 33
  - SCIRAWSTAT 35
  - SCIRETRY 19
  - SCIRXCOUNT/
    - SCIRXCOUNTCLR 26

- SCIRXTIME 24
- SCISTABLE 15
- SCISYNCDATA 34
- SCITIDE 14
- SCITXCOUNT/
  - SCITXCOUNTCLR 26
- SCIVALUE 13
  - summary 3
- Register block 4
- Registers
  - SCIPCellID0-3 38
  - SCIPeriphID0-3 35
- Response to a non-ideal card session 16
- Response to an ideal card session 9
- RXREAD interrupt 24

## S

- SCIATRSTIME 17
- SCICLK 7
- Signals to pads 5
- Stages of a card session 9
  - Answer To Reset (ATR) sequence 13
  - contact activation and cold reset sequence 12
  - contact deactivation sequence and card removal 13
  - execution of a transaction 13
  - SCI reset and initial configuration 10
  - smart card insertion and detection 11
- Summary of registers 3
- Synchronizing logic 7
- Synchronizing registers 7

## T

- Test logic 7
- Test registers
  - SCIITIP 5
  - SCIITOP 6, 8
  - SCITCR 4
  - SCITDR 9
- Transmit 22
- Transmit and receive logic 4

- Transmit FIFO 5

## W

- Warm reset sequence 15