Application Report
SPNA124A–September 2011

Configuring the Hercules™ ARM® Safety MCU SCI/LIN Module for UART Communication

Prathap Srinivasan

ABSTRACT

The purpose of this application report is to provide help for configuring the SCI/LIN module of Hercules microcontrollers for universal asynchronous receiver-transmitter (UART) communication. The document explains the SCI/LIN module configuration done in the UART driver that is used by Texas Instruments (TI™) in a safety demo of the Hercules product family. The Hercules family of microcontrollers from TI is a family of 32-bit RISC microcontrollers with an advanced safety architecture and a rich peripheral set.

Project collateral and source code can be downloaded from the following URL: http://www.ti.com/lit/zip/spna124.

Contents

List of Figures

List of Tables

1 Hercules SCI/LIN Module

1.1 Introduction

The Hercules LIN module can be configured as a full LIN mode or as an SCI (UART) mode. The SCI/LIN module is a universal asynchronous receiver-transmitter (UART) that implements the standard non-return-to-zero (NRZ) format. The serial communications interface (SCI) can be used to communicate, for example, through an RS-232 port or over a K-line.

1.2 Main Features of SCI/LIN Module in SCI Mode

The Hercules SCI/LIN module offers the following functions in SCI mode:

- Standard UART communication
- Multi-buffered receive and transmit units
- DMA capability for minimal CPU intervention

- Supports full- or half-duplex operation
- Standard NRZ format
- Double-buffered receive and transmit functions
- Configurable frame format of 3 to 13 bits per character based on the following:
  - Data word length programmable from one to eight bits
  - Additional address bit in address-bit mode
  - Parity programmable for zero or one parity bit, odd or even parity
  - Stop programmable for one or two stop bits
- Asynchronous or isosynchronous communication modes
- Two multiprocessor communication formats allow communication between more than two devices.
- Sleep mode is available to free CPU resources during multiprocessor communication.
- The 24-bit programmable baud rate supports 224 different baud rates selection.
- Four error flags and five status flags provide detailed information regarding SCI events.

1.3 Block Diagram

The Hercules SCI/LIN module has three main blocks.

- Transmitter
- Receiver
- Baud Clock Generator

For more information about the blocks, see the device-specific technical reference manual.

A general block diagram of the SCI is shown in Figure 1.

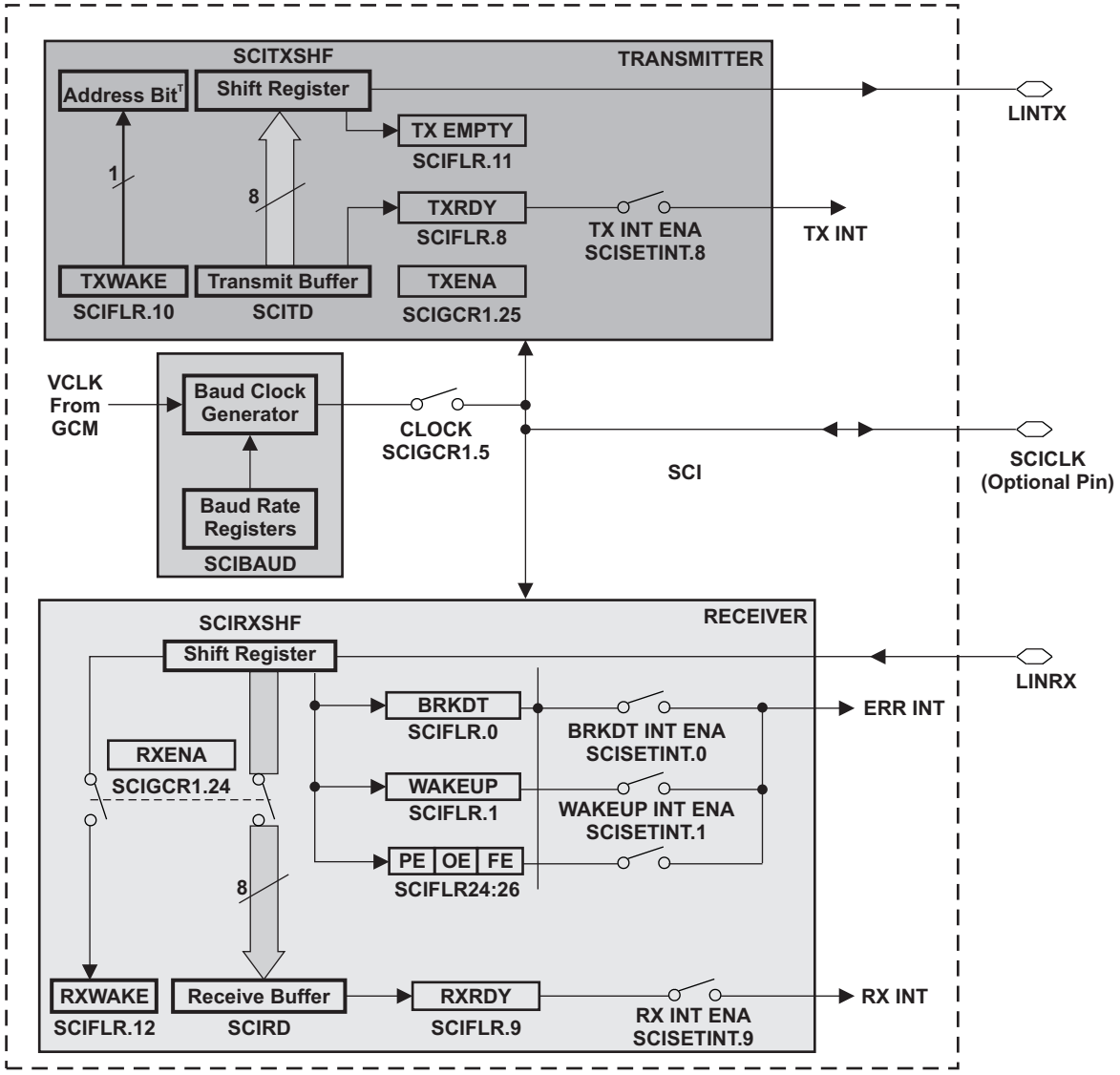

Figure 1. SCI Block Diagram

2 Configuration of SCI/LIN Module for UART Communication

This section of the application report explains the configuration of the SCI/LIN module for UART Communication between the Hercules microcontroller and the PC COM port using UART Software.

SCI/LIN module is configured to function in the SCI (UART) mode, which facilitates asynchronous communication between the microcontroller and PC.

The UART communication in the safety demo software is used as an example in this application report. The PC communicates to the safety demo software through the RS232 COM port. The safety demo software execution in the Hercules microcontroller is controlled from the PC through user-defined key words transmitted through RS232.

Table 1 lists some of the user-defined commands used in the safety demo software.

Table 1. Safety Demo Software User Commands

Sl.No	User Command	Description	Data Transmitted by MICRO
1	*DO0100!	Get the Firmware version	*VALID#!VER 1.1
2	*DO0200!	Get Hardware information – Device ID, Wafer Number, Lot Number etc.	*VALID#! 80206D0D0974100101BA000900900800
3	INVALID	Error Message is transmitted back.	WHO R U?

Windows HyperTerminal software is used to communicate from the PC to the Hercules microcontroller. You can use any terminal application software, including the one built into Code Composer Studio™ 4, with the necessary baud rate, parity, and number of stop bit configurations as described in Section 2.5. SCI/LIN Rx and Tx pins can be connected to the COM Port through an RS232 transceiver.

2.1 Circuit Diagram

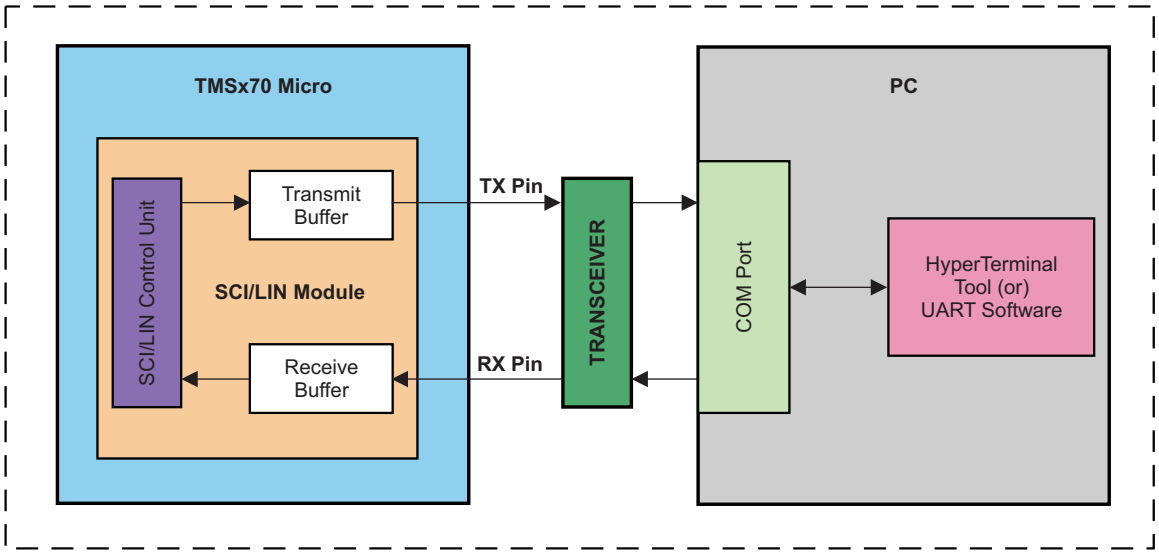

Figure 2. SCI/LIN to PC UART Communication

2.2 Initialization and Configuration of SCI/LIN Module

This section explains the different SCI/LIN module registers that have to be configured for UART communication.

NOTE: The safety demo software is developed using the HALCoGen Code generation tool. The symbols used in code snippets are the actual symbols generated by the HALCoGen tool.

sciREG1 – Pointer to the SCI/LIN module base address. It is a structure pointer for bitwise access to the register.

To initialize the SCI/LIN module for UART communication, configure the following registers. For more information on the registers and their default state after reset, see the device-specific technical reference manual.

2.2.1 Disabling SCI/LIN Module (SCIGCR0) Reset

The reset bit disables the SCI/LIN module and ensures the SCI/LIN registers are initialized to their reset values. The SCI/LIN module must be brought out of reset before any UART operation including receive or transmit. The following code snippet enables the SCI/LIN module.

```
sciREG1->GCR0 = 1U;     // Bring the Module Out of Reset
```

2.2.2 Configuration of SCI/LIN Global Control Register 1 (SCIGCR1)

The SCIGCR1 is a 32-bit register that specifies the basic functionality and configuration for the UART communication. The following attributes are defined by the SCIGCR1 register:

- Enable transmit
- Enable receive
- Internal clock (device has no clock pin)
- Number of stop bits
- Even parity or odd parity
- Asynchronous timing mode
- Enable or disable multi-buffer mode

The following code snippet shows initializes the SCIGCR1 register (Transmit and Receive with odd parity are enabled and multi-buffered mode is disabled):

```
sciREG1->GCR1 = 0x0300002A;     // Enable Transmit and Receive functionality
```

2.2.3 Configuring SCI/LIN Baud Rate Register (BRSR)

The BRSR is a 32-bit register that is used to select a baud rate for the SCI/LIN module. The below code snippet shows UART configuration for 19200 baud rate from 80 MHz VCLK.

```
sciREG1->BAUD = 260;     // baud rate --- 19200 from 80MHz VCLK
```

2.2.4 Configuring SCI/LIN Data Length

The SCI Format Control Register (SCIFORMAT) is a 32-bit register that indicates the transmitter/receiver frame length and message data length. The below code snippet configures the data length as 8 bits and frame length as 0. Standard setup for UART is a non-buffered mode so no frame length necessary. Frame length is used either in LIN Mode or buffered SCI Mode only.

```
sciREG1->LENGTH = 7;     // 8 bit Data length
```

2.2.5 Configuring SCI/LIN Pins in Functional Mode

The SCI Pin I/O Control Register 0 (SCIPIO0) defines whether the SCI/LIN pins are configured in functional mode or in GIO mode. The following code snippet shows configuration of SCI/LIN in functional mode.

```
sciReg1->FUNC = 0x00000006;     // Enabling Tx and Rx pins
```

2.2.6 Configuring PULL UP Functionality of SCI/LIN Module

The SCI Pin I/O Control Register 8 (SCIPIO8) is a 32-bit register that enables the PULL UP or PULL DOWN functionality of SCI/LIN pins. The below code snippet enables PULL UP functionality for SCI/LIN Pins.

```
sciREG1->PSL = 0x00000006;     //Enable PULL Up functionality for RX and TX Pins
```

2.2.7 Initialization and Configuration of SCI/LIN iInterrupts

The SCI Set Interrupt Level Register (SCISETINTLVL) is a 32-bit register that maps all the SCI/LIN Interrupts to either level 0 or level 1 of the vectored interrupt manager (VIM) according to their respective priority encoders.

NOTE: Level 1 interrupt has higher priority compared to Level 0.

The SCISETINT and SCICLEARINT registers are used to enable or disable the SCI/LIN interrupts by setting or clearing the respective interrupt bit. The following code snippet shows enabling of the Receive Interrupt and mapping it to Level 0 Interrupt. By default or at reset these registers are 0.

```
sciREG1->SETINTLVL = 0x00000000;     // All Interrupts are configured as Level 0

sciREG1->SETINT = 0x00000020;     // Enabling Rx Interrupt
```

2.2.8 Enabling SCI/LIN for UART Communication

The SWnRST in the SCIGCR1 initializes or enables the SCI/LIN module for UART communication. While this bit is 0, the state machine is held in reset and the module will neither transmit nor receive data. Writing a 1 releases the state machine and the UART can transmit or receive data. The following code snippet shows enabling of SCI/LIN for UART communication.

```
sciReg1->GCR1 |= 0x00000080;     //End of Module Configuration
```

NOTE: After SWnRST is set to 1, the configuration of the module should not change.

2.3 Receive Routine of UART Communication in Safety Demo Software

After SCI/LIN module initialization, the demo software waits for the user command from the PC. Each user command is defined as eight data of 8 bit. The software configures the Rx Interrupt to indicate the receipt of data. This signals the software to retrieve the data from the Rx buffer. SCI/LIN uses its state machine to receive the data and triggers the interrupt.

The Receive routine of UART driver performs the following operations:

1. Checks whether a valid user command is received or not. A valid user command starts with '*' and ends with '!' along with the task number and Sub Task Number. Example: - *DO0200!. Here 02 is task number and 00 is sub-task number.
2. Passes the task number and sub-ask Number to main software.

2.4 Transmit Routine of UART Communication in Safety Demo Software

The Transmit routine of the UART driver software follows polling method to transmit data. The software polls for the transmitter buffer register ready flag (TXRDY) to be set before writing data to the transmit data buffer register. This ensures that the previous message has been sent before writing new data in the transmit data buffer.

Once the command is verified and task numbers are identified, the following information is transmitted back by the UART driver to the PC based on the received command.

1. For a valid user command received:
   (a) First an acknowledgment is transmitted (i.e., “*VALID#“).
   (b) Based on the task number and sub-task number received, the safety demo software executes a particular task or test.
   (c) The test results are transmitted back to the PC. Example: For *DO0100! (valid user command) the Firmware Version number “Ver 1.0” is transmitted.
2. For an invalid user command received:
   (a) An error message is transmitted. Example: For !watisth (Invalid command) the error message “WHO R U?” is transmitted back to PC.

NOTE:

- The Transmit and Receive routine of UART communication described above has nothing to do with the UART protocol. It is just the way the UART driver was written in the safety demo software.
- The safety demo software has enough bandwidth to take care of interrupt latency causing missed data because of other interrupts preventing the SCI Rx interrupt from firing. In order to use this UART driver function in their application software, the user has to take care of the interrupt latency that could cause data miss.

2.5 PC-UART Software Configuration

Any UART communication program can be used to send and receive characters over the serial port. The following UART configurations have to be made to communicate with the safety demo software.

- Baud Rate : 19200
- No Parity : None
- Stop bits : 0
- Flow Control : None
- Port : Select the connected COM port.
- Transmit Data : *DO0100! , *DO0200!

After configuring, keep the UART software active to receive and transmit messages. The following snapshots have been taken while using Hyperterminal UART software for transmitting user commands and receiving responses from the microcontroller in a Windows PC.

Figure 3 and Figure 4 show the snapshot of the valid user command (*DO0100! and *DO0200!) transmission and the response it received.

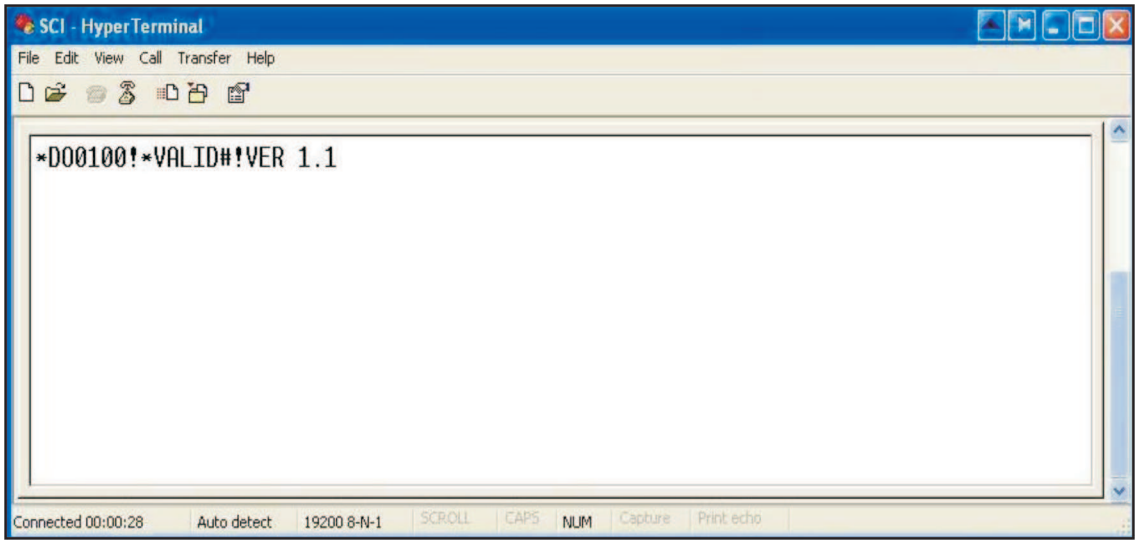

Figure 3. Transmission of Valid User Command *DO0100! and Reception of Corresponding Response.

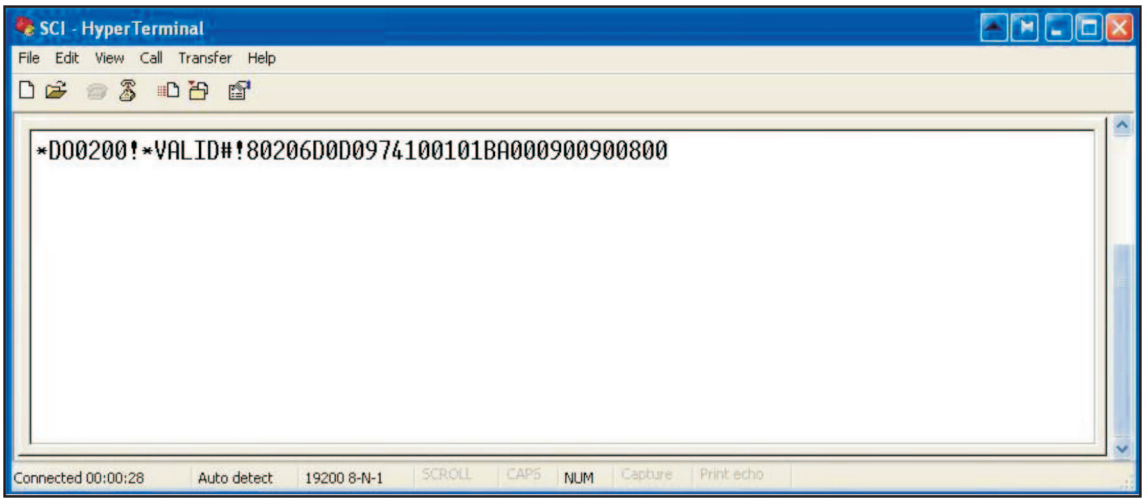

Figure 4. Transmission of Valid User Command *DO0200! and Reception of Corresponding Response

Figure 5 shows the snapshot of the invalid user command (!watishd) transmission and the response it received.

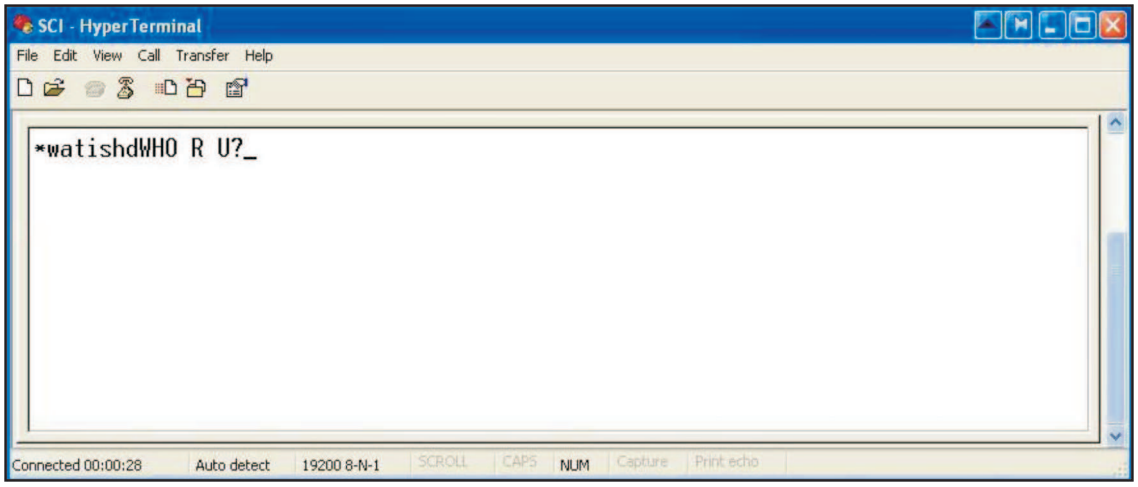

Figure 5. Transmission of Invalid User Command and Reception of Error Message.

2.6 SCI/LIN Module Configuration Steps for UART Communication

The following steps describe the SCI/LIN module configuration flow for UART communication.

1. Disable SCIGCR0 Reset.
2. Configure SCIGCR1.
   (a) Enable Transmit
   (b) Enable Receive
   (c) Configure number of stop bits (i.e., '1')
   (d) Enable asynchronous timing mode
   (e) Disable multi-buffer mode
3. Configure SCI/LIN baud rate to 19200 bits per second
4. Configure SCI/LIN data length to ‘7’ (i.e., 8 bits per character)
5. Configure SCI/LIN Pins in Functional Mode
6. Enable PULL UP functionality of SCI/LIN Module
7. Initialize and configure SCI/LIN interrupts
8. Enable SCI/LIN Module for UART communication: - Set SWnRST bit.

2.7 Software Flow Diagram

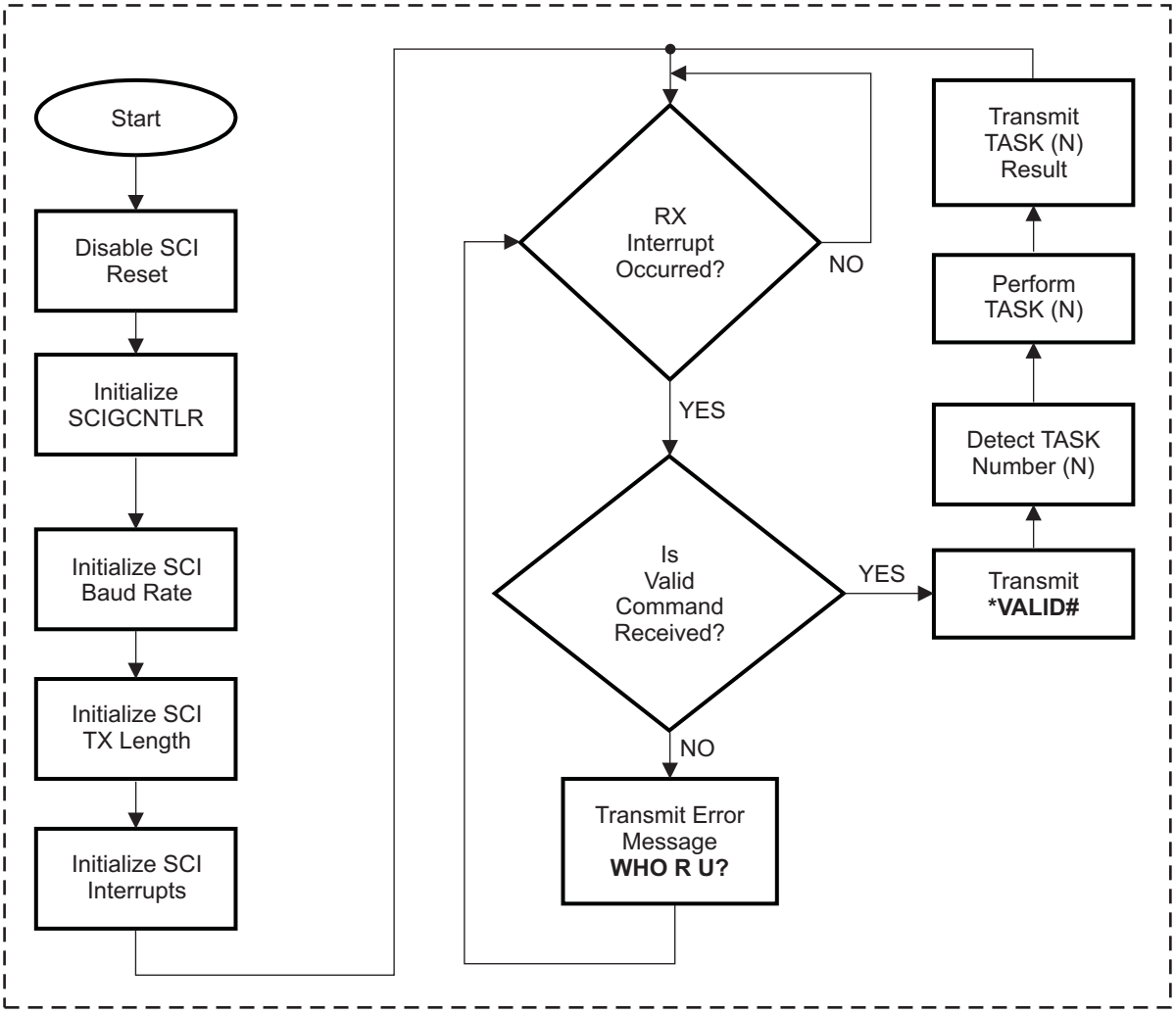

Figure 6. Flow Diagram of SCI/LIN for UART Communication

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265