# SIO4/8

**Four/Eight Channel High Speed Serial I/O**

## All SIO4 and SIO8 Models
## All Form Factors
## All Standard Zilog Versions
## All Standard SYNC Versions

# Driver Reference Manual

**Manual Revision: September 7, 2015**
**Driver Release Version 3.0.59.x.x**

**General Standards Corporation**
**8302A Whitesburg Drive**
**Huntsville, AL 35802**
**Phone: (256) 880-8787**
**Fax: (256) 880-8788**
**URL: http://www.generalstandards.com**
**E-mail: sales@generalstandards.com**
**E-mail: support@generalstandards.com**

# Preface



Additional copies of this manual or other literature may be obtained from:

**General Standards Corporation**
8302A Whitesburg Dr.
Huntsville, Alabama 35802
Phone: (256) 880-8787
FAX: (256) 880-8788
URL: http://www.generalstandards.com/
E-mail: sales@generalstandards.com

**General Standards Corporation** makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Although extensive editing and reviews are performed before release to ECO control, **General Standards Corporation** assumes no responsibility for any errors that may exist in this document. No commitment is made to update or keep current the information contained in this document.

**General Standards Corporation** does not assume any liability arising out of the application or use of any product or circuit described herein, nor is any license conveyed under any patent rights or any rights of others.

**General Standards Corporation** assumes no responsibility for any consequences resulting from omissions or errors in this manual or from the use of information contained herein.

**General Standards Corporation** reserves the right to make any changes, without notice, to this product to improve reliability, performance, function, or design.



The Purchaser of this software may use or modify in source form the subject software, but not to re-market or distribute it to outside agencies or separate internal company divisions. The software, however, may be embedded in the Purchaser's distributed software. In the event the Purchaser's customers require the software source code, then they would have to purchase their own copy of the software.

**General Standards Corporation** makes no warranty of any kind with regard to this software, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose and makes this software available solely on an "as-is" basis. **General Standards Corporation** reserves the right to make changes in this software without reservation and without notification to its users.

# Table of Contents

# Table of Figures

# 1. Introduction

This release of the driver is intended for all versions of the SIO4 and SIO8 that contain standard firmware for either Zilog Z16C30 or –SYNC based boards.

> **NOTE:** The device models listed on the front cover are those that are specifically supported by this release of the driver. Other models may be supported, though the level of support may vary.

## 1.1. Purpose

The purpose of this document is to describe the interface to the SIO4 Device Driver. This software provides the interface between "Application Software" and the SIO4 board. The interface to this board is at the device level.

## 1.2. Acronyms

The following is a list of commonly occurring acronyms used throughout this document.

| Acronyms | Description |
|---|---|
| DMA | Direct Memory Access |
| DMDMA | Demand Mode DMA |
| DPLL | Digital Phase Lock Loop |
| GSC | General Standards Corporation |
| PCI | Peripheral Component Interconnect |
| PMC | PCI Mezzanine Card |
| USC | Universal Serial Controller |

## 1.3. Definitions

The following is a list of commonly occurring terms used throughout this document.

| Term | Definition |
|---|---|
| Application | This refers to a user mode process. |
| Driver | This refers to the device driver. Depending on the OS, the driver may be a user space application, a kernel mode process, or something in between. |
| SIO4 | This is used as a general reference to any SIO4 supported by this driver. |

## 1.4. Hardware Overview

> **NOTE:** The SIO8 boards appear to the driver as two SIO4 boards.

The SIO4 is a four channel high-speed serial interface I/O board. This board provides for bi-directional serial data transfers between two computers, or one computer and an external peripheral.

This board also can transfer data indefinitely without host intervention. Once the data link between the two computers is established, the desired transfers can be performed and will become transparent to the user.

The SIO4 board includes a DMA controller and comes with a maximum of 256K Bytes of FIFO storage, which is 32K per channel side (32K * 2 * 4). The FIFO configuration can vary greatly from one SIO4 version to another (i.e. 32K * 2 * 4 to 1K * 2 *1 to none at all). The SIO4 comes in an RS232 version and an RS485/422 version. Both versions include two Universal Serial Controllers (Zilog Z16C30 USC). The DMA controller is capable of transferring data to and from host memory; whereas the FIFO memory provides a means for continuous transfer of data without interrupting the DMA transfers or requiring intervention from the host CPU. The board also provides

for interrupt generation for various states of the board like Sync Character detection, FIFO empty, FIFO full and DMA complete.

## 1.5. Reference Material

The following reference material may be of particular benefit in using the SIO4 and this driver. The specifications provide the information necessary for an in depth understanding of the specialized features implemented on this board.

- The applicable *SIO4/SIO8 Device Driver User Manual* for your operating system from General Standards Corporation.
- The applicable *SIO4/SIO8 User Manual* from General Standards Corporation.
- The *PCI Bus Master Interface Chip* data handbook for the PCI9056/9080 from PLX Technology, Inc.

  PLX Technology Inc.
  870 Maude Avenue
  Sunnyvale, California 94085 USA
  Phone: 1-800-759-3735
  WEB: http://www.plxtech.com/

- The *Z16C30 USC User's Manual* from Zilog. *
- The *Z16C30 Electronic Programmer's Manual* from Zilog (Zilog part number ZEPMDC00001). *

* The Zilog material is available from:

Zilog, Inc.
910 E Hamilton Ave
CAMPBELL, CA 95008 USA
Phone: 1-408-558-8500
WEB: Thttp://www.zilog.com/

# 2. Additional Components

The driver release is accompanied by a number of items besides the documentation. Some of these are described below.

## 2.1. Driver Interface Library

The driver interface includes a small number of functions which are exported by the static library `sio4_lib`. This library is the primary means of interacting with an SIO4. Refer to section 3.2 on page 16 for full details of these functions. The functions in the Driver Interface Library are defined in the header `sio4\lib\sio4_lib.h`. The library itself is named `sio4\docsrc\sio4_lib`. (The exact name and location of the library are OS specific.)

## 2.2. Utility Libraries

The driver is accompanied by several libraries intended to facilitate use of the driver.

### 2.2.1. Serial Protocol Libraries

The driver includes one or more serial protocol libraries designed to ease use of a specific serial protocol, or board model.

#### 2.2.1.1. The SYNC Library

The SYNC library is provided to simplify driver use for SYNC versions of the SIO4. Refer to the *SYNC Library User Manual* for additional information.

### 2.2.2. Document Source Code Examples

The Document Source Code library is provided as a means to insure that the sample code included in this reference manual compiles and builds. The functions are exported via their own header, `sio4/docsrc/sio4_dsl.h`, and are available in their own statically linkable library, `sio4/docsrc/sio4_dsl`. (The exact name and location of the library are OS specific.)

### 2.2.3. Utility Source Code

The driver archive includes utility source code suitable mostly for command line applications. A utility source file is included for every driver IOCTL service. Their purpose is to permit simple use of each service with suitable display output, if desired, where applicable, along with status. When there is an error, an error message is reported to the screen. Each service may be used to apply a setting, request the current setting, or to request support status. Other utility services are included as well. Of specific interest is `sio4_reg_list()`, which provides a detailed register dump of a channel's registers. This is especially useful for customer support when generated at a point in your code just before the problem appears. The utility function prototypes are defined in header files `sio4\docsrc\sio4_utils.h` and `sio4\docsrc\gsc_utils.h`, and are built static libraries `sio4\docsrc\sio4_utils` and `sio4\docsrc\gsc_utils`, respectively. (The exact name and location of the libraries are OS specific.) Other utility source libraries may be included with the Serial Protocol Libraries.

## 2.3. Sample Applications

Each driver release includes a small number of sample applications. Most of the sample applications are included with each OS specific release of the driver. A few however, are OS specific so are included only where applicable. Please refer to the OS specific driver user manual for information on the sample applications.

# 3. Driver Interface

The driver interface includes a small set of functions and an extensive number of IOCTL style service macros. The functions are defined via the Driver Interface Library `sio4\lib\sio4_lib.h`. The macros are defined in the files `sio4\driver\sio4.h` and `sio4\driver\sio4_usc.h`, which are included automatically via `sio4_lib.h`.

**NOTE:** Contact General Standards Corporation if additional driver functionality is required.

## 3.1. Macros

The driver interface includes as extensive number of macros centered on IOCTL style services definitions.

### 3.1.1. IOCTL

The IOCTL macros are documented in subsequent sections of this user manual. The IOCTL services are organized according to basic functional groupings, as given below. These macros are usable with the `sio4_ioctl()` function (section 3.2.2, page 17) provided in the Driver Interface Library.

- Services that are common to both Zilog based and SYNC style boards (section 4, page 21)
- Services specific to Zilog based boards (section 5, page 53)
- Services specific to the USC on the Zilog based boards (section 6, page 59)
- Services specific to SYNC boards (section 6.20, page 106)
- Other services which are specific to SIO4A boards (section 8, page 120)

The sections comprise a comprehensive list of all IOCTL services implemented by the driver. Not all services are supported on all boards. All boards support the register access services (`SIO4_IOCTL_REG_XXX`, starting in section 4.6.1 on page 47) and the wait services (`SIO4_IOCTL_WAIT_XXX`, starting in section 0 on page 34). For all other services support can be determined by passing in a value of `-1` and looking at the value returned. If the value returned is `-1`, then that service is unsupported on the board being accessed. If other than a `-1` is returned, then the service is supported. The set of supported services can also be determined by looking at the output of the *services* sample application (under the `sio4\services` directory). The application output specifically identifies the services that are supported and will conditionally list those that are not. Each service is described along with the applicable `sio4_ioctl()` function arguments.

### 3.1.2. Registers

The following table gives the complete set of SIO4 registers. The tables are divided by register categories.

#### 3.1.2.1. GSC Registers

The following table gives the complete set of GSC specific SIO4 registers. For detailed definitions of these registers refer to the *SIO4 User Manual*. As this driver supports all standard versions of the SIO4 some registers in the table below may not be supported by your board. Consult your hardware user manual for the set of registers supported by your board.

| Macros | Description | Notes |
|---|---|---|
| `SIO4_GSC_BCR` | Board Control Register | All models |
| `SIO4_GSC_BSR` | Board Status Register | SIO4A/B/BX/BXR, SIO8BXS |

| | | |
|---|---|---|
| SIO4_GSC_CCR | Clock Control Register | SIO4/A/B/BX w/ USC (legacy support only) |
| SIO4_GSC_CSR | Command/Status Register | All models |
| SIO4_GSC_FCR | FIFO Count Reg | SIO4B/BX/BXR, SIO8BXS, PCI-SIO4A |
| SIO4_GSC_FDR | FIFO Data Register | All models |
| SIO4_GSC_FR | Features Register | SIO4A/B/BX/BXR, SIO8BXS |
| SIO4_GSC_FRR | Firmware Revision Register | All models |
| SIO4_GSC_FSR | FIFO Size Reg | SIO4B/BX/BXR, SIO8BXS, PCI-SIO4A |
| SIO4_GSC_FTR | Firmware Type Register | SIO4BXR, SIO8BXS, some SIO4BX |
| SIO4_GSC_GPIOSR | GPIO Source Register | PCI-SIO4A only |
| SIO4_GSC_ICR | Interrupt Control | All models |
| SIO4_GSC_IELR | Interrupt Edge/Level Register | All but original SIO4 |
| SIO4_GSC_IHLR | Interrupt High/Low Register | All but original SIO4 |
| SIO4_GSC_IOCR | I/O Control Register | PMC-SIO4AR-SYNC only |
| SIO4_GSC_ISR | Interrupt Status | All models |
| SIO4_GSC_PCR | Prog Clock Reg | SIO4A only |
| SIO4_GSC_POCSR | Prog Osc Ctrl/Stat | SIO4B/BX/BXR, SIO8BXS |
| SIO4_GSC_PORAR | Prog Osc RAM Adrs | SIO4B/BX/BXR, SIO8BXS |
| SIO4_GSC_PORDR | Prog Osc RAM Data | SIO4B/BX/BXR, SIO8BXS |
| SIO4_GSC_PORD2R | Prog Osc RAM Data 2 | Some SIO8BXS |
| SIO4_GSC_PSRCR | Pin Source Reg | All but SIO4A-SYNC and original SIO4 |
| SIO4_GSC_PSTSR | Pin Ststus Reg | SIO4B/BX/BXR, SIO8BXS |
| SIO4_GSC_RAR | Rx Almost Register | All models |
| SIO4_GSC_RCR | Rx Count Reg | All -SYNC models |
| SIO4_GSC_SBR | Sync Byte Register | All models w/ USC |
| SIO4_GSC_TAR | Tx Almost Register | All models |
| SIO4_GSC_TCR | Tx Count Reg | All -SYNC models |
| SIO4_GSC_TSR | Time Stamp Register | SIO4BXR w/ USC |

**NOTE**: Refer to the output of the id sample application (under the `sio4\id` directory) for a complete list of the registers supported by the board being accessed.

#### 3.1.2.2. PCI Configuration Registers

The PCI registers are not listed as they are seldom required. Refer to the files `gsc_pci9056.h` and `gsc_pci9080.h` for the applicable set of PCI9080 and PCI9056 register definitions, respectively.

#### 3.1.2.3. PLX Feature Set Registers

The PLX Feature Set registers are not listed as they are seldom required. Refer to the files `gsc_pci9056.h` and `gsc_pci9080.h` for the applicable set of PCI9080 and PCI9056 register definitions, respectively.

#### 3.1.2.4. Zilog Z16C30 USC Registers

The following table gives the complete set of Zilog USC registers. These registers appear only on SIO4 models which include the Zilog Z16C30 USC chips on them. This essentially refers to all boards without the SYNC suffix in the model number.

| **Macros** | **Description** |
|---|---|
| SIO4_USC_CCAR | Channel Command/Address Register |
| SIO4_USC_CCR | Channel Control Register |
| SIO4_USC_CCSR | Channel Command/Status Register |
| SIO4_USC_CMCR | Clock Mode Control Register |
| SIO4_USC_CMR | Channel Mode Register |
| SIO4_USC_DCCR | Daisy-Chain Control Register |

| | |
|---|---|
| SIO4_USC_HCR | Hardware Configuration Register |
| SIO4_USC_ICR | Interrupt Control Register |
| SIO4_USC_IOCR | I/O Control Register |
| SIO4_USC_IVR | Interrupt Vector Register |
| SIO4_USC_MISR | Miscellaneous Interrupt Status Register |
| SIO4_USC_PRR | Primary Reserved Register |
| SIO4_USC_RCCR | Receive Character Count Register |
| SIO4_USC_RCLR | Receive Count Limit Register |
| SIO4_USC_RCSR | Receive Command Status Register |
| SIO4_USC_RDR | Receive Data Register |
| SIO4_USC_RICR | Receive Interrupt Control Register |
| SIO4_USC_RMR | Receive Mode Register |
| SIO4_USC_RSR | Receive Sync Register |
| SIO4_USC_SICR | Status Interrupt Control Register |
| SIO4_USC_SRR | Secondary Reserved Register |
| SIO4_USC_TC0R | Time Constant 0 Register |
| SIO4_USC_TC1R | Time Constant 1 Register |
| SIO4_USC_TCCR | Transmit Character Count Register |
| SIO4_USC_TCLR | Transmit Count Limit Register |
| SIO4_USC_TCSR | Transmit Command/Status Register |
| SIO4_USC_TDR | Transmit Data Register |
| SIO4_USC_TICR | Transmit Interrupt Control Register |
| SIO4_USC_TMCR | Test Mode Control Register |
| SIO4_USC_TMDR | Test Mode Data Register |
| SIO4_USC_TMR | Transmit Mode Register |
| SIO4_USC_TSR | Transmit Sync Register |

## 3.2. Functions

This driver interface includes the following functions. The functions are made available to applications via the Driver Interface Library (`sio4\docsrc\sio4_lib`, though the exact name and location are OS specific.)

### 3.2.1. sio4_close()

This function is the entry point to close a connection to an open SIO4 serial channel. This function should only be called after a successful open of the respective device.

Prototype

```
int sio4_close(int fd);
```

| Argument | Description |
|---|---|
| fd | This is the file descriptor of the device to be closed. |

| Return Value | Description |
|---|---|
| -1 | An error occurred. Consult `errno`. |
| 0 | The operation succeeded. |

Example

```
#include <errno.h>
#include <stdio.h>

#include "sio4_dsl.h"
```

```
int sio4_dsl_close(int fd)
{
    int errs;
    int sts;

    sts = sio4_close(fd);

    if (sts == -1)
    {
        printf( "ERROR: sio4_close() failure, errno = %d\n",
                errno);
    }

    errs    = (sts == -1) ? 1 : 0;
    return(errs);
}
```

### 3.2.2. sio4_ioctl()

This function is the entry point to performing setup and control operations on an open SIO4 serial channel. This function should only be called after a successful open of the respective device. The specific operation performed varies according to the `request` argument. The `request` argument also governs the use and interpretation of any additional arguments. The set of supported IOCTL services is defined in a following section.

Prototype

```
int sio4_ioctl(int fd, int cmd, void* buf);
```

| Argument | Description |
|---|---|
| `fd` | This is the file descriptor of the device to access. |
| `request` | This specifies the desired operation to be performed. |
| `buf` | This is a pointer to the variable space by which values are passed to the driver for the service and by which data is returned. |

| Return Value | Description |
|---|---|
| `-1` | An error occurred. Consult `errno`. |
| `0` | The operation succeeded. |

Example

```
#include <errno.h>
#include <stdio.h>

#include "sio4_dsl.h"

int sio4_dsl_ioctl(int fd, int cmd, void *buf)
{
    int errs;
    int sts;

    sts = sio4_ioctl(fd, cmd, buf);
```

```
    if (sts == -1)
    {
        printf( "ERROR: sio4_ioctl() failure, errno = %d\n",
                errno);
    }

    errs    = (sts == -1) ? 1 : 0;
    return(errs);
}
```

### 3.2.3. sio4_open()

This function is the entry point to open a connection to an SIO4 serial channel. Upon opening the channel, all settings and configurations are put in an initialized state.

> **NOTE:** The driver does not modify the on-board programmable oscillator as part of initializing the board when it is opened.

> **NOTE:** The SIO8 appears to the driver as two SIO4 boards.

Prototype

```
int sio4_open(int index);
```

| Argument | Description |
|---|---|
| `Index` | This is the zero based index of the board to access. |

| Return Value | Description |
|---|---|
| `-1` | An error occurred. Consult `errno`. |
| else | A valid file descriptor. |

Example

```
#include <errno.h>
#include <stdio.h>

#include "sio4_dsl.h"

int sio4_dsl_open(unsigned int index)
{
    int fd;

    fd  = sio4_open(index);

    if (fd == -1)
    {
        printf( "ERROR: sio4_open(%d) failure, errno = %d\n",
                index,
                errno);
    }
```

```
    return(fd);
}
```

### 3.2.4. sio4_read()

This function is the entry point to reading received data from an open SIO4 serial channel. This function should only be called after a successful open of the respective device. The function reads up to the specified number of bytes from the receive FIFO. If the number of bytes requested is not available within the configured time limit, the read operation times out.

Prototype

```
int sio4_read(int fd, void *dst, int bytes);
```

| Argument | Description |
|---|---|
| fd | This is the file descriptor of the device to access. |
| dst | The data read will be put here. |
| bytes | This is the desired number of bytes to read. |

| Return Value | Description |
|---|---|
| -1 | An error occurred. Consult errno. |
| 0 to bytes | The operation succeeded. If the return value is less than bytes, then the request timed out. |

Example

```
#include <errno.h>
#include <stdio.h>

#include "sio4_dsl.h"

int sio4_dsl_read(int fd, void* dst, int bytes)
{
    int sts;

    sts = sio4_read(fd, dst, bytes);

    if (sts == -1)
    {
        printf( "ERROR: sio4_read() failure, errno = %d\n",
                errno);
    }

    return(sts);
}
```

### 3.2.5. sio4_write()

This function is the entry point to writing data for transmission to an open SIO4 serial channel. This function should only be called after a successful open of the respective device. The function writes up to the specified number of

bytes to the transmit FIFO. If the number of bytes requested cannot be sent within the configured time limit, the write operation times out.

### Prototype

```
int sio4_write(int fd, const void *src, int bytes);
```

| Argument | Description |
|---|---|
| fd | This is the file descriptor of the device to access. |
| src | The data written comes from here. |
| bytes | This is the desired number of bytes to write. |

| Return Value | Description |
|---|---|
| -1 | An error occurred. Consult errno. |
| 0 to bytes | The operation succeeded. If the return value is less than bytes, then the request timed out. |

### Example

```
#include <errno.h>
#include <stdio.h>

#include "sio4_dsl.h"

int sio4_dsl_write(int fd, const void* src, int bytes)
{
    int sts;

    sts = sio4_write(fd, src, bytes);

    if (sts == -1)
    {
        printf( "ERROR: sio4_write() failure, errno = %d\n",
                errno);
    }

    return(sts);
}
```

# 4. Common IOCTL Services

This section describes the IOCTL services that are common across SYNC and Zilog based versions of the SIO4. The services are organized into groups according to their functionality and applicability.

> **NOTE**: All services whose argument data type is `s32*` accept the value of `-1` being passed to the driver. If there is a setting associated with the service, then passing in the value `-1` is a request for the current setting. If the service is supported, then the current setting is returned. If the service is not supported, then the value returned by the driver will be `-1`. If there isn't a setting associated with the service, then passing in the value `-1` is a request to determine if the service is supported. If the service is supported, then the value one is returned. If the service is not supported, then the value `-1` is returned.

## 4.1. FIFO Specific IOCTL Services

These IOCTL services relate to the operation of the Rx FIFO and the Tx FIFO.

### 4.1.1. SIO4_IOCTL_FIFO_SPACE_CFG

This service configures the relative distribution of FIFO space between the receiver and transmitter.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_FIFO_SPACE_CFG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_FIFO_SPACE_CFG_RX_2X | This refers to the Rx FIFO being twice the size of the Tx FIFO. | |
| SIO4_FIFO_SPACE_CFG_TX_2X | This refers to the Tx FIFO being twice the size of the Rx FIFO. | |

### 4.1.2. SIO4_IOCTL_RX_FIFO_AE

This service configures the Rx FIFO Almost Empty threshold level. When applying a setting the FIFO is reset and the current content is lost. The Rx FIFO Almost Empty status is asserted when the Rx FIFO contains *Almost Empty* or fewer data values.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_RX_FIFO_AE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

### 4.1.3. SIO4_IOCTL_RX_FIFO_AF

This service configures the Rx FIFO Almost Full threshold level. When applying a setting the Rx FIFO is reset and the current content is lost. The Rx FIFO Almost Full status is asserted when the Rx FIFO can receive *Almost Full* or fewer data values before becoming full.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | See table above. |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

### 4.1.4. SIO4_IOCTL_RX_FIFO_FILL_LEVEL

This service retrieves the current Rx FIFO fill level. If the fill level cannot be determined exactly, then it is approximated based on the FIFO status.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_RX_FIFO_FILL_LEVEL |
| arg | s32* |

The table below lists the options used with this service. Values returned are from zero to 0xFFFF, but will not exceed the size of the FIFO.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

### 4.1.5. SIO4_IOCTL_RX_FIFO_FULL_CFG

This service configures how a receiver responds to an Rx FIFO Full condition.

> **NOTE:** This service refers to the channel specific settings, not to the global setting. The channel specific settings are ignored of the global setting is set to the *disable* option.
>
> **NOTE:** If the *disable* setting is selected and the feature becomes activated, then this may indicate that an Rx FIFO overflow has also occurred.
>
> **NOTE:** If the *disable* setting is selected and the feature causes the receiver to become disabled, then the application must implement a recovery procedure. For –SYNC boards this means reading data from the Rx FIFO and re-enabling the receiver. For this, refer to the SIO4_IOCTL_SYNC_RX_ENABLE service (section 7.4.5, page 112). For Z16C30 boards this means reading data from the Rx FIFO, flushing the USC's Rx FIFO and re-enabling the USC receiver. For this, refer to the SIO4_IOCTL_USC_SEND_COMMAND service (section 6.17.9, page 94, use the SIO4_USC_SEND_CMD_RX_FIFO_PURGE command), and the SIO4_IOCTL_USC_RX_ENABLE service (section 6.18.7, page 97).

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_RX_FIFO_FULL_CFG |
| arg | s32* |

The table below lists the options used by this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_RX_FIFO_FULL_CFG_DISABLE | This refers to disabling the receiver. Once the receiver is disabled, it must be manually enabled. | |
| SIO4_RX_FIFO_FULL_CFG_OVER | This refers to letting the FIFO overrun. | |

### 4.1.6. SIO4_IOCTL_RX_FIFO_FULL_CFG_GLB

This service configures how the receivers respond to an Rx FIFO Full condition.

**NOTE:** This is a global setting that affects all four channels.

**NOTE:** If the *disable* option is selected, then it overrides the channel specific setting.

**NOTE:** If the *disable* setting is selected and the feature becomes activated, then this may indicate that an Rx FIFO overflow has also occurred.

**NOTE:** If the *disable* setting is selected and the feature causes the receiver to become disabled, then the application must implement a recovery procedure. For –SYNC boards this means reading data from the Rx FIFO and re-enabling the receiver. For this, refer to the SIO4_IOCTL_SYNC_RX_ENABLE service (section 7.4.5, page 112). For Z16C30 boards this means reading data from the Rx FIFO, flushing the USC's Rx FIFO and re-enabling the USC receiver. For this, refer to the SIO4_IOCTL_USC_SEND_COMMAND service (section 6.17.9, page 94, use the SIO4_USC_SEND_CMD_RX_FIFO_PURGE command), and the SIO4_IOCTL_USC_RX_ENABLE service (section 6.18.7, page 97).

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_RX_FIFO_FULL_CFG_GLB |
| arg | s32* |

The table below lists the options used by this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_RX_FIFO_FULL_CFG_GLB_DISABLE | This refers to disabling the receiver. Once a receiver is disabled, it must be manually enabled. If selected, then this overrides the channel specific settings. | |
| SIO4_RX_FIFO_FULL_CFG_GLB_OVER | This refers to letting the FIFO overrun. | |

### 4.1.7. SIO4_IOCTL_RX_FIFO_OVERRUN

This service operates on the Rx FIFO Overrun condition.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_RX_FIFO_OVERRUN |
| arg | s32* |

The table below lists the options used by this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_FIFO_OVERRUN_CLEAR | This clears the overrun status. | N/A |
| SIO4_FIFO_OVERRUN_TEST | This checks the overrun status. | N/A |
| SIO4_FIFO_OVERRUN_NO | N/A | This indicates that an overrun has not occurred. |
| SIO4_FIFO_OVERRUN_YES | N/A | This indicates that an overrun has occurred. |

### 4.1.8. SIO4_IOCTL_RX_FIFO_RESET

This service resets the Rx FIFO. The FIFO content is lost when it is reset.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_RX_FIFO_RESET |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests support status. | The service is unsupported. |
| 0 | Reset the FIFO. | The service is supported or the FIFO was reset. |

### 4.1.9. SIO4_IOCTL_RX_FIFO_STATUS

This service reports the Rx FIFO Fill Level relative to the Rx FIFO Threshold Level.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_RX_FIFO_STATUS |
| arg | s32* |

The table below lists the options used by this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests | The service is unsupported. |

| | | |
|---|---|---|
| | current setting. | |
| SIO4_FIFO_STATUS_EMPTY | N/A | The FIFO is empty. |
| SIO4_FIFO_STATUS_ALMOST_EMPTY | N/A | The FIFO is Almost Empty. The FIFO contains *Almost Empty* or fewer data values (see section 4.1.2, page 21). |
| SIO4_FIFO_STATUS_MEDIUM | N/A | The FIFO is between Almost Empty and Almost Full. |
| SIO4_FIFO_STATUS_ALMOST_FULL | N/A | The FIFO is Almost Full. The FIFO can receive *Almost Full* or fewer data value before becoming full (see section 4.1.3, page 22). |
| SIO4_FIFO_STATUS_FULL | N/A | The FIFO is full. |

### 4.1.10. SIO4_IOCTL_RX_FIFO_UNDERRUN

This service operates on the Rx FIFO Underrun condition.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_RX_FIFO_UNDERRUN |
| arg | s32* |

The table below lists the options used by this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_FIFO_UNDERRUN_CLEAR | This clears the underrun status. | N/A |
| SIO4_FIFO_UNDERRUN_TEST | This checks the underrun status. | N/A |
| SIO4_FIFO_UNDERRUN_NO | N/A | Indicates that an underrun has not occurred. |
| SIO4_FIFO_UNDERRUN_YES | N/A | Indicates that an underrun has occurred. |

### 4.1.11. SIO4_IOCTL_TX_FIFO_AE

This service configures the Tx FIFO Almost Empty threshold level. When applying a setting the FIFO is reset and the current content is lost. The Tx Almost Empty status is asserted when the Tx FIFO contains *Almost Empty* or fewer data values.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_TX_FIFO_AE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

### 4.1.12. SIO4_IOCTL_TX_FIFO_AF

This service configures the Tx FIFO Almost Full threshold level. When applying a setting the Tx FIFO is reset and the current content is lost. The Tx Almost Full status is asserted when the Tx FIFO can receive *Almost Full* or fewer data values before becoming full.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_TX_FIFO_AF |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

### 4.1.13. SIO4_IOCTL_TX_FIFO_EMPTY_CFG

This service configures how the transmitter responds to a Tx FIFO Empty condition.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_TX_FIFO_EMPTY_CFG |
| arg | s32* |

The table below lists the options used by this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_TX_FIFO_EMPTY_CFG_IGNORE | This refers to ignoring the condition. | |
| SIO4_TX_FIFO_EMPTY_CFG_TX_OFF | This refers to disabling the transmitter. | |

### 4.1.14. SIO4_IOCTL_TX_FIFO_FILL_LEVEL

This service retrieves the current Tx FIFO fill level. If the fill level cannot be determined exactly, then it is approximated based on the FIFO status.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_TX_FIFO_FILL_LEVEL |
| arg | s32* |

The table below lists the options used with this service. Values returned will not exceed the size of the FIFO.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | N/A | This is the valid range for this service. |

### 4.1.15. SIO4_IOCTL_TX_FIFO_OVERRUN

This service operates on the Tx FIFO Overrun condition.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_TX_FIFO_OVERRUN |
| arg | s32* |

The table below lists the options used by this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_FIFO_OVERRUN_CLEAR | This clears the overrun status. | N/A |
| SIO4_FIFO_OVERRUN_TEST | This checks the overrun status. | N/A |
| SIO4_FIFO_OVERRUN_NO | N/A | Indicates that an overrun has not occurred. |
| SIO4_FIFO_OVERRUN_YES | N/A | Indicates that an overrun has occurred. |

### 4.1.16. SIO4_IOCTL_TX_FIFO_RESET

This service resets the Tx FIFO. The FIFO content is lost when it is reset.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_TX_FIFO_RESET |
| arg | s32* |

The table below lists the options used by this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests support status. | The service is unsupported. |
| 0 | Reset the FIFO. | The service is supported or the FIFO was reset. |

### 4.1.17. SIO4_IOCTL_TX_FIFO_STATUS

This service reports the Tx FIFO Fill Level relative to the Tx FIFO Threshold Level.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_TX_FIFO_STATUS |
| arg | s32* |

The table below lists the options used by this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_FIFO_STATUS_EMPTY | N/A | The FIFO is empty. |
| SIO4_FIFO_STATUS_ALMOST_EMPTY | N/A | The FIFO is Almost Empty. The FIFO contains *Almost Empty* or fewer data values (see section 4.1.11, page 25). |
| SIO4_FIFO_STATUS_MEDIUM | N/A | The FIFO is between Almost Empty and Almost Full. Both the Almost Full and the Almost Empty status bits are set. |
| SIO4_FIFO_STATUS_ALMOST_FULL | N/A | The FIFO is Almost Full. The FIFO can receive *Almost Full* or fewer data values before becoming full (see section 4.1.12, page 26). |
| SIO4_FIFO_STATUS_FULL | N/A | The FIFO is full. |

## 4.2. I/O Specific IOCTL Services

These IOCTL services relate to the I/O operation when transferring data either to or from the SIO4.

### 4.2.1. SIO4_IOCTL_RX_IO_ABORT

This service aborts an active read request, if one is active.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_RX_IO_ABORT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_IO_ABORT_NO | This refers to the I/O request not being aborted. | |
| SIO4_IO_ABORT_YES | This refers to the I/O request being aborted, or the service is supported. | |

### 4.2.2. SIO4_IOCTL_RX_IO_MODE

This service configures the I/O mode used to transfer data to the board during read requests.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_RX_IO_MODE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |

| | |
|---|---|
| GSC_IO_MODE_DMA | This refers to non-Demand Mode DMA. |
| GSC_IO_MODE_DMDMA | This refers to Demand Mode DMA. |
| GSC_IO_MODE_PIO | This refers to PIO. This is the default. |

### 4.2.3. SIO4_IOCTL_RX_IO_OVERRUN

This service configures how the read service responds to overrun conditions.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_RX_IO_OVERRUN |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_IO_OVERRUN_CHECK | This refers to checking for overruns before performing I/O. | |
| SIO4_IO_OVERRUN_IGNORE | This refers to ignoring the overrun status. | |

### 4.2.4. SIO4_IOCTL_RX_IO_PIO_THRESHOLD

This service configures the read request size below which requests are automatically performed using PIO.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_RX_IO_PIO_THRESHOLD |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFFFFF | This is the valid range for this service. | |

### 4.2.5. SIO4_IOCTL_RX_IO_TIMEOUT

This service configures the read timeout period. When the duration of an I/O request takes this number of seconds or longer, the request will end. A setting of zero should be used with PIO only.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_RX_IO_TIMEOUT |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 3600 | This is the valid range for this service. | |

### 4.2.6. SIO4_IOCTL_RX_IO_UNDERRUN

This service configures how read requests will respond to Rx FIFO Underrun conditions.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_RX_IO_UNDERRUN |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_IO_UNDERRUN_CHECK | This refers to check for underruns before performing I/O. | |
| SIO4_IO_UNDERRUN_IGNORE | This refers to ignoring the underrun status. | |

### 4.2.7. SIO4_IOCTL_TX_IO_ABORT

This service aborts an active write request, if one is active.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_TX_IO_ABORT |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_IO_ABORT_NO | This refers to the I/O request not being aborted. | |
| SIO4_IO_ABORT_YES | This refers to the I/O request being aborted, or the service is supported. | |

### 4.2.8. SIO4_IOCTL_TX_IO_MODE

This service configures the I/O mode used to transfer data from the board during write requests.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_TX_IO_MODE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| GSC_IO_MODE_DMA | This refers to non-Demand Mode DMA. | |
| GSC_IO_MODE_DMDMA | This refers to Demand Mode DMA. | |
| GSC_IO_MODE_PIO | This refers to PIO. This is the default. | |

#### 4.2.9. SIO4_IOCTL_TX_IO_OVERRUN

This service configures how the write service responds to overrun conditions.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_TX_IO_OVERRUN |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_IO_OVERRUN_CHECK | This refers to check for overruns before performing I/O. | |
| SIO4_IO_OVERRUN_IGNORE | This refers to ignoring the overrun status. | |

#### 4.2.10. SIO4_IOCTL_TX_IO_PIO_THRESHOLD

This service configures the write request size below which requests are automatically performed using PIO.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_TX_IO_PIO_THRESHOLD |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFFFFF | This is the valid range for this service. | |

#### 4.2.11. SIO4_IOCTL_TX_IO_TIMEOUT

This service configures the write timeout period. When the duration of an I/O request takes this number of seconds or longer, the request will end. A setting of zero should be used with PIO only.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_TX_IO_TIMEOUT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 3600 | This is the valid range for this service. | |

## 4.3. Interrupt Specific IOCTL Services

These IOCTL services relate to the interrupt operation and detection.

### 4.3.1. SIO4_IOCTL_IRQ_GSC_CFG_HIGH

This service configures the triggering polarity of the firmware specific interrupts.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_IRQ_GSC_CFG_HIGH |
| arg | s32* |

The table below lists the options used with this service. Valid arguments values may include any combination of the below macros, or none at all. If a bit is set, then the corresponding interrupt is configured to trigger on a level-high or high going state. If a bit is clear, then the corresponding interrupt is configured to trigger on a level-low or low going state.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_IRQ_RX_ENV | This refers to the interrupt that responds to the Rx Envelope signal. (-SYNC model boards only) | |
| SIO4_IRQ_RX_FIFO_AF | This refers to the interrupt that responds to the Rx FIFO Almost Full state. | |
| SIO4_IRQ_RX_FIFO_E | This refers to the interrupt that response to the Rx FIFO Empty state. | |
| SIO4_IRQ_RX_FIFO_F | This refers to the interrupt that response to the Rx FIFO Full state. | |
| SIO4_IRQ_RX_SPARE | This refers to the interrupt that response to the Rx Spare signal. (-SYNC model boards only) | |
| SIO4_IRQ_SYNC_BYTE | This refers to the interrupt that responds to received characters matching the configured SYNC byte. (Z16C30 model boards only) | |
| SIO4_IRQ_TX_FIFO_AE | This refers to the interrupt that responds to the Tx FIFO Almost Empty state. | |
| SIO4_IRQ_TX_FIFO_E | This refers to the interrupt that responds to the Tx FIFO Empty state. | |
| SIO4_IRQ_TX_FIFO_F | This refers to the interrupt that responds to the Tx FIFO Full state. | |
| SIO4_IRQ_USC | This refers to the interrupt that response to USC generated interrupts. (Z16C30 model boards only) | |

### 4.3.2. SIO4_IOCTL_IRQ_GSC_ENABLE

This service enables or disables the firmware specific interrupts.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_IRQ_GSC_ENABLE |
| arg | s32* |

The table below lists the options used with this service. Valid arguments values may include any combination of the below macros, or none at all. If a bit is set, then the corresponding interrupt is enabled. If a bit is clear, then the corresponding interrupt is disabled.

> **NOTE**: Level triggered interrupts are disabled when they occurs (except for the USC interrupt). Edge triggered interrupts are not disabled when they occur.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_IRQ_RX_ENV | This refers to the interrupt that responds to the Rx Envelope signal. (-SYNC model boards only) | |

| | |
|---|---|
| SIO4_IRQ_RX_FIFO_AF | This refers to the interrupt that responds to the Rx FIFO Almost Full state. |
| SIO4_IRQ_RX_FIFO_E | This refers to the interrupt that response to the Rx FIFO Empty state. |
| SIO4_IRQ_RX_FIFO_F | This refers to the interrupt that response to the Rx FIFO Full state. |
| SIO4_IRQ_RX_SPARE | This refers to the interrupt that response to the Rx Spare signal. (-SYNC model boards only) |
| SIO4_IRQ_SYNC_BYTE | This refers to the interrupt that responds to received characters matching the configured SYNC byte. (Z16C30 model boards only) |
| SIO4_IRQ_TX_FIFO_AE | This refers to the interrupt that responds to the Tx FIFO Almost Empty state. |
| SIO4_IRQ_TX_FIFO_E | This refers to the interrupt that responds to the Tx FIFO Empty state. |
| SIO4_IRQ_TX_FIFO_F | This refers to the interrupt that responds to the Tx FIFO Full state. |
| SIO4_IRQ_USC | This refers to the interrupt that response to USC generated interrupts. (Z16C30 model boards only) |

### 4.3.3. SIO4_IOCTL_IRQ_USC_ENABLE

This service enables or disables USC specific interrupts.

#### Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_IRQ_USC_ENABLE |
| arg | s32* |

The table below lists the options used with this service. Valid arguments values may include any combination of the below macros, or none at all. If a bit is set, then the corresponding interrupt is enabled. If a bit is clear, then the corresponding interrupt is disabled.

> **NOTE**: Use care when enabling interrupts driven by clock signals. If the clock is present, then the influx of interrupts could make the system unresponsive.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_IRQ_USC_IOP_CTS_FALL | This refers to the interrupt that responds to a high-to-low transition on the USC's CTS pin. | |
| SIO4_IRQ_USC_IOP_CTS_RISE | This refers to the interrupt that responds to a low-to-high transition on the USC's CTS pin. | |
| SIO4_IRQ_USC_IOP_DCD_FALL | This refers to the interrupt that responds to a high-to-low transition on the USC's DCD pin. | |
| SIO4_IRQ_USC_IOP_DCD_RISE | This refers to the interrupt that responds to a low-to-high transition on the USC's DCD pin. | |
| SIO4_IRQ_USC_IOP_RXC_FALL | This refers to the interrupt that responds to a high-to-low transition on the USC's RxC pin. * | |
| SIO4_IRQ_USC_IOP_RXC_RISE | This refers to the interrupt that responds to a low-to-high transition on the USC's RxC pin. * | |
| SIO4_IRQ_USC_IOP_RXREQ_FALL | This refers to the interrupt that responds to a high-to-low transition on the USC's RxREQ pin. * | |
| SIO4_IRQ_USC_IOP_RXREQ_RISE | This refers to the interrupt that responds to a low-to-high transition on the USC's RxREQ pin. * | |
| SIO4_IRQ_USC_IOP_TXC_FALL | This refers to the interrupt that responds to a high-to-low transition on the USC's TxC pin. * | |
| SIO4_IRQ_USC_IOP_TXC_RISE | This refers to the interrupt that responds to a low-to-high transition on the USC's TxC pin. * | |
| SIO4_IRQ_USC_IOP_TXREQ_FALL | This refers to the interrupt that responds to a high-to-low | |

| | |
|---|---|
| | transition on the USC's TxReq pin. * |
| SIO4_IRQ_USC_IOP_TXREQ_RISE | This refers to the interrupt that responds to a low-to-high transition on the USC's TxReq pin. * |
| SIO4_IRQ_USC_MISC_BRG0_ZERO | This refers to the interrupt that responds to the BRG0 clocking signal. |
| SIO4_IRQ_USC_MISC_BRG1_ZERO | This refers to the interrupt that responds to the BRG1 clocking signal. |
| SIO4_IRQ_USC_MISC_DPLL_DESYNC | This refers to the interrupt that responds to the DPLL becoming unsynchronized. |
| SIO4_IRQ_USC_MISC_RCC_UNDERRUN | This refers to the interrupt that responds to an underrun of the RCC FIFO. |
| SIO4_IRQ_USC_RX_ABORT_PAR_ERROR | This refers to the interrupt that responds to the condition where the next character to be taken from the USC FIFO has the Rx Abort status or the Parity Error status. |
| SIO4_IRQ_USC_RX_BOUND | This refers to the interrupt that responds to the condition where the next character to be taken from the USC FIFO has the Rx Bound status. |
| SIO4_IRQ_USC_RX_BREAK_ABORT | This refers to the interrupt that responds to the condition where the next character to be taken from the USC FIFO has the Rx Break or Rx Abort status. |
| SIO4_IRQ_USC_RX_DATA | This refers to the interrupt that responds to the receipt of a character. * |
| SIO4_IRQ_USC_RX_EXITED_HUNT | This refers to the interrupt that responds to the condition where the receiver has exited the Hunt mode. |
| SIO4_IRQ_USC_RX_IDLE_RECEIVED | This refers to the interrupt that responds to the condition where the next character to be taken from the USC FIFO has the Rx Idle status. |
| SIO4_IRQ_USC_RX_OVERRUN | This refers to the interrupt that responds to the condition where a character has been received for placement in the USC's Rx FIFO, but the FIFO is full. |
| SIO4_IRQ_USC_TX_ABORT_SENT | This refers to the interrupt that responds to the condition where the transmitter has sent out Abort. |
| SIO4_IRQ_USC_TX_CRC_SENT | This refers to the interrupt that responds to the condition where the transmitter has sent out a CRC. |
| SIO4_IRQ_USC_TX_DATA | This refers to the interrupt generated when the transmitter has placed data in the USC Tx FIFO. * |
| SIO4_IRQ_USC_TX_END_SENT | This refers to the interrupt that responds to the condition where the transmitter has sent out the end of a message or frame. |
| SIO4_IRQ_USC_TX_IDLE_SENT | This refers to the interrupt that responds to the condition where the transmitter has sent out an Idle condition. |
| SIO4_IRQ_USC_TX_PREAMBLE_SENT | This refers to the interrupt that responds to the condition where the transmitter has sent out the Preamble. |
| SIO4_IRQ_USC_TX_UNDERRUN | This refers to the interrupt that responds to the condition where the transmitter is ready for data, but the USC Tx FIFO is empty. |

* These interrupts are disabled when they occur due to the high rate expected for these interrupt source.

### 4.3.4. SIO4_IOCTL_WAIT_CANCEL

This service resumes all threads blocked via SIO4_IOCTL_WAIT_EVENT IOCTL calls (section 4.3.5, page 35), according to the provided criteria. When a blocked thread is waiting for any event specified in the structure, then the thread is resumed.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_WAIT_CANCEL |
| arg | gsc_wait_t* |

Definition

```
typedef struct
{
    u32  flags;
    u32  main;
    u32  gsc;
    u32  alt;
    u32  io;
    u32  timeout_ms;
    u32  count;
} gsc_wait_t;
```

| Fields | Description |
|---|---|
| flags | This is unused by wait cancel operations. |
| main | This specifies the set of GSC_WAIT_MAIN_* events whose wait requests are to be cancelled. Refer to section 4.3.5.2 on page 36. |
| gsc | This specifies the set of SIO4_WAIT_GSC_* events whose wait requests are to be cancelled. Refer to section 4.3.5.3 on page 36. |
| alt | This specifies the set of SIO4_WAIT_USC_* events whose wait requests are to be cancelled. Refer to section 4.3.5.4 on page 37. |
| io | This specifies the set of GSC_WAIT_IO_* events whose wait requests are to be cancelled. Refer to section 4.3.5.5 on page 38. |
| timeout_ms | This is unused by wait cancel operations. |
| count | Upon return this indicates the number of waits that were cancelled. |

### 4.3.5. SIO4_IOCTL_WAIT_EVENT

This service blocks a thread until any one of a specified set of events occurs, or until a timeout lapses, whichever occurs first. The set of possible events to wait for are specified in the structure's main, gsc, alt and io fields. All field values must be valid and at least one event must be specified. If the thread is resumed because one of the referenced events has occurred, then the bit for the respective event is the only event bit that will be set. All other event bits and fields will be zero. (Multiple event bits will be set only if the events occur simultaneously.)

> **NOTE:** A wait timeout is reported via the gsc_wait_t structure's flags field having the GSC_WAIT_FLAG_TIMEOUT flag set, rather than via an ETIMEDOUT error.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_WAIT_EVENT |
| arg | gsc_wait_t* |

Definition

```
typedef struct
{
    u32  flags;
```

```
    u32  main;
    u32  gsc;
    u32  alt;
    u32  io;
    u32  timeout_ms;
    u32  count;
} gsc_wait_t;
```

| Fields | Description |
|---|---|
| flags | This must initially be zero. Upon return this indicates the reason that the thread was resumed. Refer to section 4.3.5.1 on page 36. |
| main | This specifies the set of GSC_WAIT_MAIN_* events whose wait requests are to be cancelled. Refer to section 4.3.5.2 on page 36. |
| gsc | This specifies the set of SIO4_WAIT_GSC_* events whose wait requests are to be cancelled. Refer to section 4.3.5.3 on page 36. |
| alt | This specifies the set of SIO4_WAIT_USC_* events whose wait requests are to be cancelled. Refer to section 4.3.5.4 on page 37. |
| io | This specifies the set of GSC_WAIT_IO_* events whose wait requests are to be cancelled. Refer to section 4.3.5.5 on page 38. |
| timeout_ms | This specified the maximum amount of time, in milliseconds, that the thread is to wait for any of the referenced events. This must be greater than zero. Upon return the value will be the approximate amount of time actually waited. |
| count | This is unused by wait event operations and must be zero. |

#### 4.3.5.1. gsc_wait_t.flags Options

Upon return from a wait request the wait structure's flags field will indicate the reason that the thread was resumed. Only one of the below option will be set.

| Values | Description |
|---|---|
| GSC_WAIT_FLAG_CANCEL | The wait request was cancelled. |
| GSC_WAIT_FLAG_DONE | One of the referenced events occurred. |
| GSC_WAIT_FLAG_TIMEOUT | The timeout period lapsed before a referenced event occurred. |

#### 4.3.5.2. gsc_wait_t.main Options

The wait structure's main field may specify any of the below primary interrupt options. These interrupt options are supported by the 24DSI and other General Standards products.

| Values | Description |
|---|---|
| GSC_WAIT_MAIN_DMA0 | This refers to the DMA Done interrupt on DMA engine number zero. |
| GSC_WAIT_MAIN_DMA1 | This refers to the DMA Done interrupt on DMA engine number one. |
| GSC_WAIT_MAIN_GSC | This refers to any of the Interrupt Control/Status Register interrupts. |
| GSC_WAIT_MAIN_OTHER | This generally refers to an interrupt generated by another device sharing the same interrupt as the SIO4. |
| GSC_WAIT_MAIN_PCI | This refers to any interrupt generated by the SIO4. |
| GSC_WAIT_MAIN_SPURIOUS | This refers to board interrupts which should never be generated. |
| GSC_WAIT_MAIN_UNKNOWN | This refers to board interrupts whose source could not be identified. |

#### 4.3.5.3. gsc_wait_t.gsc Options

The wait structure's gsc field may specify any combination of the below interrupt options. These are the interrupt options referenced in the Interrupt Control Register. Applications are responsible for selecting the desired interrupt options. Refer to SIO4_IOCTL_IRQ_GSC_ENABLE (section 0, page 32).

| Values | Description |
|---|---|
| SIO4_WAIT_GSC_RX_ENV | This refers to the interrupt that responds to the Rx Envelope signal. (-SYNC model boards only) |
| SIO4_WAIT_GSC_RX_FIFO_AF | This refers to the interrupt that responds to the Rx FIFO Almost Full state. |
| SIO4_WAIT_GSC_RX_FIFO_E | This refers to the interrupt that response to the Rx FIFO Empty state. |
| SIO4_WAIT_GSC_RX_FIFO_F | This refers to the interrupt that response to the Rx FIFO Full state. |
| SIO4_WAIT_GSC_RX_SPARE | This refers to the interrupt that response to the Rx Spare signal. (-SYNC model boards only) |
| SIO4_WAIT_GSC_SYNC_BYTE | This refers to the interrupt that responds to received characters matching the configured SYNC byte. (Z16C30 model boards only) |
| SIO4_WAIT_GSC_TX_FIFO_AE | This refers to the interrupt that responds to the Tx FIFO Almost Empty state. |
| SIO4_WAIT_GSC_TX_FIFO_E | This refers to the interrupt that responds to the Tx FIFO Empty state. |
| SIO4_WAIT_GSC_TX_FIFO_F | This refers to the interrupt that responds to the Tx FIFO Full state. |
| SIO4_WAIT_GSC_USC | This refers to the interrupt that response to USC generated interrupts. (Z16C30 model boards only) |

#### 4.3.5.4. gsc_wait_t.alt Options

The wait structure's alt field may specify any combination of the below interrupt options. These are the interrupt options generated by the USC of non-SYNC board only. Applications are responsible for selecting the desired interrupt options. Refer to SIO4_IOCTL_IRQ_USC_ENABLE (section 0, page 32).

| Values | Description |
|---|---|
| SIO4_WAIT_USC_IOP_CTS_FALL | This refers to the interrupt generated when the USC CTS pin goes from a high to a low state. |
| SIO4_WAIT_USC_IOP_CTS_RISE | This refers to the interrupt generated when the USC CTS pin goes from a low to a high state. |
| SIO4_WAIT_USC_IOP_DCD_FALL | This refers to the interrupt generated when the USC DCD pin goes from a high to a low state. |
| SIO4_WAIT_USC_IOP_DCD_RISE | This refers to the interrupt generated when the USC DCD pin goes from a low to a high state. |
| SIO4_WAIT_USC_IOP_RXC_FALL | This refers to the interrupt generated when the USC Rx Clock pin goes from a high to a low state. |
| SIO4_WAIT_USC_IOP_RXC_RISE | This refers to the interrupt generated when the USC Rx Clock pin goes from a low to a high state. |
| SIO4_WAIT_USC_IOP_RXREQ_FALL | This refers to the interrupt generated when the USC Rx Request pin goes from a high to a low state. * |
| SIO4_WAIT_USC_IOP_RXREQ_RISE | This refers to the interrupt generated when the USC Rx Request pin goes from a low to a high state. * |
| SIO4_WAIT_USC_IOP_TXC_FALL | This refers to the interrupt generated when the USC Tx Clock pin goes from a high to a low state. |
| SIO4_WAIT_USC_IOP_TXC_RISE | This refers to the interrupt generated when the USC Tx Clock pin goes from a low to a high state. |
| SIO4_WAIT_USC_IOP_TXREQ_FALL | This refers to the interrupt generated when the USC Rx Request pin goes from a high to a low state. † |
| SIO4_WAIT_USC_IOP_TXREQ_RISE | This refers to the interrupt generated when the USC Rx Request pin goes from a low to a high state. † |
| SIO4_WAIT_USC_MISC_BRG0_ZERO | This refers to the interrupt generated when BRG0 counts down to zero. |
| SIO4_WAIT_USC_MISC_BRG1_ZERO | This refers to the interrupt generated when BRG1 counts down to zero. |
| SIO4_WAIT_USC_MISC_DPLL_DESYNC | This refers to the interrupt generated when the DPLL looses sync. |
| SIO4_WAIT_USC_MISC_RCC_UNDERRUN | This refers to the interrupt generated when the RCC value is |

| | |
|---|---|
| | read, but the RCC FIFO is empty. |
| SIO4_WAIT_USC_RX_ABORT_PAR_ERROR | This refers to the interrupt generated when the receiver detects an abort over the cable interface or when it detects a parity error in the received data. |
| SIO4_WAIT_USC_RX_BOUND | This refers to the interrupt generated when a character marked with the Rx Bound status is removed from the USC Rx FIFO. |
| SIO4_WAIT_USC_RX_BREAK_ABORT | This refers to the interrupt generated when the receiver detects Break (asynchronous modes) or Abort condition (HDLC). |
| SIO4_WAIT_USC_RX_DATA | This refers to the interrupt generated when a received character brings the Rx FIFO fill level above the specified threshold level. |
| SIO4_WAIT_USC_RX_EXITED_HUNT | This refers to the interrupt generated when the receiver exits Hunt mode. |
| SIO4_WAIT_USC_RX_IDLE_RECEIVED | This refers to the interrupt generated when the receiver detects 15 or 16 consecutive ones (HDLC). |
| SIO4_WAIT_USC_RX_OVERRUN | This refers to the interrupt generated when a character marked with the Rx Overrun status is removed from the USC Rx FIFO. |
| SIO4_WAIT_USC_SPURIOUS | This refers to the condition where a USC interrupt is generated, but its source cannot be determined. |
| SIO4_WAIT_USC_TX_ABORT_SENT | This refers to the interrupt generated when the transmitter has sent out an Abort character (HDLC). |
| SIO4_WAIT_USC_TX_CRC_SENT | This refers to the interrupt generated when the transmitter has sent the CRC that ends a frame or message. |
| SIO4_WAIT_USC_TX_DATA | This refers to the interrupt generated when the number of empty spaces in the USC Tx FIFO is above the specified threshold level. |
| SIO4_WAIT_USC_TX_END_SENT | This refers to the interrupt generated when the transmitter has sent out the closing flag or SYNC sequence that ends a frame or message. |
| SIO4_WAIT_USC_TX_IDLE_SENT | This refers to the interrupt generated when the transmitter has sent out the Idle condition or SYNC characters. |
| SIO4_WAIT_USC_TX_PREAMBLE_SENT | This refers to the interrupt generated when the transmitter has sent out the Preamble (asynchronous modes). |
| SIO4_WAIT_USC_TX_UNDERRUN | This refers the interrupt generated when the USC transmitter needs another character to send, but the USC Tx FIFO is empty. |

* This pin is automatically configured by the driver to facilitate transfer of data from the USC receiver to the Rx FIFO.

† This pin is automatically configured by the driver to facilitate transfer of data from the Tx FIFO to the USC transmitter.

#### 4.3.5.5. `gsc_wait_t.io` Options

The wait structure's `io` field may specify any of the below event options. These events are generated in response to application board data read requests.

| Values | Description |
|---|---|
| GSC_WAIT_IO_RX_ABORT | This refers to read requests which have been aborted. |
| GSC_WAIT_IO_RX_DONE | This refers to read requests which have been satisfied. |
| GSC_WAIT_IO_RX_ERROR | This refers to read requests which end due to an error. |
| GSC_WAIT_IO_RX_TIMEOUT | This refers to read requests which end due to the timeout period lapse. |
| GSC_WAIT_IO_TX_ABORT | This refers to write requests which have been aborted. |
| GSC_WAIT_IO_TX_DONE | This refers to write requests which have been satisfied. |
| GSC_WAIT_IO_TX_ERROR | This refers to write requests which end due to an error. |
| GSC_WAIT_IO_TX_TIMEOUT | This refers to write requests which end due to the timeout period lapse. |

### 4.3.6. SIO4_IOCTL_WAIT_STATUS

This service count all threads blocked via the SIO4_IOCTL_WAIT_EVENT IOCTL service (section 4.3.5, page 35), according to the provided criteria. A match is made when a waiting thread's wait criteria matches any of the criteria specified in the structure passed to this service.

> **NOTE:** The driver itself makes use of the wait services for various internal operations. Driver initiated waits are ignored by application status requests.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_WAIT_STATUS |
| arg | gsc_wait_t* |

Definition

```
typedef struct
{
    u32  flags;
    u32  main;
    u32  gsc;
    u32  alt;
    u32  io;
    u32  timeout_ms;
    u32  count;
} gsc_wait_t;
```

| Fields | Description |
|---|---|
| flags | This is unused by wait status operations. |
| main | This specifies the set of GSC_WAIT_MAIN_* events whose wait requests are to be counted. Refer to section 4.3.5.2 on page 36. |
| gsc | This specifies the set of DSI_WAIT_GSC_* events whose wait requests are to be counted. Refer to section 4.3.5.3 on page 36. |
| alt | This specifies the set of SIO4_WAIT_USC_* events whose wait requests are to be counted. Refer to section 4.3.5.3 on page 36. |
| io | This specifies the set of GSC_WAIT_IO_* events whose wait requests are to be counted. Refer to section 4.3.5.4 on page 37. |
| timeout_ms | This is unused by wait status operations. |
| count | Upon return this indicates the number of waits that met any of the specified criteria. |

## 4.4. Miscellaneous IOCTL Services

### 4.4.1. SIO4_IOCTL_CBL_MODE

This service configures the routing of the signals at the cable interface.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_CBL_MODE |
| arg | s32* |

The table below lists the options used with this service.

**NOTE**: Use care when enabling interrupts driven by clock signals. If the clock is present, then the influx of interrupts could make the system unresponsive.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_CBL_MODE_DCE | This refers to routing the signals for the DCE configuration. | |
| SIO4_CBL_MODE_DTE | This refers to routing the signals for the DTE configuration. | |

### 4.4.2. SIO4_IOCTL_CBL_PIN_STATUS

This service returns the state of the signals at the cable interface. The state is provided by reading the Pin Status Register. Bits that do not reflect cable pin status are masked off and returned as zero.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_CBL_PIN_STATUS |
| arg | s32* |

The table below lists the options used with this service. The valid set of bits varies with different models and different firmware versions.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 or above | N/A | The status requested. |

### 4.4.3. SIO4_IOCTL_INITIALIZE

This service initializes a channel to the state it was in when the channel was first opened. All channel hardware and software settings are initialized. The programmable oscillator is unaffected.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_INITIALIZE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests support status. | The service is unsupported. |
| 1 | Perform initialization. | The service is supported or initialization was performed. |

### 4.4.4. SIO4_IOCTL_LED_CHANNEL

This service controls the LEDs that correspond to the channel being accessed.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_LED_CHANNEL |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_LED_CHANNEL_DISABLE | This refers to software control of the LEDs being disabled. This value is returned if either LED is disabled. | |
| SIO4_LED_CHANNEL_OFF_OFF | This refers to both LEDs being off. | |
| SIO4_LED_CHANNEL_OFF_ON | This refers to the upper LED being off and the lower LED being on. | |
| SIO4_LED_CHANNEL_ON_OFF | This refers to the upper LED being on and the lower LED being off. | |
| SIO4_LED_CHANNEL_ON_ON | This refers to both LEDs being on. | |

### 4.4.5. SIO4_IOCTL_LED_MAIN

This service controls the board's main LEDs, which are those not corresponding to any of the serial channels.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_LED_MAIN |
| arg | s32* |

The table below lists the options used with this service. Valid values are in the range 0x0 to 0x7. If a bit is set then the corresponding LED is on. If a bit is clear then the corresponding LED is off.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0x7 | This refers to the specified set of LEDs being on or off. | |

### 4.4.6. SIO4_IOCTL_LOOP_BACK

This service controls the LEDs that correspond to the channel being accessed.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_LOOP_BACK |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_LOOP_BACK_DISABLE | This refers to loopback operation being disabled. | |
| SIO4_LOOP_BACK_EXTERNAL | This refers to use of external loopback operation. | |
| SIO4_LOOP_BACK_INTERNAL | This refers to use of internal loopback operation. | |

### 4.4.7. SIO4_IOCTL_QUERY

This service provides information on an SIO4's feature set.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_QUERY |

| arg | s32* |
|---|---|

The table below lists the options used with this service. The feature in question is passed to the service. The results are returned by the service. Most feature queries return zero if the feature is not supported and non-zero if the feature is supported. Other features return a value specific to feature in question.

| Values | Description |
|---|---|
| -1 | This requests support information for this service. The value returned is zero. |
| SIO4_QUERY_BOARD_RESET | This refers to the Board Reset feature. |
| SIO4_QUERY_BUS_SPEED | This refers to the PCI Bus speed of the board. The options returned are 33 (for 33MHz) or 66 (for 66MHz). |
| SIO4_QUERY_BUS_WIDTH | This refers to the PCI Bus width of the board. The options returned are 32 (for 32-bits) or 64 (for 64-bits). |
| SIO4_QUERY_CHANNEL_QTY | This refers to the total number of channels for the board. This should be either four (for SIO4 boards) or eight (for SIO8 boards). |
| SIO4_QUERY_COUNT | This refers to the total number of query options supported by the driver. |
| SIO4_QUERY_DEVICE_QTY | This refers to the number of SIO4 devices on the board. This should be either one (for SIO4 boards) or two (for SIO8 boards). |
| SIO4_QUERY_DEVICE_TYPE | This refers to the basic board series type. This should be GSC_DEV_TYPE_SIO4. |
| SIO4_QUERY_DMDMA_SCD | This indicates if the board supports Single Cycle Disable for Demand Mode DMA operations. |
| SIO4_QUERY_FIFO_SIZE_RX | This refers to the size of the channel's Rx FIFO. |
| SIO4_QUERY_FIFO_SIZE_TOTAL | This refers to the total size for all four Tx and Rx FIFOs. |
| SIO4_QUERY_FIFO_SIZE_TX | This refers to the size of the channel's Tx FIFO. |
| SIO4_QUERY_FIFO_SPACE_CFG | This refers to the firmware having support for the FIFO Space Configuration option. |
| SIO4_QUERY_FORM_FACTOR | This refers to the board's basic form factor. See sio4_form_factor_t below. |
| SIO4_QUERY_FW_TYPE | This refers to the firmware's basic type, according to the board's model number. This should be either SIO4_FW_TYPE_Z16C30 (for USC based boards) or SIO4_FW_TYPE_SYNC (for –SYNC model board). |
| SIO4_QUERY_INDEX_CHANNEL | This refers to the index of the channel being accessed. This should be from zero to three. |
| SIO4_QUERY_INDEX_DEVICE | This refers to the zero based index of the SIO4 being accessed. |
| SIO4_QUERY_INDEX_SUBDEVICE | This refers to the sub-device index for the board being accessed. This should be zero for all SIO4 boards and zero or one for all SIO8 boards. * |
| SIO4_QUERY_IRQ_32 | This refers to the number of basic firmware interrupts supported by the firmware. This should be 32 or 16. |
| SIO4_QUERY_LEGACY_CABLE | This refers to firmware support for the legacy cable interface (i.e. upper and lower references rather than DTE or DCE). |
| SIO4_QUERY_LED_CHANNEL | This requests the number of channel specific LEDs. |
| SIO4_QUERY_LED_MAIN | This requests the number of main LEDs. |
| SIO4_QUERY_MODEL_BASE | This refers to the base model number. See sio4_model_t below. |
| SIO4_QUERY_MODEL_SYNC | This refers to the board being of the –SYNC type. |

| | |
|---|---|
| SIO4_QUERY_MODEL_Z16C30 | This refers to the board being of the USC type. |
| SIO4_QUERY_MP | This refers to the board's support for the Multi-Protocol transceiver firmware feature. |
| SIO4_QUERY_MP_CHIP | This indicates the Multi-Protocol transceiver chip type on the board. See sio4_mp_chip_t below. |
| SIO4_QUERY_MP_PROGRAM | This refers to the Multi-Protocol feature being programmable. If supported, then the transceiver type is software selectable. |
| SIO4_QUERY_OSC_CHIP | This refers to the master oscillator chip type. See sio4_osc_chip_t below. |
| SIO4_QUERY_OSC_MEASURE | This refers to the firmware's support for measuring the oscillator frequency. |
| SIO4_QUERY_OSC_PD_MAX | This refers to the firmware's Maximum Post Divider value for the oscillator programming feature. |
| SIO4_QUERY_OSC_PER_CHANNEL | This indicates if the board supports an oscillator per channel. |
| SIO4_QUERY_OSC_PROGRAM | This indicates if the on-board oscillator is programmable. |
| SIO4_QUERY_REG_BSR | This indicates support for the Board Status Register. |
| SIO4_QUERY_REG_CCR | This indicates support for the Clock Control Register. |
| SIO4_QUERY_REG_FCR | This indicates support for the FIFO Count Register. |
| SIO4_QUERY_REG_FR | This indicates support for the Features Register. |
| SIO4_QUERY_REG_FSR | This indicates support for the FIFO Size Register. |
| SIO4_QUERY_REG_FTR | This indicates support for the Firmware Type Register. |
| SIO4_QUERY_REG_GPIOSR | This indicates support for the GPIO Source Register. |
| SIO4_QUERY_REG_IELR | This indicates support for the Interrupt Edge/Level Register. |
| SIO4_QUERY_REG_IHLR | This indicates support for the Interrupt High/Low Register. |
| SIO4_QUERY_REG_IOCR | This indicates support for the I/O Control Register. |
| SIO4_QUERY_REG_PCR | This indicates support for the Programmable Clock Register. |
| SIO4_QUERY_REG_POCSR | This indicates support for the Programmable Oscillator Control/Status Register. |
| SIO4_QUERY_REG_PORAR | This indicates support for the Programmable Oscillator RAM Address Register. |
| SIO4_QUERY_REG_PORDR | This indicates support for the Programmable Oscillator RAM Data Register. |
| SIO4_QUERY_REG_PORD2R | This indicates support for the Programmable Oscillator RAM Data 2 Register. |
| SIO4_QUERY_REG_PSRCR | This indicates support for the Pin Source Register. |
| SIO4_QUERY_REG_PSRCR_BITS | This indicates the set of bits that are supported by the Pin Source Register. This is zero if the register is unsupported. |
| SIO4_QUERY_REG_PSTSR | This indicates support for the Pin Status Register. |
| SIO4_QUERY_REG_PSTSR_BITS | This indicates the set of bits that are supported by the Pin Status Register. This is zero if the register is unsupported. |
| SIO4_QUERY_REG_RCR | This indicates support for the Rx Count Register. |
| SIO4_QUERY_REG_SBR | This indicates support for the Sync Byte Register. |
| SIO4_QUERY_REG_TCR | This indicates support for the Tx Count Register. |
| SIO4_QUERY_REG_TSR | This indicates support for the Time Stamp Register. |

| | |
|---|---|
| SIO4_QUERY_RX_FIFO_FULL_CFG | This is the channel specific setting that indicates if the firmware's response to an Rx FIFO Full status is configurable. |
| SIO4_QUERY_RX_FIFO_FULL_CFG_GLB | This is the global setting that indicates if the firmware's response to an Rx FIFO Full status is configurable. |
| SIO4_QUERY_RX_FIFO_OVERRUN | This indicates if the Rx FIFO Overrun feature is supported. |
| SIO4_QUERY_RX_FIFO_UNDERRUN | This indicates if the Rx FIFO Underrun feature is supported. |
| SIO4_QUERY_RX_STATUS_WORD | This indicates if the firmware can put the USC's Rx Status Register in the Rx FIFO with each character pulled from the USC's Rx FIFO. |
| SIO4_QUERY_SIO4_TYPE | This indicates the board's basic SIO4 type See sio4_type_t below. |
| SIO4_QUERY_TIME_STAMP | This indicates if firmware supports the Time Stamp feature. |
| SIO4_QUERY_TX_FIFO_EMPTY_CFG | This indicates if the firmware's response to a Tx FIFO Empty status is configurable. |
| SIO4_QUERY_TX_FIFO_OVERRUN | This indicates if the Tx FIFO Overrun feature is supported. |
| SIO4_QUERY_USER_JUMPER_QTY | This indicates the number of user jumpers supported on the board. * |
| SIO4_QUERY_USER_JUMPER_SENSE | This indicates the bit value associated with the "jumper present" condition. |
| SIO4_QUERY_USER_JUMPER_VAL | This indicates the value read from the user jumpers. Zero is returned if the jumpers are not supported. |

* To get the board's overall user id use the Jumper Value and the sub-device index (SIO4_QUERY_USER_JUMPER_VAL and SIO4_QUERY_INDEX_SUBDEVICE, respectively).

#### 4.4.7.1. SIO4_QUERY_FW_TYPE and sio4_fw_type_t

Using the SIO4_QUERY_FW_TYPE query option returns the below values from the sio4_fw_type_t enumeration.

| Value | Description |
|---|---|
| SIO4_FW_TYPE_SYNC | The firmware is for a –SYNC board. |
| SIO4_FW_TYPE_Z16C30 | The firmware is for a Z16C30 based board. |

#### 4.4.7.2. SIO4_QUERY_FORM_FACTOR and sio4_form_factor_t

Using the SIO4_QUERY_FORM_FACTOR query option returns the below values from the sio4_form_factor_t enumeration.

| Value | Description |
|---|---|
| SIO4_FORM_FACTOR_CPCI | The board is of the Compact PCI form factor. * |
| SIO4_FORM_FACTOR_PC104P | The board is of the PC/104+ form factor. * |
| SIO4_FORM_FACTOR_PCI | The board is of the PCI form factor. * |
| SIO4_FORM_FACTOR_PCI66 | The board is of the PCI form factor. † |
| SIO4_FORM_FACTOR_PCIE | This refers to the single lane PCI Express form factor. |
| SIO4_FORM_FACTOR_PCIE4 | This refers to the four lane PCI Express form factor. |
| SIO4_FORM_FACTOR_PMC | The board is of the PMC form factor. * |
| SIO4_FORM_FACTOR_PMC66 | The board is of the PMC form factor. † |
| SIO4_FORM_FACTOR_UNKNOWN | The form factor could not be determined. |
| SIO4_FORM_FACTOR_XMC | The board is of the XMC form factor. |

* The PCI bus is 32-bits wide and runs at up to 33MHz.
† The PCI bus is 32-bits wide and runs at up to 66MHz.

#### 4.4.7.3. SIO4_QUERY_MODEL_BASE and sio4_model_t

Using the `SIO4_QUERY_MODEL_BASE` query option returns the below values from the `sio4_model_t` enumeration.

| Value | Description |
|---|---|
| `SIO4_MODEL_SIO4` | The basic model number is SIO4. |
| `SIO4_MODEL_SIO4A` | The basic model number is SIO4A. |
| `SIO4_MODEL_SIO4A_SYNC` | The basic model number is SIO4A-SYNC. |
| `SIO4_MODEL_SIO4AR` | The basic model number is SIO4AR. |
| `SIO4_MODEL_SIO4AR_SYNC` | The basic model number is SIO4ARx-SYNC. |
| `SIO4_MODEL_SIO4ARHM` | The basic model number is SIO4ARHM. |
| `SIO4_MODEL_SIO4ARHM_SYNC` | The basic model number is SIO4ARHM-SYNC. |
| `SIO4_MODEL_SIO4B` | The basic model number is SIO4B. |
| `SIO4_MODEL_SIO4B_SYNC` | The basic model number is SIO4B-SYNC. |
| `SIO4_MODEL_SIO4BX` | The basic model number is SIO4BX. |
| `SIO4_MODEL_SIO4BX_SYNC` | The basic model number is SIO4BX-SYNC. |
| `SIO4_MODEL_SIO4BX2` | The basic model number is SIO4BX2. |
| `SIO4_MODEL_SIO4BX2_SYNC` | The basic model number is SIO4BX2-SYNC. |
| `SIO4_MODEL_SIO4BXR` | The basic model number is SIO4BXR. |
| `SIO4_MODEL_SIO4BXR_SYNC` | The basic model number is SIO4BXR-SYNC. |
| `SIO4_MODEL_SIO8BX2` | The basic model number is SIO48BX2. |
| `SIO4_MODEL_SIO8BX2_SYNC` | The basic model number is SIO48BX2-SYNC. |
| `SIO4_MODEL_SIO8BXS` | The basic model number is SIO48BXS. |
| `SIO4_MODEL_SIO8BXS_SYNC` | The basic model number is SIO48BXS-SYNC. |

#### 4.4.7.4. SIO4_QUERY_MP_CHIP and sio4_mp_chip_t

Using the `SIO4_QUERY_MP_CHIP` query option returns the below values from the `sio4_mp_chip_t` enumeration.

| Value | Description |
|---|---|
| `SIO4_MP_CHIP_FIXED` | The transceivers are not programmable |
| `SIO4_MP_CHIP_SP508` | The transceivers are SP508. |

#### 4.4.7.5. SIO4_QUERY_OSC_CHIP and sio4_osc_chip_t

Using the `SIO4_QUERY_OSC_CHIP` query option returns the below values from the `sio4_osc_chip_t` enumeration.

| Value | Description |
|---|---|
| `SIO4_OSC_CHIP_FIXED` | The board has a fixed frequency oscillator. |
| `SIO4_OSC_CHIP_IDC2053B` | The board has a single IDC2053B oscillator. |
| `SIO4_OSC_CHIP_IDC2053B_4` | The board has four IDC2053B oscillators. |
| `SIO4_OSC_CHIP_CY22393` | The board has a single CY22393 oscillator. |
| `SIO4_OSC_CHIP_CY22393_2` | The board has two CY22393 oscillators. |

#### 4.4.7.6. SIO4_QUERY_SIO4_TYPE and sio4_type_t

Using the `SIO4_QUERY_SIO4_TYPE` query option returns the below values from the `sio4_type_t` enumeration.

| Value | Description |
|---|---|
| SIO4_TYPE_SIO4 | The board is a version of the original SIO4. |
| SIO4_TYPE_SIO4A | The board is a version of the SIO4A. |
| SIO4_TYPE_SIO4AR | The board is a version of the SIO4AR. |
| SIO4_TYPE_SIO4AHRM | The board is a version of the SIO4ARHM. |
| SIO4_TYPE_SIO4B | The board is a version of the SIO4B. |
| SIO4_TYPE_SIO4BX | The board is a version of the SIO4BX. |
| SIO4_TYPE_SIO4BX2 | The board is a version of the SIO4BX2. |
| SIO4_TYPE_SIO4BXR | The board is a version of the SIO4BXR. |
| SIO4_TYPE_SIO8BX2 | The board is a version of the SIO8BX2. |
| SIO4_TYPE_SIO8BXS | The board is a version of the SIO8BXS. |

## 4.5. Oscillator Specific IOCTL Services

These IOCTL services relate to operation of the on-board oscillator(s).

### 4.5.1. SIO4_IOCTL_OSC_MEASURE

This service measures the master oscillator frequency. The results are reported in hertz.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_OSC_MEASURE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 or above | N/A | This refers to the measured results. |
| 1 | Request that the measurement be made. | N/A |

### 4.5.2. SIO4_IOCTL_OSC_PROGRAM

This service programs the master oscillator for the specified frequency, in hertz.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_OSC_PROGRAM |
| arg | s32* |

The table below lists the options used with this service. The range for valid values depends on the type of oscillator on the board being accessed.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| > 0 | This refers to the requested frequency. | |

The valid range for requests are as follows, though the frequency produced through programming may be more limited.

| Oscillator Type | Fmin (Hz) | Fmax | Information |
|---|---|---|---|
| CY22393 | 0 | 20,000,000 | The minimum achievable frequency is 244 Hz for all recent firmware and 3906 for older firmware. |
| IDC2053G | 1,000,000 | 20,000,000 | Oscillator programming is not implemented. |
| Fixed | 1,000,000 | 20,000,000 | Oscillator programming is unsupported. |
| Unknown | 1,000,000 | 20,000,000 | Oscillator programming is unsupported. |

### 4.5.3. SIO4_IOCTL_OSC_REFERENCE

This service specifies the master oscillator reference frequency, in hertz. The default is 20,000,000 Hz. The purpose of this service is to let the driver know the exact frequency as it cannot be determined precisely by examination.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_OSC_REFERENCE |
| arg | s32* |

The table below lists the options used with this service. The range for valid values depends on the board being accessed.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| > 0 | This refers to the default, specified or measured frequency. | |

Valid argument values are as follows.

| Oscillator Type | Fmin (Hz) | Fmax (Hz) | Information |
|---|---|---|---|
| CY22393 | 20,000,000 | 20,000,000 | The reference frequency cannot be changed. |
| IDC2053G | 1,000,000 | 20,000,000 | Oscillator programming is not implemented. |
| Fixed | 1,000,000 | 20,000,000 | Oscillator programming is unsupported. |
| Unknown | 1,000,000 | 20,000,000 | Oscillator programming is unsupported. |

## 4.6. Register Specific IOCTL Services

These IOCTL services relate to access of the board's register.

### 4.6.1. SIO4_IOCTL_REG_MOD

This service performs a read-modify-write of an SIO4 register. This includes only the GSC firmware registers and the USC registers, when supported. The PCI and PLX Feature Set Registers are read-only. Refer to `sio4.h` for a complete list of the GSC firmware registers and to `sio4_usc.h` for a complete list of the USC registers.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_REG_MOD |
| arg | gsc_reg_t* |

Definition

```
typedef struct
{
    u32 reg;
```

```
    u32 value;
    u32 mask;
} gsc_reg_t;
```

| Fields | Description |
|---|---|
| reg | This is set to the identifier for the register to access. |
| value | This contains the value for the register bits to modify. |
| mask | This specifies the set of bits to modify. If a bit here is set, then the respective register bit is modified. If a bit here is zero, then the respective register bit is unmodified. |

### 4.6.2. SIO4_IOCTL_REG_READ

This service reads the value of an SIO4 register. This includes the PCI registers, the PLX Feature Set Registers, GSC firmware registers and the USC registers, when supported. Refer to `sio4.h` for a complete list of the GSC firmware registers and to `sio4_usc.h` for a complete list of the USC registers.

#### Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_REG_READ |
| arg | gsc_reg_t* |

#### Definition

```
typedef struct
{
    u32 reg;
    u32 value;
    u32 mask;
} gsc_reg_t;
```

| Fields | Description |
|---|---|
| reg | This is set to the identifier for the register to access. |
| value | This is the value read from the specified register. |
| mask | This is ignored for read request. |

### 4.6.3. SIO4_IOCTL_REG_READ_RAW

This service reads the value of an SIO4 firmware register without regard to the register layout or channel owner. Refer to `sio4.h` for a complete list of the GSC firmware registers.

#### Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_REG_READ_RAW |
| arg | gsc_reg_t* |

#### Definition

```
typedef struct
{
    u32 reg;
    u32 value;
    u32 mask;
```

```
} gsc_reg_t;
```

| Fields | Description |
|---|---|
| reg | This is set to the zero based index of the register. Valid values are from zero to 63. |
| value | This is the value read from the specified register. |
| mask | This is ignored for read request. |

### 4.6.4. SIO4_IOCTL_REG_WRITE

This service writes a value to an SIO4 register. This includes only the GSC firmware registers and the USC registers, when supported. The PCI and PLX Feature Set Registers are read-only. Refer to `sio4.h` for a complete list of the GSC firmware registers and to `sio4_usc.h` for a complete list of the USC registers.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_REG_WRITE |
| arg | gsc_reg_t* |

Definition

```
typedef struct
{
    u32 reg;
    u32 value;
    u32 mask;
} gsc_reg_t;
```

| Fields | Description |
|---|---|
| reg | This is set to the identifier for the register to access. |
| value | This is the value to write to the specified register. |
| mask | This is ignored for write request. |

## 4.7. Time Stamp Specific IOCTL Services

These IOCTL services relate to use of the Time Stamping feature.

### 4.7.1. SIO4_IOCTL_TIME_STAMP_COUNT

This service operates on the Time Stamp counter.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_TIME_STAMP_COUNT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_TIME_STAMP_COUNT_CLEAR | This refers to count being cleared. | |
| SIO4_TIME_STAMP_COUNT_CONTINUE | This refers to no change at all. | |

### 4.7.2. SIO4_IOCTL_TIME_STAMP_ENABLE

This service enables or disables the Time Stamp feature.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_TIME_STAMP_COUNT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_TIME_STAMP_ENABLE_NO | This refers to the feature being disabled. | |
| SIO4_TIME_STAMP_ENABLE_YES | This refers to the feature being enabled. | |

### 4.7.3. SIO4_IOCTL_TIME_STAMP_SRC

This service configures the source selection for the Time Stamp counter.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_TIME_STAMP_SRC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_TIME_STAMP_SRC_EXT | This refers to the external cable interface source. | |
| SIO4_TIME_STAMP_SRC_INT | This refers to the internal source. | |

### 4.7.4. SIO4_IOCTL_TIME_STAMP_VAL

This service configures the current Time Stamp value.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_TIME_STAMP_VAL |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFFFF | This is the valid range for this service. | |

## 4.8. Transceiver Specific IOCTL Services

These IOCTL services relate to operation of the board's transceivers.

### 4.8.1. SIO4_IOCTL_XCVR_ENABLE

This service enables and disables the transceivers.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_XCVR_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_XCVR_ENABLE_NO | This refers to the transceivers being disabled. When disabled, the legacy mode functionality is active, if it is supported. | |
| SIO4_XCVR_ENABLE_YES | This refers to the transceivers being enabled. When enabled, the legacy mode functionality is disabled, if it is supported. | |

### 4.8.2. SIO4_IOCTL_XCVR_PROTOCOL

This service enables and disables the transceivers.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_XCVR_PROTOCOL |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_XCVR_PROTOCOL_DISABLE | This refers to disabling the transceivers. | |
| SIO4_XCVR_PROTOCOL_RS232 | This refers to RS-232. | |
| SIO4_XCVR_PROTOCOL_RS422_RS423_1 | This refers to RS-422/RS-423 (mode 1). | |
| SIO4_XCVR_PROTOCOL_RS422_RS423_2 | This refers to RS-422/RS-423 (mode 2). | |
| SIO4_XCVR_PROTOCOL_RS422_RS485 | This refers to RS-422/RS-485. | |
| SIO4_XCVR_PROTOCOL_RS423 | This refers to RS-423. | |
| SIO4_XCVR_PROTOCOL_RS530 | This refers to RS-530 (mode 1). | |
| SIO4_XCVR_PROTOCOL_RS530A | This refers to RS-530 (mode 2). | |
| SIO4_XCVR_PROTOCOL_V35 | This refers to V.35 (mode 1). | |
| SIO4_XCVR_PROTOCOL_V35A | This refers to V.35 (mode 2). | |
| SIO4_XCVR_PROTOCOL_UNKNOWN | N/A | This indicates that the transceiver type is unknown. |

### 4.8.3. SIO4_IOCTL_XCVR_TERM

This service enables and disables the termination resisters that are part of the transceivers.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_XCVR_TERM |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_XCVR_TERM_DISABLE | This refers to disabling the termination resisters. | |
| SIO4_XCVR_TERM_ENABLE | This refers to enabling the termination resisters. | |

# 5. Zilog Model Specific IOCTL Services

These IOCTL services relate only to those SIO4 model boards which utilize the Zilog Z16C20 USC.

> **NOTE**: All services whose argument data type is `s32*` accept the value of `-1` being passed to the driver. If there is a setting associated with the service, then passing in the value `-1` is a request for the current setting. If the service is supported, then the current setting is returned. If the service is not supported, then the value returned by the driver will be `-1`. If there isn't a setting associated with the service, then passing in the value `-1` is a request to determine if the service is supported. If the service is supported, then the value one is returned. If the service is not supported, then the value `-1` is returned.

## 5.1. Zilog Model Cable Specific IOCTL Services

These IOCTL services relate to the configuration of the cable interface.

### 5.1.1. SIO4_IOCTL_Z16_CBL_DCD_CFG

This service configures the cable DCD signal output source for USC based SIO4 boards.

> **NOTE**: This service configures only the source for the DCD signal when configured as an output. The signal direction is configured via the `SIO4_IOCTL_USC_DCD_CFG` IOCTL service (section 6.12.3, page 77).

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_Z16_CBL_DCD_CFG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_CBL_DCD_CFG_OUT_0 | Drive the cable signal low. | |
| SIO4_Z16_CBL_DCD_CFG_OUT_1 | Drive the cable signal high. | |
| SIO4_Z16_CBL_DCD_CFG_OUT_RTS | Drive the cable signal from the receiver's Request To Send source, which is the Rx FIFO Almost Full signal. | |
| SIO4_Z16_CBL_DCD_CFG_OUT_USC_DCD | Drive the cable signal from the USC DCD signal. | |

### 5.1.2. SIO4_IOCTL_Z16_CBL_DTR_DSR_CFG

This service configures the cable DTR/DSR signal for USC based SIO4 boards. The DTR/DSR cable signal is provided for DTR/DSR flow control and is entirely under software control.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_Z16_CBL_DTR_DSR_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_CBL_DTR_DSR_CFG_OUT_0 | Drive the cable signal low. | |
| SIO4_Z16_CBL_DTR_DSR_CFG_OUT_1 | Drive the cable signal high. | |
| SIO4_Z16_CBL_DTR_DSR_CFG_IN | Configure the signal as an input. | |
| SIO4_Z16_CBL_DTR_DSR_CFG_TRI | Tristate the signal. | |

### 5.1.3. SIO4_IOCTL_Z16_CBL_RTS_CFG

This service configures the cable RTS signal for USC based SIO4 boards.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_Z16_CBL_RTS_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_CBL_RTS_CFG_OUT_0 | Drive the cable signal low. | |
| SIO4_Z16_CBL_RTS_CFG_OUT_1 | Drive the cable signal high. | |
| SIO4_Z16_CBL_RTS_CFG_OUT_RTS | Drive the cable signal from the receiver's Request To Send source, which is the Rx FIFO Almost Full signal. | |
| SIO4_Z16_CBL_RTS_CFG_OUT_USC_CTS | Drive the cable signal from the USC CTS signal. | |

### 5.1.4. SIO4_IOCTL_Z16_CBL_TXAUXC_CFG

This service configures the cable Tx Auxiliary Clock signal for USC based SIO4 boards.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_Z16_CBL_TXAUXC_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_CBL_TXAUXC_CFG_OUT_0 | Drive the cable signal low. | |
| SIO4_Z16_CBL_TXAUXC_CFG_OUT_1 | Drive the cable signal high. | |
| SIO4_Z16_CBL_TXAUXC_CFG_OUT_OSC | Drive the cable signal from the on-board oscillator. | |
| SIO4_Z16_CBL_TXAUXC_CFG_TRI | Tristate the signal. | |

### 5.1.5. SIO4_IOCTL_Z16_CBL_TXC_CFG

This service configures the cable Tx Clock signal for USC based SIO4 boards.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_Z16_CBL_TXC_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_CBL_TXC_CFG_OUT_0 | Drive the cable signal low. | |
| SIO4_Z16_CBL_TXC_CFG_OUT_1 | Drive the cable signal high. | |
| SIO4_Z16_CBL_TXC_CFG_OUT_CBL_RXA | Drive the cable signal from the Rx Auxiliary Clock signal on the cable interface. | |
| SIO4_Z16_CBL_TXC_CFG_OUT_CBL_RXC | Drive the cable signal from the Rx Clock signal on the cable interface. | |
| SIO4_Z16_CBL_TXC_CFG_OUT_OSC | Drive the cable signal from the on-board oscillator. | |
| SIO4_Z16_CBL_TXC_CFG_OUT_OSC_INV | Drive the cable signal from the on-board oscillator, but inverted. | |
| SIO4_Z16_CBL_TXC_CFG_OUT_USC_RXC | Drive the cable signal from the USC's Rx Clock pin. | |
| SIO4_Z16_CBL_TXC_CFG_OUT_USC_TXC | Drive the cable signal from the USC's Tx Clock pin. | |

### 5.1.6. SIO4_IOCTL_Z16_CBL_TXD_CFG

This service configures the cable Tx Data signal for USC based SIO4 boards.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_Z16_CBL_TXD_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_CBL_TXD_CFG_OUT_0 | Drive the cable signal low. | |
| SIO4_Z16_CBL_TXD_CFG_OUT_1 | Drive the cable signal high. | |
| SIO4_Z16_CBL_TXD_CFG_OUT_USC_TXD | Drive the cable signal from the USC's Tx Data signal. | |

## 5.2. Zilog Model Miscellaneous IOCTL Services

### 5.2.1. SIO4_IOCTL_Z16_RX_STS_WRD_ENABLE

This service configures the receiver's option of including the USC's Rx Status Register in the main FIFO with each data value extracted from the USC's Rx FIFO.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_Z16_RX_STS_WRD_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_RX_STS_WRD_ENABLE_NO | Do not enable the feature. | |
| SIO4_Z16_RX_STS_WRD_ENABLE_YES | Enable the feature. | |

### 5.2.2. SIO4_IOCTL_Z16_SYNC_BYTE

This service set the Sync Byte Value used for Sync Bytes Detection for USC based SIO4 boards.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_Z16_SYNC_BYTE |
| arg | s32* |

The table below lists the options used with this service. Valid argument values are from zero to 0xFF.

| Macros | Description |
|---|---|
| -1 | Requests current setting. |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFF | This is the valid range for this service. | |

## 5.3. Zilog Model Legacy Cable Specific IOCTL Services

These IOCTL services relate to the configuration of the legacy cable interface.

> **NOTE**: These services are available only when the legacy cable configuration is supported by firmware.

### 5.3.1. SIO4_IOCTL_Z16_LEG_RXC

This service configures the routing of the cable Rx Clock signal when using legacy cable configuration mode.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_Z16_LEG_RXC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_LEG_RXC_DISABLE | Disable the cable signal. | |
| SIO4_Z16_LEG_RXC_LOWER | Connect to the signal at the lower cable portion. | |
| SIO4_Z16_LEG_RXC_UPPER | Connect to the signal at the upper cable portion. | |

### 5.3.2. SIO4_IOCTL_Z16_LEG_RXD_DCD_CFG

This service configures the cable Rx Data and DCD signals for USC based SIO4 boards supporting the legacy cable interface.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_Z16_LEG_RXD_DCD_CFG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_LEG_RXD_DCD_CFG_LOW | This configures the signals for the lower connector pins. | |
| SIO4_Z16_LEG_RXD_DCD_CFG_TRI | This tri-states the cable signals. | |
| SIO4_Z16_LEG_RXD_DCD_CFG_UP | This configures the signals for the upper connector pins. | |

### 5.3.3. SIO4_IOCTL_Z16_LEG_TXC

This service configures the routing of the cable Tx Clock signal when using legacy cable configuration mode.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_CBL_LEG_TXC |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_LEG_TXC_DISABLE | Disable the cable signal. | |
| SIO4_Z16_LEG_TXC_BOTH | Connect the signal to both the upper and lower cable portions. | |
| SIO4_Z16_LEG_TXC_LOWER | Connect the signal to the lower cable portion. | |
| SIO4_Z16_LEG_TXC_UPPER | Connect the signal to the upper cable portion. | |

### 5.3.4. SIO4_IOCTL_Z16_LEG_TXD_CTS_CFG

This service configures the cable Tx Data and CTS signals for USC based SIO4 boards supporting the legacy cable interface.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_Z16_LEG_TXD_CTS_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_Z16_LEG_TXD_CTS_CFG_LOW | This configures the signals for the lower connector pins. | |
| SIO4_Z16_LEG_TXD_CTS_CFG_TRI | This tri-states the cable signals. | |
| SIO4_Z16_LEG_TXD_CTS_CFG_UP | This configures the signals for the upper connector pins. | |
| SIO4_Z16_LEG_TXD_CTS_CFG_UP_LOW | This configures the signals for the upper and lower connector pins. | |

# 6. USC Specific Zilog based IOCTL Services

These IOCTL services are specific to the USC on Zilog based SIO4 boards.

> **NOTE**: All services whose argument data type is `s32*` accept the value of `-1` being passed to the driver. If there is a setting associated with the service, then passing in the value `-1` is a request for the current setting. If the service is supported, then the current setting is returned. If the service is not supported, then the value returned by the driver will be `-1`. If there isn't a setting associated with the service, then passing in the value `-1` is a request to determine if the service is supported. If the service is supported, then the value one is returned. If the service is not supported, then the value `-1` is returned.

## 6.1. USC 802.3 Protocol Specific IOCTL Services

These services are specific to the USC's 802.3 protocol support.

### 6.1.1. SIO4_IOCTL_USC_8023_RX_ADRS_SRCH

This service configures the Address Search feature for the receiver when using the 802.3 serial protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_8023_RX_ADRS_SRCH |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_8023_RX_ADRS_SRCH_NO | This disables use of the feature. | |
| SIO4_USC_8023_RX_ADRS_SRCH_YES | This enables use of the feature. | |

### 6.1.2. SIO4_IOCTL_USC_8023_TX_UNDERRUN

This service configures the Underrun reaction of the transmitter when using the 802.3 serial protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_8023_TX_UNDERRUN |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_8023_TX_UNDERRUN_CRC | Send out the current CRC value. | |
| SIO4_USC_8023_TX_UNDERRUN_NONE | Take no additional action. | |

## 6.2. USC Asynchronous Protocol Specific IOCTL Services

These services are specific to the USC's Asynchronous protocol support.

### 6.2.1. SIO4_IOCTL_USC_ASYNC_RX_CLK_RATE

This service configures the Clock Rate (the oversampling rate) for the receiver when using the Asynchronous protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_ASYNC_RX_CLK_RATE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ASYNC_CLK_RATE_16X | Divide the source clock by 16. | |
| SIO4_USC_ASYNC_CLK_RATE_32X | Divide the source clock by 32. | |
| SIO4_USC_ASYNC_CLK_RATE_64X | Divide the source clock by 64. | |

### 6.2.2. SIO4_IOCTL_USC_ASYNC_TX_CLK_RATE

This service configures the Clock Rate for the transmitter when using the Asynchronous protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_ASYNC_TX_CLK_RATE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ASYNC_CLK_RATE_16X | Divide the source clock by 16. | |
| SIO4_USC_ASYNC_CLK_RATE_32X | Divide the source clock by 32. | |
| SIO4_USC_ASYNC_CLK_RATE_64X | Divide the source clock by 64. | |

### 6.2.3. SIO4_IOCTL_USC_ASYNC_TX_STOP_BIT

This service configures the number of Stop Bits for the transmitter when using the Asynchronous protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_ASYNC_TX_STOP_BIT |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ASYNC_TX_STOP_BIT_0__9_16 | Set the Stop Bit period to 9/16$^{th}$ of a bit period. | |
| SIO4_USC_ASYNC_TX_STOP_BIT_0_10_16 | Set the Stop Bit period to 10/16$^{th}$ of a bit period. | |

| | |
|---|---|
| SIO4_USC_ASYNC_TX_STOP_BIT_0_11_16 | Set the Stop Bit period to 11/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_0_12_16 | Set the Stop Bit period to 12/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_0_13_16 | Set the Stop Bit period to 13/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_0_14_16 | Set the Stop Bit period to 14/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_0_15_16 | Set the Stop Bit period to 15/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1 | Set the Stop Bit period to one bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1__1_16 | Set the Stop Bit period to 1 + 1/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1__2_16 | Set the Stop Bit period to 1 + 2/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1__3_16 | Set the Stop Bit period to 1 + 3/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1__4_16 | Set the Stop Bit period to 1 + 4/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1__5_16 | Set the Stop Bit period to 1 + 5/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1__6_16 | Set the Stop Bit period to 1 + 6/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1__7_16 | Set the Stop Bit period to 1 + 7/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1__8_16 | Set the Stop Bit period to 1 + 8/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1__9_16 | Set the Stop Bit period to 1 + 9/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1_10_16 | Set the Stop Bit period to 1 + 10/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1_11_16 | Set the Stop Bit period to 1 + 11/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1_12_16 | Set the Stop Bit period to 1 + 12/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1_13_16 | Set the Stop Bit period to 1 + 13/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1_14_16 | Set the Stop Bit period to 1 + 14/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_1_15_16 | Set the Stop Bit period to 1 + 15/16th of a bit period. |
| SIO4_USC_ASYNC_TX_STOP_BIT_2 | Set the Stop Bit period to two bit periods. |

## 6.3. USC Asynchronous with Code Violations Protocol Specific IOCTL Services

These services are specific to the USC's Asynchronous with Code Violations protocol support.

### 6.3.1. SIO4_IOCTL_USC_ACV_RX_EXT_W

This service configures the Extended Word feature for the receiver when using the Asynchronous with Code Violation protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_ACV_RX_EXT_W |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ACV_EXT_W_NO | Do not use Extended Words. | |
| SIO4_USC_ACV_EXT_W_YES | Use Extended Words. | |

### 6.3.2. SIO4_IOCTL_USC_ACV_TX_CV_POL

This service configures the Code Violation Polarity for the transmitter when using the Asynchronous with Code Violation protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_ACV_TX_CV_POL |

| | |
|---|---|
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ACV_TX_CV_POL_0 | This selects the zero or low polarity. | |
| SIO4_USC_ACV_TX_CV_POL_1 | This selects the one or high polarity. | |

### 6.3.3. SIO4_IOCTL_USC_ACV_TX_EXT_W

This service configures the Extended Word feature for the transmitter when using the Asynchronous with Code Violation protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_ACV_TX_EXT_W |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ACV_EXT_W_NO | Do not use Extended Words. | |
| SIO4_USC_ACV_EXT_W_YES | Use Extended Words. | |

### 6.3.4. SIO4_IOCTL_USC_ACV_TX_STOP_BIT

This service configures the number of Stop Bits for the transmitter when using the Asynchronous with Code Violation protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_ACV_TX_STOP_BIT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ACV_TX_STOP_BIT_1 | Use one Stop Bit. | |
| SIO4_USC_ACV_TX_STOP_BIT_2 | Use two Stop Bits. | |
| SIO4_USC_ACV_TX_STOP_BIT_NONE | Use no Stop Bits. | |

## 6.4. USC BSC Protocol Specific IOCTL Services

These services are specific to the USC's BSC protocol support. BSC stands for Bisynchronous Serial Communications and refers to the Bisync protocol.

### 6.4.1. SIO4_IOCTL_USC_BSC_RX_SHORT

This service configures the Short Sync Character selection for the receiver when using the Bisynchronous Serial Communications protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BSC_RX_SHORT |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BSC_SHORT_NO | Do not use a Short Sync Character. | |
| SIO4_USC_BSC_SHORT_YES | Use a Short Sync Character. | |

### 6.4.2. SIO4_IOCTL_USC_BSC_RX_STRIP

This service configures the receiver to strip detected sync characters from the input stream when using the Bisynchronous Serial Communications protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BSC_RX_STRIP |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BSC_RX_STRIP_NO | Do not Strip the Sync Character(s) from the input stream. | |
| SIO4_USC_BSC_RX_STRIP_YES | Strip the Sync Character(s) from the input stream. | |

### 6.4.3. SIO4_IOCTL_USC_BSC_RX_SYN0

This service configures the SYN0 Sync Character value for the receiver when using the Bisynchronous Serial Communications protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BSC_RX_SYN0 |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFF | This is the valid range for this service. | |

### 6.4.4. SIO4_IOCTL_USC_BSC_RX_SYN1

This service configures the SYN1 Sync Character value for the receiver when using the Bisynchronous Serial Communications protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BSC_RX_SYN1 |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFF | This is the valid range for this service. | |

### 6.4.5. SIO4_IOCTL_USC_BSC_TX_PREAMBLE

This service configures the Preamble output selection for the transmitter when using the Bisynchronous Serial Communications protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BSC_TX_PREAMBLE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BSC_TX_PREAMBLE_NO | This refers to sending no Preamble. | |
| SIO4_USC_BSC_TX_PREAMBLE_YES | This refers to sending a Preamble. | |

### 6.4.6. SIO4_IOCTL_USC_BSC_TX_SHORT

This service configures the Short Sync Character selection for the transmitter when using the Bisynchronous Serial Communications protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BSC_TX_SHORT |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BSC_SHORT_NO | Do not use a Short Sync Character. | |
| SIO4_USC_BSC_SHORT_YES | Use a Short Sync Character. | |

### 6.4.7. SIO4_IOCTL_USC_BSC_TX_SYN0

This service configures the SYN0 Sync Character value for the transmitter when using the Bisynchronous Serial Communications protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BSC_TX_SYN0 |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFF | This is the valid range for this service. | |

### 6.4.8. SIO4_IOCTL_USC_BSC_TX_SYN1

This service configures the SYN1 Sync Character value for the transmitter when using the Bisynchronous Serial Communications protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BSC_TX_SYN1 |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFF | This is the valid range for this service. | |

### 6.4.9. SIO4_IOCTL_USC_BSC_TX_UNDERRUN

This service configures the Underrun reaction selection for the transmitter when using the Bisynchronous Serial Communications protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BSC_TX_UNDERRUN |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BSC_TX_UNDERRUN_CRC_S01 | Send the CRC, then the SYN0 and SYN1 characters. | |
| SIO4_USC_BSC_TX_UNDERRUN_CRC_S1 | Send the CRC, then the SYN1 character. | |
| SIO4_USC_BSC_TX_UNDERRUN_S01 | Send the SYN0 and SYN1 characters. | |
| SIO4_USC_BSC_TX_UNDERRUN_S1 | Send the SYN1 character. | |

## 6.5. USC HDLC Protocol Specific IOCTL Services

These services are specific to the USC's HDLC protocol support. HDLC stands for High-Level Data Link Control. For detailed information on the HDLC protocol refer to ISO/IEC 13239.

### 6.5.1. SIO4_IOCTL_USC_HDLC_RX_ADRS_CTRL

This service configures the receiver's Address and Control field detection method when using the HDLC protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_HDLC_RX_ADRS_CTRL |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_HDLC_RX_ADRS_CTRL_OFF | This disabled Address and Control field detection. | |
| SIO4_USC_HDLC_RX_ADRS_CTRL_16 | This specifies 16-bits of combined content. | |
| SIO4_USC_HDLC_RX_ADRS_CTRL_24 | This specifies 24-bits of combined content. | |
| SIO4_USC_HDLC_RX_ADRS_CTRL_32 | This specifies 32-bits of combined content. | |
| SIO4_USC_HDLC_RX_ADRS_CTRL_EA_16 | This specifies Extended Address plus 16-bits of Control. | |
| SIO4_USC_HDLC_RX_ADRS_CTRL_EA_24 | This specifies Extended Address plus 24-bits of Control. | |
| SIO4_USC_HDLC_RX_ADRS_CTRL_EAC8 | This specifies Extended Address plus 8-bits with Extended Control. | |
| SIO4_USC_HDLC_RX_ADRS_CTRL_EAC16 | This specifies Extended Address plus 16-bits with Extended Control. | |

### 6.5.2. SIO4_IOCTL_USC_HDLC_TX_L_CHR_LEN

This service configures the transmitter's Last Character Length for a frame when using the HDLC protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_HDLC_TX_L_CHR_LEN |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_HDLC_TX_L_CHR_LEN_1 | This sets the last character length to 1-bit. | |
| SIO4_USC_HDLC_TX_L_CHR_LEN_2 | This sets the last character length to 2-bits. | |
| SIO4_USC_HDLC_TX_L_CHR_LEN_3 | This sets the last character length to 3-bits. | |
| SIO4_USC_HDLC_TX_L_CHR_LEN_4 | This sets the last character length to 4-bits. | |
| SIO4_USC_HDLC_TX_L_CHR_LEN_5 | This sets the last character length to 5-bits. | |
| SIO4_USC_HDLC_TX_L_CHR_LEN_6 | This sets the last character length to 6-bits. | |
| SIO4_USC_HDLC_TX_L_CHR_LEN_7 | This sets the last character length to 7-bits. | |
| SIO4_USC_HDLC_TX_L_CHR_LEN_8 | This sets the last character length to 8-bits. | |

### 6.5.3. SIO4_IOCTL_USC_HDLC_TX_PREAMBLE

This service configures the transmitter to include, or not, a preamble as part of a frame when using the HDLC protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_HDLC_TX_PREAMBLE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_HDLC_TX_PREAMBLE_NO | Do not include a preamble. | |
| SIO4_USC_HDLC_TX_PREAMBLE_YES | Include a preamble. | |

### 6.5.4. SIO4_IOCTL_USC_HDLC_TX_SHARE_0

This service configures the transmitter to share the zero between consecutive flag sequences when using the HDLC protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_HDLC_TX_SHARE_0 |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_HDLC_TX_SHARE_0_NO | Do not share the zero. (Send two zeroes.) | |
| SIO4_USC_HDLC_TX_SHARE_0_YES | Share the zero. (Only send one zero.) | |

### 6.5.5. SIO4_IOCTL_USC_HDLC_TX_UNDERRUN

This service configures the transmitter's reaction to a Tx FIFO Underrun condition when using the HDLC protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_HDLC_TX_UNDERRUN |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_HDLC_TX_UNDERRUN_ABORT | Send an Abort sequence. | |
| SIO4_USC_HDLC_TX_UNDERRUN_CRC_F | Send a CRC then a Flag sequence. | |
| SIO4_USC_HDLC_TX_UNDERRUN_EXT_A | Send an Extended Abort sequence. | |
| SIO4_USC_HDLC_TX_UNDERRUN_FLAG | Send a Flag sequence. | |

## 6.6. USC HDLC Loop Protocol Specific IOCTL Services

These services are specific to the USC's HDLC Loop protocol support.

### 6.6.1. SIO4_IOCTL_USC_HDLCL_TX_ACTIVE

This service configures the transmitter to insert its own into the data stream when using the HDLC protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_HDLCL_TX_ACTIVE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_HDLCL_TX_ACTIVE_NONE | Do not insert the transmitter's own data into the stream. | |
| SIO4_USC_HDLCL_TX_ACTIVE_POLL | Request that the transmitter insert its own data into the stream. | |

### 6.6.2. SIO4_IOCTL_USC_HDLCL_TX_SHARE_0

This service configures the transmitter to share the zero between consecutive flag sequences when using the HDLC Loop protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_HDLCL_TX_SHARE_0 |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_HDLCL_TX_SHARE_0_NO | Do not share the zero. (Send two zeroes.) | |
| SIO4_USC_HDLCL_TX_SHARE_0_YES | Share the zero. (Only send one zero.) | |

### 6.6.3. SIO4_IOCTL_USC_HDLCL_TX_UNDERRUN

This service configures the transmitter's reaction to a Tx FIFO Underrun condition when using the HDLC Loop protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_HDLCL_TX_UNDERRUN |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_HDLCL_TX_UNDERRUN_ABORT | Send an Abort sequence. | |
| SIO4_USC_HDLCL_TX_UNDERRUN_CRC_F | Send a CRC then a Flag sequence. | |
| SIO4_USC_HDLCL_TX_UNDERRUN_E_ABT | Send an Extended Abort sequence. | |
| SIO4_USC_HDLCL_TX_UNDERRUN_FLAG | Send a Flag sequence. | |

## 6.7. USC Isochronous Protocol Specific IOCTL Services

These services are specific to the USC's Isochronous protocol support.

### 6.7.1. SIO4_IOCTL_USC_ISOC_TX_STOP_BIT

This service configures the transmitter's number of Stop Bits when using the Isochronous protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_ISOC_TX_STOP_BIT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ISOC_TX_STOP_BIT_1 | Use one Stop Bit. | |
| SIO4_USC_ISOC_TX_STOP_BIT_2 | Use two Stop Bits. | |

## 6.8. USC Monosync Protocol Specific IOCTL Services

These services are specific to the USC's Monosync protocol support.

### 6.8.1. SIO4_IOCTL_USC_MONO_RX_SHORT

This service configures the Short Sync Character selection for the receiver when using the Monosync protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_MONO_RX_SHORT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_MONO_SHORT_NO | Do not use a Short Sync Character. Use 8-bits. | |
| SIO4_USC_MONO_SHORT_YES | Use a Short Sync Character. Use less than 8-bits. | |

### 6.8.2. SIO4_IOCTL_USC_MONO_RX_STRIP

This service configures the receiver to strip detected sync characters from the input stream when using the Monosync protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_MONO_RX_STRIP |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_MONO_RX_STRIP_NO | Do not Strip the Sync Character from the input stream. | |
| SIO4_USC_MONO_RX_STRIP_YES | Strip the Sync Character from the input stream. | |

### 6.8.3. SIO4_IOCTL_USC_MONO_RX_SYNC

This service configures the receiver Sync character when using the Monosync protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_MONO_RX_SYNC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFF | This is the valid range for this service. | |

### 6.8.4. SIO4_IOCTL_USC_MONO_TX_CRC_UNDER

This service configures the transmitter's reaction to an underrun when operating in Monosync mode.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_MONO_TX_CRC_UNDER |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_MONO_TX_CRC_UNDER_NO | Do not output a CRC when an underrun occurs. | |
| SIO4_USC_MONO_TX_CRC_UNDER_YES | Output a CRC when an underrun occurs. | |

### 6.8.5. SIO4_IOCTL_USC_MONO_TX_PREAMBLE

This service configures the transmitter's generation of a preamble sequence when operating in Monosync mode.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_MONO_TX_PREAMBLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_MONO_TX_PREAMBLE_NO | Do not produce a Preamble sequence. | |
| SIO4_USC_MONO_TX_PREAMBLE_YES | Produce a Preamble sequence. | |

### 6.8.6. SIO4_IOCTL_USC_MONO_TX_SHORT

This service configures the Short Sync Character selection for the transmitter when using the Monosync protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_MONO_TX_SHORT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_MONO_SHORT_NO | Do not use a Short Sync Character. Use 8-bits. | |
| SIO4_USC_MONO_SHORT_YES | Use a Short Sync Character. Use less than 8-bits. | |

### 6.8.7. SIO4_IOCTL_USC_MONO_TX_SYNC

This service configures the transmitter Sync character when using the Monosync protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_MONO_TX_SYNC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFF | This is the valid range for this service. | |

## 6.9. USC NBIP Protocol Specific IOCTL Services

These services are specific to the USC's NBIP protocol support. NBIP stands for Nine-Bit Interprocessor Protocol.

### 6.9.1. SIO4_IOCTL_USC_NBIP_RX_CLK_RATE

This service configures the Clock Rate (the oversampling rate) for the receiver when using the Nin-Bit Interprocessor Protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_NBIP_RX_CLK_RATE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_NBIP_CLK_RATE_16X | Divide the source clock by 16. | |
| SIO4_USC_NBIP_CLK_RATE_32X | Divide the source clock by 32. | |
| SIO4_USC_NBIP_CLK_RATE_64X | Divide the source clock by 64. | |

### 6.9.2. SIO4_IOCTL_USC_NBIP_RX_PARITY

This service configures the receiver's use of parity when using the Nin-Bit Interprocessor Protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_NBIP_RX_PARITY |
| arg | s32* |

The table below lists the options used with this service.

| **Macros** | **Description** |
|---|---|
| -1 | Requests the current setting. |
| SIO4_USC_NBIP_PARITY_NO | Do not use Parity. |
| SIO4_USC_NBIP_PARITY_YES | Use Parity. |

### 6.9.3. SIO4_IOCTL_USC_NBIP_TX_ADRS_BIT

This service configures the setting of the address bit for the transmitter when using the Nin-Bit Interprocessor Protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_NBIP_TX_ADRS_BIT |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_NBIP_TX_ADRS_BIT_NO | Do not set the address bit. | |
| SIO4_USC_NBIP_TX_ADRS_BIT_YES | Set the address bit. | |

### 6.9.4. SIO4_IOCTL_USC_NBIP_TX_CLK_RATE

This service configures the Clock Rate for the transmitter when using the Nin-Bit Interprocessor Protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_NBIP_TX_CLK_RATE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_NBIP_CLK_RATE_16X | Divide the source clock by 16. | |
| SIO4_USC_NBIP_CLK_RATE_32X | Divide the source clock by 32. | |
| SIO4_USC_NBIP_CLK_RATE_64X | Divide the source clock by 64. | |

### 6.9.5. SIO4_IOCTL_USC_NBIP_TX_PARITY

This service configures the transmitter's use of parity when using the Nin-Bit Interprocessor Protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_NBIP_TX_PARITY |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_NBIP_PARITY_NO | Do not use Parity. | |
| SIO4_USC_NBIP_PARITY_YES | Use Parity. | |

## 6.10. USC Slaved Monosync Protocol Specific IOCTL Services

These services are specific to the USC's Slaved Monosync protocol support.

### 6.10.1. SIO4_IOCTL_USC_SMONO_RX_SYNC

This service configures the receiver Sync character when using the Slaved Monosync protocol.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_SMONO_RX_SYNC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFF | This is the valid range for this service. | |

### 6.10.2. SIO4_IOCTL_USC_SMONO_TX_ACTIVE

This service configures the USC transmitter activation when using the Slaved Monosync protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_SMONO_TX_ACTIVE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_SMONO_TX_ACTIVE_ON_SYNC | This refers to activation when the transmitter is synchronized. | |
| SIO4_USC_SMONO_TX_ACTIVE_WAIT | This refers to waiting for activation. | |

### 6.10.3. SIO4_IOCTL_USC_SMONO_TX_SHORT

This service configures the Short Sync Character selection for the transmitter when using the Slaved Monosync protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_SMONO_TX_SHORT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_SMONO_TX_SHORT_NO | This refers to the use of normal sync character lengths. Use 8-bits. | |
| SIO4_USC_SMONO_TX_SHORT_YES | This refers to the use of short sync character lengths. Use less than 8-bits. | |

### 6.10.4. SIO4_IOCTL_USC_SMONO_TX_SYNC

This service configures the transmitter Sync character when using the Slaved Monosync protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_SMONO_TX_SYNC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFF | This is the valid range for this service. | |

### 6.10.5. SIO4_IOCTL_USC_SMONO_TX_UNDERRUN

This service configures the Underrun reaction of the transmitter when using the Slaved Monosync protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_SMONO_TX_UNDERRUN |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_SMONO_TX_UNDERRUN_CRC | This refers to the sending of a CRC sequence, | |
| SIO4_USC_SMONO_TX_UNDERRUN_NONE | This refers to no action taking place. | |

## 6.11. USC Transparent BSC Protocol Specific IOCTL Services

These services are specific to the USC's Transparent BSC protocol support.

### 6.11.1. SIO4_IOCTL_USC_TBSC_RX_ENCODING

This service configures the receiver's character encoding format when using the Transparent Bisynchronous Serial Communications protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TBSC_RX_ENCODING |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TBSC_ENCODING_ASCII | This refers to ASCII encoding. | |
| SIO4_USC_TBSC_ENCODING_EBCDIC | This refers to EBCDIC encoding. | |

### 6.11.2. SIO4_IOCTL_USC_TBSC_TX_ENCODING

This service configures the transmitter's character encoding format when using the Transparent Bisynchronous Serial Communications protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TBSC_TX_ENCODING |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TBSC_ENCODING_ASCII | This refers to ASCII encoding. | |
| SIO4_USC_TBSC_ENCODING_EBCDIC | This refers to EBCDIC encoding. | |

### 6.11.3. SIO4_IOCTL_USC_TBSC_TX_PREAMBLE

This service configures the Preamble output selection for the transmitter when using the Transparent Bisynchronous Serial Communications protocol (Transparent Bisync).

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TBSC_TX_PREAMBLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TBSC_TX_PREAMBLE_NO | This refers to the Preamble being disabled. | |
| SIO4_USC_TBSC_TX_PREAMBLE_YES | This refers to the Preamble being enabled. | |

### 6.11.4. SIO4_IOCTL_USC_TBSC_TX_UNDERRUN

This service configures the Underrun reaction selection for the transmitter when using the Transparent Bisynchronous Serial Communications protocol.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TBSC_TX_UNDERRUN |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TBSC_TX_UNDERRUN_C_D_S | This refers to the sequence CRC, DLE and SYN. | |
| SIO4_USC_TBSC_TX_UNDERRUN_C_S | This refers to the sequence CRC and SYN. | |
| SIO4_USC_TBSC_TX_UNDERRUN_D_S | This refers to the sequence DLE and SYN. | |
| SIO4_USC_TBSC_TX_UNDERRUN_S | This refers to the sequence SYN only. | |

## 6.12. USC Signal Routing Specific IOCTL Services

These services are specific to the USC's signal routing support.

### 6.12.1. SIO4_IOCTL_USC_CTS_CFG

This service configures the USC's CTS (Clear To Send) signal functionality.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_CTS_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CTS_CFG_IN_CBL_CTS | Configure the signal as an input for CTS operation. | |
| SIO4_USC_CTS_CFG_OUT_0 | Configure the signal as an output driven low. | |
| SIO4_USC_CTS_CFG_OUT_1 | Configure the signal as an output driven high. | |
| SIO4_USC_CTS_CFG_TRI | Tri-state the signal. | |

### 6.12.2. SIO4_IOCTL_USC_CTS_LEG

This service configures the USC's CTS (Clear To Send) pin for legacy cable interface configurations.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_CTS_LEG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CTS_LEG_IN | Configure the signal as an input. | |
| SIO4_USC_CTS_LEG_OUT_0 | Configure the signal as an output driven low. | |
| SIO4_USC_CTS_LEG_OUT_1 | Configure the signal as an output driven high. | |

### 6.12.3. SIO4_IOCTL_USC_DCD_CFG

This service configures the USC's DCD (Data Carrier Detect) signal functionality.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_DCD_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_DCD_CFG_DISABLE | Disable the signal. | |
| SIO4_USC_DCD_CFG_IN_DCD_CBL_DCD | Use the cable DCD signal for DCD operation. | |
| SIO4_USC_DCD_CFG_IN_SYNC_CBL_DCD | Use the cable DCD signal for SYNC operation. | |
| SIO4_USC_DCD_CFG_OUT_0 | Configure the signal as an output driven low. * | |
| SIO4_USC_DCD_CFG_OUT_1 | Configure the signal as an output driven high. * | |

* This option enables the cable DCD signal to be driven, though the SIO4_IOCTL_Z16_CBL_DCD_CFG IOCTL service (section 5.1.1, page 53) may configure the cable to output an alternate selection.

### 6.12.4. SIO4_IOCTL_USC_DCD_LEG

This service configures the USC's DCD (Data Carrier Detect) pin for legacy cable interface configurations.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_DCD_LEG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_DCD_LEG_IN_DCD | Use the pin as the DCD input. | |
| SIO4_USC_DCD_LEG_IN_SYNC | Use the pin as the SYNC input. | |
| SIO4_USC_DCD_LEG_OUT_0 | Configure the signal as an output driven low. | |
| SIO4_USC_DCD_LEG_OUT_1 | Configure the signal as an output driven high. | |

### 6.12.5. SIO4_IOCTL_USC_RXC_CFG

This service configures the clock source and signal routing of the USC's RxC pin.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_RXC_CFG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_RXC_CFG_IN_0 | This refers to RxC being an input which is driven low. | |
| SIO4_USC_RXC_CFG_IN_1 | This refers to RxC being an input which is driven high. | |
| SIO4_USC_RXC_CFG_IN_CBL_RXAUXC | This refers to RxC being an input which is driven from the Rx Auxiliary Clock signal at the cable interface. | |
| SIO4_USC_RXC_CFG_IN_CBL_RXC | This refers to RxC being an input which is driven from the Rx Clock signal at the cable interface. | |
| SIO4_USC_RXC_CFG_IN_OSC | This refers to RxC being an input which is driven from the on-board oscillator. | |
| SIO4_USC_RXC_CFG_IN_OSC_INV | This refers to RxC being an input which is driven from the inverted form of the on-board oscillator. | |
| SIO4_USC_RXC_CFG_OUT_BRG0 | This refers to RxC being an output which is driven from BRG0. | |
| SIO4_USC_RXC_CFG_OUT_BRG1 | This refers to RxC being an output which is driven from BRG1. | |
| SIO4_USC_RXC_CFG_OUT_CTR0 | This refers to RxC being an output which is driven from CTR0. | |
| SIO4_USC_RXC_CFG_OUT_DPLL_RX | This refers to RxC being an output which is driven from the DPLL's receive output clock. | |
| SIO4_USC_RXC_CFG_OUT_RX_BYTE_CLK | This refers to RxC being an output which is driven from the receiver's Byte Clock. | |

| | |
|---|---|
| SIO4_USC_RXC_CFG_OUT_RX_CLK | This refers to RxC being an output which is driven from the Receive Clock. |
| SIO4_USC_RXC_CFG_OUT_SYNC | This refers to RxC being an output which is driven from the Sync signal. |

### 6.12.6. SIO4_IOCTL_USC_RXC_LEG

This service configures the signal routing of the USC's RxC pin for legacy cable interface configurations.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RXC_LEG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_RXC_LEG_IN | This refers to RxC being an input. | |
| SIO4_USC_RXC_LEG_OUT_BRG0 | This refers to RxC being an output which is driven from BRG0. | |
| SIO4_USC_RXC_LEG_OUT_BRG1 | This refers to RxC being an output which is driven from BRG1. | |
| SIO4_USC_RXC_LEG_OUT_CTR0 | This refers to RxC being an output which is driven from CTR0. | |
| SIO4_USC_RXC_LEG_OUT_DPLL_RX | This refers to RxC being an output which is driven from the DPLL's receive output clock. | |
| SIO4_USC_RXC_LEG_OUT_RX_BYTE_CLK | This refers to RxC being an output which is driven from the receiver's Byte Clock. | |
| SIO4_USC_RXC_LEG_OUT_RX_CLK | This refers to RxC being an output which is driven from the Receive Clock. | |
| SIO4_USC_RXC_LEG_OUT_SYNC | This refers to RxC being an output which is driven from the Sync signal. | |

### 6.12.7. SIO4_IOCTL_USC_TXC_CFG

This service configures the clock source and signal routing of the USC's TxC pin.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TXC_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TXC_CFG_IN_0 | This refers to TxC being an input which is driven low. | |
| SIO4_USC_TXC_CFG_IN_1 | This refers to TxC being an input which is driven high. | |
| SIO4_USC_TXC_CFG_IN_CBL_RXAUXC | This refers to TxC being an input which is driven from | |

| | |
|---|---|
| | the Rx Auxiliary Clock signal at the cable interface. |
| SIO4_USC_TXC_CFG_IN_CBL_RXC | This refers to TxC being an input which is driven from the Rx Clock signal at the cable interface. |
| SIO4_USC_TXC_CFG_IN_OSC | This refers to TxC being an input which is driven from the on-board oscillator. |
| SIO4_USC_TXC_CFG_IN_OSC_INV | This refers to TxC being an input which is driven from the inverted form of the on-board oscillator. |
| SIO4_USC_TXC_CFG_OUT_BRG0 | This refers to TxC being an output which is driven from BRG0. |
| SIO4_USC_TXC_CFG_OUT_BRG1 | This refers to TxC being an output which is driven from BRG1. |
| SIO4_USC_TXC_CFG_OUT_CTR1 | This refers to TxC being an output which is driven from CTR1. |
| SIO4_USC_TXC_CFG_OUT_DPLL_TX | This refers to TxC being an output which is driven from the DPLL's transmitter output clock. |
| SIO4_USC_TXC_CFG_OUT_TX_BYTE_CLK | This refers to TxC being an output which is driven from the transmitter's Byte Clock. |
| SIO4_USC_TXC_CFG_OUT_TX_CLK | This refers to TxC being an output which is driven from the Transmit Clock. |
| SIO4_USC_TXC_CFG_OUT_TX_COMP | This refers to TxC being an output which is driven from the Complete signal. |

### 6.12.8. SIO4_IOCTL_USC_TXC_LEG

This service configures the signal routing of the USC's TxC pin for legacy cable interface configurations.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_TXC_LEG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TXC_LEG_IN | This refers to TxC being an input. | |
| SIO4_USC_TXC_LEG_OUT_BRG0 | This refers to TxC being an output which is driven from BRG0. | |
| SIO4_USC_TXC_LEG_OUT_BRG1 | This refers to TxC being an output which is driven from BRG1. | |
| SIO4_USC_TXC_LEG_OUT_CTR1 | This refers to TxC being an output which is driven from CTR1. | |
| SIO4_USC_TXC_LEG_OUT_DPLL_TX | This refers to TxC being an output which is driven from the DPLL's transmitter output clock. | |
| SIO4_USC_TXC_LEG_OUT_TX_BYTE_CLK | This refers to TxC being an output which is driven from the transmitter's Byte Clock. | |
| SIO4_USC_TXC_LEG_OUT_TX_CLK | This refers to TxC being an output which is driven from the Transmit Clock. | |
| SIO4_USC_TXC_LEG_OUT_TX_COMP | This refers to TxC being an output which is driven from the Complete signal. | |

### 6.12.9. SIO4_IOCTL_USC_TXD_CFG

This service configures the clock source and signal routing of the USC's TxD pin.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_TXD_CFG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TXD_CFG_OUT_0 | This refers to TxD being an output which is driven low. | |
| SIO4_USC_TXD_CFG_OUT_1 | This refers to TxD being an output which is driven high. | |
| SIO4_USC_TXD_CFG_OUT_TXD | This refers to TxD being an output which is driven to the Tx Data signal at the cable interface. | |
| SIO4_USC_TXD_CFG_TRI | This refers to TxD being tri-stated. | |

## 6.13. USC Bit Rate Configuration Specific IOCTL Services

These services are specific to the USC's bit rate configuration support.

### 6.13.1. SIO4_IOCTL_USC_BRG0_CLK_SRC

This service configures the Baud Rate Generator Zero Clock Source.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BRG0_CLK_SRC |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BRG_CLK_SRC_CTR0 | Drive BRG0 from the output of Counter Zero (CTR0). | |
| SIO4_USC_BRG_CLK_SRC_CTR1 | Drive BRG0 from the output of Counter One (CTR1). | |
| SIO4_USC_BRG_CLK_SRC_RXC_PIN | Drive BRG0 from the signal on the USC RxC pin. | |
| SIO4_USC_BRG_CLK_SRC_TXC_PIN | Drive BRG0 from the signal on the USC TxC pin. | |

### 6.13.2. SIO4_IOCTL_USC_BRG0_ENABLE

This service enables or disables Baud Rate Generator Zero.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_BRG0_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BRG_ENABLE_NO | This disables BRG0. | |
| SIO4_USC_BRG_ENABLE_YES | This enables BRG0. | |

### 6.13.3. SIO4_IOCTL_USC_BRG0_MODE

This service configures the Baud Rate Generator Zero operating mode.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_BRG0_MODE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BRG_MODE_CONT | Run BRG0 continuously. | |
| SIO4_USC_BRG_MODE_SINGLE | Run BRG0 once then stop when it rolls over to zero. | |

### 6.13.4. SIO4_IOCTL_USC_BRG1_CLK_SRC

This service configures the Baud Rate Generator One Clock Source.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_BRG1_CLK_SRC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BRG_CLK_SRC_CTR0 | Drive BRG1 from the output of Counter Zero (CTR0). | |
| SIO4_USC_BRG_CLK_SRC_CTR1 | Drive BRG1 from the output of Counter One (CTR1). | |
| SIO4_USC_BRG_CLK_SRC_RXC_PIN | Drive BRG1 from the signal on the USC RxC pin. | |
| SIO4_USC_BRG_CLK_SRC_TXC_PIN | Drive BRG1 from the signal on the USC TxC pin. | |

### 6.13.5. SIO4_IOCTL_USC_BRG1_ENABLE

This service enables or disables Baud Rate Generator One.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_BRG1_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BRG_ENABLE_NO | This disables BRG1. | |
| SIO4_USC_BRG_ENABLE_YES | This enables BRG1. | |

### 6.13.6. SIO4_IOCTL_USC_BRG1_MODE

This service configures the Baud Rate Generator One operating mode.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_BRG1_MODE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_BRG_MODE_CONT | Run BRG1 continuously. | |
| SIO4_USC_BRG_MODE_SINGLE | Run BRG1 once then stop when it rolls over to zero. | |

### 6.13.7. SIO4_IOCTL_USC_CTR0_CLK_SRC

This service configures the Counter Zero Clock Source.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_CTR0_CLK_SRC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CTR_CLK_SRC_DISABLE | Disable CTR0. | |
| SIO4_USC_CTR_CLK_SRC_RXC_PIN | Drive CTR0 from the signal on the USC RxC pin. | |
| SIO4_USC_CTR_CLK_SRC_TXC_PIN | Drive CTR0 from the signal on the USC TxC pin. | |

### 6.13.8. SIO4_IOCTL_USC_CTR0_RATE

This service configures the Counter Zero dividing Rate.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_CTR0_RATE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CTR0_RATE_4X | Divide the source clock by 4. | |
| SIO4_USC_CTR0_RATE_8X | Divide the source clock by 8. | |
| SIO4_USC_CTR0_RATE_16X | Divide the source clock by 16. | |
| SIO4_USC_CTR0_RATE_32X | Divide the source clock by 32. | |

### 6.13.9. SIO4_IOCTL_USC_CTR1_CLK_SRC

This service configures the Counter One Clock Source.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_CTR1_CLK_SRC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CTR_CLK_SRC_DISABLE | Disable CTR1. | |
| SIO4_USC_CTR_CLK_SRC_RXC_PIN | Drive CTR1 from the signal on the USC RxC pin. | |
| SIO4_USC_CTR_CLK_SRC_TXC_PIN | Drive CTR1 from the signal on the USC TxC pin. | |

### 6.13.10. SIO4_IOCTL_USC_CTR1_RATE

This service configures the Counter One dividing Rate.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_CTR1_RATE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CTR1_RATE_CTR0 | Use the rate configured for CTR0. | |
| SIO4_USC_CTR1_RATE_DPLL | Use the rate configured for the DPLL. | |

### 6.13.11. SIO4_IOCTL_USC_TC0

This service configures the Time Constant Zero values used by BRG0.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TC0 |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

### 6.13.12. SIO4_IOCTL_USC_TC1

This service configures the Time Constant One values used by BRG1.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TC1 |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

## 6.14. USC DPLL Configuration Specific IOCTL Services

These services are specific to the USC's DPLL configuration support.

### 6.14.1. SIO4_IOCTL_USC_DPLL_ADJ_SYNC

This service configures the DPLL's synchronization functionality.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_DPLL_ADJ_SYNC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_DPLL_ADJ_SYNC_BOTH_EDGE | Have the DPLL synchronize on both falling and rising edges. | |
| SIO4_USC_DPLL_ADJ_SYNC_FALL_EDGE | Have the DPLL synchronize on falling edges only. | |
| SIO4_USC_DPLL_ADJ_SYNC_INHIBIT | Inhibit the DPLL from synchronizing. | |
| SIO4_USC_DPLL_ADJ_SYNC_RISE_EDGE | Have the DPLL synchronize on rising edges only. | |

### 6.14.2. SIO4_IOCTL_USC_DPLL_CLK_SRC

This service configures the DPLL's clock source.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_DPLL_CLK_SRC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_DPLL_CLK_SRC_BRG0 | Drive the DPLL from the output of BRG0. | |
| SIO4_USC_DPLL_CLK_SRC_BRG1 | Drive the DPLL from the output of BRG1. | |
| SIO4_USC_DPLL_CLK_SRC_RXC_PIN | Drive the DPLL from the signal on the USC RxC pin. | |
| SIO4_USC_DPLL_CLK_SRC_TXC_PIN | Drive the DPLL from the signal on the USC TxC pin. | |

### 6.14.3. SIO4_IOCTL_USC_DPLL_MISS_1

This service operates on the DPLL's status reported when it misses one clock cycle.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_DPLL_MISS_1 |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current status. | The service is unsupported. |
| SIO4_USC_DPLL_MISS_1_CLEAR | This clears the status. | N/A |
| SIO4_USC_DPLL_MISS_1_NO | N/A | The DPLL did not miss a clock cycle. |
| SIO4_USC_DPLL_MISS_1_YES | N/A | The DPLL missed a clock cycle. |

### 6.14.4. SIO4_IOCTL_USC_DPLL_MISS_2

This service operates on the DPLL's status reported when it misses two clock cycles.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_DPLL_MISS_2 |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current status. | The service is unsupported. |
| SIO4_USC_DPLL_MISS_2_CLEAR | This clears the status. | N/A |
| SIO4_USC_DPLL_MISS_2_NO | N/A | The DPLL missed less than two clock cycles. |
| SIO4_USC_DPLL_MISS_2_YES | N/A | The DPLL missed two or more clock cycles. |

### 6.14.5. SIO4_IOCTL_USC_DPLL_MODE

This service configures the DPLL's operating mode, which corresponds to the encoding of the source clock.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_DPLL_MODE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_DPLL_MODE_BIPH_LVL | The source signal is encoded as Biphase Level. | |
| SIO4_USC_DPLL_MODE_BIPH_MS | The source signal is encoded as Biphase Mark or Space. | |
| SIO4_USC_DPLL_MODE_DISABLE | Disable the DPLL. | |
| SIO4_USC_DPLL_MODE_NRZ_NRZI | The source signal is encoded as NRZ or NRZ Inverted. | |

### 6.14.6. SIO4_IOCTL_USC_DPLL_RATE

This service configures the DPLL's clock dividing rate.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_DPLL_RATE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_DPLL_RATE_CTR1_4X | Divide the source by four. This is for CTR1 operation only. The DPLL should be disabled when this option is used. | |
| SIO4_USC_DPLL_RATE_8X | Divide the source by eight. | |
| SIO4_USC_DPLL_RATE_16X | Divide the source by 16. | |
| SIO4_USC_DPLL_RATE_32X | Divide the source by 32. | |

### 6.14.7. SIO4_IOCTL_USC_DPLL_SYNC

This service operates on the DPLL's synchronization status.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_DPLL_SYNC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_DPLL_SYNC_START | This requests that the DPLL synchronize on its source signal. | N/A |
| SIO4_USC_DPLL_SYNC_NO | The DPLL is not synchronized. | |
| SIO4_USC_DPLL_SYNC_YES | The DPLL is synchronized. | |

## 6.15. USC CRC Specific IOCTL Services

These services are specific to the USC's CRC support.

### 6.15.1. SIO4_IOCTL_USC_RX_CRC_ENABLE

This service configures the receiver's use of CRC.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_RX_CRC_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CRC_ENABLE_NO | This refers to the receiver not using CRCs. | |
| SIO4_USC_CRC_ENABLE_YES | This refers to the receiver using CRCs. | |

### 6.15.2. SIO4_IOCTL_USC_RX_CRC_PRESET

This service configures the receiver's CRC starting value when CRC use is enabled.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_RX_CRC_PRESET |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CRC_PRESET_ALL_0 | This refers to a starting value in which each bit is zero. | |
| SIO4_USC_CRC_PRESET_ALL_1 | This refers to a starting value in which each bit is one. | |

### 6.15.3. SIO4_IOCTL_USC_RX_CRC_TYPE

This service configures the receiver's CRC type when CRC use is enabled.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_RX_CRC_TYPE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CRC_TYPE_16 | This refers to a 16-bit CRC. | |
| SIO4_USC_CRC_TYPE_32 | This refers to a 32-bit CRC. | |
| SIO4_USC_CRC_TYPE_CCITT | This refers to the CCITT style CRC. | |

### 6.15.4. SIO4_IOCTL_USC_TX_CRC_ENABLE

This service configures the transmitter's use of CRC.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_CRC_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CRC_ENABLE_NO | This refers to the transmitter not using CRCs. | |
| SIO4_USC_CRC_ENABLE_YES | This refers to the transmitter using CRCs. | |

### 6.15.5. SIO4_IOCTL_USC_TX_CRC_ON_END

This service configures the transmitter's sending of a CRC at the end of a frame or message.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_CRC_ON_END |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TX_CRC_ON_END_NO | This refers to the transmitter not sending a CRC. | |
| SIO4_USC_TX_CRC_ON_END_YES | This refers to the transmitter sending a CRC. | |

### 6.15.6. SIO4_IOCTL_USC_TX_CRC_PRESET

This service configures the transmitter's CRC starting value when CRC use is enabled.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_CRC_PRESET |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CRC_PRESET_ALL_0 | This refers to a starting value in which each bit is zero. | |
| SIO4_USC_CRC_PRESET_ALL_1 | This refers to a starting value in which each bit is one. | |

### 6.15.7. SIO4_IOCTL_USC_TX_CRC_TYPE

This service configures the transmitter's CRC type when CRC use is enabled.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_CRC_TYPE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CRC_TYPE_16 | This refers to a 16-bit CRC. | |
| SIO4_USC_CRC_TYPE_32 | This refers to a 32-bit CRC. | |
| SIO4_USC_CRC_TYPE_CCITT | This refers to the CCITT style CRC. | |

## 6.16. USC Parity Specific IOCTL Services

These services are specific to the USC's parity support.

### 6.16.1. SIO4_IOCTL_USC_RX_PAR_ENABLE

This service configures the USC receiver's use of Parity.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_PAR_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_PAR_ENABLE_NO | This refers to Parity use being disabled. | |
| SIO4_USC_PAR_ENABLE_YES | This refers to Parity use being enabled. | |

### 6.16.2. SIO4_IOCTL_USC_RX_PAR_TYPE

This service configures the type of parity used by the USC receiver when Parity use is enabled.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_PAR_TYPE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_PAR_TYPE_EVEN | This refers to Even Parity. | |
| SIO4_USC_PAR_TYPE_ODD | This refers to Odd Parity. | |
| SIO4_USC_PAR_TYPE_ONE | This refers to One Parity (parity bit is always one). | |
| SIO4_USC_PAR_TYPE_ZERO | This refers to Zero Parity (parity bit is always zero). | |

### 6.16.3. SIO4_IOCTL_USC_TX_PAR_ENABLE

This service configures the USC transmitter's use of Parity.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_PAR_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_PAR_ENABLE_NO | This refers to Parity use being disabled. | |
| SIO4_USC_PAR_ENABLE_YES | This refers to Parity use being enabled. | |

### 6.16.4. SIO4_IOCTL_USC_TX_PAR_TYPE

This service configures the type of parity used by the USC transmitter when Parity use is enabled.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_PAR_TYPE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_PAR_TYPE_EVEN | This refers to Even Parity. | |
| SIO4_USC_PAR_TYPE_ODD | This refers to Odd Parity. | |
| SIO4_USC_PAR_TYPE_ONE | This refers to One Parity (parity bit is always one). | |
| SIO4_USC_PAR_TYPE_ZERO | This refers to Zero Parity (parity bit is always zero). | |

## 6.17. USC Miscellaneous IOCTL Services

### 6.17.1. SIO4_IOCTL_USC_ACCEPT_CV

This service configures the Code Violations. This is not applicable to all protocols.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_ACCEPT_CV |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ACCEPT_CV_NO | Do not accept Code Violations. | |
| SIO4_USC_ACCEPT_CV_YES | Accept Code Violations. | |

### 6.17.2. SIO4_IOCTL_USC_LOOP_SENDING

This service reports the channel's Loop Sending status.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_LOOP_SENDING |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_LOOP_SENDING_NO | N/A | The Loop Sending status is negated. |
| SIO4_USC_LOOP_SENDING_YES | N/A | The Loop Sending status is asserted. |

### 6.17.3. SIO4_IOCTL_USC_ON_LOOP

This service reports the On Loop status for the USC.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_ON_LOOP |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current status. | The service is unsupported. |
| SIO4_USC_ON_LOOP_NO | N/A | The On Loop status is negated. |
| SIO4_USC_ON_LOOP_YES | N/A | The On Loop status is asserted. |

### 6.17.4. SIO4_IOCTL_USC_OPER_MODE

This service configures the basic operating mode of the USC.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_OPER_MODE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_OPER_MODE_AUTO_ECHO | The activity at the receiver is duplicated at the transmitter. | |
| SIO4_USC_OPER_MODE_EXT_LOOPBACK | The USC operates in External Loopback mode for testing. | |
| SIO4_USC_OPER_MODE_INT_LOOPBACK | The USC operates in Internal Loopback mode for testing. | |
| SIO4_USC_OPER_MODE_NORMAL | The USC operates in its normal mode where data is received and transmitted per the protocols selected. | |

### 6.17.5. SIO4_IOCTL_USC_RCC_FIFO_CLEAR

This service operates on the Receive Character Counter FIFO.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RCC_FIFO_CLEAR |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 | The service is supported. | N/A |
| SIO4_USC_RCC_FIFO_CLEAR_NO | This refers to the FIFO not being cleared. | |
| SIO4_USC_RCC_FIFO_CLEAR_YES | This refers to the FIFO being cleared. | |

### 6.17.6. SIO4_IOCTL_USC_RCC_FIFO_OVERRUN

This service reports on the Overrun status of the Receive Character Counter FIFO.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RCC_FIFO_OVERRUN |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current status. | The service is unsupported. |
| SIO4_USC_RCC_FIFO_OVERRUN_NO | N/A | An overrun has not occurred. |
| SIO4_USC_RCC_FIFO_OVERRUN_YES | N/A | An overrun has occurred. |

### 6.17.7. SIO4_IOCTL_USC_RCC_FIFO_VALID

This service reports on the validity of the Receive Character Counter FIFO content.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RCC_FIFO_VALID |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_RCC_FIFO_VALID_NO | N/A | The RCC FIFO does not contain valid data. |
| SIO4_USC_RCC_FIFO_VALID_YES | N/A | The RCC FIFO does contain valid data. |

### 6.17.8. SIO4_IOCTL_USC_RESET

This service resets the USC. The USC is also reset when the channel is initialized (see SIO4_IOCTL_INITIALIZE, section 4.4.3, page 40).

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RESET |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 | Reset the USC. | The service is supported or the USC was reset. |

### 6.17.9. SIO4_IOCTL_USC_SEND_COMMAND

This service sends a command to the USC.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_SEND_COMMAND |
| arg | s32* |

The table below lists the options used with this service. Refer to the USC hardware manual for additional information on sending commands to the USC.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_SEND_CMD_FIFOS_PURGE | Purges the USC FIFOs. | |
| SIO4_USC_SEND_CMD_LD_CHAR_CNTS | Load the Receive and Transmit Character Counts. | |
| SIO4_USC_SEND_CMD_LD_RX_CHAR_CNT | Load the Receive Character Count. | |
| SIO4_USC_SEND_CMD_LD_TC0 | Load the TC0 count. | |
| SIO4_USC_SEND_CMD_LD_TC0_TC1 | Load the TC0 and TC1 counts. | |
| SIO4_USC_SEND_CMD_LD_TC1 | Load the TC1 count. | |
| SIO4_USC_SEND_CMD_LD_TX_CHAR_CNT | Load the Transmit Character Count. | |
| SIO4_USC_SEND_CMD_NONE | This is ignored. | |
| SIO4_USC_SEND_CMD_RESET_H_IUS | Reset the Highest Interrupt Under Service. | |
| SIO4_USC_SEND_CMD_RX_FIFO_PURGE | Purge the USC receive FIFO. | |
| SIO4_USC_SEND_CMD_RX_PURGE | Purge the receiver. | |
| SIO4_USC_SEND_CMD_SEL_LSB_FIRST | Select Least Significant Byte First processing. | |
| SIO4_USC_SEND_CMD_SEL_MSB_FIRST | Select Most Significant Byte First processing. | |
| SIO4_USC_SEND_CMD_SEL_STRAIT_MEM | Select Strain memory access. | |
| SIO4_USC_SEND_CMD_SEL_SWAP_MEM | Select Swapped memory access. | |
| SIO4_USC_SEND_CMD_TRIG_LD_DMA | Trigger channel loading via DMA. | |
| SIO4_USC_SEND_CMD_TRIG_RX_DMA | Trigger the receiver to resume data transfer. | |
| SIO4_USC_SEND_CMD_TRIG_TX_DMA | Trigger the transmitter to resume data transfer. | |
| SIO4_USC_SEND_CMD_TRIG_TX_RX_DMA | Trigger the receiver and transmitter to resume data transfer. | |
| SIO4_USC_SEND_CMD_TX_FIFO_PURGE | Purge the USC transmit FIFO. | |

## 6.18. USC Receiver Specific IOCTL Services

These services are specific to the USC's receiver.

### 6.18.1. SIO4_IOCTL_USC_RX_CHAR_CNT

This service reports the receive Character Count, which is the size of the most recent frame or message.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_CHAR_CNT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | N/A | This is the valid range for this service. |

### 6.18.2. SIO4_IOCTL_USC_RX_CHAR_CNT_LIM

This service configures the upper size limit for received frames or messages.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_CHAR_CNT_LIM |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

### 6.18.3. SIO4_IOCTL_USC_RX_CHAR_LEN

This service configures the bit length of received characters.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_CHAR_LEN |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CHAR_LEN_1 | This refers to a character length of 1-bit. | |
| SIO4_USC_CHAR_LEN_2 | This refers to a character length of 2-bits. | |
| SIO4_USC_CHAR_LEN_3 | This refers to a character length of 3-bits. | |
| SIO4_USC_CHAR_LEN_4 | This refers to a character length of 4-bits. | |
| SIO4_USC_CHAR_LEN_5 | This refers to a character length of 5-bits. | |
| SIO4_USC_CHAR_LEN_6 | This refers to a character length of 6-bits. | |
| SIO4_USC_CHAR_LEN_7 | This refers to a character length of 7-bits. | |
| SIO4_USC_CHAR_LEN_8 | This refers to a character length of 8-bits. | |

### 6.18.4. SIO4_IOCTL_USC_RX_CLK_SRC

This service configures receiver clock source selection.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_CLK_SRC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CLK_SRC_BRG0 | This refers to the output from BRG0. | |
| SIO4_USC_CLK_SRC_BRG1 | This refers to the output from BRG1. | |
| SIO4_USC_CLK_SRC_CTR0 | This refers to the output from CTR0. | |
| SIO4_USC_CLK_SRC_CTR1 | This refers to the output from CTR1. | |
| SIO4_USC_CLK_SRC_DPLL | This refers to the output from DPLL. | |
| SIO4_USC_CLK_SRC_DISABLE | This disables the receiver's operation. | |

| | |
|---|---|
| SIO4_USC_CLK_SRC_RXC_PIN | This refers to the USC's RxC pin. |
| SIO4_USC_CLK_SRC_TXC_PIN | This refers to the USC's TxC pin. |

### 6.18.5. SIO4_IOCTL_USC_RX_CMD

This service sends a command to the USC receiver.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_RX_CMD |
| arg | s32* |

The table below lists the options used with this service. Refer to the USC hardware manual for additional information on sending commands to the receiver.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_RX_CMD_ENTER_HUNT_MODE | This refers to the USC Enter Hunt Mode operation. | |
| SIO4_USC_RX_CMD_NULL | This refers to no command at all. | |
| SIO4_USC_RX_CMD_PRESET_CRC | This refers to presetting the CRC. | |
| SIO4_USC_RX_CMD_SEL_FIFO_STATUS | This refers to selection of the USC's Rx FIFO status. | |
| SIO4_USC_RX_CMD_SEL_FIFO_INT_LVL | This refers to USC Rx FIFO fill threshold level for interrupt generation. | |
| SIO4_USC_RX_CMD_SEL_FIFO_STS_LVL | This refers to USC Rx FIFO fill level. | |

### 6.18.6. SIO4_IOCTL_USC_RX_DATA_ENCODE

This service configures the data encoding format used by the USC receiver.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_RX_DATA_ENCODE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_DATA_ENCODE_BI_LEVEL | This refers to Biphase-Level encoding (Manchester). | |
| SIO4_USC_DATA_ENCODE_BI_MARK | This refers to Biphase-Mark encoding (FM1). | |
| SIO4_USC_DATA_ENCODE_BI_SPACE | This refers to Biphase-Space encoding (FM0). | |
| SIO4_USC_DATA_ENCODE_D_BI_LEVEL | This refers to Differential Biphase Level encoding (Differential Manchester). | |
| SIO4_USC_DATA_ENCODE_NRZ | This refers to Non-Return to Zero encoding. | |
| SIO4_USC_DATA_ENCODE_NRZB | This refers to Inverted NRZ encoding. | |
| SIO4_USC_DATA_ENCODE_NRZI_MARK | This refers to NRZI-Mark encoding. | |
| SIO4_USC_DATA_ENCODE_NRZI_SPACE | This refers to NRZI-Space encoding. | |

### 6.18.7. SIO4_IOCTL_USC_RX_ENABLE

This service configures the USC receiver enabled state.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ENABLE_NO_AFTER | This refers to the receiver being disabled after receipt of the current character, if one is being received. | |
| SIO4_USC_ENABLE_NO_NOW | This refers to the receiver being disabled immediately. | |
| SIO4_USC_ENABLE_YES_NOW | This refers to the receiver being enabled immediately. | |
| SIO4_USC_ENABLE_YES_W_AE | This refers to the receiver being enabled immediately along with the hardware flow control signals. | |

### 6.18.8. SIO4_IOCTL_USC_RX_MODE

This service configures the protocol used by the USC receiver for data reception.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_MODE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_MODE_8023 | This refers to the 802.3 protocol. | |
| SIO4_USC_MODE_ASY_CV | This refers to the Asynchronous with Code Violations protocol. | |
| SIO4_USC_MODE_ASYNC | This refers to the Asynchronous protocol. | |
| SIO4_USC_MODE_BSC | This refers to the Bisynchronous Serial Communications protocol. | |
| SIO4_USC_MODE_E_SYNC | This refers to the External Sync protocol. | |
| SIO4_USC_MODE_HDLC | This refers to the HDLC protocol. | |
| SIO4_USC_MODE_ISOC | This refers to the Isochronous protocol. | |
| SIO4_USC_MODE_MONO | This refers to the Monosync protocol. | |
| SIO4_USC_MODE_NBIP | This refers to the Nine-Bit Interprocessor Protocol. | |
| SIO4_USC_MODE_TBSC | This refers to the Transparent Bisynchronous Serial Communications protocol. | |

### 6.18.9. SIO4_IOCTL_USC_RX_QUEUE_ABORT

This service operates on the USC receiver to queue Abort detection status through the Rx FIFO.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_QUEUE_ABORT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_RX_QUEUE_ABORT_NO | This specifies to not queue the status through the Rx FIFO. | |
| SIO4_USC_RX_QUEUE_ABORT_YES | This specifies to queue the status through the Rx FIFO. | |

### 6.18.10. SIO4_IOCTL_USC_RX_STATUS

This service retrieves the USC receiver status, which is essentially the value read from the Receive Command/Status Register.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_STATUS |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | N/A | This is the valid range for this service. |

### 6.18.11. SIO4_IOCTL_USC_RX_STATUS_BLOCK

This service configures the USC receiver's use of Receive Status Blocks.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_STATUS_BLOCK |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_STATUS_BLOCK_1_WORD | This refers to Receive Status Blocks consisting of one word (two bytes). | |
| SIO4_USC_STATUS_BLOCK_2_WORD | This refers to Receive Status Blocks consisting of two words (four bytes). | |
| SIO4_USC_STATUS_BLOCK_NO | This refers to not using Receive Status Blocks. | |

### 6.18.12. SIO4_IOCTL_USC_RX_WAIT_DMA_TRIG

This service configures the USC receiver's pausing of data transfer to the channel's external Rx FIFO.

> **NOTE**: By default each channel is configured to automatically transfer data to the channel's external Rx FIFO as data is received by the USC. This service deals with the pausing of such data transfer until commanded to resume the transfer.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_RX_WAIT_DMA_TRIG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_WAIT_DMA_TRIG_NO | This refers to data transfer being unimpeded. | |
| SIO4_USC_WAIT_DMA_TRIG_YES | This refers to data transfer being paused. | |

## 6.19. USC Transmitter Specific IOCTL Services

These services are specific to the USC's transmitter.

### 6.19.1. SIO4_IOCTL_USC_TX_CHAR_CNT

This service configures the size of the next transmission frame or message.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_CHAR_CNT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

### 6.19.2. SIO4_IOCTL_USC_TX_CHAR_CNT_LIM

This service configures the upper size limit for transmission frames or messages.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_CHAR_CNT_LIM |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

### 6.19.3. SIO4_IOCTL_USC_TX_CHAR_LEN

This service configures the bit length of transmitted characters.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_CHAR_LEN |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CHAR_LEN_1 | This refers to a character length of 1-bit. | |
| SIO4_USC_CHAR_LEN_2 | This refers to a character length of 2-bits. | |
| SIO4_USC_CHAR_LEN_3 | This refers to a character length of 3-bits. | |
| SIO4_USC_CHAR_LEN_4 | This refers to a character length of 4-bits. | |
| SIO4_USC_CHAR_LEN_5 | This refers to a character length of 5-bits. | |
| SIO4_USC_CHAR_LEN_6 | This refers to a character length of 6-bits. | |
| SIO4_USC_CHAR_LEN_7 | This refers to a character length of 7-bits. | |
| SIO4_USC_CHAR_LEN_8 | This refers to a character length of 8-bits. | |

### 6.19.4. SIO4_IOCTL_USC_TX_CLK_SRC

This service configures transmitter clock source selection.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_CLK_SRC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CLK_SRC_BRG0 | This refers to the output from BRG0. | |
| SIO4_USC_CLK_SRC_BRG1 | This refers to the output from BRG1. | |
| SIO4_USC_CLK_SRC_CTR0 | This refers to the output from CTR0. | |
| SIO4_USC_CLK_SRC_CTR1 | This refers to the output from CTR1. | |
| SIO4_USC_CLK_SRC_DPLL | This refers to the output from DPLL. | |
| SIO4_USC_CLK_SRC_DISABLE | This disables the receiver's operation. | |
| SIO4_USC_CLK_SRC_RXC_PIN | This refers to the USC's RxC pin. | |
| SIO4_USC_CLK_SRC_TXC_PIN | This refers to the USC's TxC pin. | |

### 6.19.5. SIO4_IOCTL_USC_TX_CMD

This service sends a command to the USC transmitter.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_CMD |
| arg | s32* |

The table below lists the options used with this service. Refer to the USC hardware manual for additional information on sending commands to the transmitter.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TX_CMD_NULL | This refers to no command at all. | |
| SIO4_USC_TX_CMD_PRESET_CRC | This refers to presetting the CRC. | |
| SIO4_USC_TX_CMD_ENTER_HUNT_MODE | This refers to the USC Enter Hunt Mode operation. | |
| SIO4_USC_TX_CMD_RST_DLE_INHIBIT | This refers to resetting the DLE inhibit status. | |
| SIO4_USC_TX_CMD_RST_EOF_EOM | This refers to resetting the End of Frame or End of Message status. | |
| SIO4_USC_TX_CMD_SEL_FIFO_INT_LVL | This refers to USC Tx FIFO fill threshold level for interrupt generation. | |
| SIO4_USC_TX_CMD_SEL_FIFO_STATUS | This refers to selection of the USC's Tx FIFO status. | |
| SIO4_USC_TX_CMD_SEL_FIFO_STS_LVL | This refers to USC Tx FIFO fill level. | |
| SIO4_USC_TX_CMD_SEND_ABORT | This refers to sending an abort sequence. | |
| SIO4_USC_TX_CMD_SEND_FRM_MSG | This refers to sending a frame or message. | |
| SIO4_USC_TX_CMD_SET_DLE_INHIBIT | This refers to setting the DLE inhibit status. | |
| SIO4_USC_TX_CMD_SET_EOF_EOM | This refers to setting the End of Frame or End of Message status. | |

### 6.19.6. SIO4_IOCTL_USC_TX_CTRL_BLOCK

This service configures the USC transmitter's use of Transmit Control Blocks.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_CTRL_BLOCK |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_CTRL_BLOCK_1_WORD | This refers to Transmit Status Blocks consisting of one word (two bytes). | |
| SIO4_USC_CTRL_BLOCK_2_WORD | This refers to Transmit Status Blocks consisting of two words (four bytes). | |
| SIO4_USC_CTRL_BLOCK_NO | This refers to not using Transmit Status Blocks. | |

### 6.19.7. SIO4_IOCTL_USC_TX_DATA_ENCODE

This service configures the data encoding format used by the USC transmitter.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_DATA_ENCODE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_DATA_ENCODE_BI_LEVEL | This refers to Biphase-Level encoding (Manchester). | |
| SIO4_USC_DATA_ENCODE_BI_MARK | This refers to Biphase-Mark encoding (FM1). | |
| SIO4_USC_DATA_ENCODE_BI_SPACE | This refers to Biphase-Space encoding (FM0). | |
| SIO4_USC_DATA_ENCODE_D_BI_LEVEL | This refers to Differential Biphase Level encoding (Differential Manchester). | |
| SIO4_USC_DATA_ENCODE_NRZ | This refers to Non-Return to Zero encoding. | |
| SIO4_USC_DATA_ENCODE_NRZB | This refers to Inverted NRZ encoding. | |
| SIO4_USC_DATA_ENCODE_NRZI_MARK | This refers to NRZI-Mark encoding. | |
| SIO4_USC_DATA_ENCODE_NRZI_SPACE | This refers to NRZI-Space encoding. | |

### 6.19.8. SIO4_IOCTL_USC_TX_ENABLE

This service configures the USC transmitter enabled state.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_ENABLE_NO_AFTER | This refers to the transmitter being disabled after sending of the current character, if one is being sent. | |
| SIO4_USC_ENABLE_NO_NOW | This refers to the transmitter being disabled immediately. | |
| SIO4_USC_ENABLE_YES_NOW | This refers to the transmitter being enabled immediately. | |
| SIO4_USC_ENABLE_YES_W_AE | This refers to the transmitter being enabled immediately along with the hardware flow control signals. | |

### 6.19.9. SIO4_IOCTL_USC_TX_IDLE_COND

This service configures the USC transmitter output on the Tx Data signal when no data is available to send.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_IDLE_COND |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TX_IDLE_COND_0 | This refers to a continuous low. | |
| SIO4_USC_TX_IDLE_COND_0_1 | This refers to a continuous alternating low and high sequence. | |
| SIO4_USC_TX_IDLE_COND_1 | This refers to a continuous high. | |
| SIO4_USC_TX_IDLE_COND_DEFAULT | This refers to the default, which varies with each protocol. | |
| SIO4_USC_TX_IDLE_COND_MARK | This refers to a continuous Mark. | |

| | |
|---|---|
| SIO4_USC_TX_IDLE_COND_MARK_SPACE | This refers to a continuous alternating Mark and Space sequence. |
| SIO4_USC_TX_IDLE_COND_SPACE | This refers to a continuous Space. |

### 6.19.10. SIO4_IOCTL_USC_TX_MODE

This service configures the protocol used by the USC transmitter for data transmission.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_TX_MODE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_MODE_8023 | This refers to the 802.3 protocol. | |
| SIO4_USC_MODE_ASY_CV | This refers to the Asynchronous with Code Violations protocol. | |
| SIO4_USC_MODE_ASYNC | This refers to the Asynchronous protocol. | |
| SIO4_USC_MODE_BSC | This refers to the Bisynchronous Serial Communications protocol. | |
| SIO4_USC_MODE_HDLC | This refers to the HDLC protocol. | |
| SIO4_USC_MODE_HDLC_L | This refers to the HDLC Loop protocol. | |
| SIO4_USC_MODE_ISOC | This refers to the Isochronous protocol. | |
| SIO4_USC_MODE_MONO | This refers to the Monosync protocol. | |
| SIO4_USC_MODE_NBIP | This refers to the Nine-Bit Interprocessor Protocol. | |
| SIO4_USC_MODE_S_MONO | This refers to the Slaved Monosync protocol. | |
| SIO4_USC_MODE_TBSC | This refers to the Transparent Bisynchronous Serial Communications protocol. | |

### 6.19.11. SIO4_IOCTL_USC_TX_PREAMBLE_FLAG

This service configures the Preamble Flag output selection for the transmitter.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_TX_PREAMBLE_FLAG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TX_PREAMBLE_FLAG_NO | This refers to sending no Preamble Flag. | |
| SIO4_USC_TX_PREAMBLE_FLAG_YES | This refers to sending a Preamble Flag. | |

### 6.19.12. SIO4_IOCTL_USC_TX_PREAMBLE_LEN

This service configures the Preamble length selection for the transmitter.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_PREAMBLE_LEN |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TX_PREAMBLE_LEN_8_BITS | This refers to sending a Preamble length of 8-bits. | |
| SIO4_USC_TX_PREAMBLE_LEN_16_BITS | This refers to sending a Preamble length of 16-bits. | |
| SIO4_USC_TX_PREAMBLE_LEN_32_BITS | This refers to sending a Preamble length of 32-bits. | |
| SIO4_USC_TX_PREAMBLE_LEN_64_BITS | This refers to sending a Preamble length of 64-bits. | |

### 6.19.13. SIO4_IOCTL_USC_TX_PREAMBLE_PAT

This service configures the Preamble pattern selection for the transmitter.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_PREAMBLE_PAT |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TX_PREAMBLE_PAT_0 | This refers to continuous low output. | |
| SIO4_USC_TX_PREAMBLE_PAT_01 | This refers to continuous low then high pattern. | |
| SIO4_USC_TX_PREAMBLE_PAT_1 | This refers to continuous low high. | |
| SIO4_USC_TX_PREAMBLE_PAT_10 | This refers to continuous high then low pattern. | |

### 6.19.14. SIO4_IOCTL_USC_TX_STATUS

This service retrieves the USC transmitter status, which is essentially the value read from the lower byte of the Transmit Command/Status Register.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_USC_TX_STATUS |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFF | This is the valid range for this service. | |

### 6.19.15. SIO4_IOCTL_USC_TX_WAIT_DMA_TRIG

This service configures the USC transmitter's pausing of data transfer from the channel's external Tx FIFO.

**NOTE**: By default each channel is configured to automatically transfer data from the channel's external Tx FIFO as data is received over the PCI bus. This service deals with the pausing of such data transfer until commanded to resume the transfer.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_TX_WAIT_DMA_TRIG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_WAIT_DMA_TRIG_NO | This refers to data transfer being unimpeded. | |
| SIO4_USC_WAIT_DMA_TRIG_YES | This refers to data transfer being paused. | |

### 6.19.16. SIO4_IOCTL_USC_TX_WAIT_UNDERRUN

This service configures the USC transmitter's Wait On Underrun feature.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_USC_TX_WAIT_UNDERRUN |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_USC_TX_WAIT_UNDERRUN_NO | This refers to not waiting. | |
| SIO4_USC_TX_WAIT_UNDERRUN_YES | This refers to waiting. | |

## 6.20. USC Receive Character Count FIFO Specific IOCTL Services

The USC maintains a small Receive Character Count (RCC) FIFO that records frame size information at the end of each received frame. This information is used to determine the size of the received frame, but the FIFO is only four elements deep. The driver implements a replacement software FIFO that is 256 elements deep. The below services are used to interact with the RCC software FIFO.

### 6.20.1. SIO4_IOCTL_RCC_SW_FIFO_ENABLE

This service enables or disables the RCC software FIFO.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_RCC_SW_FIFO_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_RCC_SW_FIFO_ENABLE_NO | This refers to the service being disabled. | |
| SIO4_RCC_SW_FIFO_ENABLE_YES | This refers to the service being enabled. | |

### 6.20.2. SIO4_IOCTL_RCC_SW_FIFO_FLUSH

This service flushes the content of the driver's RCC software FIFO as well as the USC's RCC hardware FIFO. The service also clears the USC RCC FIFO Overrun status.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_RCC_SW_FIFO_FLUSH |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_RCC_SW_FIFO_FLUSH_NO | This refers to the FIFO(s) not being flushed. | |
| SIO4_RCC_SW_FIFO_FLUSH_YES | This refers to the FIFO(s) being flushed. The feature is disabled before the flush is performed. | |

### 6.20.3. SIO4_IOCTL_RCC_SW_FIFO_READ

This service reads a single record from the driver's RCC software FIFO, if a record is available. If a record is not immediately available, then the driver will wait up to the specified amount of time for a record to become available.

> **NOTE:** The value passed to the driver is the amount of time the application is willing to wait for the arrival of a record when a record is not immediately available. The time is specified in milliseconds.

> **NOTE:** The value returned is zero if no records are available. Returned records are composed of two fields. The upper 16-bits include status flags. The lower 16-bits include the frame size, if a frame was received. The frame size is set to zero if a frame was not received.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_RCC_SW_FIFO_READ |
| arg | s32* |

The table below lists the options passed to the driver used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0-60000 | This is the valid | |

The table below lists the status flags returned in the upper 16-bits of each record. The error conditions generally indicate a need for recovery and resynchronization.

| Values | Description |
|---|---|
| SIO4_RCC_SW_FIFO_READ_ABORT_PE | Either an Abort condition was received at the cable |

| | |
|---|---|
| | interface or there was a data Parity Error. The specific meaning depends on the serial protocol in use. |
| SIO4_RCC_SW_FIFO_READ_BREAK_ABRT | Either a Break condition or an Abort condition was received at the cable interface. The specific meaning depends on the serial protocol in use. |
| SIO4_RCC_SW_FIFO_READ_DPLL_DESYN | The DPLL became desynchronized. Data may have been lost. |
| SIO4_RCC_SW_FIFO_READ_OVERRUN | There was either a data overrun or an RCC FIFO overrun (either hardware or software). |
| SIO4_RCC_SW_FIFO_READ_FRAME_END | The frame size is indicated in the lower 16-bits of the record. If this flag is not set, then the record is not reporting a completed frame. |

# 7. SYNC Model Specific IOCTL Services

These IOCTL services relate to SIO4-SYNC model boards only.

> **NOTE**: All services whose argument data type is `s32*` accept the value of `-1` being passed to the driver. If there is a setting associated with the service, then passing in the value `-1` is a request for the current setting. If the service is supported, then the current setting is returned. If the service is not supported, then the value returned by the driver will be `-1`. If there isn't a setting associated with the service, then passing in the value `-1` is a request to determine if the service is supported. If the service is supported, then the value zero is returned. If the service is not supported, then the value `-1` is returned.

## 7.1. SYNC Model Rx Clock Specific IOCTL Services

These IOCTL services configure the operation of the Rx Clock signal.

### 7.1.1. SIO4_IOCTL_SYNC_RXC_CFG

This service configures the polarity of the Rx Clock cable signal.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_RXC_CFG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_RXC_CFG_FALL_EDGE | Clock in data on the falling edge. | |
| SIO4_SYNC_RXC_CFG_RISE_EDGE | Clock in data on the rising edge. | |

### 7.1.2. SIO4_IOCTL_SYNC_RXC_POL

This service configures the polarity of the Rx Clock signal on which data bits are clocked into the receiver.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_RXC_POL |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_CLOCK_POL_FALL | Data is clocked in on the clock's falling edge. | |
| SIO4_SYNC_CLOCK_POL_RISE | Data is clocked in on the clock's rising edge. | |

## 7.2. SYNC Model Rx Data Specific IOCTL Services

These IOCTL services configure the operation of the Rx Data signal.

### 7.2.1. SIO4_IOCTL_SYNC_RXD_CFG

This service configures the polarity of the Rx Data cable signal.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_RXD_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_RXD_CFG_ACTIVE_HI | This refers to the Active High configuration. | |
| SIO4_SYNC_RXD_CFG_ACTIVE_LO | This refers to the Active Low configuration. | |

## 7.3. SYNC Model Rx Envelope Specific IOCTL Services

These IOCTL services configure the operation of the Rx Envelope signal.

### 7.3.1. SIO4_IOCTL_SYNC_RXE_CFG

This service configures the Rx Envelope cable signal.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_RXE_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_RXE_CFG_ACTIVE_HI | This refers to the Active High configuration. | |
| SIO4_SYNC_RXE_CFG_ACTIVE_LO | This refers to the Active Low configuration. | |
| SIO4_SYNC_RXE_CFG_DISABLE | This disables the signal. | |

### 7.3.2. SIO4_IOCTL_SYNC_RXE_POL

This service configures the polarity of the Rx Envelope cable signal.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_RXE_POL |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_ENV_POL_ACTIVE_HI | This refers to the Active High configuration. | |

| | |
|---|---|
| SIO4_SYNC_ENV_POL_ACTIVE_LO | This refers to the Active Low configuration. |

## 7.4. SYNC Model Receiver Specific IOCTL Services

These IOCTL services configure the operation of the receiver.

### 7.4.1. SIO4_IOCTL_SYNC_RX_BIT_COUNT

This service retrieves the current received Bit Count.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_RX_BIT_COUNT |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | NA | This is the valid range for this service. |

### 7.4.2. SIO4_IOCTL_SYNC_RX_BIT_ORDER

This service configures the order in which data bits are received over the cable interface.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_RX_BIT_ORDER |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_BIT_ORDER_LSB | This refers to the Least Significant Bit arriving first. | |
| SIO4_SYNC_BIT_ORDER_MSB | This refers to the Most Significant Bit arriving first. | |

### 7.4.3. SIO4_IOCTL_SYNC_RX_COUNT_ERROR

This service reports the occurrence of an Rx Bit Count error.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_RX_COUNT_ERROR |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_RX_COUNT_ERROR_NO | N/A | There was not an error. |
| SIO4_SYNC_RX_COUNT_ERROR_YES | N/A | There was an error. |

### 7.4.4. SIO4_IOCTL_SYNC_RX_COUNT_RESET

This service clears the Rx Bit Count to zero.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_RX_COUNT_RESET |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests support information. | The service is unsupported. |
| 0 | Reset the count. | The service is supported or the count was reset. |

### 7.4.5. SIO4_IOCTL_SYNC_RX_ENABLE

This service enables and disables the receiver.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_RX_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_ENABLE_NO | Disable the receiver. | |
| SIO4_SYNC_ENABLE_YES | Enable the receiver. | |

### 7.4.6. SIO4_IOCTL_SYNC_RX_GAP_ENABLE

This service enables and disables the Rx Gap feature.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_RX_GAP_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_RX_GAP_ENABLE_NO | Disable the feature. | |
| SIO4_SYNC_RX_GAP_ENABLE_YES | Enable the feature. | |

## 7.5. SYNC Model Tx Clock Specific IOCTL Services

These IOCTL services configure the operation of the Tx Clock signal.

### 7.5.1. SIO4_IOCTL_SYNC_TXC_CFG

This service configures the source selection for the Tx Clock signal.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_TXC_CFG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_TXC_CFG_EXT | This refers to an external clock source. | |
| SIO4_SYNC_TXC_CFG_INT | This refers to the on-board clock source. | |

### 7.5.2. SIO4_IOCTL_SYNC_TXC_IDLE

This service configures the operation of the Tx Clock signal when no data is being transmitted. (This service is unavailable when the SIO4_IOCTL_SYNC_TXC_IDLE_CFG service is available.)

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_TXC_IDLE |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_TXC_IDLE_NO | No, the clock is not to be idle. The clock will be active. | |
| SIO4_SYNC_TXC_IDLE_YES | Yes, the clock is to be idle. The clock will be idle. | |

### 7.5.3. SIO4_IOCTL_SYNC_TXC_IDLE_CFG

This service configures the operation of the Tx Clock signal when no data is being transmitted. (This service is unavailable when the SIO4_IOCTL_SYNC_TXC_IDLE service is available.)

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_TXC_IDLE_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_TXC_IDLE_CFG_ACTIVE | This refers to the clock being fully active. | |
| SIO4_SYNC_TXC_IDLE_CFG_IDLE_0 | This refers to the signal being idle, or when supported, driven low. | |
| SIO4_SYNC_TXC_IDLE_CFG_IDLE_1 | This refers to the signal being driven high. If the board does not support this option, then the *IDLE_0* option is used. | |

### 7.5.4. SIO4_IOCTL_SYNC_TXC_POL

This service configures the polarity of the Tx Clock signal on which data bits are clocked out of the transmitter.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_TXC_POL |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_CLOCK_POL_FALL | Data is clocked out on the clock's falling edge. | |
| SIO4_SYNC_CLOCK_POL_RISE | Data is clocked out on the clock's rising edge. | |

### 7.5.5. SIO4_IOCTL_SYNC_TXC_SRC

This service configures the source selection for the Tx Clock signal.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_TXC_SRC |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_TXC_SRC_0 | This refers to the signal being driven low. | |
| SIO4_SYNC_TXC_SRC_1 | This refers to the signal being driven high. | |
| SIO4_SYNC_TXC_SRC_EXT_FALL | This refers to the falling edge of the external Tx Clock signal. | |
| SIO4_SYNC_TXC_SRC_EXT_RISE | This refers to the rising edge of the external Tx Clock signal. | |
| SIO4_SYNC_TXC_SRC_OSC_HALF_FALL | This refers to the falling edge of the on-board clock (divided by two). | |
| SIO4_SYNC_TXC_SRC_OSC_HALF_RISE | This refers to the rising edge of the on-board clock (divided by two). | |

## 7.6. SYNC Model Tx Data Specific IOCTL Services

These IOCTL services configure the operation of the Tx Data signal.

### 7.6.1. SIO4_IOCTL_SYNC_TXD_CFG

This service configures the operation of the Tx Data cable signal.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_TXD_CFG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_TXD_CFG_OUT_0 | This refers to the signal being driven low. | |
| SIO4_SYNC_TXD_CFG_OUT_1 | This refers to the signal being driven high. | |
| SIO4_SYNC_TXD_CFG_ACTIVE_HI | This refers to the Active High configuration. | |
| SIO4_SYNC_TXD_CFG_ACTIVE_LO | This refers to the Active Low configuration. | |

### 7.6.2. SIO4_IOCTL_SYNC_TXD_IDLE_CFG

This service configures the operation of the Tx Data cable signal while no data is being transmitted.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_TXD_IDLE_CFG |
| arg | s32* |

The table below lists the options used with this service.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_TXD_IDLE_CFG_OUT_0 | This refers to the signal being driven low. | |
| SIO4_SYNC_TXD_IDLE_CFG_OUT_1 | This refers to the signal being driven high. | |

## 7.7. SYNC Model Tx Envelope Specific IOCTL Services

These IOCTL services configure the operation of the Tx Envelope signal.

### 7.7.1. SIO4_IOCTL_SYNC_TXE_CFG

This service configures the operation of the Tx Envelope cable signal.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_SYNC_TXE_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_TXE_CFG_ACTIVE_HI | This refers to the Active High configuration. | |
| SIO4_SYNC_TXE_CFG_ACTIVE_LO | This refers to the Active Low configuration. | |
| SIO4_SYNC_TXE_CFG_OUT_0 | This refers to the signal being driven low. | |
| SIO4_SYNC_TXE_CFG_OUT_1 | This refers to the signal being driven high. | |

#### 7.7.2. SIO4_IOCTL_SYNC_TXE_POL

This service configures the polarity of the Tx Envelope cable signal.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_TXE_POL |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_ENV_POL_ACTIVE_HI | Data is transmitted while the envelope signal is high. | |
| SIO4_SYNC_ENV_POL_ACTIVE_LO | Data is transmitted while the envelope signal is low. | |

### 7.8. SYNC Model Tx Auxiliary Clock Specific IOCTL Services

These IOCTL services configure the operation of the Tx Auxiliary Clock signal.

#### 7.8.1. SIO4_IOCTL_SYNC_TXAUXC_CFG

This service configures the Tx Auxiliary Clock signal at the cable interface.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_TXAUXC_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_TXAUXC_CFG_OSC_HALF | This refers to the on-board oscillator divided by two. | |
| SIO4_SYNC_TXAUXC_CFG_OUT_0 | This refers to the signal being driven low. | |
| SIO4_SYNC_TXAUXC_CFG_OUT_1 | This refers to the signal being driven high. | |
| SIO4_SYNC_TXAUXC_CFG_TRI | This refers to the signal being tri-stated. | |

### 7.9. SYNC Model Tx Spare Specific IOCTL Services

These IOCTL services configure the operation of the Tx Spare signal.

### 7.9.1. SIO4_IOCTL_SYNC_TXSP_CFG

This service configures the operation of the Tx Spare cable signal.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_TXSP_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_TXSP_CFG_DISABLE | This refers to the signal being disabled. | |
| SIO4_SYNC_TXSP_CFG_INPUT | This refers to the signal operating as an input.. | |
| SIO4_SYNC_TXSP_CFG_OUT_0 | This refers to the signal being driven low. | |
| SIO4_SYNC_TXSP_CFG_OUT_1 | This refers to the signal being driven high. | |

## 7.10. SYNC Model Transmitter Specific IOCTL Services

These IOCTL services configure the operation of the transmitter.

### 7.10.1. SIO4_IOCTL_SYNC_TX_BIT_ORDER

This service configures the order in which data bits are transmitted over the cable interface.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_TX_BIT_ORDER |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_BIT_ORDER_LSB | This refers to the Least Significant Bit being transmitted first. | |
| SIO4_SYNC_BIT_ORDER_MSB | This refers to the Most Significant Bit being transmitted first. | |

### 7.10.2. SIO4_IOCTL_SYNC_TX_ENABLE

This service enables and disables the transmitter.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_TX_ENABLE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_ENABLE_NO | Disable the transmitter. | |
| SIO4_SYNC_ENABLE_YES | Enable the transmitter. | |

### 7.10.3. SIO4_IOCTL_SYNC_TX_GAP_SIZE

This service configures the length of the gap between successive Enavelope active periods.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_TX_GAP_SIZE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

### 7.10.4. SIO4_IOCTL_SYNC_TX_WORD_SIZE

This service configures the Word Size for the data to be transmitted.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_TX_WORD_SIZE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0xFFFF | This is the valid range for this service. | |

## 7.11. SYNC Model Miscellaneous IOCTL Services

### 7.11.1. SIO4_IOCTL_SYNC_MODE

This service configures the SYNC Mode's operating mode.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_MODE |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_MODE_DUAL | This refers to the dual mode of operation. | |
| SIO4_SYNC_MODE_NORMAL | This refers to the normal mode of operation. | |

## 7.12. SYNC Model Legacy Cable Interface IOCTL Services

These IOCTL services relate to legacy cable operating services for SIO4-SYNC model boards only.

> **NOTE**: These services are available only when the legacy cable configuration is supported by firmware.

### 7.12.1. SIO4_IOCTL_SYNC_LEG_RXD_CFG

This service configures the routing of the cable Rx Data signal when using legacy cable configuration mode.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_LEG_RXD_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_LEG_RXD_CFG_LOW | Connect to the signal at the lower cable portion. | |
| SIO4_SYNC_LEG_RXD_CFG_TRI | Tri-state the cable signal. | |
| SIO4_SYNC_LEG_RXD_CFG_UP | Connect to the signal at the upper cable portion. | |

### 7.12.2. SIO4_IOCTL_SYNC_LEG_TXD_CFG

This service configures the routing of the cable Tx Data signal when using legacy cable configuration mode.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_SYNC_LEG_TXD_CFG |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_SYNC_LEG_TXD_CFG_LOW | Connect to the signal at the lower cable portion. | |
| SIO4_SYNC_LEG_TXD_CFG_TRI | Tri-state the cable signal. | |
| SIO4_SYNC_LEG_TXD_CFG_UP | Connect to the signal at the upper cable portion. | |
| SIO4_SYNC_LEG_TXD_CFG_UP_LOW | Connect to the signal at the upper and lower cable portions. | |

# 8. SIO4A Model Specific IOCTL Services

These IOCTL services are specific to the SIO4A versions of the board.

> **NOTE**: All services whose argument data type is `s32*` accept the value of `-1` being passed to the driver. If there is a setting associated with the service, then passing in the value `-1` is a request for the current setting. If the service is supported, then the current setting is returned. If the service is not supported, then the value returned by the driver will be `-1`. If there isn't a setting associated with the service, then passing in the value `-1` is a request to determine if the service is supported. If the service is supported, then the value zero is returned. If the service is not supported, then the value `-1` is returned.

### 8.1.1. SIO4_IOCTL_GPIO_DIRECTION_OUT

This service configures the direction of the General Purpose I/O pins.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_GPIO_DIRECTION_OUT |
| arg | s32* |

The table below lists the options used with this service. If a bit is set, then the pin is an output. If a bit is clear, then the pin is an input.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0x3F | This is the valid range for this service. | |

### 8.1.2. SIO4_IOCTL_GPIO_INPUT_LATCHING

This service configures the General Purpose I/O input pins as latching or non-latching.

Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_GPIO_INPUT_LATCHING |
| arg | s32* |

The table below lists the options used with this service. If a bit is set, then the pin is a latching when it is an input. If a bit is clear, then the pin is non-latching when it is an input.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0x3F | This is the valid range for this service. | |

### 8.1.3. SIO4_IOCTL_GPIO_INPUT_READ

This service returns the state of the General Purpose I/O pins configured as inputs.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_GPIO_INPUT_READ |
| arg | s32* |

The table below lists the options used with this service. Bits are set if the corresponding pins are configured as inputs and the input pins are at a high state. Bits are clear if the corresponding pins are configured as inputs and the input pins are at a low state. Bits are also clear if the corresponding pins are configured as outputs.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0x3F | N/A | This is the valid range for this service. |

### 8.1.4. SIO4_IOCTL_GPIO_OUTPUT_WRITE

This service sets the state of the General Purpose I/O pins configured as outputs.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_GPIO_OUTPUT_WRITE |
| arg | s32* |

The table below lists the options used with this service. If a bit is set and configured as an output, then the cable pin is driven high. If a bit is clear and configured as an output, then the cable pin is driven low. Cable pins configured as inputs are unaffected.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0x3F | This is the valid range for this service. | |

### 8.1.5. SIO4_IOCTL_GPIO_POLARITY

This service configures the latching polarity of the General Purpose I/O pins configured as inputs.

Usage

| ioctl() Argument | Description |
|---|---|
| request | SIO4_IOCTL_GPIO_POLARITY |
| arg | s32* |

The table below lists the options used with this service.

| Values | Passed to Driver | Returned By Driver |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| SIO4_GPIO_POLARITY_HIGH | Latch on a level-high or a high going edge. | |
| SIO4_GPIO_POLARITY_LOW | Latch on a level-low or a low going edge. | |

### 8.1.6. SIO4_IOCTL_GPIO_SENSE_EDGE

This service sets the latching sense of the General Purpose I/O pins configured as latching inputs.

### Usage

| **ioctl() Argument** | **Description** |
|---|---|
| request | SIO4_IOCTL_GPIO_SENSE_EDGE |
| arg | s32* |

The table below lists the options used with this service. If a bit is set and the corresponding pin is configured as a latching input, then the latch responds to changes in cable signal state. If a bit is clear and the corresponding pin is configured as a latching input, then the latch responds to the level of the cable signal state. Cable pins configured as non-latching inputs or as outputs are unaffected.

| **Values** | **Passed to Driver** | **Returned By Driver** |
|---|---|---|
| -1 | Requests current setting. | The service is unsupported. |
| 0 to 0x3F | This is the valid range for this service. | |

# 9. Operation

This section explains some operational procedures on using the driver. This is in no way intended to be a comprehensive guide on using the SIO4 and makes no attempt at explaining configuration of the Zilog Z16C30. This is simply to address a very few issues relating to GSC specific features of the SIO4.

## 9.1. I/O Modes

The following describes the three supported I/O modes used for data transfer between the host and the SIO4. All three modes are available using the routines `sio4_read()` and `sio4_write()`. (There may be OS specific limitations.) Applications select the desired mode using IOCTL services. Use the `SIO4_TX_IO_MODE_CONFIG` IOCTL service (section 4.2.1, page 28) to configure the `sio4_write()` data transfer mode and use the `SIO4_RX_IO_MODE_CONFIG` IOCTL service (section 4.2.7, page 30) to configure the `sio4_read()` data transfer mode.

### 9.1.1. DMDMA

This refers to Demand Mode DMA. This mode transfers data with the least amount of CPU overhead. It accommodates transfers that exceed the size of the installed FIFOs and uses the FIFO fill level to throttle data movement over the PCI bus. This permits efficient data movement over the PCI bus and also permits the transfer to remain active while data is being transferred over the cable interface. Since the SIO4 can have up to eight data streams (four Rx and four Rx) but has only two DMA engines, applications must make selective use of DMA and non-DMA I/O requests. Applications can make DMDMA mode I/O requests without having to monitor FIFO fill levels.

### 9.1.2. DMA

This refers to Non-Demand Mode DMA. This mode transfers data with little CPU overhead, but is suitable only for requests that do not exceed the size of the installed FIFOs. Using this mode, applications must monitor a FIFO's fill level to insure that it can accommodate desired requests. Calling `sio4_read()` when the Rx FIFO contains insufficient data will result in inefficient transfers as the driver may make several smaller requests while waiting for the FIFO to receive the additional required data. Calling `sio4_write()` when the Tx FIFO contains insufficient free space will result in inefficient transfers as the driver may make several smaller requests while waiting for the FIFO to contain the additional required space. Since the SIO4 can have up to eight data streams (four Rx and four Rx) and only two DMA engines, applications must make selective use of DMA and non-DMA I/O requests.

### 9.1.3. PIO

This mode uses repetitive register accesses. While it is the least efficient method it accommodates simultaneous transfers on any number of channels and in both directions. Applications can make PIO mode I/O requests without having to monitor FIFO fill levels.

## 9.2. Onboard DMA with the USC

For Zilog based SIO4 boards, the SIO4 is designed to automatically transfer data between the USC and the channel FIFOs. This is done using DMA, which is a feature built-in to the USC and supported by SIO4 circuitry and firmware. This feature is configured automatically when using the Initialize IOCTL service (`SIO4_IOCTL_INITIALIZE`, section 4.4.3, page 40) and the USC Reset service (`SIO4_IOCTL_USC_RESET`, section 6.17.8, page 94). Doing this manually requires that register fields be set as follows.

| Register | Setting |
|---|---|
| `USC.IOCR.TxRMode` | `1` |
| `USC.IOCR.RxRMode` | `1` |
| `USC.HCR.TxAMode` | `1` |

| USC.HCR.RxAMode | 1 |
|---|---|

## 9.3. Configuration Aid For The SIO4B

The figure below is intended as an aid to those trying to configure an SIO4B (or later) with a USC.

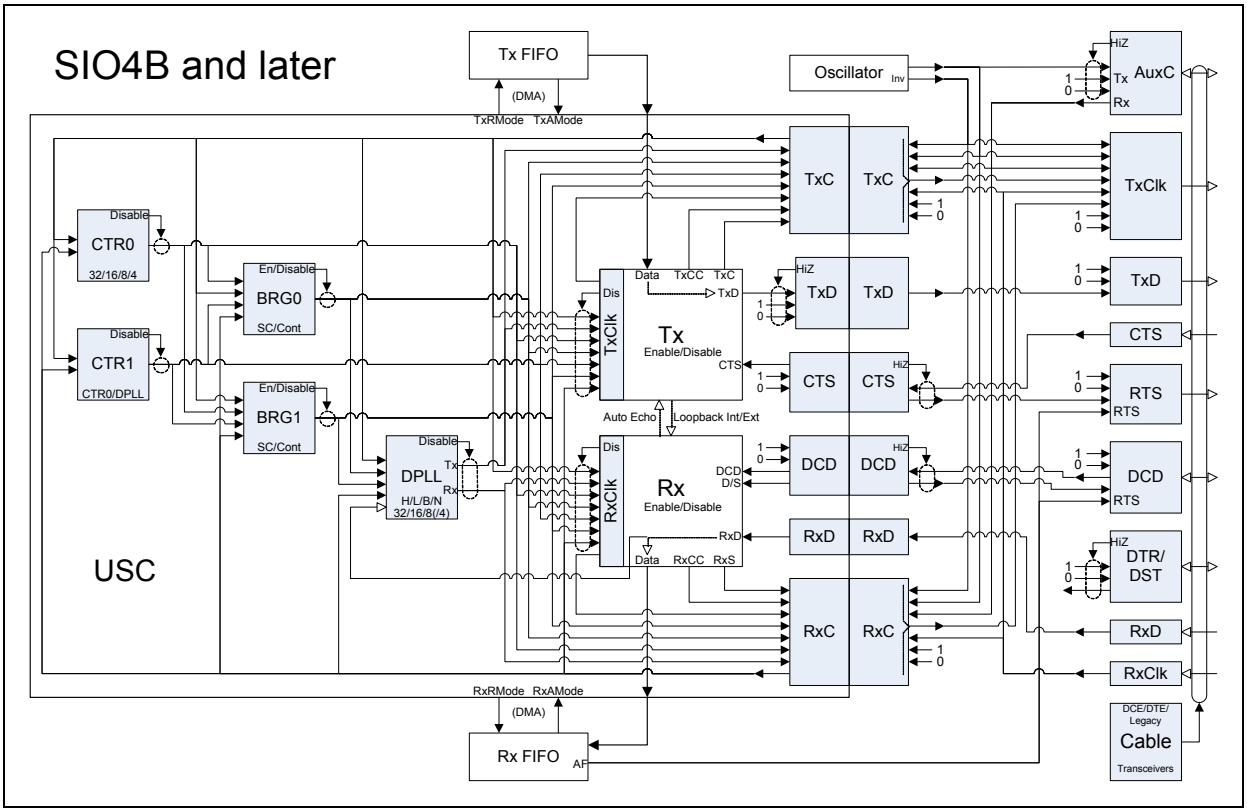


**Figure 1** A configuration aid for Zilog based SIO4B and later boards.

## 9.4. Configuration Aid For The SIO4B-SYNC

The figure below is intended as an aid to those trying to configure an SIO4B-SYNC (or later).

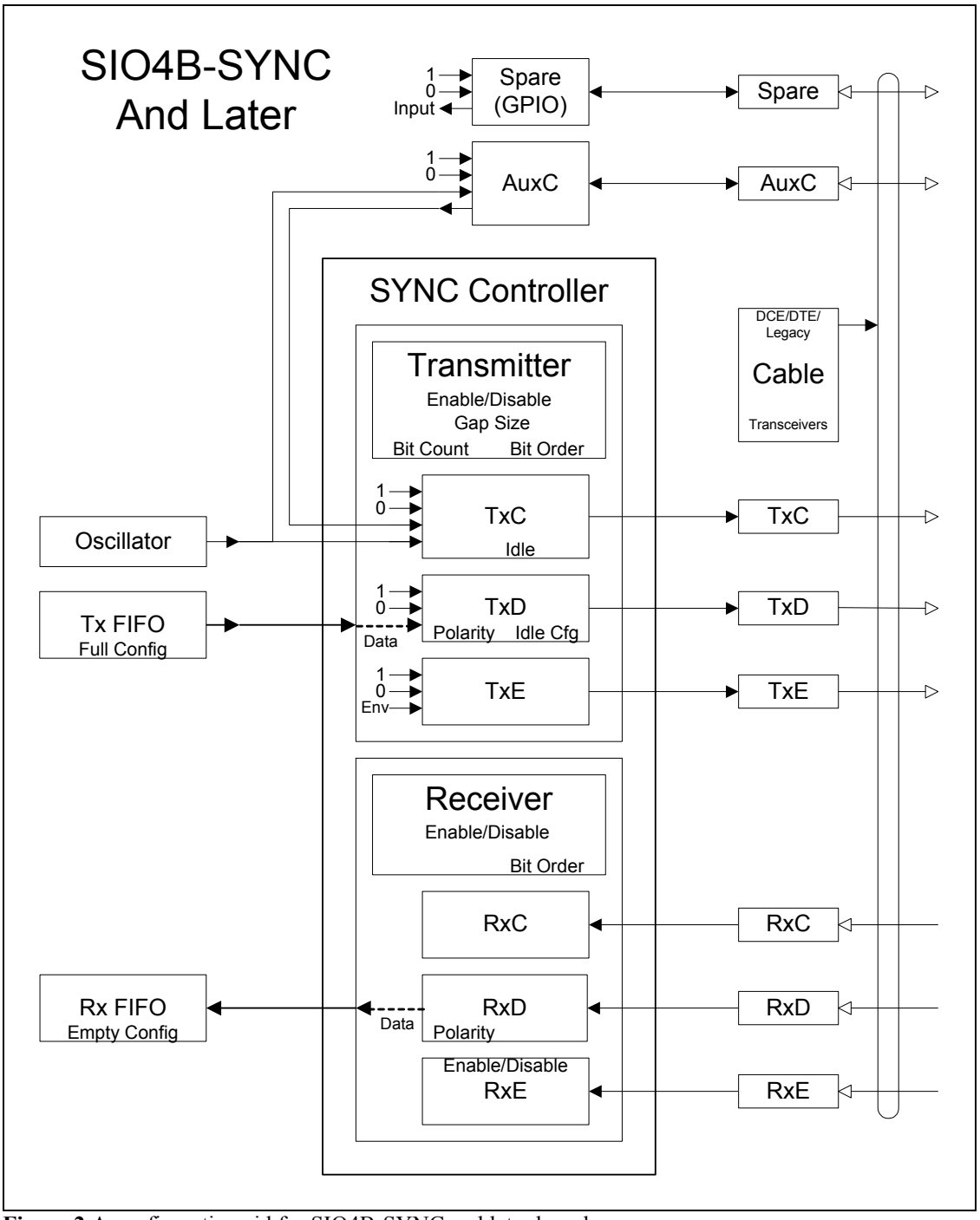


**Figure 2** A configuration aid for SIO4B-SYNC and later boards.

## 9.5. Configuration Aid For The SIO4A

The figure below is intended as an aid to those trying to configure an SIO4A with a USC.

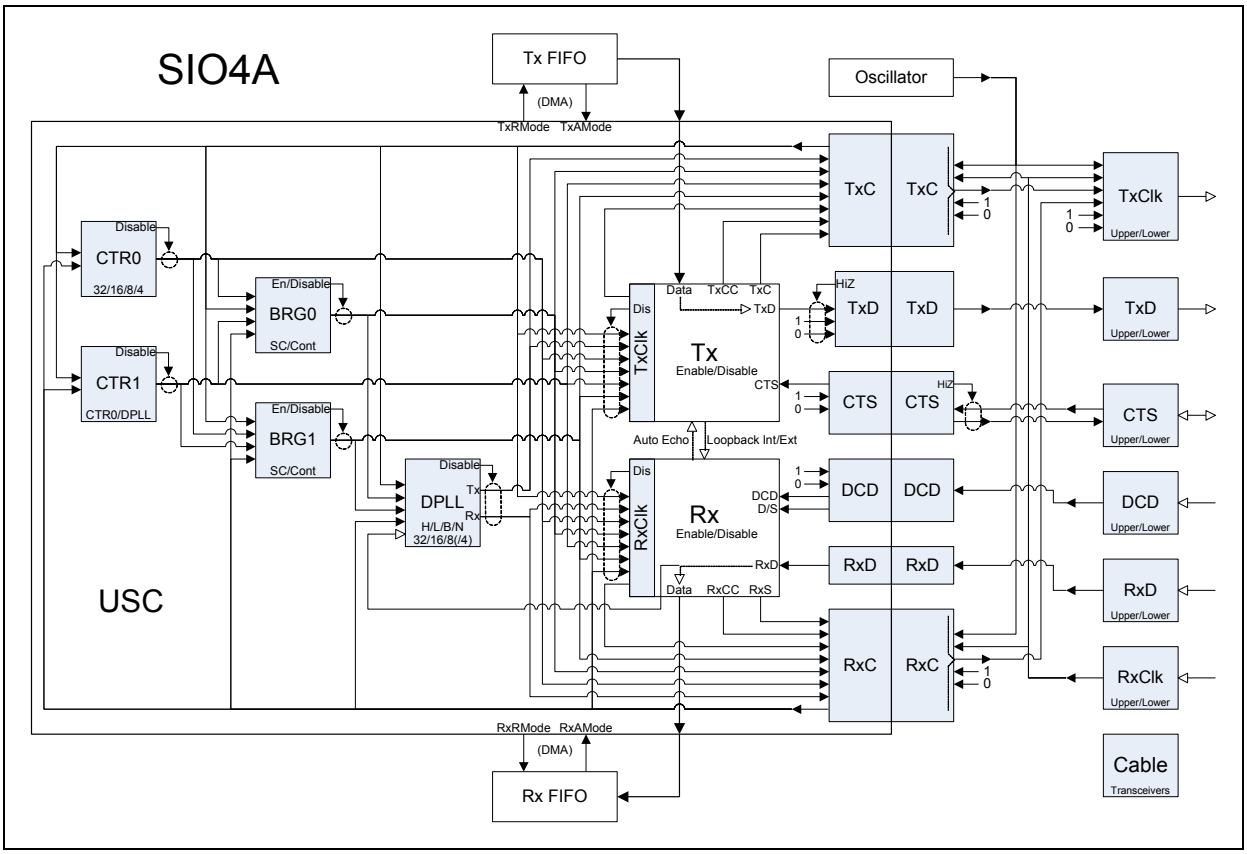


**Figure 3** A configuration aid for Zilog based SIO4A boards.

## 9.6. Configuration Aid For The Original SIO4

The figure below is intended as an aid to those trying to configure an original SIO4 with a USC.

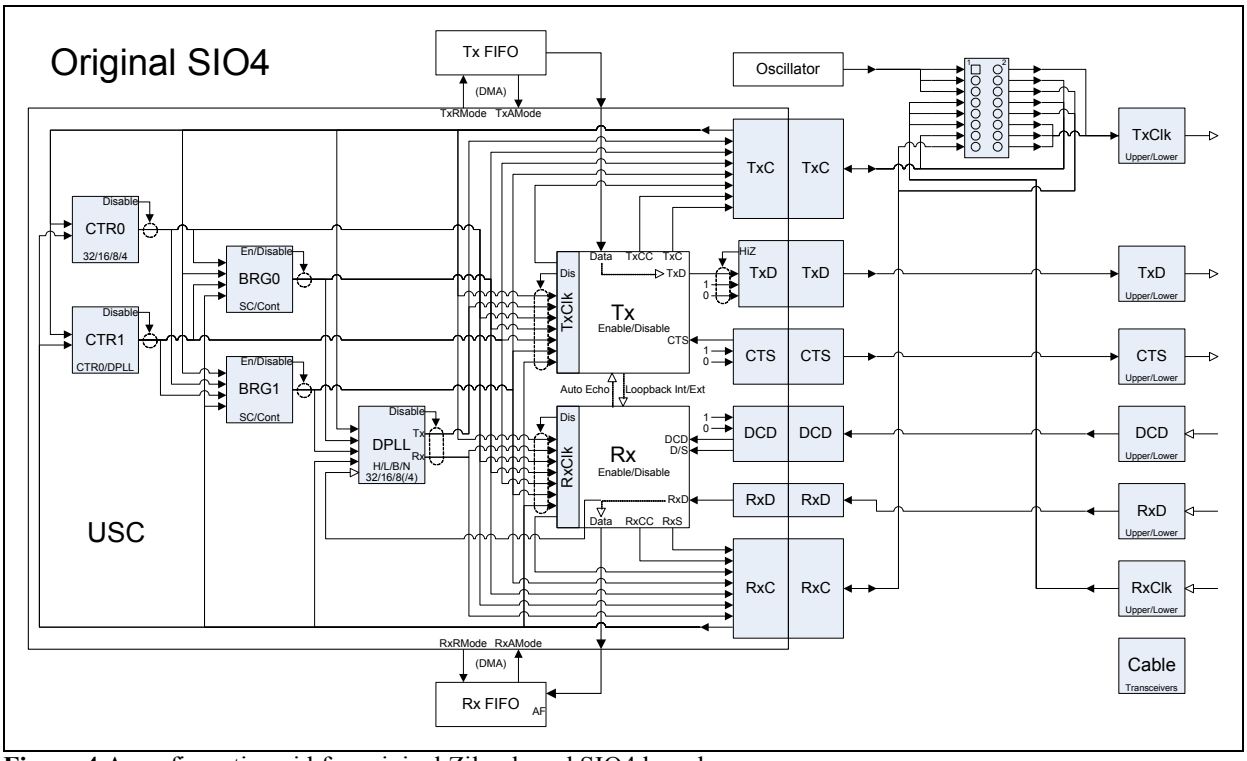


**Figure 4** A configuration aid for original Zilog based SIO4 boards.

# Document History

| Revision | Description |
|---|---|
| September 7, 2015 | Updated to release version 3.0.59.x.x. Added information about the FIFO Almost Empty and Almost Full states. |
| December 9, 2014 | Updated the release date. |
| December 4, 2014 | Updated to release version 2.8.47.x.x. All occurrences of "isoch" have been changed to "isoc", both lower and upper case, including file names. Updated the configuration aids and added figures for the SIO4A and the original SIO4. |
| July 31, 2014 | Updated to release version 2.7.44.5.0. Updated the SIO4-SYNC configuration aid figure. |
| May 17, 2014 | Updated to release version 2.7.44.x.0. |
| May 17, 2014 | Updated to release version 2.6.44.5.0. |
| April 16, 2014 | Updated to release version 2.5.43.4.0. |
| October 22, 2013 | Updated to release version 2.4.42.3.1. |
| October 22, 2013 | Updated to release version 2.4.42.3.0. |
| October 15, 2013 | Updated to release version 2.3.42.3.0. |
| August 27, 2013 | Updated to release version 2.2.42.x.x. Updated the `SIO4_IOCTL_OSC_REFERENCE` service for the CY22393 programmable oscillator. Updated the oscillator measurement service description. |
| October 11, 2012 | Updated to release version 2.1.41.1.0. Renamed `SIO4_SYNC_TX_BIT_COUNT` to `SIO4_SYNC_TX_WORD_SIZE`. Renamed `SIO4_USC_TX_STATUS_BLOCK` to `SIO4_USC_TX_CTRL_BLOCK`. Added the Rx and Tx I/O Abort services. |
| September 9, 2012 | Initial release of the 2.x series driver. |