24DSI

24-bit, 4 to 32 channel, 200KS/S/Ch Delta-Sigma A/D Input

PCI-24DSI32
CPCI-24DSI32

Linux Device Driver Porting Guide from The 24DSI32-PLL Linux Driver

Manual Revision: November 29, 2023

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

Table of Contents

1. Introduction

This porting guide provides information on porting applications from using the 24DSI32-PLL Linux device driver (release 1.0.6) to using the 24DSI Linux device driver (release 3.0.1.0).

1.1. Purpose

The purpose of the two drivers is essentially the same, except that the 24DSI driver supports the 24DSI32 and the 24DSI12, along with their variants.

1.2. Acronyms

The following is a list of commonly occurring acronyms which may appear throughout this document.

Acronyms	Description
API	Application Programming Interface
GSC	General Standards Corporation
PCI	Peripheral Component Interconnect
PCIe	PCI Express
PMC	PCI Mezzanine Card

1.3. Definitions

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

Term	Definition
24DSI	This is used as a general reference to any 24DSI based board, including the 24DSI32 and its variants.
24DSI32	This is used as a general reference to any 24DSI32 based board.
Application	Application means the user mode process, which runs in user space with user mode privileges.
Driver	Driver means the kernel mode device driver, which runs in kernel space with kernel mode privileges.

1.4. Software Overview

The software overview of the two drivers is essentially the same.

1.5. Hardware Overview

The hardware overview of the devices supported by the two drivers is essentially the same.

1.6. Reference Material

The reference material for the two drivers is essentially the same, except that the reference material for the 24DSI32 driver includes only 24DSI32 user manuals.

1.7. Licensing

For licensing information please refer to the text file `LICENSE.txt` in the root installation directory.

2. Installation

2.1. CPU and Kernel Support

Both drivers are designed to support the same processors. Explicit kernel support for the 24DSI driver is more extensive than that for the 24DSI32 driver. Both drivers operate under 32-bit and 64-bit environments.

2.1.1. 32-bit Support Under 64-bit Environments

The 24DSI driver provides support for 32-bit applications under 64-bit environments. The 24DSI32 driver does not provide this support.

2.2. The /proc File System

Both drivers provide driver information under the `/proc` file system. Refer to the below table for a summary.

Item	24DSI32	24DSI
File Name	`/proc/gsc24dsi32pll`	`/proc/24dsi`
`version`	`x.x`	`x.x.x.x`
`built`	This entry is identical for both drivers.	
`32-bit support:`	Not Supported	Supported
`boards`	This entry is identical for both drivers.	
`models:`	Not Supported	Supported

2.3. File List

The files provided as part of the release are similar for both drivers. Refer to the below table for a summary.

Item	24DSI32	24DSI
Driver Archive	`gsc_24DSI32.tar.gz`	`24dsi.linux.tar.gz`
User Manual	`GSC_24DSI32pll_linux_driver_user_manual.pdf`	`24dsi_linux_um.pdf`

2.4. Directory Structure

The directory structure is quite different for the two drivers. Refer to the below tables for a summary.

24DSI32 Directory	Content
`gsc24dsi32pll_release`	This is the root directory, which contains the sources for the driver and a single sample application.

24DSI Directory	Content
`24dsi`	This is the root directory.
`24dsi\2bsync`	This subdirectory contains a sample application.
`24dsi\billion`	This subdirectory contains a sample application.
`24dsi\docsrc`	This subdirectory contains utility sources from the user manual.
`24dsi\driver`	This subdirectory contains the driver sources.
`24dsi\fsamp`	This subdirectory contains a sample application.
`24dsi\rxrate`	This subdirectory contains a sample application.
`24dsi\savedata`	This subdirectory contains a sample application.
`24dsi\sbtest`	This subdirectory contains a sample application.
`24dsi\utils`	This subdirectory contains utility sources.

2.5. Installation

Installation is essentially the same for both drivers, except that each driver's archive file has a different name. Refer to the below table for a summary.

Item	24DSI32	24DSI
Driver Archive	`gsc_24DSI32.tar.gz`	`24dsi.linux.tar.gz`

2.6. Removal

Removal is basically the same for both drivers, except that the names are different for the archives, the install directories and the device nodes. Refer to the below table for a summary.

Item	24DSI32	24DSI
Driver Archive	`gsc_24DSI32PLLDriver.tar.gz`	`24dsi.linux.tar.gz`
Root Directory	`gsc24dsi32pll_release`	`24dsi`
Device Nodes	`/dev/gsc24dsi32pll*`	`/dev/24dsi.*`

2.7. Overall Make Script

The 24DSI driver includes a script that builds all driver archive targets and loads the driver. The 24DSI32 driver includes no such script. Refer to the below table for a summary.

Item	24DSI32	24DSI
Script	Not Provided	`make_all`

3. The Driver

Both drivers are loadable kernel mode drivers executing under control of the kernel. And, while each driver has an interface defined in one or more header files, the interface and the files differ. The interface files are summarized in the below table.

Item	24DSI32	24DSI
Primary Header	`gsc24dsi32pll_ioctl.h` †	`24dsi.h` †
PLX Register Definitions	`plx_regs.h` †	`gsc_pci9080.h` ††
Additional Headers	None	`gsc_common.h` ††

† Must be explicitly included.
†† This file is included implicitly by including the primary header.

3.1. Build

The build instructions are essentially the same for both drivers. Refer to the below table for a summary.

Item	24DSI32	24DSI
Starting Directory	`gsc24dsi32pll_release`	`24dsi/driver`
Clean Command	`make clean`	`make clean`
Make Command	`make`	`make all`

3.2. Startup

The startup procedures, both manual and automatic, are essentially the same for both drivers. The only differences are the location and name of the startup script. Refer to the below table for a summary.

Item	24DSI32	24DSI
Starting Directory	`gsc24dsi32pll_release`	`24dsi/driver`
Startup Script	`gsc_start`	`start`
Device Nodes	`/dev/gsc24dsi32pll*`	`/dev/24dsi.*`

3.3. Verification

The verification steps are not identical. The 24DSI32 driver invokes the sample application for verification, while the 24DSI driver uses the proc file entry for verification. Refer to the below table for a summary.

Item	24DSI32	24DSI
Command	`./testapp <board>`	`cat /proc/24dsi`

> **NOTE:** Verification for each driver can be accomplishes by verifying the presence of the corresponding `/proc` file. The file for the 24DSI32 driver is named `/proc/gsc24dsi32pll`.

3.4. Version

The driver version number can be obtained in essentially the same ways with both drivers. The difference is in the format of the version number and the name of the `/proc` file. Refer to the below table for a summary.

Item	24DSI32	24DSI
Version Format	`x.x`	`x.x.x.x`
`/proc` File	`/proc/gsc24dsi32pll`	`/proc/24dsi`

3.5. Shutdown

Shutdown is essentially the same for both drivers. The only difference is the name of the driver module. Refer to the below table for a summary.

Item	24DSI32	24DSI
Module Name	`gsc24dsi32pll`	`24dsi`

4. Document Source Code Examples

The 24DSI driver provides the user manual code examples in source form. The 24DSI32 driver does not. Refer to the below table for a summary.

Item	24DSI32	24DSI
Subdirectory	None Provided	`24dsi/docsrc`

5. Sample Applications

The 24DSI32 driver provides only a single sample application, while the 24DSI driver provides several. Refer to the below table for a summary.

Application	**24DSI32**	**24DSI**
testapp	`gsc24dsi32pll_release/testapp`	Not provided.
Billion	Not provided.	`24dsi/billion/billion`
Identify Board	Not provided.	`24dsi/id/id`
Receive Rate	Not provided.	`24dsi/rxrate/rxrate`
Register Access	Not provided.	`24dsi/regs/regs`
Sample Rate	Not provided.	`24dsi/fsamp/fsamp`
Save Acquired Data	Not provided.	`24dsi/savedata/savedata`
Single Board Test	Not provided.	`24dsi/sbtest/sbtest`
Software Sync	Not provided.	`24dsi/sw_sync/sw_sync`
Two Board Sync	Not provided.	`24dsi/2bsync/2bsync`

6. Driver Interface

Both drivers provide a unique interface based on the features of the boards that each driver supports. This section provides details relating to the similarities and differences.

6.1. Data Types

6.1.1. IOCTL Unary Types

With the 24DSI32 driver virtually all IOCTL services that exchanged a unary type value with the driver did so using an unsigned long pointer (`unsigned long*`). The 24DSI driver uses a signed 32-bit type (`s32*`).

6.1.2. device_register_params

The 24DSI32 driver uses this structure to read and write individual registers. The corresponding 24DSI driver structure is `gsc_reg_t`. The 24DSI driver uses this structure to read, write, and perform read-modify-write operation on registers. Refer to the below table for a comparison.

Item	24DSI32	24DSI
Structure	`struct device_register_params`, `DEVICE_REGISTER_PARAMS` and `PDEVICE_REGISTER_PARAMS`	`gsc_reg_t`
Register	`__u32 ulRegister`	`u32 reg`
Value	`__u32 ulValue`	`u32 value`
Mask	Not Supported	`u32 mask`

6.1.3. gen_assign_params

The 24DSI32 driver uses this structure to assign a clock source to a specified channel group. The 24DSI driver instead makes channel group clock source assignments by using different IOCTL services. Refer to the below table for a comparison.

Item	24DSI32	24DSI
Channel Group 0	`gen_assign_params.eGroup = GRP_0`	`DSI_IOCTL_CH_GRP_0_SRC`
Channel Group 1	`gen_assign_params.eGroup = GRP_1`	`DSI_IOCTL_CH_GRP_1_SRC`
Channel Group 2	`gen_assign_params.eGroup = GRP_2`	`DSI_IOCTL_CH_GRP_2_SRC`
Channel Group 3	`gen_assign_params.eGroup = GRP_3`	`DSI_IOCTL_CH_GRP_3_SRC`

6.2. Macros

6.2.1. IOCTL

Both drivers provide device specific support through a set of driver specific IOCTL services.

6.2.2. Registers

The set of supported registers and their macro names are given in the below subsections. However, there are other differences as well. The 24DSI32 driver defines registers as indexes in the region where the registers are located as if each region were an array of 32-bit values. This produces one series of zero-based indexes for the PLX Feature Set Registers and another series of zero-based indexes for the firmware registers. The 24DSI32 driver does not provide access to the PCI registers. The 24DSI driver derives each register macro based on an encoding that includes the register's region, size and offset. One result of these differences is that all 24DSI32 driver registers are accessed as 32-bit values, while all 24DSI driver registers are accessed by their native size. Also, because the 24DSI32 driver

access registers as 32-bit entities, there may be multiple 24DSI register definitions that correspond to a single 24DSI32 definition.

6.2.2.1. GSC Registers

Both drivers provide access to all firmware-based registers. Refer to the below table for a summary.

Register	24DSI32	24DSI
Auto Calibration Values Register	AUTOCAL_VALUES_REG	DSI_GSC_AVR
Board Configuration Register	BOARD_CONFIG_REG	DSI_GSC_BCFGR
Board Control Register	BOARD_CTRL_REG	DSI_GSC_BCTLR
Input Buffer Control Register	BUFFER_CONTROL_REG	DSI_GSC_IBCR
Buffer Size Register	BUFFER_SIZE_REG	DSI_GSC_BSR
Input Data Buffer Register	INPUT_DATA_BUFFER_REG	DSI_GSC_IDBR
NREF Control Register (PLL specific)	NREF_PLL_CONTROL_REG	DSI_GSC_NREFCR
NVCO Control Register (PLL specific)	NVCO_PLL_CONTROL_REG	DSI_GSC_NVCOCR
PLL Reference Frequency Register	PLL_REF_FREQ_REG	DSI_GSC_PRFR
Range and Filter Control Register	RANGE_FILTER_CONTROL	DSI_GSC_RFCR
Rate Assignments Register	RATE_ASSIGN_REG	DSI_GSC_RAR
Rate Divisors Register	RATE_DIVISOR_REG	DSI_GSC_RDR
Reserved	RESERVED_1	Not Supported

6.2.2.2. PCI Configuration Registers

The 24DSI32 driver does not provide macro definitions for the PCI registers. The 24DSI driver does provide macro definitions for these registers. Refer to `gsc_pci9080.h` for the definitions of these registers.

6.2.2.3. PLX PCI9080 Feature Set Registers

Both drivers provide macro for these registers. The table below identifies where the definitions are declared. Refer to the following tables for the macro names for the defined registers.

Item	24DSI32	24DSI
Header File	`plx_regs.h` †	`gsc_pci9080.h` ††

† This header file must be included explicitly.
†† This header file is included implicitly when including the primary header `24dsi.h`.

Local Configuration Registers: The offsets are given relative to the base of the BAR0 memory space region.

Offset	24DSI32	24DSI
0x00	PCI_TO_LOC_ADDR_0_RNG	GSC_PLX_9080_LAS0RR
0x04	LOC_BASE_ADDR_REMAP_0	GSC_PLX_9080_LAS0BA
0x08	MODE_ARBITRATION	GSC_PLX_9080_MARBR
0x0C	BIG_LITTLE_ENDIAN_DESC	GSC_PLX_9080_BIGEND
0x10	PCI_TO_LOC_ROM_RNG	GSC_PLX_9080_EROMRR
0x14	LOC_BASE_ADDR_REMAP_EXP_ROM	GSC_PLX_9080_EROMBA
0x18	BUS_REG_DESC_0_FOR_PCI_LOC	GSC_PLX_9080_LBRD0
0x1C	DIR_MASTER_TO_PCI_RNG	GSC_PLX_9080_DMRR
0x20	LOC_ADDR_FOR_DIR_MASTER_MEM	GSC_PLX_9080_DMLBAM
0x24	LOC_ADDR_FOR_DIR_MASTER_IO	GSC_PLX_9080_DMLBAI
0x28	PCI_ADDR_REMAP_DIR_MASTER	GSC_PLX_9080_DMPBAM
0x2C	PCI_CFG_ADDR_DIR_MASTER_IO	GSC_PLX_9080_DMCFGA
0xF0	PCI_TO_LOC_ADDR_1_RNG	GSC_PLX_9080_LAS1RR
0xF4	LOC_BASE_ADDR_REMAP_1	GSC_PLX_9080_LAS1BA

0xF8	BUS_REG_DESC_1_FOR_PCI_LOC	GSC_PLX_9080_LBRD1

Runtime Registers: The offsets are given relative to the base of the BAR0 memory space region.

Offset	24DSI32	24DSI
0x40	MAILBOX_REGISTER_0	GSC_PLX_9080_MBOX0
0x44	MAILBOX_REGISTER_1	GSC_PLX_9080_MBOX1
0x48	MAILBOX_REGISTER_2	GSC_PLX_9080_MBOX2
0x4C	MAILBOX_REGISTER_3	GSC_PLX_9080_MBOX3
0x50	MAILBOX_REGISTER_4	GSC_PLX_9080_MBOX4
0x54	MAILBOX_REGISTER_5	GSC_PLX_9080_MBOX5
0x58	MAILBOX_REGISTER_6	GSC_PLX_9080_MBOX6
0x5C	MAILBOX_REGISTER_7	GSC_PLX_9080_MBOX7
0x60	PCI_TO_LOC_DOORBELL	GSC_PLX_9080_P2LDBELL
0x64	LOC_TO_PCI_DOORBELL	GSC_PLX_9080_L2PDBELL
0x68	INT_CTRL_STATUS	GSC_PLX_9080_INTCSR
0x6C	PROM_CTRL_CMD_CODES_CTRL	GSC_PLX_9080_CNTRL
0x70	DEVICE_ID_VENDOR_ID	GSC_PLX_9080_PCIVIDR GSC_PLX_9080_PCIDIDR
0x74	REVISION_ID	GSC_PLX_9080_PCIHREV
0x78	MAILBOX_REG_0	Not Defined
0x7C	MAILBOX_REG_1	Not Defined

DMA Registers: The offsets are given relative to the base of the BAR0 memory space region.

Offset	24DSI32	24DSI
0x80	DMA_CH_0_MODE	GSC_PLX_9080_DMAMODE0
0x84	DMA_CH_0_PCI_ADDR	GSC_PLX_9080_DMAPADR0
0x88	DMA_CH_0_LOCAL_ADDR	GSC_PLX_9080_DMALADR0
0x8C	DMA_CH_0_TRANS_BYTE_CNT	GSC_PLX_9080_DMASIZ0
0x90	DMA_CH_0_DESC_PTR	GSC_PLX_9080_DMADPR0
0x94	DMA_CH_1_MODE	GSC_PLX_9080_DMAMODE1
0x98	DMA_CH_1_PCI_ADDR	GSC_PLX_9080_DMAPADR1
0x9C	DMA_CH_1_LOCAL_ADDR	GSC_PLX_9080_DMALADR1
0xA0	DMA_CH_1_TRANS_BYTE_CNT	GSC_PLX_9080_DMASIZ1
0xA4	DMA_CH_1_DESC_PTR	GSC_PLX_9080_DMADPR1
0xA8	DMA_CMD_STATUS_0 †	GSC_PLX_9080_DMACSR0
0xA9	DMA_CMD_STATUS_1 †	GSC_PLX_9080_DMACSR1
0xAC	DMA_MODE_ARBITRATION	GSC_PLX_9080_DMAARB
0xB0	DMA_THRESHOLD_REG	GSC_PLX_9080_DMATHR

† The 24DSI32 definition for this register is invalid for 24DSI32 application use.

Messaging Queue Registers: The offset is given relative to the base of the BAR0 memory space region.

Offset	24DSI32	24DSI
0x30	OUT_POST_Q_INT_STATUS	GSC_PLX_9080_OPLFIS
0x34	OUT_POST_Q_INT_MASK	GSC_PLX_9080_OPLFIM
0x40	IN_Q_PORT	GSC_PLX_9080_IQP
0x48	OUT_Q_PORT	GSC_PLX_9080_OQP
0xC0	MSG_UNIT_CONFIG	GSC_PLX_9080_MQCR
0xC4	Q_BASE_ADDR	GSC_PLX_9080_QBAR
0xC8	IN_FREE_HEAD_PTR	GSC_PLX_9080_IFHPR
0xCC	IN_FREE_TAIL_PTR	GSC_PLX_9080_IFTPR
0xD0	IN_POST_HEAD_PTR	GSC_PLX_9080_IPHPR

0xD4	IN_POST_TAIL_PTR	GSC_PLX_9080_IPTPR
0xD8	OUT_FREE_HEAD_PTR	GSC_PLX_9080_OFHPR
0xDC	OUT_FREE_TAIL_PTR	GSC_PLX_9080_OFTPR
0xE0	OUT_POST_HEAD_PTR	GSC_PLX_9080_OPHPR
0xE4	OUT_POST_TAIL_PTR	GSC_PLX_9080_OPTPR
0xE8	Q_STATUS_CTRL_REG	GSC_PLX_9080_QSR

6.3. Functions

The driver interface includes the following functions.

6.3.1. close()

Both drivers use the `close()` function in an identical manner.

6.3.2. ioctl()

Both drivers use the `ioctl()` function in an identical manner.

6.3.3. open()

Both drivers use the `open()` function in a similar manner. The only difference is in the device node prefix. Refer to the below table for a summary. (The trailing *n* is the zero-based index of the board to access.)

Item	24DSI32	24DSI
Device Node	/dev/gsc24dsi32pll*n*	/dev/24dsi.*n*

6.3.4. read()

Both drivers use the `read()` function in an identical manner.

> **NOTE:** There are internal driver differences with regard to the Buffer Threshold and the I/O mode. With both drivers one may experience improved responsiveness with read requests by coordinating the Buffer Threshold with the number of samples requested.

6.3.5. write()

Neither driver supports this operation.

6.4. IOCTL Services

The following sections provide information on porting applications using 24DSI32 driver IOCTL services to using the corresponding functionality in the 24DSI driver.

6.4.1. IOCTL_ASSIGN_RATE_GROUP

The 24DSI32 driver uses this service to assign a clock source to a channel group. Both the clock source and the channel group are specified via fields in the structure passed to the driver with this service. The 24DSI driver provides similar functionality by using a different IOCTL service for each channel group. Refer to the below tables for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_ASSIGN_RATE_GROUP	Supported by a separate IOCTL service for each channel group, as listed below.

Argument Type	struct gen_assign_params*	s32*
Channel Group 0	struct->eGroup = GRP_0	DSI_IOCTL_CH_GRP_0_SRC
Channel Group 1	struct->eGroup = GRP_1	DSI_IOCTL_CH_GRP_1_SRC
Channel Group 2	struct->eGroup = GRP_2	DSI_IOCTL_CH_GRP_2_SRC
Channel Group 3	struct->eGroup = GRP_3	DSI_IOCTL_CH_GRP_3_SRC
Direct External Source	ASN_DIRECT_EXTERNAL	DSI_CH_GRP_SRC_DIR_EXTERN
Disable Channel Group	ASN_DISABLED †	DSI_CH_GRP_SRC_DISABLE †
Enable Channel Group	Not Supported	DSI_CH_GRP_SRC_ENABLE †
External Source	ASN_EXTERNAL	DSI_CH_GRP_SRC_EXTERN
Internal Rate Generator	ASN_RATE_INTERNAL	DSI_CH_GRP_SRC_GEN_A ††

† Some versions of the 24DSI32 permit source selection with only the first channel group, while the remaining channel groups are limited to the enable and disable options.

†† The 24DSI32 has only a single Rate Generator. The "A" in the 24DSI Rate Generator service code refers to this Rate Generator. The alphabetic index is present in the service code because the 24DSI driver supports the 24DSI12, which has two Rate Generators.

6.4.2. IOCTL_AUTO_CAL

The 24DSI32 driver uses this service to initiate an Autocalibration cycle. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_AUTO_CAL	DSI_IOCTL_AUTOCAL
Argument Type	Not Used	Not Used

6.4.3. IOCTL_CLEAR_BUFFER

The 24DSI32 driver uses this service to clear the input buffer. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_CLEAR_BUFFER	DSI_IOCTL_AIN_BUF_CLEAR
Argument Type	Not Used	Not Used

6.4.4. IOCTL_CLEAR_BUFFER_SYNC

The 24DSI32 driver uses this service to control the functionality of a Software Sync operation. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_CLEAR_BUFFER_SYNC	DSI_IOCTL_SW_SYNC_MODE
Argument Type	unsigned long*	s32*
Clear Buffer	TRUE	DSI_SW_SYNC_MODE_CLR_BUF
Synchronize	FALSE	DSI_SW_SYNC_MODE_SW_SYNC

6.4.5. IOCTL_FILL_BUFFER

The 24DSI32 driver uses this service to permit an application to tell the driver that it is to wait indefinitely, if needed, when a `read()` request needs additional time to complete the request. The 24DSI driver has no similar service. The 24DSI driver uses the I/O Timeout service (`DSI_IOCTL_RX_IO_TIMEOUT`) to control the maximum amount of time the driver should wait for `read()` requests to complete.

6.4.6. IOCTL_GET_REF_FREQUENCY

The 24DSI32 driver uses this service to retrieve the board's approximation of the reference oscillator frequency used by the PLL feature. The 24DSI driver uses the corresponding service listed below for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_GET_REF_FREQUENCY	DSI_IOCTL_PLL_REF_FREQ_HZ
Argument Type	unsigned long*	s32*

6.4.7. IOCTL_GSC_SET_RANGE_FILTER

The 24DSI32 driver uses this service to write to the Range and Filter Control Register (RANGE_FILTER_CONTROL). The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_GSC_SET_RANGE_FILTER	DSI_IOCTL_REG_WRITE
Argument Type	unsigned long*	gsc_reg_t*
Register Identifier	Implied by the service.	struct->reg = DSI_GSC_RFCR
Register Value	Passed via the pointer.	struct->value

While applications can write directly to the Range and Control Register with both drivers, the 24DSI driver provides separate services for manipulation of the fields in this register. Refer to the below table for a summary.

Description	24DSI
Enable or disable the selective filter settings.	DSI_IOCTL_AIN_FILTER_SEL
Set the filtering option for channels 0 through 3.	DSI_IOCTL_AIN_FILTER_00_03
Set the filtering option for channels 4 through 7.	DSI_IOCTL_AIN_FILTER_04_07
Set the filtering option for channels 8 through 11.	DSI_IOCTL_AIN_FILTER_08_11
Set the filtering option for channels 12 through 15.	DSI_IOCTL_AIN_FILTER_12_15
Set the filtering option for channels 16 through 19.	DSI_IOCTL_AIN_FILTER_16_19
Set the filtering option for channels 20 through 23.	DSI_IOCTL_AIN_FILTER_20_23
Set the filtering option for channels 24 through 27.	DSI_IOCTL_AIN_FILTER_24_27
Set the filtering option for channels 28 through 31.	DSI_IOCTL_AIN_FILTER_28_31
Enable or disable the selective voltage range settings.	DSI_IOCTL_AIN_RANGE_SEL
Set the voltage range option for channels 0 through 3.	DSI_IOCTL_AIN_RANGE_00_03
Set the voltage range option for channels 4 through 7.	DSI_IOCTL_AIN_RANGE_04_07
Set the voltage range option for channels 8 through 11.	DSI_IOCTL_AIN_RANGE_08_11
Set the voltage range option for channels 12 through 15.	DSI_IOCTL_AIN_RANGE_12_15
Set the voltage range option for channels 16 through 19.	DSI_IOCTL_AIN_RANGE_16_19
Set the voltage range option for channels 20 through 23.	DSI_IOCTL_AIN_RANGE_20_23
Set the voltage range option for channels 24 through 27.	DSI_IOCTL_AIN_RANGE_24_27
Set the voltage range option for channels 28 through 31.	DSI_IOCTL_AIN_RANGE_28_31

6.4.8. IOCTL_INITIALIZE

The 24DSI32 driver uses this service to initialize the board. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_INITIALIZE	DSI_IOCTL_INITIALIZE
Argument Type	Not Used	Not Used

6.4.9. IOCTL_NO_COMMAND

The 24DSI driver has no corresponding IOCTL service. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_NO_COMMAND	None
Argument Type	Not Used	Not Applicable

6.4.10. IOCTL_READ_REGISTER

The 24DSI32 driver uses this service to read firmware registers. The 24DSI driver uses a corresponding service to read PCI Register, PLX Feature Set Registers, and GSC Firmware Registers. Refer to the below table for a summary. For register macro definitions refer to section 6.2.2 on page 12.

Item	24DSI32	24DSI
Service Macro	IOCTL_READ_REGISTER	DSI_IOCTL_REG_READ
Argument Type	struct device_register_params*	gsc_reg_t*
Register	ulRegister	reg
Value	ulValue	value

6.4.11. IOCTL_SELECT_IMAGE_FILTER

The 24DSI32 driver uses this service to configure the board filtering option. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SELECT_IMAGE_FILTER	DSI_IOCTL_AIN_FILTER
Argument Type	unsigned long*	s32*
High Frequency Filter	IMAGE_FILTER_HI_FREQ	DSI_AIN_FILTER_HI
Low Frequency Filter	IMAGE_FILTER_LO_FREQ	DSI_AIN_FILTER_LOW

6.4.12. IOCTL_SET_ACQUIRE_MODE

The 24DSI32 driver uses this service to control the flow of sampled data into the input buffer. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_ACQUIRE_MODE	DSI_IOCTL_AIN_BUF_INPUT
Argument Type	unsigned long*	s32*
Disable Input	STOP_ACQUIRE	DSI_AIN_BUF_INPUT_DISABLE
Enable Input	START_ACQUIRE	DSI_AIN_BUF_INPUT_ENABLE

6.4.13. IOCTL_SET_BUFFER_THRESHOLD

The 24DSI32 driver uses this service to set the input buffer threshold level. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_BUFFER_THRESHOLD	DSI_IOCTL_AIN_BUF_THRESH
Argument Type	unsigned long*	s32*

6.4.14. IOCTL_SET_DATA_FORMAT

The 24DSI32 driver uses this service to set the data encoding format. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_DATA_FORMAT	DSI_IOCTL_DATA_FORMAT
Argument Type	unsigned long*	s32*
Offset Binary	FORMAT_OFFSET_BINARY	DSI_DATA_FORMAT_OFF_BIN
Twos Compliment	FORMAT_TWOS_COMPLEMENT	DSI_DATA_FORMAT_2S_COMP

6.4.15. IOCTL_SET_DATA_WIDTH

The 24DSI32 driver uses this service to configure the resolution of the Analog-to-Digital converters. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_DATA_WIDTH	DSI_IOCTL_DATA_WIDTH
Argument Type	unsigned long*	s32*
16-Bit	DATA_WIDTH_16	DSI_DATA_WIDTH_16
18-Bit	DATA_WIDTH_18	DSI_DATA_WIDTH_18
20-Bit	DATA_WIDTH_20	DSI_DATA_WIDTH_20
24-Bit	DATA_WIDTH_24	DSI_DATA_WIDTH_24

6.4.16. IOCTL_SET_DMA_STATE

The 24DSI32 driver uses this service to set the data transfer mode user for `read()` requests. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_DMA_STATE	DSI_IOCTL_RX_IO_MODE
Argument Type	unsigned long*	s32*
Demand Mode DMA	DMA_DEMAND_MODE	GSC_IO_MODE_DMDMA
DMA	DMA_ENABLE	GSC_IO_MODE_BMDMA
PIO	DMA_DISABLE	GSC_IO_MODE_PIO

6.4.17. IOCTL_SET_INITIATOR_MODE

The 24DSI32 driver uses this service to configure the device as an initiator or as a target. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_INITIATOR_MODE	DSI_IOCTL_INIT_MODE
Argument Type	unsigned long*	s32*
Initiator Mode	INITIATOR_MODE	DSI_INIT_MODE_INITIATOR
Target Mode	TARGET_MODE	DSI_INIT_MODE_TARGET

6.4.18. IOCTL_SET_INPUT_MODE

The 24DSI32 driver uses this service to set the analog input voltage range. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_INPUT_MODE	DSI_IOCTL_AIN_MODE

Argument Type	unsigned long*	s32*
Differential Inputs	MODE_DIFFERENTIAL	DSI_AIN_MODE_DIFF
VREF Input Test	MODE_VREF_TEST	DSI_AIN_MODE_VREF
Zero Input Test	MODE_ZERO_TEST	DSI_AIN_MODE_ZERO

6.4.19. IOCTL_SET_INPUT_RANGE

The 24DSI32 driver uses this service to set the analog input voltage range. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_INPUT_RANGE	DSI_IOCTL_AIN_RANGE
Argument Type	unsigned long*	s32*
±2.5 Volts	RANGE_2p5V	DSI_AIN_RANGE_2_5V
±5 Volts	RANGE_5V	DSI_AIN_RANGE_5V
±10 Volts	RANGE_10V	DSI_AIN_RANGE_10V

6.4.20. IOCTL_SET_NREF

The 24DSI32 driver uses this service to set the NREF value for the Rate Generator. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_NREF	DSI_IOCTL_RATE_GEN_A_NREF †
Argument Type	unsigned long*	s32*

† The 24DSI32 has only a single Rate Generator. The "A" in the 24DSI Rate Generator service code refers to this Rate Generator. The alphabetic index is present in the service code because the 24DSI driver supports the 24DSI12, which has two Rate Generators.

6.4.21. IOCTL_SET_NVCO

The 24DSI32 driver uses this service to set the NVCO value for the Rate Generator. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_NVCO	DSI_IOCTL_RATE_GEN_A_NVCO †
Argument Type	unsigned long*	s32*

† The 24DSI32 has only a single Rate Generator. The "A" in the 24DSI Rate Generator service code refers to this Rate Generator. The alphabetic index is present in the service code because the 24DSI driver supports the 24DSI12, which has two Rate Generators.

6.4.22. IOCTL_SET_OVERFLOW_CHECK

The 24DSI32 driver uses this service to control whether or not the `read()` function checks for buffer overflows. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_OVERFLOW_CHECK	DSI_IOCTL_RX_IO_OVERFLOW
Argument Type	unsigned long*	s32*
Check	TRUE	DSI_IO_OVERFLOW_CHECK
Ignore	FALSE	DSI_IO_OVERFLOW_IGNORE

In addition to the above listed service, the 24DSI driver provides the below listed related services.

Item	24DSI
Check for a buffer overflow condition.	DSI_IOCTL_AIN_BUF_OVERFLOW
Check for a buffer underflow condition.	DSI_IOCTL_AIN_BUF_UNDERFLOW
Check for an underflow when reading data.	DSI_IOCTL_RX_IO_UNDERFLOW

6.4.23. IOCTL_SET_RATE_DIVISOR

The 24DSI32 driver uses this service to set the NDIV value for the Rate Divider. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_RATE_DIVISOR	DSI_IOCTL_RATE_DIV_0_NDIV
Argument Type	unsigned long*	s32*

> **NOTE:** The 24DSI32 has only a single Rate Divider. The zero in the 24DSI service code refers to this Rate Divider. The index is present in the service code because the 24DSI driver supports the 24DSI12, which has two Rate Dividers.

6.4.24. IOCTL_SET_SW_SYNC

The 24DSI32 driver uses this service to initiate a Software Sync pulse at the cable interface. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_SW_SYNC	DSI_IOCTL_SW_SYNC
Argument Type	Not Used	Not Used

6.4.25. IOCTL_SET_TIMEOUT

The 24DSI32 driver uses this service to set the I/O timeout limit in seconds for read requests. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SET_TIMEOUT	DSI_IOCTL_RX_IO_TIMEOUT
Argument Type	unsigned long*	s32*

> **NOTE:** The 24DSI32 has no upper limit on the I/O timeout period. The 24DSI driver has an upper limit of one hour, which is 3600 seconds.

6.4.26. IOCTL_SYNCRONIZE_SCAN

The 24DSI32 driver uses this service to control the channel order of the sample data entering the input buffer. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary.

Item	24DSI32	24DSI
Service Macro	IOCTL_SYNCRONIZE_SCAN	DSI_IOCTL_CHANNEL_ORDER
Argument Type	unsigned long*	s32*
Asynchronous	FALSE	DSI_CHANNEL_ORDER_ASYNC
Synchronous	TRUE	DSI_CHANNEL_ORDER_SYNC

6.4.27. IOCTL_WRITE_REGISTER

The 24DSI32 driver uses this service to write to firmware registers. The 24DSI driver uses a corresponding service for the same purpose. Refer to the below table for a summary. For register macro definitions refer to section 6.2.2 on page 12.

Item	24DSI32	24DSI
Service Macro	IOCTL_WRITE_REGISTER	DSI_IOCTL_REG_WRITE
Argument Type	struct device_register_params*	gsc_reg_t*
Register	ulRegister	reg
Value	ulValue	value

6.5. Additional IOCTL Services

The 24DSI driver provides the following additional IOCTL services pertinent to 24DSI32 devices.

Description	24DSI
Select the source for the external clock output.	DSI_IOCTL_EXT_CLK_SRC
Configure the board's external burst trigger input feature.	DSI_IOCTL_EXT_TRIG
Query the driver for device and driver information.	DSI_IOCTL_QUERY
Set the Nrate value for non-PLL model boards.	DSI_IOCTL_RATE_GEN_A_NRATE †
Perform a read-modify-write operation on a GSC firmware register.	DSI_IOCTL_REG_MOD
Route the Threshold Flag to the cable interface.	DSI_IOCTL_THRES_FLAG_CBL

† The 24DSI32 has only a single Rate Generator. The "A" in the 24DSI Rate Generator service code refers to this Rate Generator. The alphabetic index is present in the service code because the 24DSI driver supports the 24DSI12, which has two Rate Generators.

Document History

Revision	Description
November 29, 2023	Updated release date. All Auto_Cal content for the new driver has been renamed to Autocal.
January 9, 2023	Updated release date. Minor editorial changes.
November 21, 2022	Minor editorial updates.
April 25, 2022	Updated release date.
May 19, 2021	Updated release date.
July 15, 2019	Added a licensing subsection.
December 14, 2018	Minor editorial changes. Updated release version.
November 5, 2018	Updated the inside cover page. Updated Block Mode DMA macro and associated information. Minor editorial changes.
September 15, 2015	Updated the release date.
July 31, 2015	Updated the driver version format. Changed the spelling of various Autocalibration related macros. Removed double underscore that prefaced various data types.
December 24, 2008	This is the initial release of this porting guide.