Application Report
SLVAE25–August 2018

Daisy Chain Implementation for Serial Peripheral Interface

Ishtiaque Amin *Analog Motor Drives*

ABSTRACT

Computers and other electronic systems use serial interfaces to provide for transfer of data between connected devices. Serial peripheral interface (SPI) is one type of serial communication interface that provides synchronous transfer of data between a master device, such as a micro-controller (MCU), and one or more peripherals (slave devices). In SPI, the master device generates a clock signal, a select signal and an input data signal (e.g., data transferred to the slave devices). The slave devices receive the input data signal synchronous with the clock signal while the select signal is active, and generate, synchronous with the clock signal, a data output signal for reception by the master device. SPI is a common form of interface in automotive applications for better flexibility, configurability, and fault reporting by the electronic components. The report describes the method for synchronous serial communication between a master device (MCU) and multiple slave devices (motor driver devices for example).

Contents

List of Figures

Trademarks

All trademarks are the property of their respective owners.

1 Functional Overview

Figure 1 shows daisy chain configuration to keep GPIO ports available when multiple devices are communicating to the same MCU by sharing the SPI bus. The MCU must be configured to generate a chip select signal, a header signal, and multiple address fields depending on the number of devices connected in series. In addition, the MCU generates a clock signal that allows the data to be synchronously transmitted through the chain. From Figure 1 the MCU sends out a master signal via SDI1 which gets decoded by each device in the chain, and the appropriate commands are executed. Details about the decoding method is described herein.

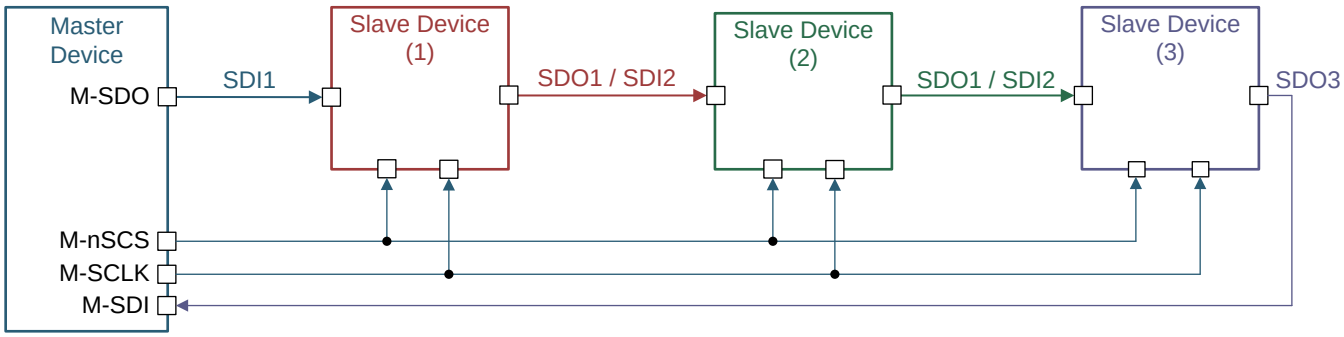

Figure 1. Three Motor Driver Devices Connected in Daisy Chain

The chip select signal (nSCS) defines a frame interval for provision to the motor driver devices. The clock signal is to control synchronous transfer of serial data between the MCU and the motor driver devices. The M-SDO is the data being sent from the MCU to the first motor driver device in the chain, and M-SDI is the data being received from the last motor driver device in the chain. The M-SDO consists of the header bytes, address bytes, and data bytes. The M-SDI, generated by the last motor driver device in the chain, consists of status bytes, header bytes (same as M-SDO), and report bytes.

From MCU to the motor driver devices, the header field is the first field to be transmitted in the frame interval, and specifies a number of slave devices communicatively coupled to the MCU. The header field is followed by address fields. The multiple address fields are to be transmitted in the frame interval. Each of the address fields corresponds to a different motor driver device. A first of the address fields transmitted by the MCU in the frame interval corresponds to the last motor driver device to receive the header field, and vice versa. The address fields are followed by data fields which carry the information to be executed in a given motor driver device.

2 Implementation Details

The traditional daisy chain configuration allows for a reduced number of terminals on the master device, but limits communication bandwidth. For example, in some implementations, two transactions (two communication frames) are required to read from a slave device. In systems that allow single frame reads, transfer speed of information through the serially connected slave devices is reduced as the number of slave devices increases.

The synchronous serial communication system described here employs the daisy chain configuration to reduce the number of terminals required on the master device, allows reads in a single frame, and provides a transfer bit rate that is independent of the number of slave devices in the chain. Some implementations provide operation with a single slave device without introducing additional protocol overhead.

Figure 1 shows a block diagram of an example of a synchronous serial communication system in accordance with the daisy chain implementation described in this section. The SPI system includes a MCU and one or more motor driver devices in series to communicate with the MCU.

For example, the MCU outputs data signal and the first motor driver device in the chain receives the data signal (SDI1). After performing any relevant operation, it passes along the outputs signal SDO1/SDI2, which includes a portion of the data provided via SDI1. The same sequencing is repeated throughout the entire chain until the final motor driver device is reached.

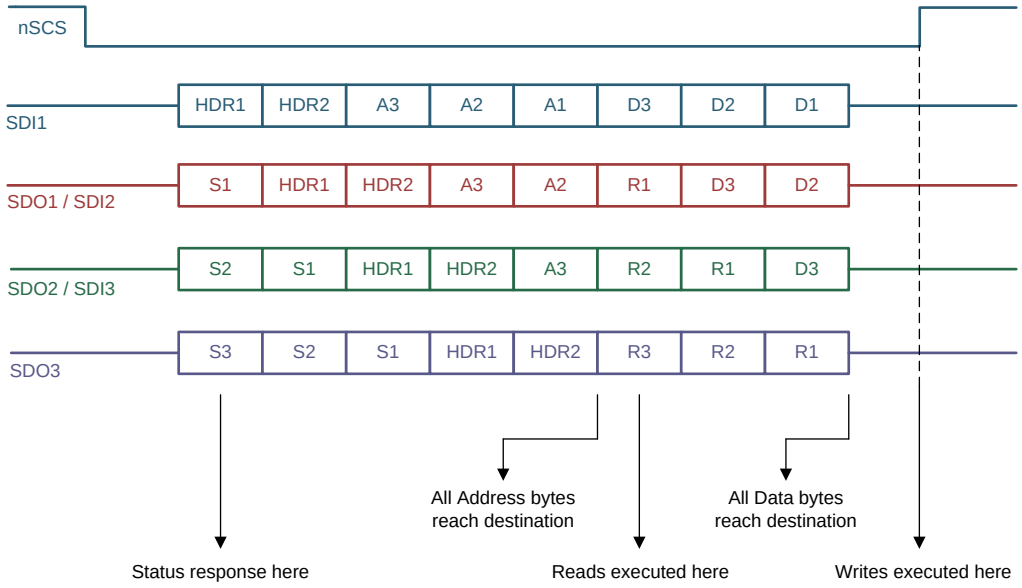

Figure 2. SPI Frame With Three Motor Driver (Slave) Devices

Figure 2 shows the data transmit and receive structure between the MCU and three motor driver devices. The first device in the chain receives data from the MCU in the following format for 3-device configuration: 2 bytes of header (HDRx) followed by 3 bytes of address (Ax) followed by 3 bytes of data (Dx). After the data has been transmitted through the chain, the MCU receives the data string in the following format for 3-device configuration: 3 bytes of status (Sx) followed by 2 bytes of header followed by 3 bytes of report (Rx).

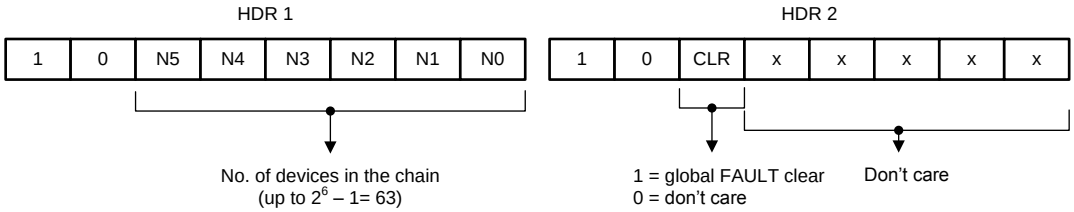

Figure 3. Header Bytes

The header bytes (HDRx), shown in Figure 3 contain information of the number of devices connected in the chain, and a global clear fault command that will clear the fault registers of all the devices on the rising edge of the chip select (nSCS) signal. Header bytes must start with 1 and 0 for the two MSBs. The header identification value is disposed at the start of the header field and is therefore the first value transmitted by the MCU that is received by each of the motor driver devices in a sequence determined by its position in the chain. Header values N5 through N0 are 6 bits dedicated to show the number of devices in the chain. Up to 63 devices can be connected in series for each daisy chain connection. The 5 LSBs of the HDR2 register are don't care bits that can be used by the MCU to determine integrity of the daisy chain connection.

The address field (Ax) is an implementation of the address field from the MCU that needs to be accessed for a particular motor driver device. The address field includes an identification value of 0 that identifies that byte as an address field. It also includes a read/write control value and an address value in the byte. The identification value specifies whether a location of the motor driver device corresponding to the address value is to be read or written. For example, if the read/write control value is logic 1, then the address corresponding address value is to be read, and if the read/write control value 334 is a logic 0, then the address corresponding to the address value is to be written.

The data field (Dx) specifies a value to be written at the address value of the motor driver device. For example, a motor driver device writes the value contained in the data field to the address specified in the address field corresponding to the motor driver device at the termination of the frame in which the data field is received.

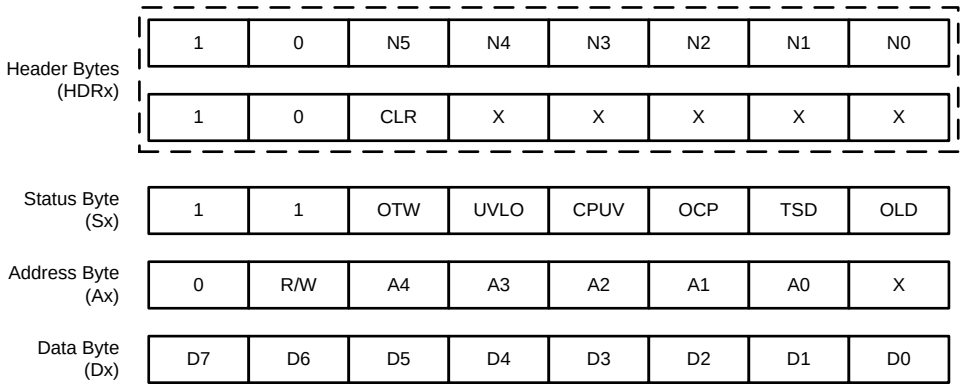

Figure 4. Contents of Header, Status, Address, and Data Bytes

When the chip-select signal is active, the motor driver devices begin to receive clock signal, and at each clock pulse, the motor driver devices transmit a status value. For example, while the motor driver device is receiving the header field, it is transmitting a status bits that provide information about the fault status register for each device in the daisy chain so that the MCU does not have to re-initiate another read command to read the fault status from a given motor driver device. This reduces the number of read commands from the MCU and makes the system more efficient to determining fault conditions flagged by a device.

The status field includes an identification value and a status value. The identification value must start with 1 and 1 for the two MSBs. The status field contains global fault bit identification for the motor driver device from which the status byte was generated. In Figure 4 the global fault bits shown refer to the DRV8873-Q1 device as an example. These six global fault bits, following the two identification bits, can vary depending on the motor driver device.

When data passes through a device, it determines the position of itself in the chain by counting the number of status bytes it receives followed by the first header byte. For example, in this 3-device configuration, device 2 in the chain receives two status bytes before receiving the HDR1 byte which is then followed by the HDR2 byte. From the two status bytes, the motor driver device can determine that its position is second in the chain. From the HDR2 byte, each device can determine how many devices are connected in the chain. In this way, the data only loads the relevant address and data byte in its buffer and bypasses the other bits. This protocol allows for faster communication without adding latency to the system for up to 63 devices in the chain. Figure 4 shows the encoding of a status byte.

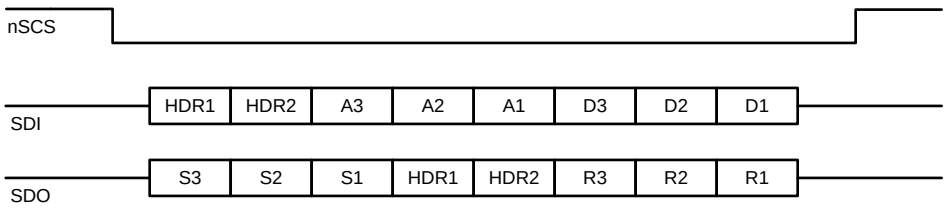

Figure 5. SPI Data Sequence between MCU and Three motor driver devices

Figure 5 shows how the data sequence should look like between the MCU and motor driver devices for three slave units connected in the chain. The number of header bytes will always remain 2, but the other bytes (status, address, data, and report) will scale according to the number of devices in series.

Such a daisy chain configuration also allows for only one motor driver device connected with the MCU without having to change the SDI data structure. Therefore, whether there is only one device or more than one device connected in a daisy chain, the SPI data transaction structure can remain the same. Only change would happen in the header byte 2 which carries the information of the total number of devices connected in the chain.

3 Summary

There are certain advantages of implementing daisy chain as described in this report. They include:

- Speed: large number of SPI slave devices (up to 63) can be connected on a single SPI bus without having to reduce frequency of the SPI transaction.
- Same-frame response: For read transactions, response from a given motor driver device is sent in the same frame, without having to command a second read transaction.
- Robustness: Header bytes sent by the micro-controller returns back to the micro-controller, after going through the entire chain, providing the ability to the micro-controller to continuously check for integrity of the chain connection.
- Broadcast global command: Header 2 byte can be used to broadcast a commands to all the slave devices to perform a certain action at the exact same time.

Refer to the *DRV8873-Q1 Automotive H-Bridge Motor Driver* data sheet for more information.

Glossary

A.1 Nomenclature Used in this Document

The following acronyms and initialisms are used in ths document:

MCU — Microcontroller unit

SPI — Serial peripheral Interface

MSB — Most significant bit

LSB — Least significant bit

SDO — Serial data out

SDI — Serial data in

GPIO — General purpose input-output

For a more comprehensive list of terms, acronyms, and definitions, refer to the *TI Glossary*.

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

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