Application Report
SPNA231–April 2016

High Speed Serial Bus Using the MibSPIP Module on Hercules™-Based Microcontrollers

Christian Herget

ABSTRACT

This application report describes how the Multi-Buffered Serial Peripheral Interface Module (MibSPIP) and the DMA modules found on many devices of the Hercules MCU family can be used to autonomously transfer data from and into the device embedded RAM. In addition to this, the parallel pin option of the MibSPIP module will be covered and used to increase the speed of the SPI link. This technique could be used to implement a high-speed SPI link between two or more Hercules devices, for example, FPGAs, ASICs, or other devices offering a SPI interface.

Project collateral and source code discussed in this application report can be downloaded from the following URL: http://www.ti.com/lit/zip/spna231.

NOTE: TI assumes no liability that the discussed implementation and provided code are free from faults, compliant to certain coding guidelines, nor was it developed in accordance with any standards. The implementer has to ensure and verify that the code conforms to appropriate rules and standards, and he has the sole responsibility to ensure and verify correct functionality in his application.

Contents

List of Figures

List of Tables

1 MibSPIP Basics

The multi-buffer is a feature-rich FIFO with dedicated RAM for transmit and receive buffers that can be assigned to individual transfer groups. A transfer group is a collection of buffers and can be assigned to certain chip selects and can be configured for a specific data format. A data format configures the baud-rate, transfer length and clocking mode for the serial peripheral interface (SPI) module.

The parallel pin option utilizes up to 8 parallel pins for transmission and reception to increase the bandwidth. More details about these features are discussed in the following sections.

The MibSPIP module is available on many devices of the Hercules device family (as shown in Table 1).

Table 1. Hercules Family MibSPIP Module and DMA Matrix (1) (2) (3) (4)

	MibSPIP1	MibSPIP2	MibSPIP3	MibSPIP4	MibSPIP5	DMA Controller
RM42	NPPO	SPI	SPI	NA	NA	NA
RM44	2 PPs	SPI	NPPO	SPI	4 PPs	Yes
RM46	2 PPs	SPI	NPPO	SPI	4 PPs	Yes
RM48	2 PPs	SPI	NPPO	SPI	4 PPs	Yes
RM57	2 PPs	NPPO	NPPO	NPPO	4 PPs	Yes
TMS570LS0432	NPPO	SPI	SPI	NA	NA	NA
TMS570LS0714	2 PPs	SPI	NPPO	SPI	4 PPs	Yes
TMS570LS1227	2 PPs	SPI	NPPO	SPI	4 PPs	Yes
TMS570LS3137	2 PPs	SPI	NPPO	SPI	4 PPs	Yes
TMS570LC4357	2 PPs	NPPO	NPPO	NPPO	4 PPs	Yes

(1) NA - Module not available
(2) NPPO - No parallel pin option available
(3) PPs - Amount of parallel pins available
(4) SPI - Standard SPI module without FIFO RAM and parallel pins

Table 1 only shows the superset devices of the different device families. For more details on the individual devices, see the device-specific data sheet.

1.1 Data Formats

To support different types of slaves in a SPI network, four independent SPI data formats are implemented. Each SPI data format includes the following features:

- SPI data word length, the data word length can be between 2 and 16 bits
- Prescaler to set the SPI baud rate (only necessary in master mode)
- SPI clock phase and polarity
- Disable chip select delays. The SPI module has the possibility to automatically add CS delays in master more to address the needs of certain SPI slaves.
- Enable of half duplex mode, this bit controls the I/O functionality of the Slave Out Master In (SOMI) and Save In Master Out (SIMO) lines.
- Shift direction can be used to select MSB first or LSB first, whereas, the position of the MSB depends on the configured data word length.
- Wait for the enable signal in master mode; this can be used to implement a hardware handshaking mode for the SPI communication.
- To increase fault detection of data transmission and reception, an odd or even parity bit can automatically be added and evaluated at the end of a data word.

- Configuration of a delay between subsequent data word transmissions.

1.2 5-Pin Operation (Hardware Handshaking)

The MibSPIP module has the option to enable a hardware handshaking mode, where the master waits for the slave to assert an enable signal. This can be used by the slave to delay a transmission until it has new data ready. The master implements a time out counter in hardware to automatically detect situations where the slave does not respond in time. Figure 1 shows the principle of the 5-pin operation.

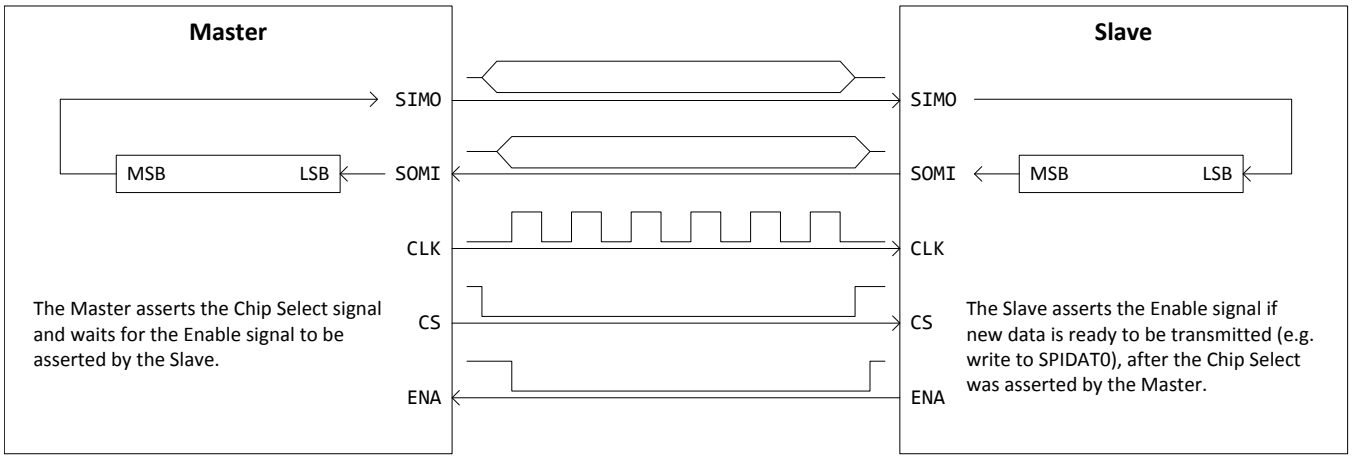

Figure 1. SPI in 5-Pin Mode

The 5-pin mode is used in the examples provided with this application report. However, this feature is not required. The usage is determined by the requirements of the application. The main advantage would be if one device acts as a slave it could delay the SPI transmission until new data was written into the multi-buffer RAM.

1.3 Parallel Pin Mode

The MibSPIP module has the option to use parallel SIMO and SOMI lines to increase the bandwidth of the communication. In addition to single SIMO and SOMI lines, two, four or eight lines can be used. Currently, only devices with up to four SIMO and SOMI lines are available. For more details, see Table 1 and the device-specific data sheet.

Figure 2 shows the MibSPIP in 4-pin parallel mode.

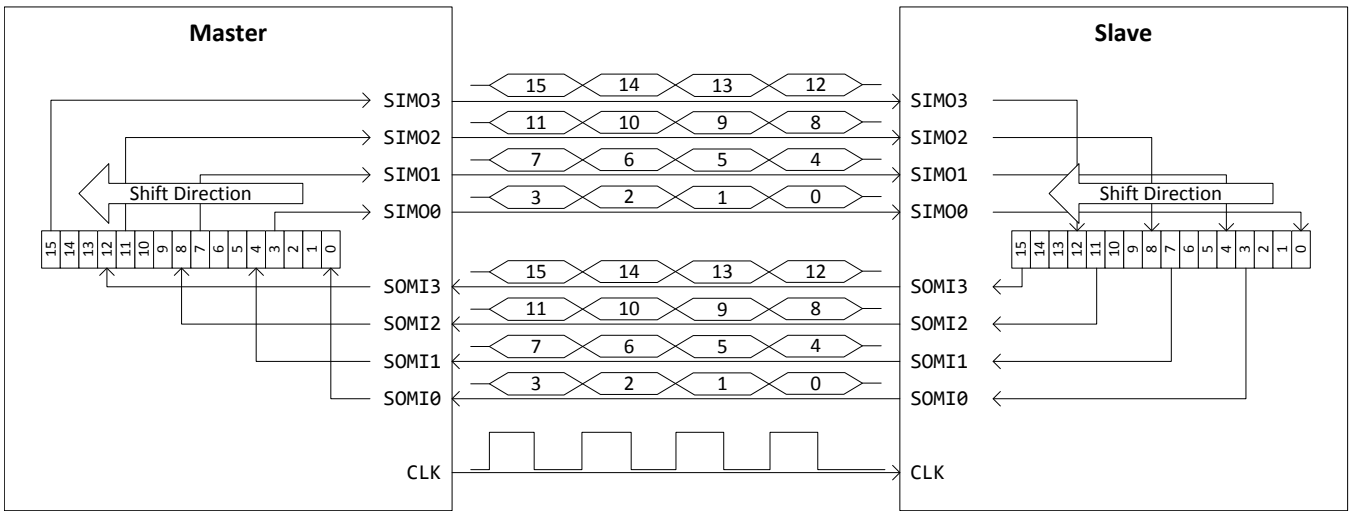

Figure 2. SPI With 4 Parallel Pins (MSB first)

If parity is enabled, the parity bit will be inserted in and expected at bit position 0. If, for example, 16 data bit are transmitted per word and parity is enabled, an extra clock cycle will be inserted to transmit the parity bit. This results in a significant performance drop in parallel pin mode (for example, 16 bit in the 4 parallel pin mode, 1 - 4/5 = 20%).

1.4 Transfer Groups and Multi-Buffer RAM

The MibSPIP module has a large RAM of up to 1kB associated, which can act as a FIFO buffer for the SPI communication. The RAM can be split up into up to eight transfer groups. The size of each transfer group is configurable and it is possible to assign individual data formats and chip selects to each transfer group.

The RAM is split into two banks of the same size up to 512 byte each, one for TX and one for RX. Each TX and RX buffer is 4 byte in size, which means that up to 128 buffers for both TX and RX are available. Figure 3 shows how the RAM is organized.

	Offset	Parity 35..32	Bit 31..16	Bit 15..0
TX Buffers	0x000	Parity 0	TX Control 0	TX Data 0
	0x004	Parity 1	TX Control 1	TX Data 1
	0x008	Parity 2	TX Control 2	TX Data 2
	. . .	...	...	...
	0x1FC	Parity 127	TX Control 127	TX Data 127
RX Buffers	0x200	Parity 0	RX Status 0	RX Data 0
	0x204	Parity 1	RX Status 1	RX Data 1
	0x208	Parity 2	RX Status 2	RX Data 2
	. . .	...	...	...
	0x3FC	Parity 127	RX Status 127	RX Data 127

Figure 3. Multi-Buffer RAM Configuration

Note that the parity bits in the RAM are optional, not every device has parity integrated. Some have ECC instead or might have no data protection at all.

1.5 DMA Trigger From Multi-Buffer RAM

The MibSPIP module has several options to trigger DMA transfers. This document only describes how to use the DMA in conjunction with the multi-buffer RAM.

The MibSPIP offers up to eight DMA channels (for TX and RX). All of the DMA channels are individually programmable and can be hooked up to any buffer in the multi-buffer RAM. The MibSPIP provides up to 16 DMA request lines, DMA requests from any channel can be programmed to be routed through any of these 16 lines. A DMA transfer can be triggered by both TX and RX events.

The DMA channels are assigned (mapped) to buffers in the multi-buffer RAM and not to transfer groups. This means that in the case that the SPI state machine has finished the read out of a TX buffer, which is configured to trigger a TX DMA transfer, the DMA will be triggered. The same is true for RX buffers, after the SPI state machine has finished writing to a RX buffer, which is configured to trigger a RX DMA transfer, a DMA will be triggered.

Figure 4 shows how this mapping could look like.

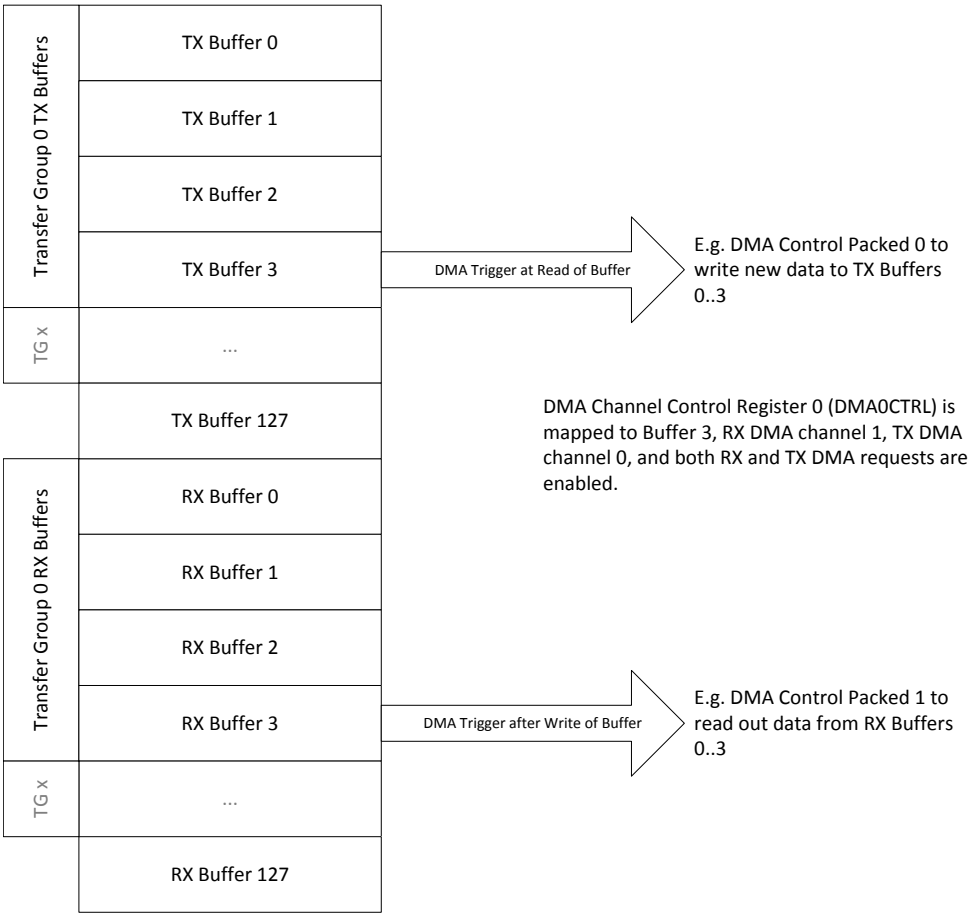

Figure 4. DMA Channel Mapping in Multi-Buffer RAM

The Transfer Group 0, shown in Figure 4, is configured to the size of 4 buffers. The DMA should be triggered at both TX and RX events, when the last buffer has been processed (Read and Write) by the SPI state machine. Therefore, the DMA Channel Mapping in the MibSPIP module has to be configured to react on buffer 3, as this is the fourth (last) buffer in Transfer Group 0.

In order to relax time constraints on reading and writing of the buffers by the DMA, it could be beneficial to trigger the DMA at two points: after the first half of the Transfer Group is processed and after the second half is processed. With this and the right DMA settings, it can be ensured that no TX or RX data will be overwritten by either the DMA or the SPI state machine.

As the largest SPI data size is 16-bit and the buffer size is 32-bit, the DMA can be programed in a way that it points to the first data (offset 0x2) and that it will perform an increment of 4 byte per read or write in order to get to the next buffers data. This can be combined with the data packing and unpacking capability of the DMA to group four 16-bit transfers in or out of the multi-buffer RAM into a single 64-bit transfer to the main SRAM, in order to maximize throughput and to minimize the amount of SRAM accesses. For more details on these concepts, see Section 2.1.2 and Section 2.2.2 and for more details about the DMA controller, see the device-specific Technical Reference Manual.

2 Implementation

The setup is split into two sides:

- Master (transmitter) side
- Slave (receiver) side

Both sides have individual Hercules MCU's with MibSPIP support.

The goal of the described setup is to perform burst data transfers of 1024 bit (128 byte) from the master to the slave side. The setup is unidirectional for the sake of simplicity and the slave only transmits constant dummy data, whereas, the master transmits randomized data to simulate a more realistic scenario.

2.1 Master Transmitter Side Single Buffer

The master side is configured to transmit data bursts of 1024 bit (128 byte). For this purpose, the DMA is used to copy a prepared buffer holding 128 byte of data from the embedded SRAM into the MibSPIP TX RAM. The MibSPIP starts a new SPI transmission as soon as new data is copied into a TX buffer. The sequence in this example is controlled by software, as the software has to trigger the DMA request for each 128 byte data set to be transmitted.

2.1.1 MibSPIP Setup

The MibSPIP5 module is used in the example application. The module configuration is described in the following list:

- Data Format
  - The data is transmitted in 16-bit chunks with the 4 parallel pin mode to increase the bandwidth as described in Section 1.3.
- Transfer Group
  - One transfer group with 64 buffers is used in this example. This allows the reception and storage of up to 128 byte (128 x 2 byte) of data.
- Interrupts
  - No interrupts are used
- DMA Trigger
  - The DMA is set to hardware trigger type the MibSPIP DMA Channel Control Register will be used to trigger the DMA. Strictly, this is not necessary as software could also directly trigger the DMA transfer.

2.1.2 DMA Control Packet

The DMA controller has to write 2 x 4 TX data buffers into the MibSPIP5 RAM. For this, a single 64-bit read from the SRAM will be unpacked to four 16-bit writes to the TX Data buffers. To only write the TX data to the MibSPIP5 RAM and not overwrite the TX Control bit, the following setup was chosen:

- Read Element Size
  - 64-bit, to reduce the number of reads performed by the DMA from the SRAM, in order to minimize bandwidth usage, as the RX Data portion is 16-bit wide.
- Write Element Size
  - Write element size
- Element Transfer Count
  - Has to be 64 as 64 transmit data buffers should used
- Frame Transfer Count
  - Can simply be one as software will take care about buffer usage
- Transfer Type
  - Is don't care, as in this case, the frame and bock sizes are equal

- Source Address Element Index (Offset)
  - Has to be 8 byte, for 64-bit reads
- Source Address Frame Index (Offset)
  - Can be zero as only one source buffer is used
- Destination Address Element Index (Offset)
  - Is irrelevant as long as only one write per frame is performed (Set to Zero)
- Destination Address Frame Index (Offset)
  - Has to be 4 byte so that only the transmit data portion of the MibSPIP transmit buffers are written to by the DMA
- Address Modes
  - Have to be set to indexed write, can be set to Index or auto increment for read

Figure 5 shows how the data from the SRAM is copied and packed into the MibSPIP5 RAM.

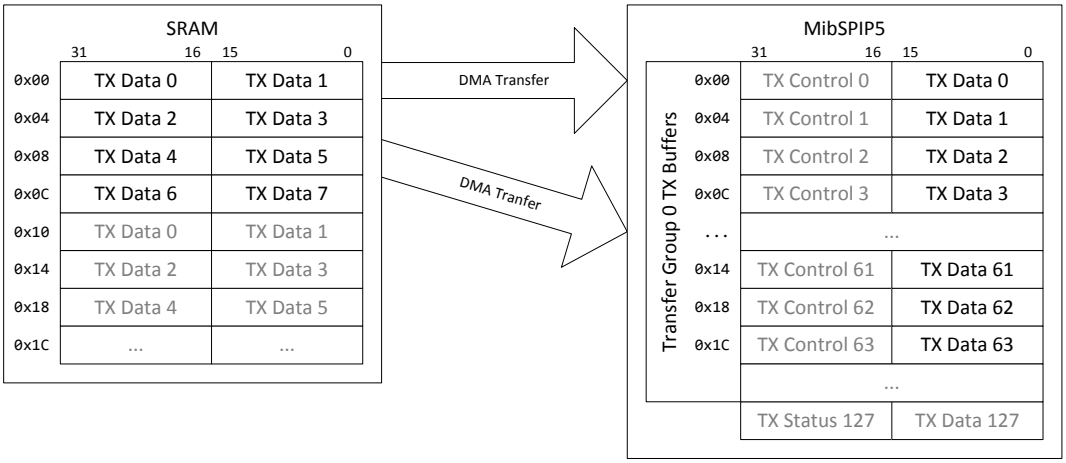

Figure 5. DMA Data Unpacking (64 to 4 x 16-bit)

The complete buffer size in the SRAM can be calculated as follows:

- One Control Packet (CP)
  - Read element size 64-bit (8 byte)
  - Write element size 16-bit (2 byte)
  - Element count 64 read elements (64 x 2 byte = 128 byte)
  - Frame count 1 element (1 x 128 byte = 128 byte)

2.2 Slave Receiver Side

Figure 6 shows how the MibSPIP and DMA setup is done in principle. The individual aspects will be discussed in the subsequent sections.

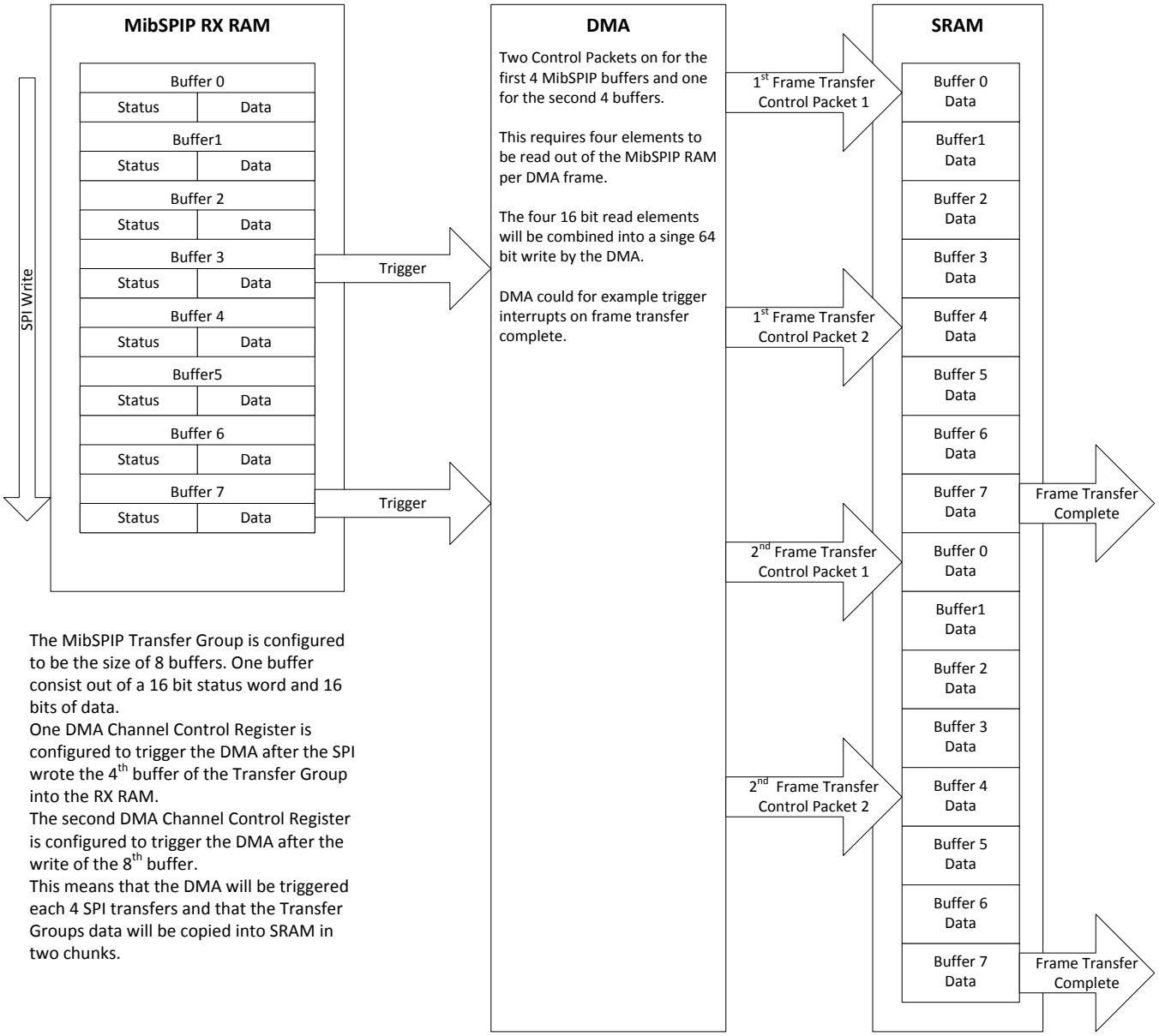

Figure 6. Receiver Side Operational Block Diagram

The slave has two 128 byte buffers implemented and checks the data integrity after each buffer has been received.

2.2.1 MibSPIP Setup

The MibSPIP5 module is used in the example application. The module configuration is described in the following list:

- Data Format
  - The data is transmitted in 16-bit chunks with the 4 parallel pin mode to increase the bandwidth as described in Section 1.3.
- Transfer Group
  - One transfer group with 8 buffers is used in this example. This allows the reception and storage of up to 16 byte (8 x 2 byte) of data until the buffer overflows. To prevent this, the DMA is used to copy the data portions of the buffers into the SRAM.
- Interrupts
  - As the complete data transfer is managed via the DMA controller, no interrupts for SPI data processing are necessary. However, the MibSPIP module offers some interrupts for fault detection, which should be enabled to be able to react on faults during the data transmission and on the unlikely case of a data overflow.
- DMA Trigger
  - The DMA will trigger 4 SPI transfers (4x16 bit). Then, it will read (four times) 16 bit of data from the multi-buffer RX RAM and pack it into a single 64-bit write into the SRAM. This means that the initial (read) element transfer count has to be 4. Two DMA control packets are used: one for the first and one for the second four buffers. The use of two DMA control packets will relax the timing requirements, reducing the likelihood that the RX data gets overwritten. The DMA setup is described in detail in Section 2.2.2.

2.2.2 DMA Control Packet

The DMA has to read 2 x 4 RX data buffers from the MibSPIP5 RAM; pack it into a 64-bit word and write it into a buffer in the SRAM. Two control packets are used: one for the first four RX Data buffers and one for the second four RX Data buffers as described in Section 2.2.1.

To only read the RX Data from the MibSPIP5 RAM and not the RX Status bit, the following setup was chosen:

- Read Element Size
  - 16 bit, as the RX data portion is 16-bit wide
- Write Element Size
  - 64 bit, to reduce the number of writes performed by the DMA to the SRAM and to minimize the necessary bandwidth
- Element Transfer Count
  - Has to be 4 as 4 RX data buffers should be read
- Frame Transfer Count
  - Has to be adjusted in a way to fit the reserved buffer space in the SRAM. It is 16, in this example, which leads to 16 x 64-bit or 128 byte of data. Remember that two DMA control packets are used, thus, the total buffer size will be 2 x 128 byte or 256 byte.
- Transfer Type
  - Has to be set to frame transfer.
- Source Address Element Index (Offset)
  - Has to be 4 byte, to only read out every second 16-bit or in other words only the RX data portion of the RX buffers
- Source Address Frame Index (Offset)
  - Has to be zero, as always the same buffers should be read
- Destination Address Element Index (Offset)
  - Is irrelevant as long as only one write per frame is performed (set to zero)

- Destination Address Frame Index (Offset)
  - Has to be 16 byte so that only each second 8 byte buffer is written by each DMA Control Packed. Keep in mind that two DMA control packets are used.
- Address Modes
  - Has to be set to indexed for both Read and Write

Figure 7 shows how the data from the MibSPIP5 RAM is copied and packed into the SRAM.

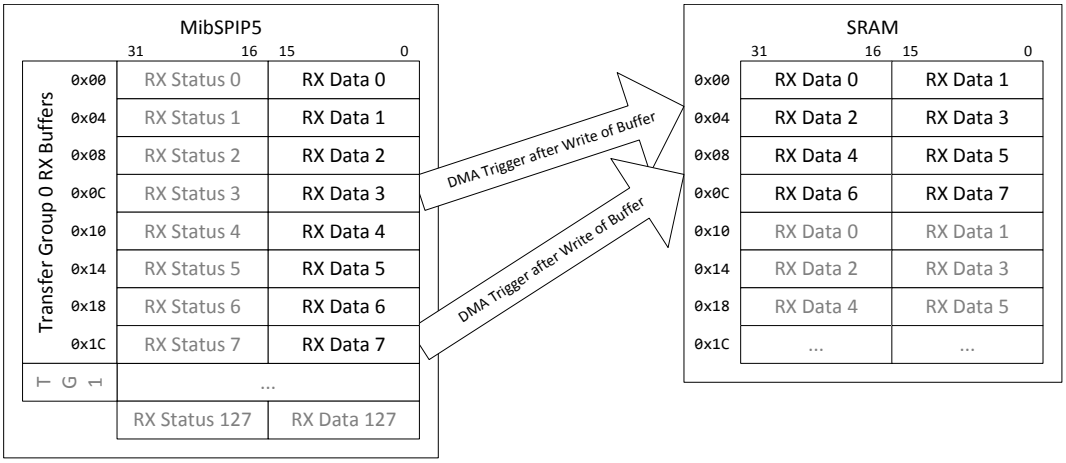

Figure 7. DMA Data Packing (4*16 to 64-bit)

The complete buffer size in the SRAM can be calculated as follows:

- One Control Packet (CP)
  - Read element size 16-bit (2 byte)
  - Write element size 64-bit (8 byte)
  - Element count 4 read elements (4 x 2 byte = 8 byte)
  - Frame count 16 elements (16 *x 8 byte = 128 byte)
- Two alternating CP's 2 x 128 byte = 256 byte

2.2.3 Interrupts

The second DMA control packet (CP) will trigger two interrupts to the main CPU.

One after the first half of a block has been transmitted, or in other words after the first 8 frames have been transmitted. This interrupt is called first half of block complete (HBC) interrupt. The first 8 frames of both DMA control packets do represent one complete 128 byte block (2 CP's x 8 frames x 8 byte = 128 byte).

And, the second interrupt after the second 8 frames, or in other words, after a complete block transfer. This interrupt is called block transfer complete (BTC) interrupt. The first 8 frames of both DMA control packets represent another complete 128 byte block.

With this setup, the CPU receives an interrupt after each 128 byte block has been received and copied into the SRAM. Figure 8 shows how the 16 frames are copied from the MibSPIP5 RAM into the SRAM by the DMA controller and when the HBC and BTC interrupts are triggered.

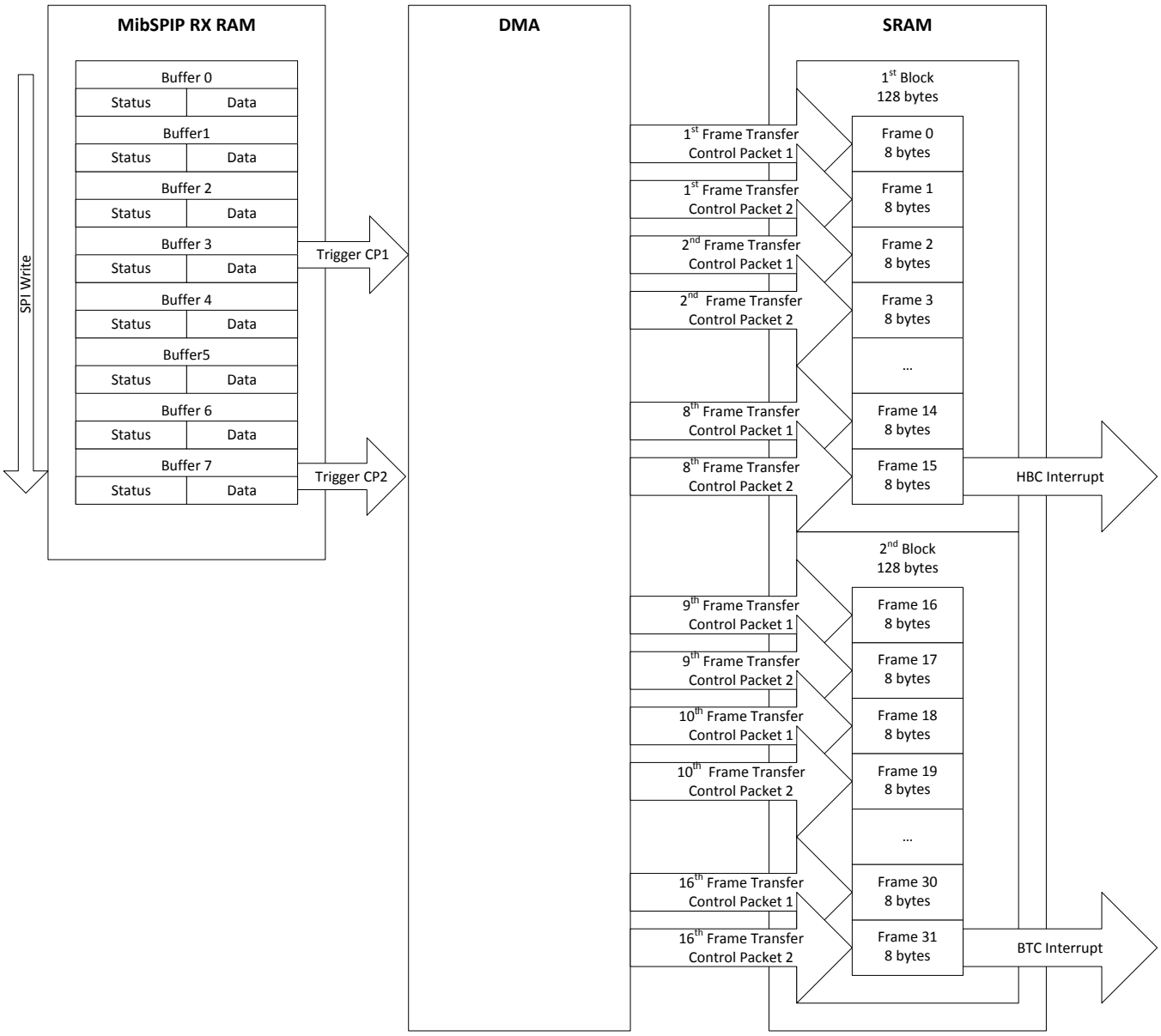

Figure 8. Receiver Side Block Diagram DMA Interrupts

3 Test Setup and Results

This document comes with three Code Composer Studio™ projects: one for the Master side and two for the Slave side.

1. MibSPIP_Master_Single_DMA (as described in Section 2.1)
2. MibSPIP_Slave_Double_DMA (as described in Section 2.2)
3. MibSPIP_Slave_Single_DMA

The results presented in Section 3.2 are based on projects 1 and 2.

3.1 Test Setup

Two TMDSRM48HDK boards were connected via a blue wired adapter PCB: one TMDSRM48HDK acts as the master and the second one as the slave for the SPI communication.

Figure 9 shows the connections made by the adapter PCB.

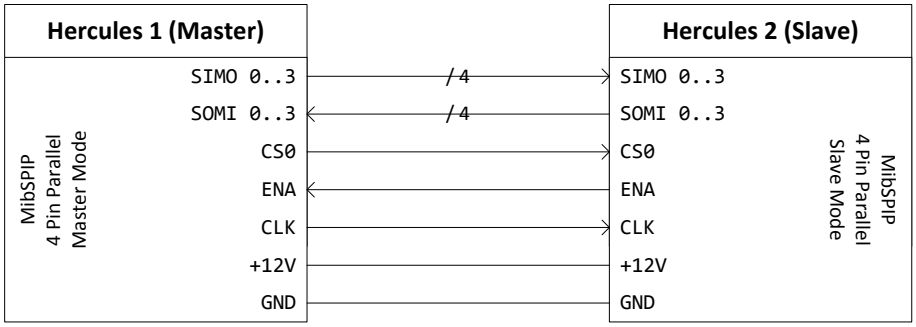

Figure 9. Board Connection Schematic

Figure 10 shows a picture of the whole setup.

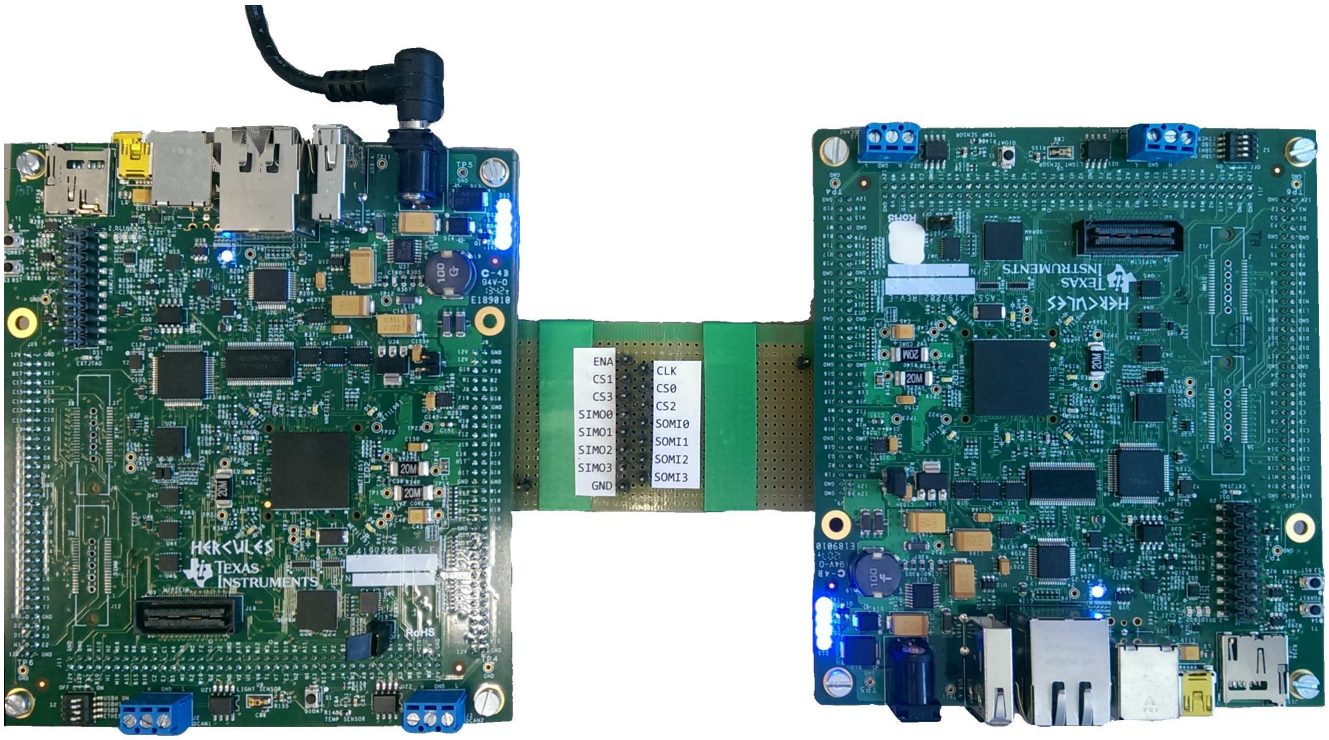

Figure 10. Board Connection Test Setup

3.2 Results

3.2.1 Using the 5-Pin Operation (ENA pin)

Blocks of 1024 bits were transmitted periodically, the ENA pin was used to delay the master until the slave is ready for data reception. This lead to the following waveform:

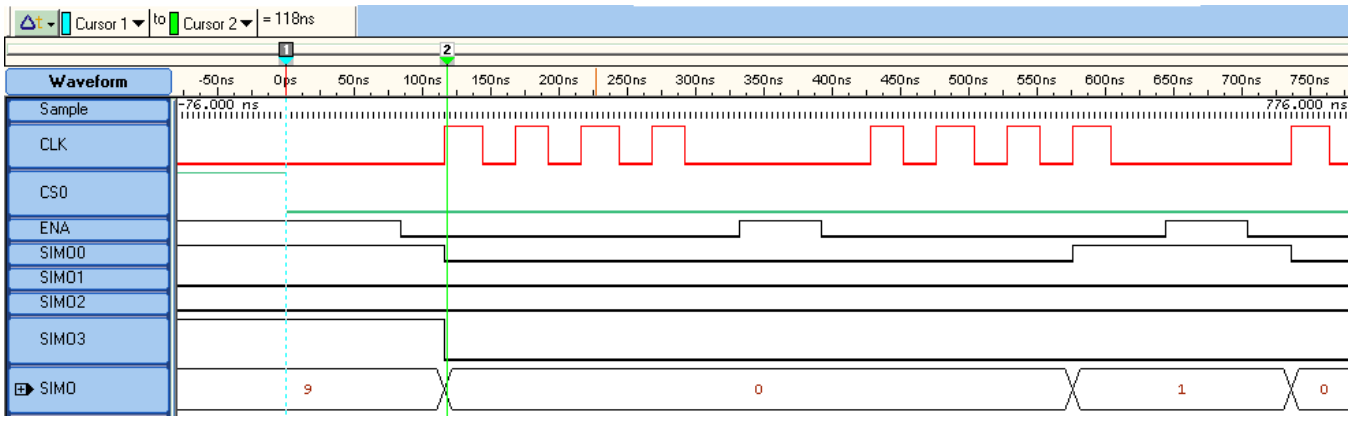

Figure 11. C2TDELAY, 5-Pin Operation (ENA pin)

Figure 11 shows how the 16 bits per frame are transferred in just 4 clock cycles using the 4 parallel pin feature (SIMO0..3). The third signal shows the ENA signal from the slave to the master. It can be seen that the master delays its next transmission to the ENA signal until it goes low again.

The measurement feature of the logic analyzer was used to measure the data transfer rate. The low pulse of the CS pin stays low for the entire 1024 bit block. It was measured with 20.172 µs, which leads to the following data rate:

$$\frac{1024\ bit}{20.172\ \mu s} = 50.8\ Mbit/s \tag{1}$$

The SPI clock was set to 20 MHz, so the theoretical maximum data rate is:

$$20M\frac{1}{s} \times 4\frac{bit}{clock} = 80\ Mbit/s \tag{2}$$

This means that the efficiency is about 63%.

The average length of the ENA pulses are measured with 247.489 ns. That means that the data rate within the 16 bit frames is:

$$\frac{16\ bit}{247.489\ ns} = 64.6\ Mbit/s \tag{3}$$

3.2.2 Using the 5-Pin Operation and Insert Parity

Blocks of 1024 bits were transmitted periodically, the ENA pin was used to delay the master until the slave is ready for data reception and a parity bit was added in addition to the 16 data bits. This leads to the following waveform:

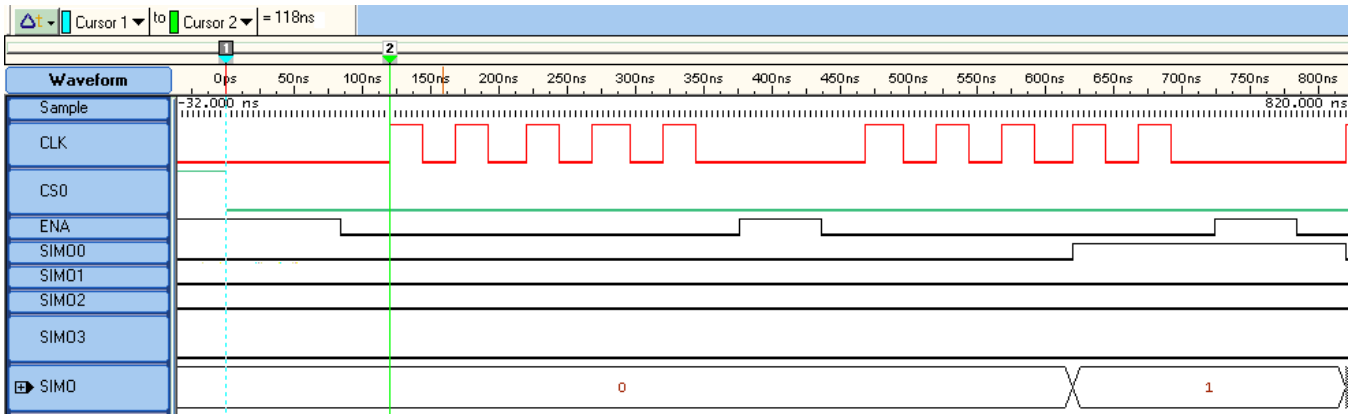

Figure 12. C2TDELAY, 5-Pin Operation and Insert Parity

Figure 12 shows how the 16 bits per frame are transferred in 5 clock cycles using the 4 parallel pin feature (SIMO0..3). An additional clock cycle is needed compared to the previous example due to the extra parity bit.

The measurement feature of the logic analyzer was used to measure the data transfer rate. The low pulse of the CS pin stays low for the entire 1024 bit block. It was measured with 22.529 µs, which leads to the following data rate:

$$\frac{1024\ bit}{22.529\ \mu s} = 45.5\ Mbit/s \tag{4}$$

The SPI clock was set to 20 MHz, the theoretical maximum data rate taking the extra clock cycle into consideration is:

4405016

$$20M\frac{1}{s} \times 4\frac{bit}{clock} \times \frac{4}{5} = 64\ Mbit/s \tag{5}$$

This means that the efficiency is about 71%.

The average length of the ENA pulses is measured with 288.091 ns. That means that the data rate within the 16 bit frames is:

$$\frac{16\ bit}{288.091\ ns} = 55.5\ Mbit/s \tag{6}$$

The expected and measured data transfer rates drop significantly when inserting an extra parity bit, as the extra clock cycle needed has a significant impact (5 cycles for 17 bits vs. 4 cycles for 16 bits).

3.2.3 Using the 4-Pin Operation With Optimized Delays

Blocks of 1024 bits were transmitted periodically, the master sets the timing for the transmission, the ENA pin was not used.

The Chip Select 2 Transmission Start (C2TDELAY) was set to 7 and CSHOLD was enabled, which translates to the following timing:

$$t_{C2TDELAY} = \frac{C2TDELAY + 3}{f_{VCLK}} = \frac{7 + 3}{100\ MHz} = 100\ ns \tag{7}$$

This fits well with the measurement shown in Figure 13.

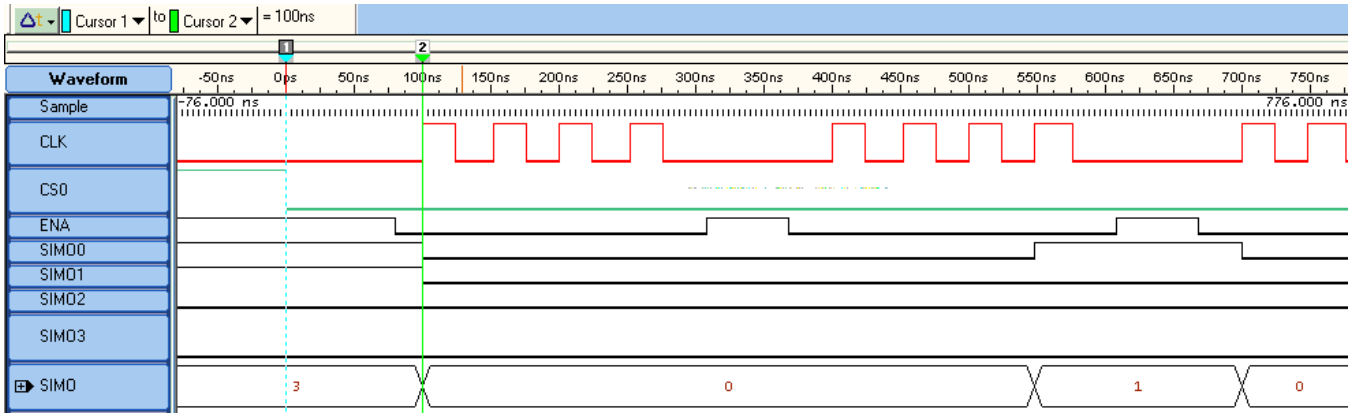

Figure 13. C2TDELAY, 4-Pin Operation With Optimized Delays

The WDELAY (delay between individual frames) was set to 5.

$$t_{WDELAY} = \frac{WDELAY + 2}{f_{VCLK}} = \frac{5 + 2}{100\ MHz} = 70\ ns \tag{8}$$

However, the delay between two frames is not only defined by WDELAY. In addition, the minimum C2TDELAY and T2CDELAY timings also have to be considered:

$$t_{T2CDELAY\_min\_} = \frac{T2CDELAY + 1}{f_{VCLK}} = \frac{0 + 2}{100\ MHz} = 20\ ns \tag{9}$$

$$t_{T2CDELAY\_min\_} = \frac{T2CDELAY + 1}{f_{VCLK}} = \frac{0 + 1}{100\ MHz} = 10\ ns \tag{10}$$

Depending on the clock phase setting, an additional delay might be added to either C2T or T2C. In this case, clock phase = 0, an additional 0.5 SPICLK periods will be added to T2CDELAY. As SPICLK is 20 MHz, the period time is 50 ns and 0.5 SPICLK periods, which equals 25 ns. The means that the total delay between two SPI frames sums up to, 70 ns + 20 ns + 10 ns + 25 ns = 125 ns. This fits well with the measurement (124 ns) shown in Figure 14.

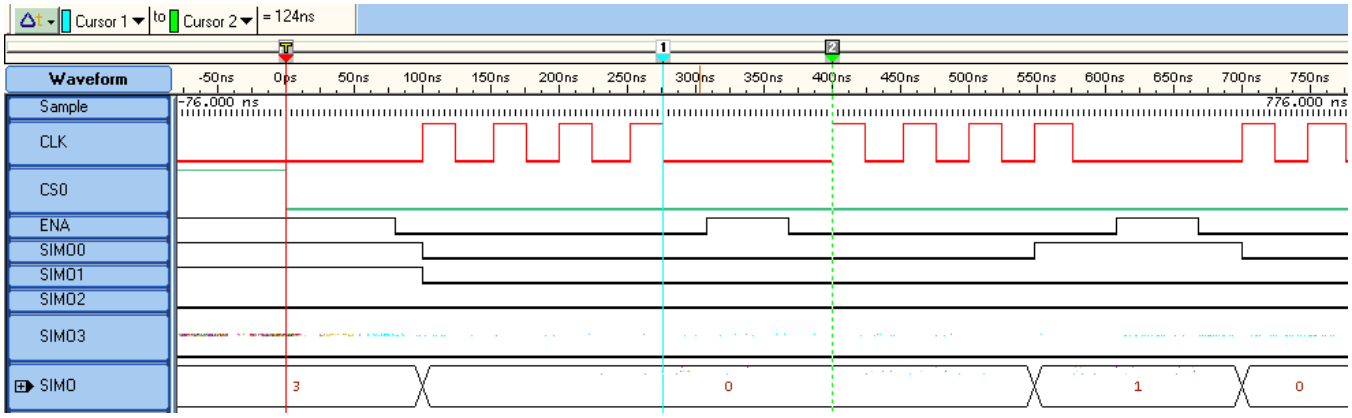

Figure 14. Delay Between to SPI Frames, 4-Pin Operation With Optimized Delays

The MibSPIP on RM48 can run with up to 25 MHz, so the maximum data rate could be improved by about 25%. However, the resulting data rate and the overhead will stay about the same. The delay time between individual frames (WDELAY) is necessary to give the MibSPIP slave enough time to copy the received data from the shift register to the MibSPIP RAM and vice versa, and also to give the DMA time to copy the data from the MibSPIP RAM to the SRAM. If the timing had been too aggressive, data loss would have be the consequence. The MibSPIP is able to signal this fault condition to the CPU via an interrupt.

The measurement feature of the logic analyzer was used to measure the data transfer rate. The low pulse of the CS pin stays low for the entire 1024 bit block. It was measured with 19.284 µs, which leads to the following data rate:

$$\frac{1024\ bit}{19.284\ \mu s} = 53.1\ Mbit/s \tag{11}$$

The SPI clock was set to 20 MHz, so the theoretical maximum data rate is:

$$20M\frac{1}{s} \times 4\frac{bits}{clock} = 80\ Mbit/s \tag{12}$$

This means that the efficiency is about 66%.

4 Conclusion

The maximum data rate can be achieved when fixed chip select and frame delay timings are used on the master side. The slave side limits the maximum throughput as a minimum and frame delay has to be inserted, to give the MibSPIP enough time to copy its data from and to the MibSPIP RAM. The ENA pin can be used to automatically ensure, the right timings are used. However, using the ENA pin will result in a slight jitter of the transfers, as the clocks between master and slave aren't synchronized.

Table 2. Summary of the Measurement Results

Mode	1024 Bit Block
5 Pin	50.8 MBPS
5 Pin + Parity	45.5 MBPS
4 Pin	53.1 MBPS

When connecting a Hercules MCU to another slave like a FPGA, the Hercules MCU should be configured as master to get the best performance out of the system. In such a case, the ENA pin could still be used to delay the master until the slave has data ready. Another option to start the transmission from the slave side is to use the hardware trigger feature of the MibSPIP. For a list of the available trigger sources (Event Trigger Hookup), see the device-specific data sheet.

The described method with using the parallel pin feature of the MibSPIP module on Hercules provides a simple and scalable solution to connect different devices on a board with a high speed link across the entire Hercules portfolio. Other methods like using Controller Area Network (CAN), External Memory Interface (EMIF) or the MII interface, would not be that easy to implement, not using the peripheral in the intended way (CAN or MII without phy) or with a higher board level complexity (EMIF).

5 References

- RM48L952 16- and 32-Bit RISC Flash Microcontroller Data Manual
- RM48x 16/32-Bit RISC Flash Microcontroller Technical Reference Manual

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information 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.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

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