Freescale Semiconductor
Application Note

Document Number: AN3862
Rev. 0, 05/2009

Configuring and Using the Phantom Port Feature on the MPC5668

by: Daniel McKenna
East Kilbride, Scotland

The Phantom Port feature on the MPC5668 microcontroller facilitates a low-cost solution for the addition of up to 116 extra General-Purpose Output pins by using up to four Deserial-Serial Peripheral Interface (DSPI) modules. Once initialized, Phantom Ports can be treated in software as standard ports and require no CPU overhead.

A phantom port uses an external Shift Register in order to provide more output pins. The MCU sends the port values to the register via the DSPI module, as shown in Figure 1.

Contents

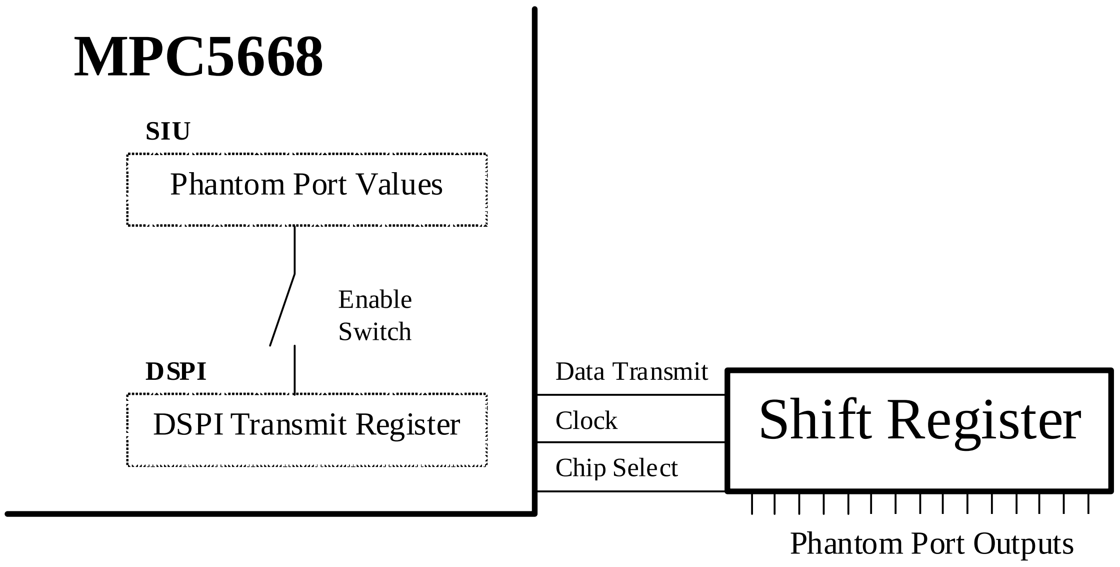


Figure 1.

This application note will discuss both the hardware and software required to create the 32-bit phantom port seen on the MPC5668 Evaluation Board.

1 DSPI Overview

A Serial Peripheral Interface (SPI) is a full-duplex synchronous serial data link used to facilitate communications between the microcontroller and external devices. A SPI connection is set up in a master-slave configuration, with all communications initiated by the master. The maximum speed of the connection is limited by the clock and pad speed restrictions of the communicating devices.

Interdevice communications are carried out using four separate signals as shown in Figure 2.

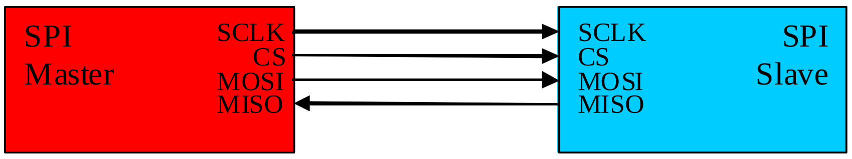


Figure 2.

Signal	Description
SCLK	Serial Clock — driven by master.
CS	Peripheral Chip Select — master can have multiple CSs to allow communication with several slaves.
MOSI	Master Out Slave In — carries data from master to slave.
MISO	Master In Slave Out — carries data from slave to master.

The MPC5668 contains four Deserial-Serial Peripheral Interface (DSPI) modules. The deserial element allows the module to automatically serialize/deserialize input/output signals to memory-mapped registers. Each DSPI module has six chip selects; as an external device will only transmit/receive if its chip select has been activated, each DSPI module can communicate independently with six individual external devices.

A timing diagram of a standard DSPI transmission between two nodes can be seen below:

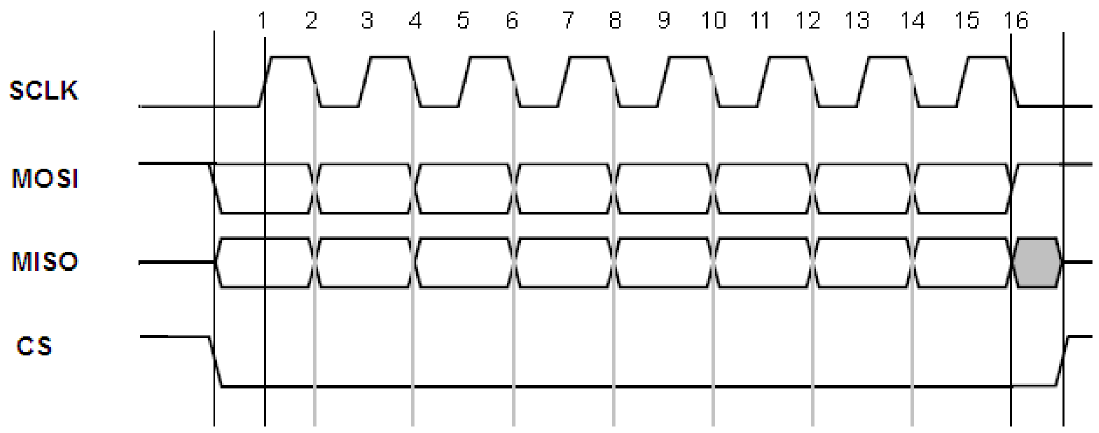


Figure 3.

Note that the transfer is fully synchronous, with data lines changing on a negative SCLK edge, and that the SCLK is only active, and data is only exchanged, when the Chip Select Signal is active.

For further information on the DSPI module, consult *AN2867: Using the DSPI Module on the MPC5500 Family.*

2 Phantom Ports

The creation of phantom ports requires the use of external shift registers along with the on-chip DSPI and System Integration Unit (SIU). After initialization, phantom ports can be treated in the same manner as standard ports in software. The MPC5668 allows for up to four phantom ports, each with up to 32 output pins.

The SIU contains memory-mapped phantom port output registers which hold the current value of the 32 signals of each port. Once enabled, any change in these registers causes internal logic to trigger a corresponding DSPI transmission which will serially transmit the new value of that port. This is done with no intervention from the core. The data is serially transmitted to the external shift registers that perform a serial-to-parallel transition and output the result.

3 Hardware Implementation

The external shift register receives the DSPI data serially, and then outputs it in parallel. It does not have to support full DSPI, as not all pins will be required, in other words there will be no return transmission on the MISO line from the shift register to the MCU.

The shift register also has to support latching of the result through the use of double buffers. This ensures that the output data (stored in the secondary buffer) will not fluctuate whilst new serial data is arriving into the primary buffer. Thus, the previous port value will continue to be output until the DSPI transmission is complete and new data has been fully received.

For the MPC5668 EVB, four Fairchild MM74HC594 ‘8-bit Shift Register with Output Registers’ were selected, as they represent better value than a single 32-bit part and can operate at both 3.3 V and 5 V, making them fully compatible with the MPC5668’s outputs.

A table of the devices’ pins can be seen below:

Pin	Description
SER	Serial Input
Qa-Qh	Parallel outputs, MSB=Qa
Qh'	Qh value at previous clock pulse — used to chain shift registers together
GND	Ground
Vcc	Supply voltage (2–6 V)
SCK	Serial Clock
RCK	Transfer input value to parallel output storage register on positive edge
SCLR	Shift register clear (active low)
RCLR	Storage register clear (active low)

The SER pin is connected to the DSPI Transmit pin on the MCU and is used to serially receive the port values. Each bit of the port is clocked by the DSPI Serial Clock pin which is connected to SCK on the shift register. The DSPI Chip Select pin is connected to the RCK pin to trigger the shift register to move the newly received data to the output pins. Both the SCLR and RCLR pins can either be connected to a GPIO pin on the MCU to allow software reset of the shift registers, or they can be connected to the reset circuitry to allow hardware resets only.

This can be seen in Figure 4.

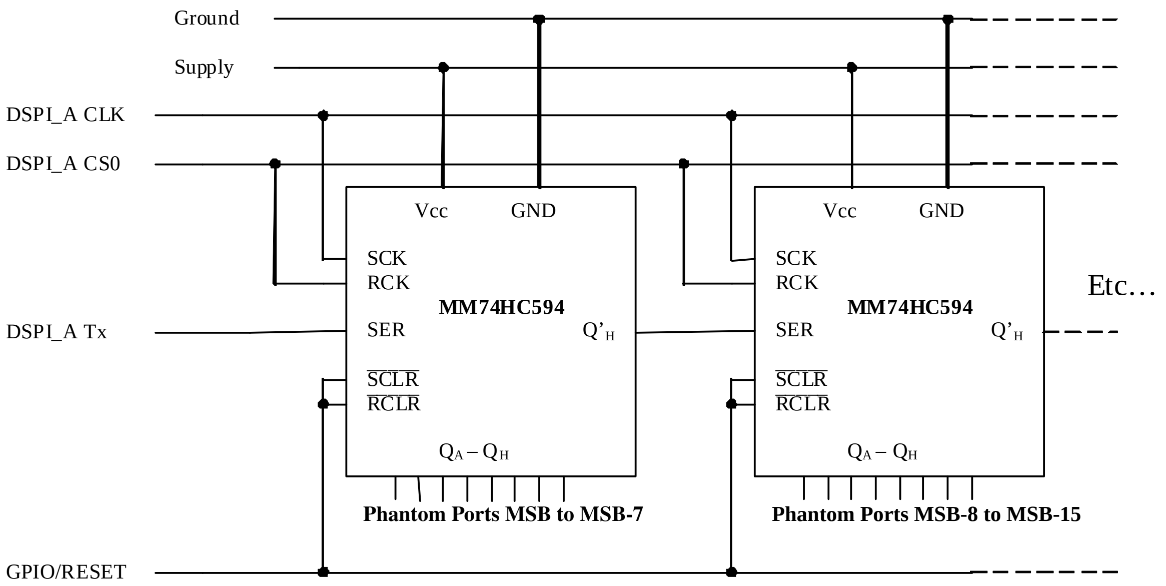


Figure 4.

In order to make use of all 32 pins of a phantom port, four 8-bit shift registers are required to be chained together.

4 Software Implementation

The main steps to initialize the phantom ports in software are:

1. Configure DSPI pins correctly.
2. Initialize the DSPI module.
3. Configure the internal logic.
4. Enable the DSPI module and set the output value of the port.

These steps will now be examined in detail.

This section can be read along with the DSPI chapter in the *MPC5668 Reference Manual* for a detailed description of the individual bits being altered.

4.1 Pin Configuration

```
SIU.PCR[81].R = 0x60C;           /* PF1-SOUT, output, fast slew */
SIU.PCR[83].R = 0x60C;           /* PF3-PCS0, output, fast slew */
SIU.PCR[80].R = 0x60C;           /* PF0-SCK, output, fast slew */
SIU.PCR[91].R = 0x20C;           /* PF11-GPIO, output, fast slew. Drive low to reset
                                 all shift
                                 registers */
```

The first step is to configure the pins used for communication to the shift registers. This is done using the SIU.PCR registers. For the Fado board, the shift registers are connected to the DSPI_A module with the pins connected to Port F11. The slew rate for the pads should be carefully selected to ensure compatibility with the specification of the shift registers.

4.2 Initialize the DSPI Module

```
DSPI_A.MCR.B.MSTR = 0x1;           /* select master mode */
DSPI_A.MCR.B.DCONF = 0b01;         /* select DSI mode */
```

In order to operate as a phantom port, the DSPI module must be put into Deserial-Serial Interface (DSI) mode. In this mode, the module will serialize parallel input signals or bits from memory-mapped registers. The module should also be put into master mode, as the MCU will be initializing all transfers.

```
DSPI_A.MCR.B.PCSIS0 = 1;           /* set inactive CS0 state to high */
```

The Peripheral Chip Select has to be configured to “idle high” so that there is a negative edge upon the completion of transmission for compatibility with the RCK pin on the shift registers.

```
DSPI_A.DSICR.R = 0x00110001;       /* 32 bit transfers, source:SDR,
                                   transfer upon change, use CS0 */
```

The next step is to configure the DSI transfers and the communication speed. The phantom port logic makes use of the Serialization Data Register (SDR), so this should be selected as the transmit source. The CID bit should also be set to allow transmission to occur automatically as soon as the contents of the SDR changes, in other words as soon as a phantom port bit is changed by software.

```
DSPI_A.DSICR1.R = 0x1f000001;      /* set payload to 32bits, use CS0 */
```

The DSICR1 register can be used to select the number of bits to be output, in other words how many phantom output ports to write to. This can be any number between four and 32. This affects the size of the DSPI frame sent each time a port is toggled, and therefore can affect the refresh time of the phantom ports. For example, by using a 16-bit shift register and sending 16-bit frames, the time taken from start of frame transmission to change an output state would be half that of a 32-bit shift register/frame.

The selection of which Chip Select to trigger upon a transfer is controlled by the DPCS bits in the DSICR and DSICR1 registers. The DSICR is used to select which chip select pin should be triggered for the first 16-bits of the frame, the DSICR1 selects that which will be used for the remaining bits. In this instance we shall use Chip Select zero for both halves of the frame.

```
DSPI_A.CTAR[0].R = 0x79010000;     /* frame size 32bits, transfer LSB
                                   first, speed: 21.3MHz */
```

The final step is to configure the speed of the communication. As there is no speed restriction in DSPI, the maximum transfer speed is dictated by either the pad speeds on the MPC5668, or the maximum clock speed on the external shift register. For this application, the communication speed has been set to 21.3 MHz, just under the 24 MHz maximum clock speed of the shift register at 4.5 V.

4.3 Configure the Internal Logic

```
SIU.DSPIAHLA.R = 0xffffffff;       /* enable data path between SIU and
                                   DSPI_A */
```

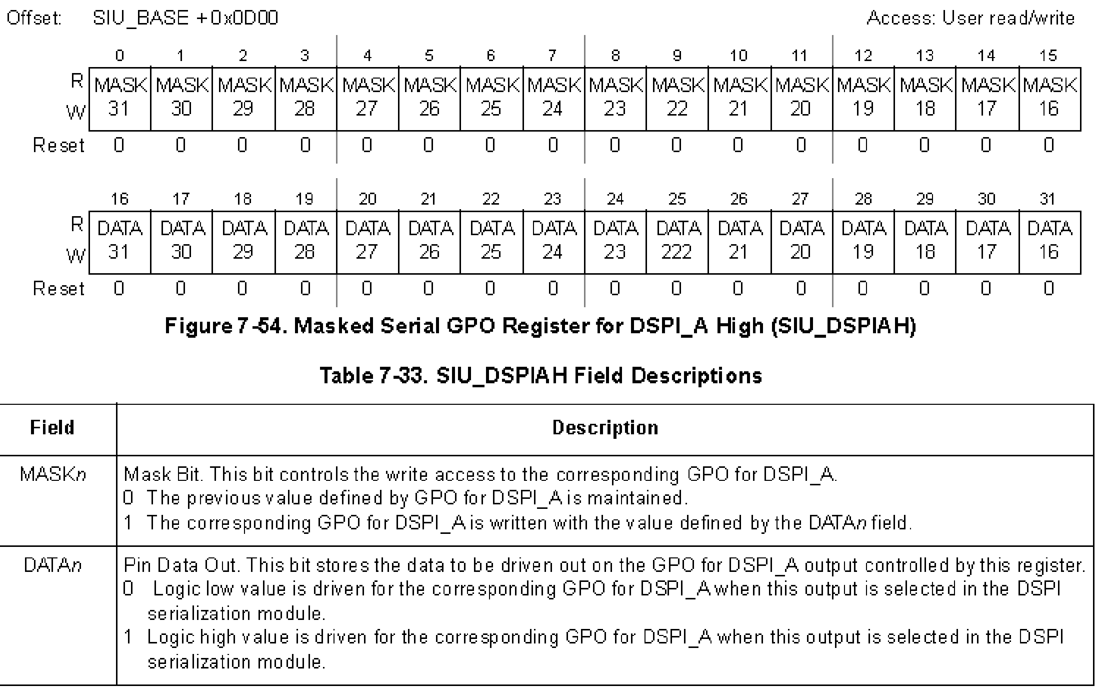
The Phantom Port function makes use of both the SIU and the DSPI module. With the DSPI now configured, the next step is to write to the Select Register for DSPI_A in the SIU (DSPIAHLA) in order to enable internal logic which ensures all 32 phantom port bits are transferred internally from the SIU to DSPI_A.

4.4 Enable DSPI Module and Set Output Values

```
DSPI_A.MCR.B.HALT = 0x0;          /* exit HALT mode in master */
DSPI_A.MCR.B.MDIS = 0x0;          /* enable module in master */
SIU.GPDO[91].R = 1;               /* put all shift registers OUT of
                                  reset */
```

Now that both the SIU and DSPI modules have been configured, the phantom ports can be enabled. This is done by enabling the DSPI, also the shift register reset pins should be tied to a pin on the MPC5668, this should also be driven high now to bring them out of reset.

The phantom port values can now be set by writing to the Masked Serial GPO Register in the SIU. The field descriptor for this register is shown in Figure 5.



Offset: SIU_BASE +0x0D00 Access: User read/write

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	MASK 31	MASK 30	MASK 29	MASK 28	MASK 27	MASK 26	MASK 25	MASK 24	MASK 23	MASK 22	MASK 21	MASK 20	MASK 19	MASK 18	MASK 17	MASK 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	DATA 31	DATA 30	DATA 29	DATA 28	DATA 27	DATA 26	DATA 25	DATA 24	DATA 23	DATA 222	DATA 21	DATA 20	DATA 19	DATA 18	DATA 17	DATA 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-54. Masked Serial GPO Register for DSPI_A High (SIU_DSPIAH)

Table 7-33. SIU_DSPIAH Field Descriptions

Field	Description
MASK*n*	Mask Bit. This bit controls the write access to the corresponding GPO for DSPI_A. 0 The previous value defined by GPO for DSPI_A is maintained. 1 The corresponding GPO for DSPI_A is written with the value defined by the DATA*n* field.
DATA*n*	Pin Data Out. This bit stores the data to be driven out on the GPO for DSPI_A output controlled by this register. 0 Logic low value is driven for the corresponding GPO for DSPI_A when this output is selected in the DSPI serialization module. 1 Logic high value is driven for the corresponding GPO for DSPI_A when this output is selected in the DSPI serialization module.

Figure 5.

As each data bit is masked, there are two such register for each Phantom Port — DSPIAH contains the upper 16 bits, DSPIAL contains the lower 16 bits. In this example, a 32-bit packet is sent whenever either of these registers is altered, thus to change the Phantom Port output from 0x0000_0000 to 0xFFFF_FFFF, two 32-bit packets will be sent, one when the upper 16 bits are altered, one when the lower.

The Phantom Port Data registers are written to as shown below:

```
SIU.DSPIAH.R = 0xFFFF5555;        /* data written to Phantom out –
                                  PH_A16-31 */
```

The above command triggers a DSPI message which sets the upper 16 bits of the phantom port to 0x5555, the lower 16 bits remain in their previous state. A scope trace of this is shown in Figure 6.

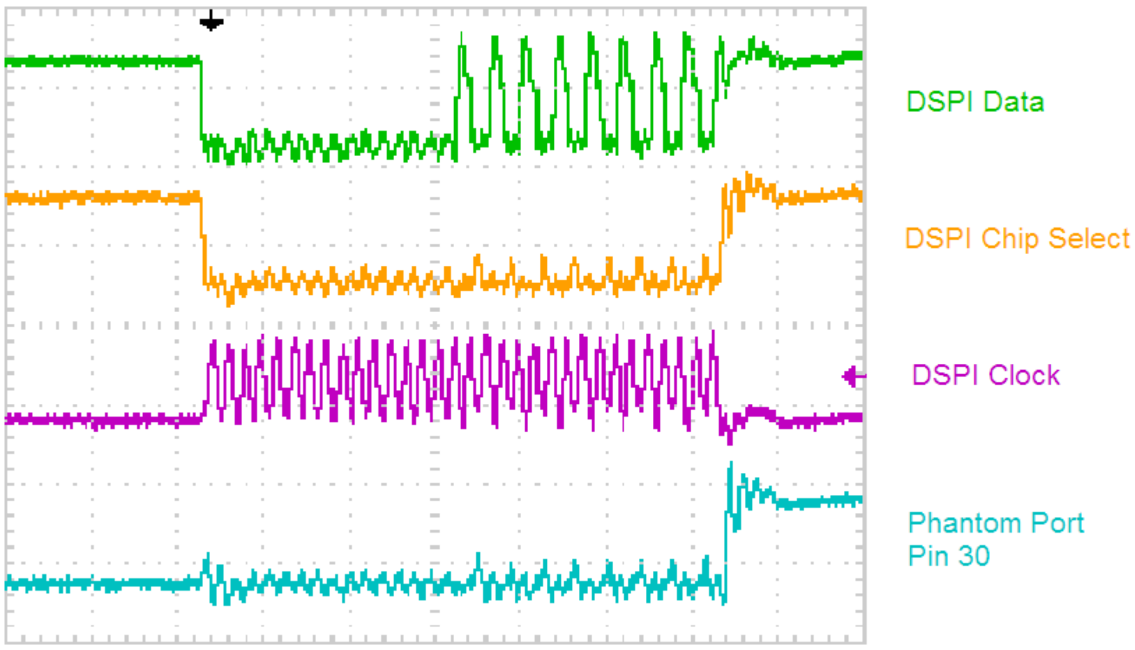


Figure 6.

The lower 16 bits of the phantom ports can be written to with the following command:

```
        SIU.DSPIAL.R = 0xFFFF0FF0;          /* data written to Phantom out –
                                            PH_A0-15 */
```

A scope trace of the resultant transmission can be seen in Figure 7.

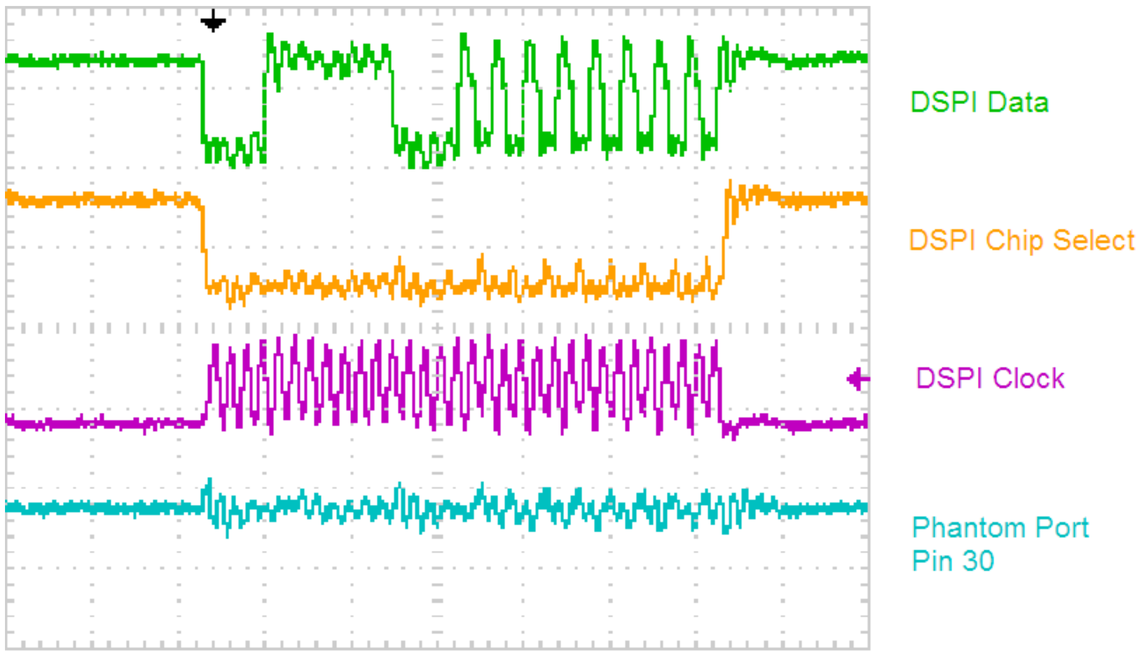


Figure 7.

Note that, although only the bottom 16 bits have been changed, all 32 bits of the phantom port are still transmitted. Also note that changing the lower 16 bits has no effect on Phantom Port Pin 30.

It is important to ensure that the phantom port is not written to before the transmission of the previous DSPI frame is complete. In this example, the DSPI runs at 21.3 MHz, thus the time required to send a 32-bit DSPI message is 1.5 µs. Therefore, the code must ensure that a period greater than this has passed before another message is sent. This gives a maximum Phantom Port output frequency of approximately 333 KHz when changing 16 bits per cycle. This can be increased by selecting a shift register which supports higher clock speeds.

5 Conclusion

This application note has shown that the number of general-purpose outputs on the MPC5668 can be greatly increased by making use of the phantom port functionality. The phantom port requirements have been discussed in depth, which should provide the user with the knowledge to design the hardware and software required to create them.

THIS PAGE IS INTENTIONALLY BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN3862
Rev. 0
05/2009