225 Cedar Hill Street • Suite 112•Marlborough, MA 01752, USA • Telephone: 508-658-8360 • www.qualtre.com

Accessing the ADC Clock on the QGYR330HD"

by Robert Shuler

Introduction

The Qualtré QGYR330HD Gyroscope has the ability to output its internal Analog to Digital Converter (ADC) Clock on its Serial Data Output (SDO) pin. Accessing the QGYR330HD ADC Clock can be useful to systems that need to synchronize clocking operations.

To drive the ADC Clock out of the SDO pin of the QGYR330HD, a Three Wire Master Serial Peripheral Interface (SPI) on host controller needs to interfaced to the Serial Data Input (SDI) pin of the QGYR330HD. In this configuration, the SDI pin of the QGYR330HD is used for both data output and data input. Once in the QGYR330HD Gyroscope is in Three SPI mode, the internal registers of the QGYT330HD also need to be accessed via the same Three Wire SPI mode.

Three Wire SPI mode is a non-standard serial communication mode of SPI and may not be supported on most Micro-controller Units (MCU) or Embedded Systems currently available. However, it is possible to implement a Three Wire SPI using General Purpose Input Output (GPIO) pins on a MCU or Embedded System via a Bit Bang technique. Bit Banging uses software to control the high and low states of a GPIO pin over a period of time. It is also possible to design a custom Three Wire SPI interface via digital logic hardware or a Field Programmable Gate Array (FPGA).

The QGYR330HD normal mode of operation for internal register access and control is via Inter-Integrated Circuit (I2C) or Four Wire SPI, so the QGYR330HD must be put into Three Wire SPI mode to allow the output of the internal ADC clock. Putting the QGYR330HD into Three Wire SPI Mode can be accomplished by writing logic high ('1') to bit 0 of register address 0x7F within the QGYR330HD.

Once the QGYR330HD is in Three Wire SPI mode, all future operations needing to access the internal QGYR330HD registers will need to use Three Wire SPI. The I2C interface of the QGYR330HD will be disabled in the Three Wire SPI mode.

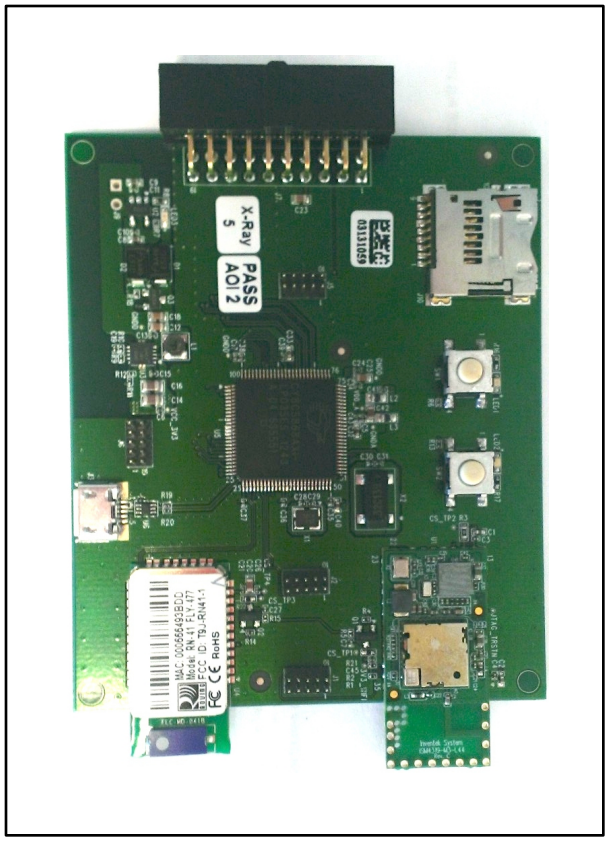

Figure 1
Qualtre Evaluation Board (CommBoard)

After QGYR330HD is in Three Wire SPI mode and if the ADC Modulator (Digital Mode) is not powered up, then the QGYR330HD needs to be put into Digital Mode by powering up the ADC Modulator before the ADC Clock can be driven to the SDO pin. The QGYR330HD can be put in Digital Mode by writing a '1' to bit 4 of register address 0x69. With the QGYR330HD in Digital Mode, writing a

0x05 (decimal value 5) to register address 0x79 will drive the ADC Clock out the SDO pin.
The Qualtre Evaluation Board (CommBoard) for the QGYR330HD gyroscope has the ability to implement a Three Wire SPI on top of its Four Wire SPI. Figure 1 shows a Picture of the CommBoard.

Three Wire SPI Mode Hardware Implementation

Figure 2 shows how the Four Wire to Three Wire Master Mode SPI is implemented via the CommBoard's internal programable logic array. The CommBoard is able to switch between Four Wire and Three Wire SPI modes on the fly via writes to internal control registers within the CommBoard. The CommBoard is based on the Cypress PSoC 5, System on Chip, ARM MCU with internal Universal Digital Blocks (UDB) that can be used to create any number of digital logic functions.

Reviewing Figure 2, we see a standard Master SPI interface block with Serial Data Input (SDI) pin, Serial Data Output (SDO) pin, an input multiplexer (MUX) on the Master In Slave Out (MISO) input pin, and a tristate buffer on the Master Out Slave In (MOSI) Output pin. Also, shown is the Master Mode SPI Chip Select (CS) signal that enables a slave device for access.

Along with the Master SPI controller implemented in the CommBoard's PSoC5 UDBs there are two control registers spiModeCr and spiTxEnableCr. These registers allow software control for the switching between Four Wire and Three Wire SPI Modes. The spiModeCr register sets the Master SPI mode to Four Wire or Three Wire. The spiTxEnableCr register is used to switch the SDI pin of the CommBoard's Master SPI from input to output for Three Wire receive or transmit.

Three Wire SPI Mode Software Implementation

Listing 1 shows a 'C' code segment which lists the steps necessary to enable or disable the ADC Clock Out on the SDO pin of the QGYR330HD using the CommBoard's Master SPI Four Wire to Three Wire Interface.

As shown in Listing 1, there are a simple number of software instructions or steps needed to enable or

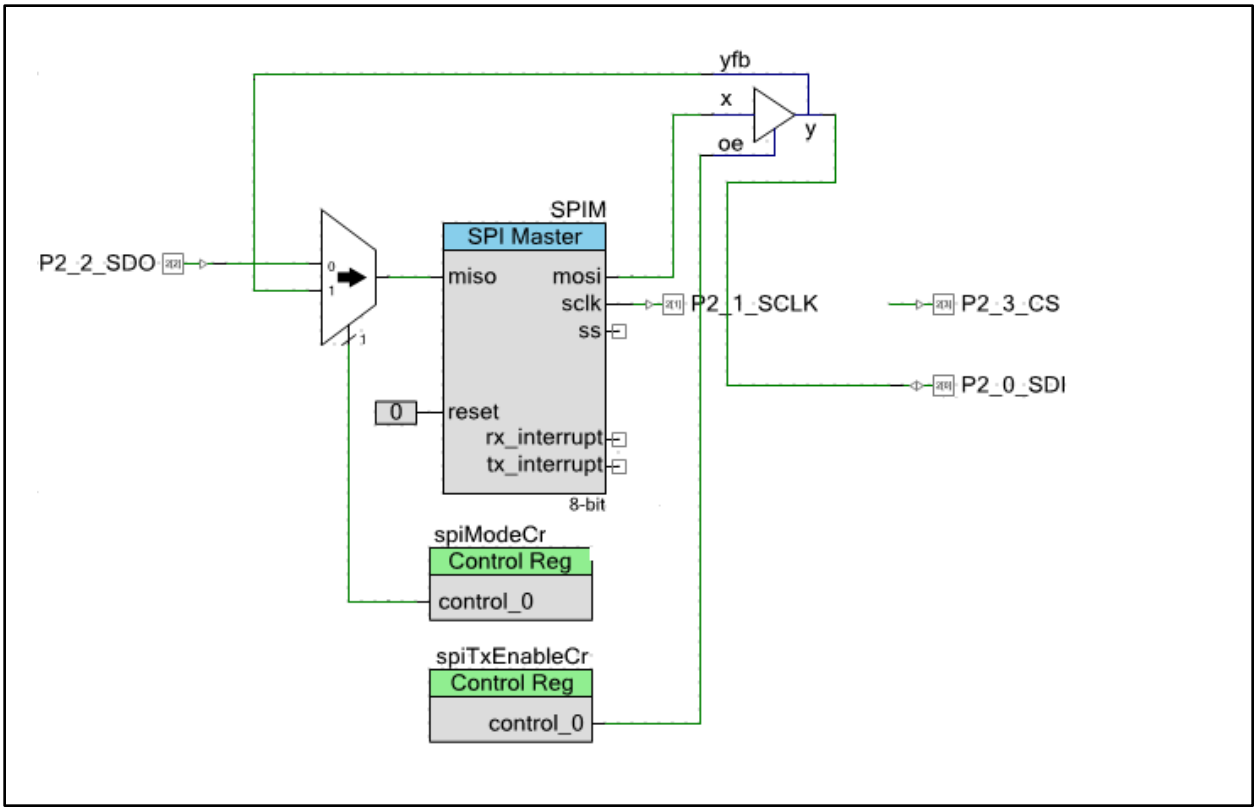

Figure 2
CommBoard 4-Wire/3-Wire SPI interface

disable ADC Clock Out of on the SDO pin. These steps are summarized below.

Enable ADC Clock Out to SDO pin:
1) Enable Three Wire SPI mode on the QGYR330HD by writing a 1 to bit 0 of register 0x7F.
2) Enable Digital Mode by writing a 1 to bit 4 of register 0x69
3) Drive the ADC Clock out by writing a 0x05 to register 0x79.

Disable ADC Clock Out to the SDO pin:
1) Enable Four Wire SPI mode on the QGYR330HD by writing a 0 to bit 0 of register 0x7F.
2) Enable Digital Mode by writing a 0 to bit 4 of register 0x69. This step is optional if part needs to be left in Digital Mode.
3) Stop driving the ADC Clock out by writing a 0x00 to register 0x79.

```
// Enable ADC Clock Out
 if (ADC_CLK_OUT_ENABLE)
 {
     commMode = SPI_3WIRE;
     i2cOrSpiMode(commMode);
     sampleMode = DIGITAL;
     spiWrite(0x7F, SPI_3WIRE, SPI_4WIRE);

     temp = spiRead(0x69, SPI_3WIRE);
     spiWrite(0x69, temp | 0x10, SPI_3WIRE);

     spiWrite(0x79, 0x05, SPI_3WIRE);
     temp = spiRead(0x79, SPI_3WIRE);
     commMode = SPI_3WIRE;
 }
 else  //Disable ADC Clock Out
 {
     sampleMode = DIGITAL;
     spiWrite(0x7F, SPI_4WIRE, SPI_4WIRE);

     temp = spiRead(0x69, SPI_4WIRE);
     spiWrite(0x69, temp & ~0x10, SPI_4WIRE);

     spiWrite(0x79, 0x00, SPI_4WIRE);
     temp = spiRead(0x79, SPI_4WIRE);
     commMode = SPI_4WIRE;
 }
```

Listing 1
How to Enable and Disable ADC Clock Out

The spiWrite and spiRead instructions shown in Listings 1 are built on top of the lower level commands for controlling the Master SPI interface of the Cypress PSoC5. Listing 2 and Listing 3 detail the lower level functions - spiWire and spiRead.

```
// write to device
void spiWrite(uint8 address, uint8 data, uint8 spiMode)
 {
    if (spiMode == SPI_4WIRE)
    {
       // enable 4 wire spi
       spiModeCr_Write(SPI_4WIRE);
    }
    else
    {
       // enable 3 wire spi
       spiModeCr_Write(SPI_3WIRE);
    }

    spiTxEnableCr_Write(ENABLE);  // enable output
     SPIM_ClearTxBuffer();
     SPIM_ClearFIFO();
     P2_3_CS_Write(LOW);
     SPIM_WriteTxData(address & SPI_WRITE_REG);
     while (!(SPIM_ReadTxStatus() &
(SPIM_STS_SPI_DONE)));

      SPIM_WriteTxData(data);
      while (!(SPIM_ReadTxStatus() &
(SPIM_STS_SPI_DONE)));

       P2_3_CS_Write(HIGH);
 }
```

Listing 2
spiWrite function

```
// read from device
  uint8 spiRead(uint8 address, uint8 spiMode)
 {
    if (spiMode == SPI_4WIRE)
    {
       // enable 4 wire spi
       spiModeCr_Write(SPI_4WIRE);
    }
    else
    {
       // enable 3 wire
       spiModeCr_Write(SPI_3WIRE);
    }
```

Listing 3
spiRead function

```
    // enable output
    spiTxEnableCr_Write(ENABLE);

    SPIM_ClearTxBuffer();
    SPIM_ClearFIFO();

    P2_3_CS_Write(LOW);
    SPIM_WriteTxData(address |
SPI_READ_REG);
    while (!(SPIM_ReadTxStatus() &
(SPIM_STS_SPI_DONE)));

   if (spiMode == SPI_3WIRE)
   {
       // enable output
       spiTxEnableCr_Write(DISABLE);
   }

    SPIM_ClearTxBuffer();
    SPIM_ClearFIFO();

    SPIM_WriteTxData(0x00);
    while (!(SPIM_ReadTxStatus() &
           (SPIM_STS_SPI_DONE)));
    P2_3_CS_Write(HIGH);

    return SPIM_ReadRxData();
 }
```

Listing 3 Continued
spiRead function

TABLE OF CONTENTS

AN1001-8/14(1.1)

www.qualtre.com