SPI ROM interface to McBSP

Contributed by Shaku Anjanaiah
04/14/98

Design Problem

How do I interface the Serial Peripheral Interface (SPI) ROM to the ‘C6201?

Solution

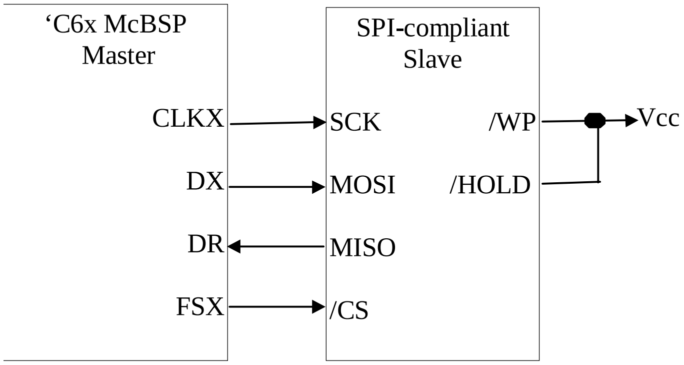
The multi-channel buffered serial port (McBSP) in the ‘C6201 can be interfaced to a SPI ROM with no glue logic. A SPI system is typically a 4-wire interface comprising serial data in, serial data out, serial clock, and device select. The McBSP provides this 4-wire interface via DR, DX, CLKX, and FSX pins respectively.

The SPI interface is supported in the ‘C6201 McBSP for a synchronous, full-duplex, variable element length (element length is fixed for a given transfer), master or slave mode back-to-back transmission and reception. This is achieved by using the clock-stop (CLKSTP) feature of the McBSP. This document uses the Atmel™ SPI serial CMOS EEPROM, which can only be a slave. Also, McBSP is typically used as a master in a SPI system. The McBSP as a SPI master is responsible to generate the required control signals and clocking to the slave.

Pin Configuration

In order that the McBSP behave as a master, it is necessary to configure the bi-directional (except data pins) serial port pins as outputs only. Hence, CLKX and FSX pins have to be outputs only. CLKX can be generated either via the ‘C6201 CPU clock or by the CLKS pin which serves as an external clock source. In SPI mode, a SPI system clock or any other clock source can drive CLKS if present. The clock divide down can be programmed as per application needs.

Figure 1. McBSP Master interface to SPI slave device



The signal connectivity shown in Figure 1 is for connecting a Atmel™ SPI serial CMOS EEPROM AT25 series which has a maximum clock rate of 2.1 MHz for Vcc range from 2.7V to 5.5V. This slave device is organized as 1k/2k/4k/8k of 8-bit data and only supports SPI modes 0 and 3.

As shown in Figure 1, CLKSTP scheme supports back-to-back transmission and reception utilizing signals that correspond to the transmitter. But the McBSP also simultaneously receives data by utilizing the CLKX and FSX outputs as CLKR and FSR signals internally. As a good practice, CLKR and FSR should be programmed as inputs.

McBSP Initialization

The various McBSP control registers shown in Figures 2 through 6 have to be initialized for SPI operation. The serial port initialization procedure for SPI mode is as follows:

1. If McBSP is not in reset state, set /XRST = /RRST = 0 in SPCR.
2. Now program the McBSP configuration registers XCR, RCR, SRGR, PCR, and SPCR for all parameters as required *except* for CLKSTP bits in SPCR, which should be 0Xb.
3. Set /GRST=1 in SPCR to get the sample rate generator out of reset.
4. Wait two CLKG clocks for the McBSP to re-initialize.
5. Write the desired value into the CLKSTP bit-fields in the SPCR. Figure 7 shows the various CLKSTP modes that are supported by the McBSP.
6. Either (a) or (b) should be followed.

(a) This step should be performed if the CPU is used to service the McBSP. Set /XRST = /RRST = 1 to enable the serial port. Note that the value written to the SPCR at this time should have only the reset bits changed to 1 and the remaining bit-fields should have the same value as in Step 2 and 4 above.

(b) If DMA is used to perform data transfers, the DMA should be initialized first with the appropriate read/write syncs and the start bit set to run. The DMA waits for the synchronization events to occur. Now, pull the McBSP out of reset.

Figure 2. Receive Control Register (RCR for SPI Master)

31	30 24	23 21	20 19	18	17 16
0	0	0	0	0	01
RPHASE	RFRLEN2	RWDLEN2	RCOMPAND	RFIG	RDATDLY

15	14 8	7 5	4 0
0	0	0	0
reserved	RFRLEN1	RWDLEN1	reserved

Figure 3. Transmit Control Register (XCR for SPI Master)

31	30 24	23 21	20 19	18	17 16
0	0	0	0	0	01
XPHASE	XFRLEN2	XWDLEN2	XCOMPAND	XFIG	XDATDLY

15	14 8	7 5	4 0
0	0	0	0
reserved	XFRLEN1	XWDLEN1	reserved

Figure 4. Sample Rate Generator Register (SRGR for SPI Master)

31	30	29	28	27 26 25 24 23 22 21 20 19 18 17 16
0	0	1	0	0
GSYNC	CLKSP	CLKSM	FSGM	FPER

15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
0	0x5F
FWID	CLKGDV

Figure 5. Pin Control Register (PCR for SPI Master)

31 16
0x0000
reserved

15 14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	1	0	0	0	0	0	1	1	0	0
rsv	XIOEN	RIOEN	FSXM	FSRM	CLKXM	CLKRM	rsv	CLKS_STAT	DX_STAT	DR_STAT	FSXP	FSRP	CLKXP	CLKRP

Figure 6. Serial Port Control Register (SPCR for SPI Master)

31 24	23	22	21 20	19	18	17	16
0x00	0	1	0	0	0	0	0
R, +0	FRST-	GRST-	XINTM	XSYNCERR	XEMPTY-	XRDY	XRST-

15	14 13	12 11	10 8	7	6	5 4	3	2	1	0
0	0	11	0	0	0	0	0	0	0	0
DLB	RJUST	CLKSTP	reserved	reserved	reserved	RINTM	RSYNCERR	RFULL	RRDY	RRST-

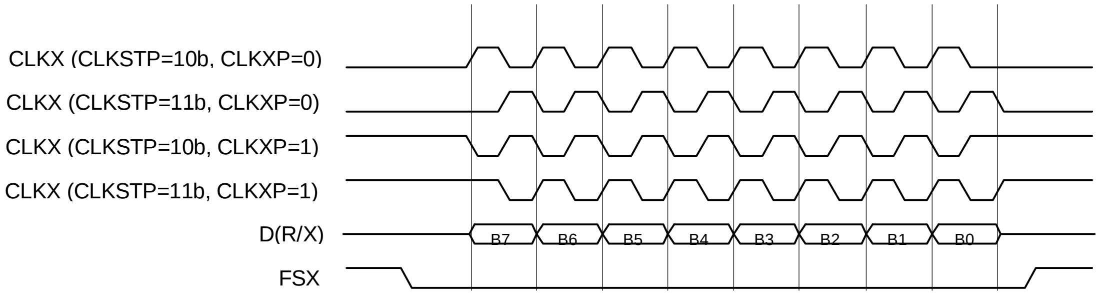
Table 1. McBSP Register values for 200MHz CPU clock

Register	**Value in hex**	**Description**
RCR	0x00010000	single phase, one 8-bit element per frame, one bit-clock delay
XCR	0x00010000	single phase, one 8-bit element per frame, one bit-clock delay
SRGR	0x0200005F	serial clock CLKX generated by CPU clock (CLKSM=1), frame sync FSX generated due to DXR-to-XSR transfer (FSGM=0), clock divide down is 95 for 200 MHz clock to generate 2.1 MHz shift clock (CLKGDV=0x5F)
PCR	0x00000A0C	FSX is an active low (FSXP=1) output (FSXM=1), FSR is an active low (FSRP=1) input (FSRM=0), CLKX is an output (CLKXM=1) and starts with a rising edge (CLKXP=0)
SPCR[12:10]	0x3	CLKSTP=11b. Since CLKXP=0, this refers to data transmitted on rising edge and received on falling edge of CLKX by the master. This parameter can be changed as per application needs.

The following macros initialize the McBSP in the correct order for SPI mode communication between the McBSP and a SPI serial EEPROM. These macros (except your_*) can be found in the TMS320C6x Peripheral Support Library.

```
#include <regs.h>
#include <mcbsp.h>
/* Refer Table 1 for a description of the register initialization
values */
REG_WRITE (MCBSP_RCR_ADDR(0), 0x00010000);
REG_WRITE (MCBSP_XCR_ADDR(0), 0x00010000);
REG_WRITE (MCBSP_SRGR_ADDR(0), 0x0200005F);
REG_WRITE (MCBSP_PCR_ADDR(0), 0x00000A0C);
MCBSP_SAMPLE_RATE_ENABLE (0); /* set /GRST=1 */
your_wait (2 bitclocks); /* wait for 2 bit-clocks */
LOAD_FIELD  (MCBSP_SPCR_ADDR(0),  3,  CLKSTP,  CLKSTP_SZ);  /*  Set
CLKSTP=11b */
your_wait (2 bitclocks); /* wait for 2 bit-clocks */
your_dma_setup(); /* Set up DMA for data acquisition if required.
See Step 5 */
MCBSP_ENABLE (0, 3); /* enable transmit and receive side of
McBSP0 */
```

Figure 7. Clock Stop Mode Options


CLKX (CLKSTP=10b, CLKXP=0)
CLKX (CLKSTP=11b, CLKXP=0)
CLKX (CLKSTP=10b, CLKXP=1)
CLKX (CLKSTP=11b, CLKXP=1)
D(R/X)
FSX
B7
B6
B5
B4
B3
B2
B1
B0

Conclusion

The TMS320C6201 can be interfaced to any SPI-type device without requiring any glue logic. If the McBSP is used as a slave, please ensure that the internal clock, CLKG runs at least eight times that of the master clock. Typically, programming CLKGDV=1 and using CPU clock (CLKSM=1) (when McBSP is a SPI slave) should suffice since SPI clocks are very slow.