ADC AND DAC INTERFACING WITH PT 51 USING SPI COMMUNICATION

Introduction:

In this experiment we are using ADC – MCP 3008 and DAC – MCP4921. The ADC can be given single ended or differential analog input and it produces 10 bit digital output. This 10 bit digital output is available on the Dout pin of the ADC serially. We are interfacing this with Pt 51 microcontroller . The microcontroller takes this 10 bit input and modifies data appropriately and provides it to 12 bit DAC . Here we are using DAC in which 12 bit digital input is given serially through microcontroller and corresponding analog output can be obtained. We will carry out the entire experiment in various steps:

Step 1: DAC interfacing with Pt 51

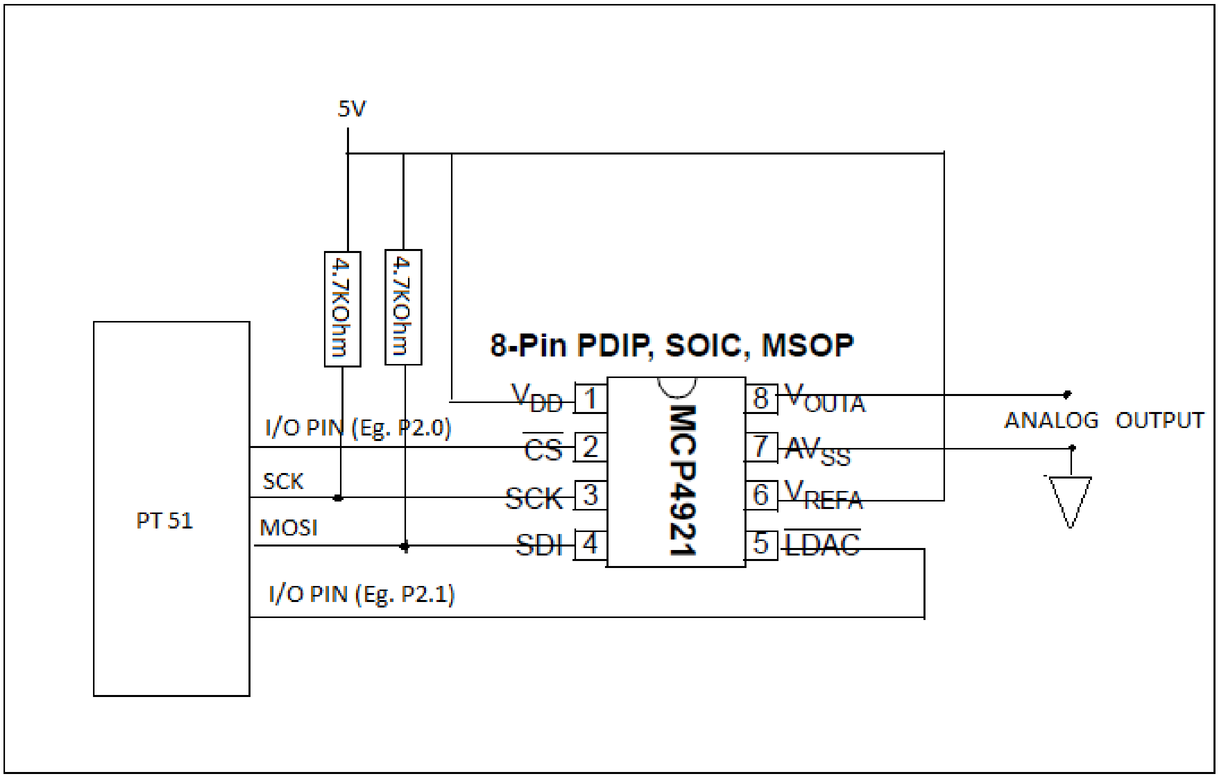
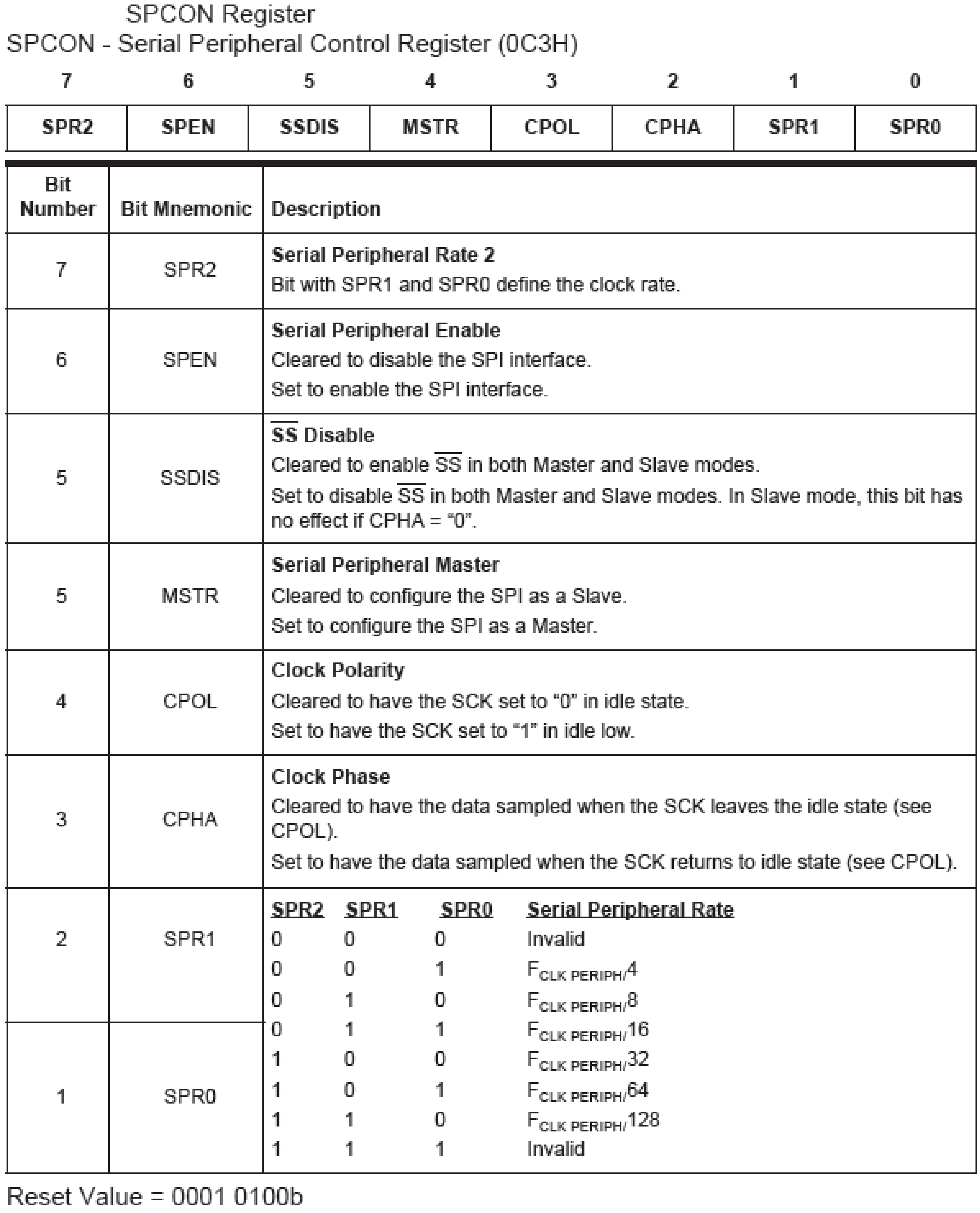


FIGURE 1: DAC CONNECTION DIAGRAM

For the SPI communication between Pt 51 and DAC , we use Pt51 as master and DAC as slave. The connections are done as shown in figure 1 . Inorder to make Pt51 master we need to configure SPCON(serial peripheral control) register.



SPCON Register

SPCON - Serial Peripheral Control Register (0C3H)

7	6	5	4	3	2	1	0
SPR2	**SPEN**	**SSDIS**	**MSTR**	**CPOL**	**CPHA**	**SPR1**	**SPR0**

Bit Number	Bit Mnemonic	Description
7	SPR2	**Serial Peripheral Rate 2** Bit with SPR1 and SPR0 define the clock rate.
6	SPEN	**Serial Peripheral Enable** Cleared to disable the SPI interface. Set to enable the SPI interface.
5	SSDIS	**$\overline{SS}$ Disable** Cleared to enable $\overline{SS}$ in both Master and Slave modes. Set to disable $\overline{SS}$ in both Master and Slave modes. In Slave mode, this bit has no effect if CPHA = “0”.
5	MSTR	**Serial Peripheral Master** Cleared to configure the SPI as a Slave. Set to configure the SPI as a Master.
4	CPOL	**Clock Polarity** Cleared to have the SCK set to “0” in idle state. Set to have the SCK set to “1” in idle low.
3	CPHA	**Clock Phase** Cleared to have the data sampled when the SCK leaves the idle state (see CPOL). Set to have the data sampled when the SCK returns to idle state (see CPOL).
2	SPR1	SPR2 SPR1 SPR0 Serial Peripheral Rate 0 0 0 Invalid 0 0 1 $F_{CLK\ PERIPH}/4$ 0 1 0 $F_{CLK\ PERIPH}/8$ 0 1 1 $F_{CLK\ PERIPH}/16$ 1 0 0 $F_{CLK\ PERIPH}/32$ 1 0 1 $F_{CLK\ PERIPH}/64$ 1 1 0 $F_{CLK\ PERIPH}/128$ 1 1 1 Invalid
1	SPR0	

Reset Value = 0001 0100b

Steps to configure SPCON register:

-- Frees the SS pin for a general-purpose
-- Selects one of the Master clock rates
– Configure the SPI module as Master (in this case)
– Selects serial clock polarity and phase(1,1) or (0,0)
– Enable the SPI module(SPEN = 1)

The digital data is transferred (MSB first) from Pt51 serially 8 bits at a time . The data to be transferred through the MOSI pin is given to SPDAT register. The successful transmission of 8 bits is indicated by SPIF flag in SPSTA register. The bitwise configuration of these registers is given on next page.

Steps for DAC interfacing:

1. Configure SPCON register. Here we are using CPHA =1, CPOL=1 ; so data new data bit is send through MOSI pin at falling edge of SCK and data can be sampling in PT51 at rising edge of SCK
2. Make LDAC pin for DAC always logic 0 so that analog signal will appear at the output as soon as CS pin is made logic 1.
3. Enable SPI communication through microcontroller(SPEN= 1) .
4. To select DAC make its CS pin logic 0.
5. Once DAC is selected new data bit is transferred from Pt51 at falling edge which gets sampled in DAC at next rising edge of SCK.
6. The 8 bit successful data transfer can be verified by checking SPIF flag.
7. The clocking diagram for (1,1) mode is shown in figure 2. For other modes refer data sheet.
8. We need to transfer 16 bits in total. First 4 bits are to configure DAC and next 12 bits for data as shown in figure 3 .(refer to MCP4921 data sheet)
9. After successful transmission of 16 bits make CS = 1.

SPSTA - Serial Peripheral Status and Control register (0C4H)

7	6	5	4	3	2	1	0
SPIF	**WCOL**	**SSERR**	**MODF**	-	-	-	-

Bit Number	Bit Mnemonic	Description
7	SPIF	**Serial Peripheral data transfer flag** Cleared by hardware to indicate data transfer is in progress or has been approved by a clearing sequence. Set by hardware to indicate that the data transfer has been completed.
6	WCOL	**Write Collision flag** Cleared by hardware to indicate that no collision has occurred or has been approved by a clearing sequence. Set by hardware to indicate that a collision has been detected.
5	SSERR	**Synchronous Serial Slave Error flag** Set by hardware when $\overline{SS}$ is de- asserted before the end of a received data. Cleared by disabling the SPI (clearing SPEN bit in SPCON).
4	MODF	**Mode Fault** Cleared by hardware to indicate that the $\overline{SS}$ pin is at appropriate logic level, or has been approved by a clearing sequence. Set by hardware to indicate that the $\overline{SS}$ pin is at inappropriate logic level.
3	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit
2	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit
1	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
0	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.

Reset Value = 00X0 XXXXb
Not Bit addressable

SPDAT - Serial Peripheral Data Register (0C5H)

7	6	5	4	3	2	1	0
R7	**R6**	**R5**	**R4**	**R3**	**R2**	**R1**	**R0**

Reset Value = Indeterminate

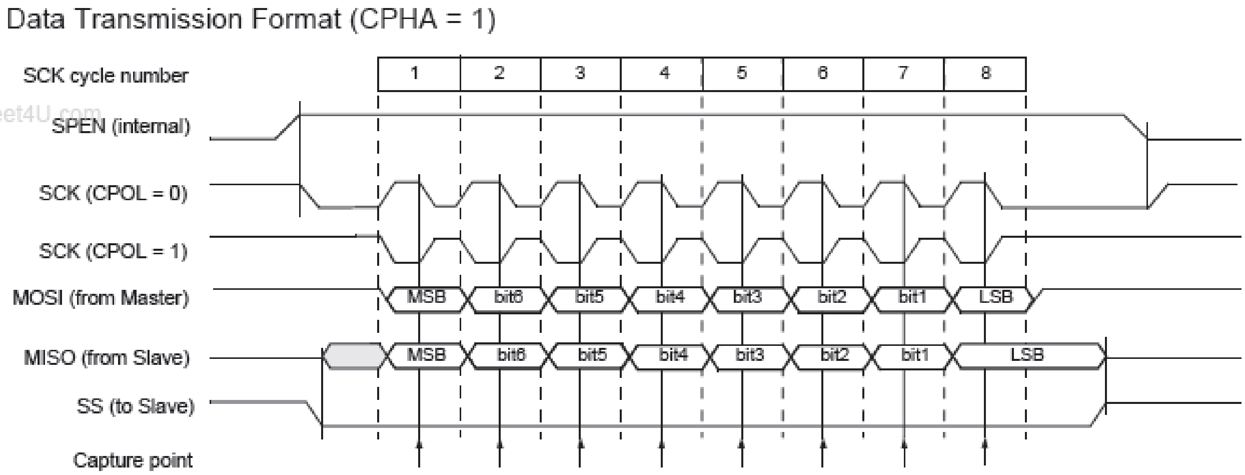
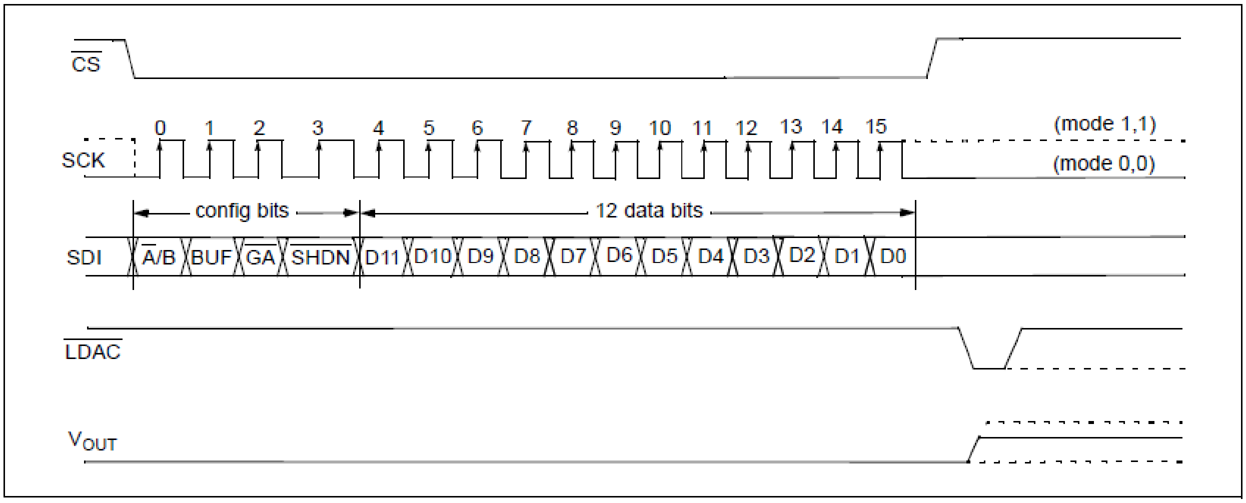


Figure 3a: Pt51 clock diagram



bit 15 **$\overline{A}$/B:** DAC$_A$ or DAC$_B$ Select bit

1 = Write to DAC$_B$

0 = Write to DAC$_A$

bit 14 **BUF:** V$_{REF}$ Input Buffer Control bit

1 = Buffered

0 = Unbuffered

bit 13 **$\overline{GA}$:** Output Gain Select bit

1 = 1x (V$_{OUT}$ = V$_{REF}$ * D/4096)

0 = 2x (V$_{OUT}$ = 2 * V$_{REF}$ * D/4096)

bit 12 **$\overline{SHDN}$:** Output Power Down Control bit

1 = Output Power Down Control bit

0 = Output buffer disabled, Output is high impedance

bit 11-0 **D11:D0:** DAC Data bits

12 bit number "D" which sets the output value. Contains a value between 0 and 4095.

Figure 3b:write command DAC

Step 2:ADC interfacing with Pt51

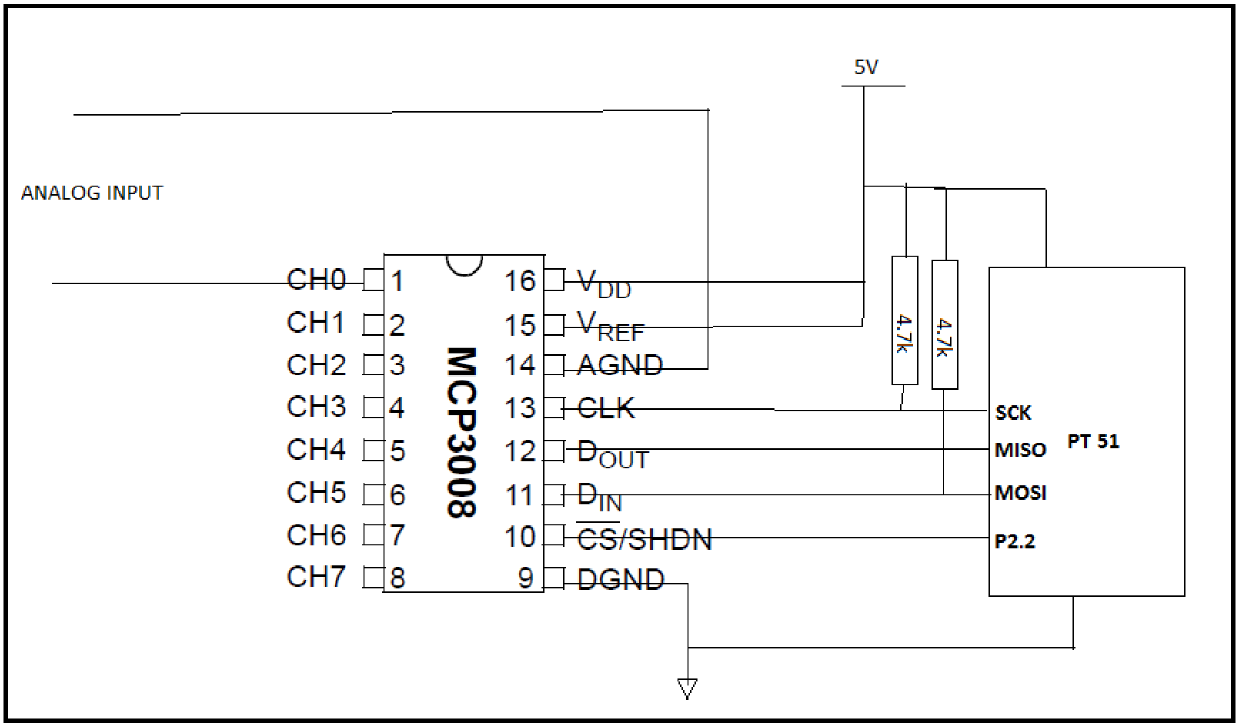


Figure 4 : connection diagram

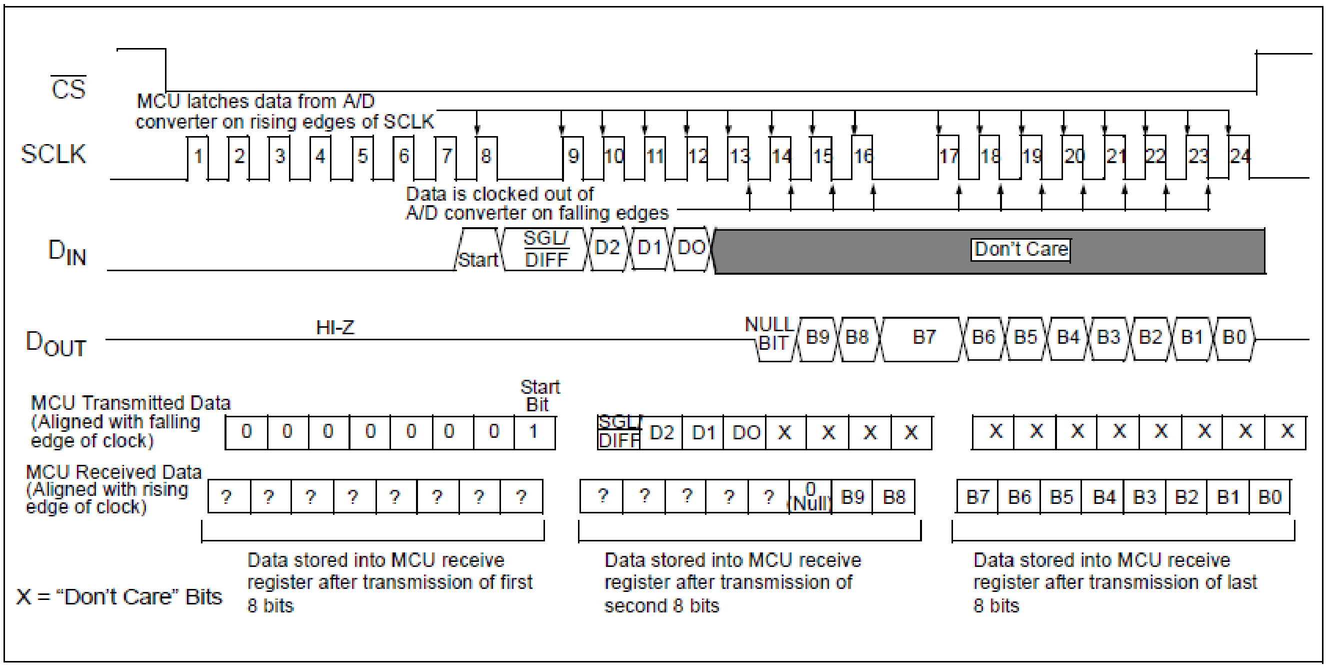

$\overline{CS}$
SCLK
MCU latches data from A/D converter on rising edges of SCLK
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
Data is clocked out of A/D converter on falling edges
D_{IN}
Start
SGL/ DIFF
D2
D1
DO
Don't Care
D_{OUT}
HI-Z
NULL BIT
B9
B8
B7
B6
B5
B4
B3
B2
B1
B0
MCU Transmitted Data (Aligned with falling edge of clock)
Start Bit
0 0 0 0 0 0 0 1
SGL/ DIFF D2 D1 DO X X X X
X X X X X X X X
MCU Received Data (Aligned with rising edge of clock)
? ? ? ? ? ? ? ?
? ? ? ? ? 0 (Null) B9 B8
B7 B6 B5 B4 B3 B2 B1 B0
Data stored into MCU receive register after transmission of first 8 bits
Data stored into MCU receive register after transmission of second 8 bits
Data stored into MCU receive register after transmission of last 8 bits
X = "Don't Care" Bits

Figure 5 : Timing diagram

Steps for ADC interfacing:

1. Configure the SPCON register in the same way as in the case of DAC.
2. Enable SPI communication.
3. Make CS pin of ADC low to select it.
4. As shown in timing diagram send logic high to Din of ADC to indicate start of conversion , followed by single/differential mode selection ,followed by channel selection for ADC input.
5. Thus the corresponding digital data is obtained at Dout pin null bit followed by 10 bits(MSB first). Refer datasheet MCP3008.
6. After the successful reception of 10 bit of data CS pin of ADC is made high.
7. The data bits transmission and reception from microcontroller should be done as per the alignment shown in timing diagram.

Step 3: GIVING ANALOG SIGNAL TO ADC AND REPRODUCING CORRESPONDING SIGNAL THROUGH DAC USING PT51 MICROCONTROLLER

The analog signals can be easily reproduced by integrating Step 1 & 2.

IMPORTANT FORMULAE FOR VOLTAGE CONVERSION CALCULATION:

ADC:

$$LSB\ Size = \frac{V_{REF}}{1024}$$

$$Digital\ Output\ Code = \frac{1024 \times V_{IN}}{V_{REF}}$$

Where:

V_{IN} = analog input voltage

V_{REF} = analog input voltage

DAC:

$$V_{OUT} = \frac{V_{REF} G D_N}{2^n}$$

Here n = 12 D_N = digital 12 bit number value & G = gain (to be set to 1)