Lab-7: A to D and D to A conversion

In the last lab, we had used asynchronous serial communication which used extra bits per frame for synchronization. This is not necessary in synchronous communication as the communicating devices use the same clock. SPI and I2C are two popular synchronous serial protocols. These protocols are widely used because these allow one to connect peripherals to a micro-computer without using up too many port pins and to send and receive data at fairly high speeds.

The objective of this lab is to learn data transmission using a synchronous protocol, as also to learn about Digital to Analog Converters (DACs) and Analog to Digital Converters (ADCs). We shall use the 10 bit ADC MCP3008 and the 12 bit DAC MCP4921 from Microchip. Both of these are connected to the micro-controller through SPI interfaces. Data sheets for these chips and a mini-tutorial on SPI were uploaded to moodle last week. Instructions for configuring the micro-controller and for connecting these chips to it are also being uploaded as a pdf document prepared by Zeal Sheth. An excerpt of Zeal's document is attached here for quick review. Do read the full document before attempting your homework or labwork problems.

We shall use this lab using assembly programming. (You are encouraged to try it in C afterwards and see how close you can get to the speed the assembly program using C).

Homework

1. Write the code for configuring the peripheral clock and the SPI interface of Pt51 as the master device.
2. Write the code for repeatedly reading a 10 bit word from the ADC, formatting it according to the requirements of the 12 bit DAC and writing it to the DAC through the SPI interface.

When you show this software to your TA in the lab, use a variable DC value as the input and check the output of the DAC.

Labwork

In the lab, we shall set up a system which samples a given waveform using the ADC and then recreates it using the DAC.

1. Connect just the DAC chip to Pt51 using the SPI interface. Write a program which will generate a stair-step approximation to a triangular wave of 1 KHz. Your program should use at least 8 steps on the rising as well as on the falling ramp of the triangular wave.

2. Now connect the ADC chip also to Pt. 51. Write a program which will sample the input connected to the ADC and will write it out to the DAC in an endless loop. (Re-use the code you wrote as homework). Connect a waveform from the signal generator to the ADC and observe the output from the DAC on a scope.

IMPORTANT: Make sure you do not damage the ADC by applying a voltage which lies outside the permissible range for the ADC as you have connected it. For example, the sine wave output from a signal generator will be bi-polar. You need to shift it to make it unipolar.

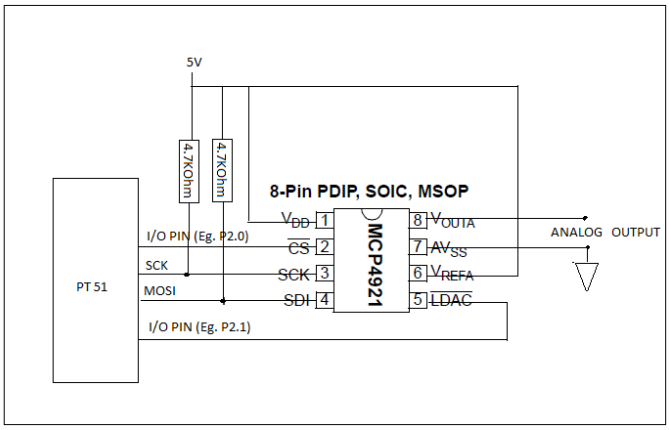
Show the output of the signal generator on the scope to your TA before connecting it to the ADC.

Report the highest frequency sine wave that you are able to re-create, using at least 8 points per cycle.

ADC AND DAC INTERFACING WITH PT 51 USING SPI COMMUNICATION

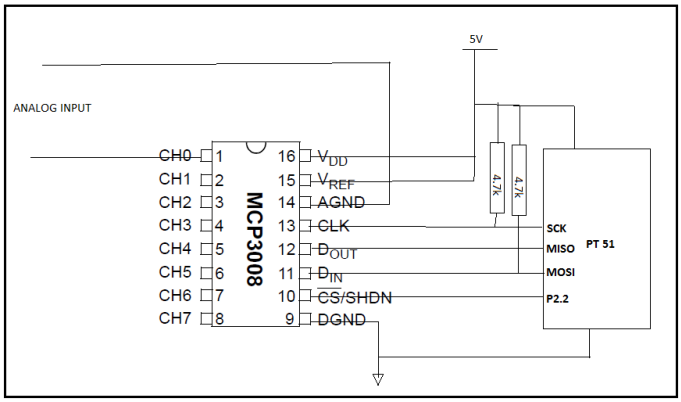
From the document written by Zeal Sheth
diagrams are from the data sheets of 89C51, MCP 4921 and MCP 3008

1 DAC interfacing with Pt 51



The figure on the left shows the connection between Pt51 and the DAC. Pt51 acts as the master, while the DAC acts as the slave. We need to enable the DAC and write to SPDAT register to send data to DAC.

2 ADC interfacing with Pt 51



This figure shows the connection between Pt51 and the ADC. Pt51 acts as the master, while the ADC acts as the slave. We need to do a write operation indicating start of conversion to SPDAT in order to read the ADC. See the document from Zeal for details.

The operation of the SPI interface in 89C5131 is described in the data sheet which was uploaded on moodle at the beginning of this course. Description of SPI begins from page 93 in this document.

Functioning of the SPI is configured using several registers which lie in the SFR area. SPCON is the SFR which controls the operation of SPI interface. SPDAT is the data register. The status of data transfer is contained in SPSTA.

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

To configure the SPI, we need to set up bits in this register. We need to enable SPI (SPEN = 1), configure Pt51 as the SPI master and select the serial clock polarity and phase in which data will be read/written. When configured as the master, the slave select pin (SS) is not required and can be freed for general IO.

You will have to use two port pins to provide slave select signals to the DAC and to the ADC. Make sure the correct peripheral is enabled when sending/receiving data.

Please refer to the document by Zeal Sheth for details.