No	Rev	Date	Repo		Page
0002	A	2011-09-07	MSPS		1 of 16
Title	High-Level System Design				
File	MSPS_0002_LM_matlabSystem_A.odt				
Type	EX -- Laboratory Manual 2, MSPS		Area	ES : docs : courses : msps	
Created	Per Löwenborg		Approved	J Jacob Wikner	
Issued	J Jacob Wikner, jacwi50		Class	Public	

Laboratory Manual 2, MSPS

High-Level System Design

Description

The purpose of this laboratory exercise is mainly to provide the student with hands-on experience of the MATLAB design tool used in the TSTE16 course: Mixed-Signal Processing Systems.

The laboratory also gives an illustration of top-down design methodology, which is a very important aspect of this course.

Outline

History

Rev	Date	Comment	Created/Issued by
1.0	1973-07-13	Creation. New document template. Changed the descriptions.	Per Löwenborg
2.0	2009-09-18	Updated the system figure and changed the target specification.	Per Löwenborg
3	2009-09-24	Changed the SNR/SNDR margins in table 8.1.	J Jacob Wikner
PA4	2010-10-01	Updated for class 2010.	J Jacob Wikner
PA5	2011-09-07	Touch-up for class 2011. Changed name of document too.	J Jacob Wikner
A	2011-09-07	Released for class of 2011.	J Jacob Wikner

1. OVERVIEW / INTRODUCTION

This is the manual of the second laboratory work in the course Mixed-Signal Processing Systems. Before starting the laboratory work, please read this manual as well as Chapter 1 in the course book. It is also recommended to look through Chapters 2 - 5 in the same book.

- *This lab is performed in groups of 2 or 3 students.*
- *Please also read through the material before the lab and try to answer as many questions as possible on before-hand.*

One copy of your manual, containing answers, must be handed in to Dr. Per Löwenborg, so that you can be approved on the course. Please fill in the table below with your contact details.

Names:	**Email addresses:**

In this laboratory, the functional behaviour of the system shown in Figure 1.1 is studied. The system is one segment of a mixed-signal part typically found in the physical layer of a communication system implementation. The upper part, i.e., oversampled digital-to-analogue conversion is typically situated before the radio frequency (RF) electronics on the transmitter side of a digital radio system. The lower part, on the other hand, oversampled analogue-to-digital conversion is typically used to digitize the received signals after the radio frequency electronics in a digital radio receiver. To the right in the figure, the radio frequency electronics, antennas, and channel, etc., are situated. To the left, the digital channel, source coding/decoding, synchronization, equalization, and symbol detection, etc., are typically performed. In this lab (and in the figure) the analogue parts are connected back-to-back simulating a very simple channel (as added noise).

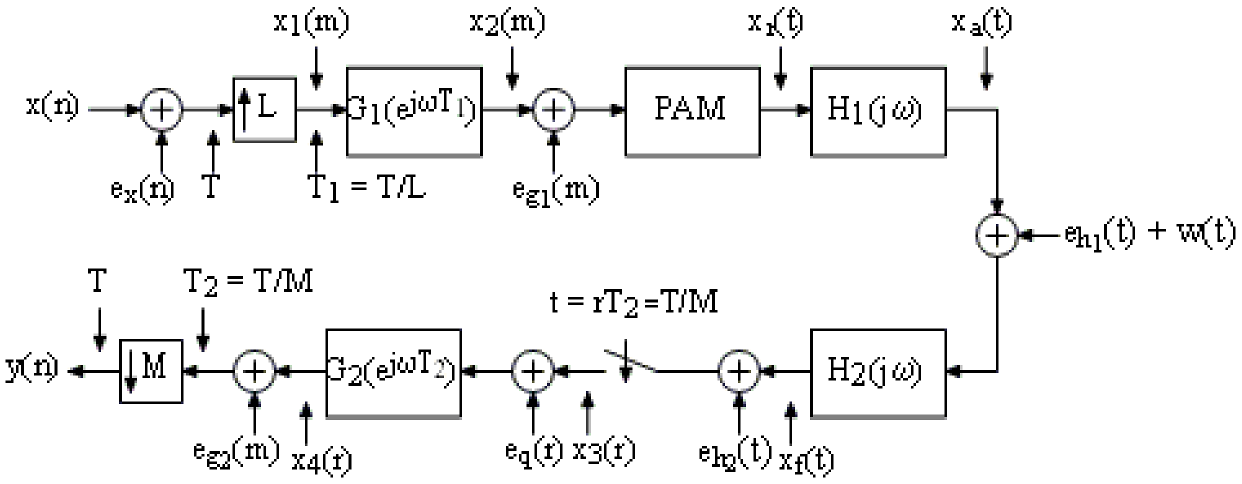

Figure 1.1: Mixed-signal system.

1.1 Initiating your Linux Environment

In this laboratory we will use a simulation tool from the vendor Mathworks, called MATLAB. (There are free clones, such as GNU Octave, which you could also use. However, some of the routines created for this laboratory work is not fully compatible with the octave environment.) During the lab, we will run a MATLAB executable model of the system shown in Figure 1.1 and the MATLAB files needed for this are found in the project setup directory in your `$WORKAREA`:

```
$WORKAREA/daisyProjSetup/m
```

The main file that will be used repeatedly during the laboratory work is called `tste16Lab2.m`. Other files found in the directory are subroutines and definitions which are called by `tste16Lab2.m`. The file requires that you, as a user, set some variables/parameters that are used to run different segments of the code. Throughout the manual we will highlight what variables to set.

However, before you go there, make sure you have loaded the module for the course:

```
module load TSTE16
```

and sourced the startup file

```
source ~/.TSTE16_rc
```

so that you can use, e.g., the `cde` command to jump to the `$WORKAREA`.

> ✗ *If the any of the two above commands fail, please read through the manual for laboratory 1! Notice that you must have created your project directory first!*

Once you have loaded the module, you can launch MATLAB without java manager by typing

```
mlab
```

(You will see some warnings if you run without java manager, but they can be ignored.) If you want MATLAB with the java manager, you can instead use:

```
mlabjava
```

You can open and look at the lab script by typing

```
edit tste16lab2
```

and you get some first-order help by typing

```
help tste16lab2
```

Please feel free to copy the file to your local directory

```
$WORKAREA/work_$USER/m
```

in case you want to do some extended tests on your own. Use another name to not confuse MATLAB too much.

- ✗ *Notice that you are supposed to work in your project directory just as you did with your Cadence exercises in the first lab. This is also how you are going to work during your project, and then be sharing data with the whole project group.*
- ✗ *Try to understand the organization of the files and avoid adding files in e.g. $HOME or in other directories than the project directory. This is vital for your project work!*

Now start with the exercises that essentially takes you through the signal chain shown in Figure 1.1.

2. UP-SAMPLING

Interpolation consists of an up-sampling followed by digital filtering. Here it is used to increase the sampling frequency of the digital input signal before the data is reconstructed by the PAM and reconstruction filter.

Why is the sampling frequency increased despite the fact that this will also increase the operating speed of the PAM (D/A converter)?

We will now study the up-sampling, which increases the sampling frequency. In MATLAB, start this by set ting a variable to run correct part of the code:

```
labSection = 2;
```

and select a suitable interpolation factor `L` *(such as an integer in the range 2 - 8).* Before you run the code, you also have to set the frequency scaling constant, `k`, to `k = 1e6`; Run the code by typing

```
tste16Lab2
```

✗ *If you have forgotten to set some of the parameters, MATLAB will complain and produce an error message. Set the variable/parameter MATLAB refers to and then retry.*

The generated figure will now show two amplitude spectra. The top spectrum is that of a sinusoid signal at $\omega_0 T = 0.8\pi$ and the lower spectrum shows an upsampled version of the sinusoid.

What has happened in the lower spectrum plot?

What has happened to the signal power of the desired signal?

Why do we need a digital filter after the up-sampling?

3. INTERPOLATION FILTER

We will now design an interpolation filter. We assume that the information bandwidth of the input signal is $\omega_0 = 0.9\pi/T$. This will determine the passband and stopband edges of the interpolation filter. The remaining design parameters are the passband ripple and the stopband ripple.

Set

```
labSection = 3;
```

and select suitable values for the passband ripple to and the stopband ripple (For example `dcG1 = 0.01` and `dsG1 = 0.001`, respectively). Run the code by typing `tste16Lab2` again.

The previously upsampled sinusoid is now filtered by the designed interpolation filter. What is the amplitude of the desired sinusoidal signal?

What is the amplitude of the dominant aliasing tone after interpolation?

How would the interpolator be designed to yield an SFDR of 84 dB when the passband ripple is 0.01?

4. PULSE AMPLITUDE MODULATION (PAM)

The interpolated signal is then reconstructed into an analogue form, i.e., the signal will become a continuous-time signal. This is done by pulse-amplitude modulation (PAM).

In practice the PAM is realized by a digital-to-analog converter (DAC) followed by an analog filter. Why?

We will now study the effect of the PAM and DAC. Change the section of the script by

```
labSection = 4;
```

and run the code. In the resulting figure, three plots are shown. The *x*-axis does now correspond to angular frequency in [rad/s], i.e., we are studying continuous-time responses.

The top and middle plots show artificial responses which do not correspond to any response of the system in figure 1.1 but they are nevertheless useful when understanding the system response calculated from the input to the output of the DAC (PAM).

The top plot show the continuous repetitions of the magnitude spectrum of the interpolation filter. This is the shape by which the input signal spectrum will be shaped by and shows how the input spectrum is be repeated indefinitely.

The middle plot shows the magnitude response corresponding to the pulse of the PAM. For regular DACs, the pulse spectrum is defined by a so called sinc function. The sinc function assumes its peak value around DC, i.e., around the main lobe. It also consists of an infinite number of sidelobes. Between every lobe there is a zero.

The bottom plot shows the actual response from the system input to the output of the DAC (PAM).

At what frequencies are the zeros of the sinc function located?

Consider the bottom plot - what happens to the desired signal spectrum in the passband?

The repetitions are called images. How large is the largest image?

What is the worst case SFDR?

The DAC (PAM) has a DC gain equal to the sampling period for which it operates. The higher the sampling frequency, the lower the gain. This is a consequence of preservation of signal energy. The total energy of the signal being reconstructed cannot depend on the sampling frequency of the DAC. In the MATLAB model, set the frequency scaling constant, *k*, to 1 and rerun.

```
k = 1;
tste16Lab2
```

Now the sampling frequency of the DAC is set to *L* samples/s As the frequency axis is scaled down, the amplitude scales up.

5. RECONSTRUCTION FILTER

The reconstruction filter is an analog filter situated after the DAC (PAM). The purpose of this filter is to smoothen the output signal of the DAC. In the frequency domain this corresponds to attenuating the images created by the DAC.

Set the upsampling factor `L = 2;` and run. Study the magnitude response after the DAC (PAM). Set the upsampling factor `L = 5;` and run again.

What happens to the images and the requirements of the reconstruction filter after you change L from 2 to 5?

What is the drawback of increasing the interpolation factor?

We are now going to design an analog reconstruction filter. Set

```
labSection = 5;
```

We choose a Chebyshev-I filter. The passband and stopband edge frequencies are determined by the choice of the interpolation factor *L*, *T*, and the system information bandwidth. The design parameters are the passband and stopband attenuation. First, set the interpolation filter stopband attenuation requirements to achieve an SFDR of 84 dB. Secondly, choose a passband ripple of the analog reconstruction filter to `dcH1 = 0.0559` corresponding to a passband attenuation of 0.5 dB and the stopband ripple to `dsH1 = 0.01` corresponding to 40 dB attenuation in the stopband. Finally, set `L = 2` and run.

The upper figure shows the magnitude response of the designed Chebyshev-I filter. In the passband, the response is equiripple with 0.5 dB attenuation. The stopband response is monotonically decreasing The lower plot shows the magnitude response after the reconstruction filter. As can be seen from the figure, the images are not sufficiently attenuated in order to achieve 84 dB spurious-free dynamic range. The stopband attenuation of the reconstruction filter as well as the filter order need to be increased. The required filter order is calculated from the specification of the filter. When increasing the stopband attenuation, the filter order will also increase.

Keep the passband ripple of 0.5 dB. Design the reconstruction filter iteratively such that the worst-case SFDR is better than 84 dB and fill out the table below.

Interpolation factor, L	Required filter order	Achieved SFDR [dB]
2		
3		
4		
5		
6		

Table 5.1: Reconstruction filter parameters

Set the frequency scaling constant, *k*, back to `k = 1e6;` and select design parameters for the reconstruction filter corresponding to L = 2 in table 5.1.

Here we have designed the reconstruction filter with the purpose to smoothen the reconstructed signal, i.e., to suppress image spectra. In for example digital radio communication systems, the reconstructed signal is typically frequency-upconverted and amplified. In this process, additional filtering is usually performed with the purpose to band limit the spectra of the transmitted radio frequency signal. Due to the fact that components are nonlinear, such as the DAC and the amplifier (power amplifier, PA), the signals processed tends to produce a wider frequency spectrum which needs to be limited to prevent it from interfering with other systems and standards in other frequency bands. Therefore, radio communication standards specify so called spectral masks that specify the maximum signal power at certain frequencies and bandwidths of the transmitted signals.

6. UNIFORM SAMPLING

We will now consider the sampling of the analog signal coming from the channel (see figure 1.1). Before doing this some comments on the channel are in place. It is modelled by adding an error to the reconstructed signal. In the simplest case of this model, the signal added is an uncorrelated noise signal. Better models take into consideration correlated noise and distortion as well as so called multi-path propagation and frequency dependent channel attenuation (fading).

Before the sampling, an anti-aliasing filter is used to limit the signal and noise bandwidth. If these are not bandlimited, the higher frequency contents will alias (interfere and add to) the desired signal. In this section we do not add any noise or distortion signal and therefore the reconstruction filter acts also as an anti-aliasing filter. The undesired signals that are aliased during the sampling is then the remaining image spectra caused by the DAC. It is pointed out, however, that an anti-aliasing filter is always necessary along with the sampling (integral part of the ADC). Here, in this example, one analog filter serves as both reconstruction filter and anti-aliasing filter which cannot be in practical systems when interfering noise and distortion will emerge between the reconstruction filter and the sampler.

Set

```
   labSection = 6;
```

and run the code.

The signal is now sampled uniformly with a sampling frequency that is determined by $1/T$ and the oversampling ratio *M.* The resulting figure shows three plots of the system magnitude responses from the input to the output of the sampler. The top and middle figure are artificial responses in the sense that they are not in them selves responses of any part in the system in figure 1.1. The top plot shows the response for $p = 0$ in Poisson's summation formula (Poisson's summation formula describes the frequency response of a uniformly sampled signal). This gives the fundamental term containing the desired signal part. The second plot shows the response from all other terms in the Poisson's summation formula. For frequencies where the useful information is located, the middle plot shows the level of aliasing that is introduced which is useful when determining the SFDR of the system after sampling.

The lower plot show the total response from the system input to the output of the sampler (the sum of the responses in the top and middle figure). Here all terms are included, making the spectrum periodic as it should according to the definition of the discrete-time Fourier transform.

Try some different values for the oversampling ratio, *M*, of the sampler. For each value run the code and study the magnitude response.

Is the location of the aliasing dependent on *M*? Explain!

What is the effect of the anti-aliasing filter as *M* is changed?

7. DECIMATION FILTER

Hitherto, we have studied the functionality and the magnitude response of the system in figure 1.1. From this one can estimate the expected SFDR of the system where the spurious tones are coming from aliasing distortion. When the nonlinear behavior of the data converters and the analog filters the SFDR can be degraded due to the distortion generated and added by the nonlinear operation of these blocks such as static nonlinearity of the ADC, nonlinear gain of operational amplifiers used in the active filter e t c. Here in this high-level MATLAB model of the system, these latter nonlinear errors sources have not been considered.

The signal after the analog-to-digital conversion is decimated by *M* to retain the original sampling rate 1/*T*. The decimation filter bandlimits the signal prior to the downsampling. The inband SFDR will in practice not change as a consequence of this filtering since the digital filter can be made to introduce very little distortion to the signal. This means that it the SFDR in many cases is set during the sampling. We therefore do no study the decimation further in the context of SFDR.

The passband and stopband edges of the decimation filter is determined by the information bandwidth, the sampling period of the original data, *T*, and the oversampling ratio of the ADC, *M*. The free parameters are the passband and stopband ripples.

Set

```
  labSection = 7
```

and the pass- and stopband ripples to `dcG2 = 0.01;` and `dsG2 = 0.001;`, respectively. Run the code and compare the system response before and after decimation filtering.

How does the decimation filter affect the SFDR regarding tones originating from the input signal itself (aliasing)?

How does the decimation filter affect the SFDR regarding tones generated inside the system due to nonlinear operation?

8. SYSTEM-LEVEL DESIGN

We now turn our attention also to the noise generated in the system in figure 1.1. In a practical implementation of the system, many noise sources will interfere with the useful signal of the system. Some examples are quantization noise, thermal noise, phase noise, flicker noise, channel noise etc. In this lab we will focus on the quantization noise only.

The signal-to-noise ratio (SNR) is of interest. It relates the signal power to the noise power and is usually estimated using a single-tone input signal. One can also calculate the SNR for a single-tone input in case the noise power spectral densities are known. When testing a physical device, this is not possible. Then, the SNR is estimated from the output data produced. However, when modelling a system as is done here, one can also use models of the noise and calculate the noise power. This is done in this lab by numerically integrating the modelled noise power spectral densities.

Quantization noise is in the considered system added by finite arithmetic in the representation of data in the digital filters and of the input signal itself. The ADC is also contributing to quantization noise. Some parameters that influence the quantization noise are the number of bits in the input data, `Wx`, the number of bits of the decimation filter output data, `Wg2`, and the number of bits of the ADC and the DAC, `WADC` and `WDAC`, respectively. Besides these, the filtering of the digital and analog filters (`dcG1`, `dsG1`, `dcG2`, `dsG2`, `dcH1`, `dsH2`) as well as the oversampling ratios `L` and `M` affect the achievable SNR.

Set

```
labSection = 8;
```

and consider the following system sub specification:

Requirement	Value
Number of input bits (N)	12
System signal-to-noise-and-distortion-ratio (SNDR)	> 60 dB
Noise and distortion margin	6 dB
System signal-to-quantization-noise ratio (SQNR)	> 66 dB
System spurious-free dynamic range (SFDR)	> 60 dB

Table 8.1: Specification

The SNDR will depend on the errors introduced from many different sources. Here we consider the system design from a quantization noise perspective and since more noise sources will be added in the realization, the SQNR will be an upper bound for the SNDR. In order to account for this, a design margin of 12 dB is added.

Use the MATLAB model iteratively to find the system parameters so that the above specification is fulfilled and fill in the table below. Try to think about the potential implementation cost of each subsystem when choosing the parameters and make suitable choices. Also, try to share the design margin, i.e., how much the performance of your design exceeds the specified requirements, as evenly as possible.

Parameter	Value	Comment
Wx	12	Do not change, this is a specification item.
*Wg*2		Word length of digital filters
WADC		ADC number of bits
WDAC		DAC number of bits
L		Interpolation order
M		Decimation order
*dcG*1		Pass-band ripple in digital interpolation filter
*dsG*1		Stop-band ripple in digital interpolation filter
*dcH*1		Pass-band ripple in analog reconstruction and anti-aliasing filter
*dsH*1		Stop-band ripple in analog reconstruction and anti-aliasing filter
*dcG*2		Pass-band ripple in digital decimation filter
*dsG*2		Stop-band ripple in digital decimation filter
SNR at input [dB]		
SNR after interpolation [dB]		
SNR after anti-aliasing filter [dB]		
SNR after ADC [dB]		
System SNR [dB]		
System SFDR [dB]		

Table 8.2: System design parameters