# Experiment # 5

# Pulse Code Modulation

## 1 Purpose

The purpose of this experiment is to introduce Pulse Code Modulation (PCM) by approaching this technique from two individual fronts: *sampling* and *quantization*. As any textbook can attest, PCM is a technique widely used in communication systems, in particular in the conversion of analog signals into their digital representation.

PCM will be explored in this experiment in three manners:

- First, the validity of the sampling theorem will be verified through a simulation in `Simulink`.
- Second, the input sampling rate will be modified and the resulting output will be analyzed on the DSP target hardware. In this part of the experiment, you will observe *aliasing* occurring, in practice, on the output signal.
- Third, the number of quantization levels utilized to convert the analog signal into a digital signal will be modified. This will be done by altering the number of bits of the data word sent to the DAC from the DSP. You will verify the consequences of this limitation on the output signal.

## 2 Background Reading and Preparation

It is fair to say that every textbook in digital signal processing starts by describing how an analog signal is converted into digital prior to being processed. Sampling and quantization are described in detail in every one of them. Therefore, rather than looking for a single reference for background reading, you can refer to any of the textbooks cited in the bibliography presented at the end of this outline. Keep in mind, however, that these should be seen as introductory reading, since they only touch on PCM as applied to A/D conversion. The textbook for this course ([1]) looks at PCM from the point of view of communication systems going well beyond A/D conversion. Another good reference is [2].

*Before* coming to the lab, complete the **lab preparation** and hand it to the T.A.. If you *still* are not comfortable with `Matlab` and `Simulink`, come to the lab and practice on the workstations.

## 3 Equipment

Hardware:

1. One Signal Generator;
2. One Two-Channel Oscilloscope;
3. One Spectrum Analyzer
4. One TMS320C6711DSK Module (with audio daughtercard) attached to a workstation;
5. Three Coaxial cables BNC-to-BNC.

Software:

1. `Matlab` Release 12
2. `Simulink` with ECE416 Toolbox
3. `Code Composer Studio`, v.2.1

## 4 Experiment

The experiment is divided in three parts: the simulation of a system presenting aliasing; the modification of sampling rates to verify the occurence of aliasing on real signals; and the verification of the relation between signal integrity and word length in the quantization process. Differently than the previous experiments where you verified the concepts by designing a "whole" system, this experiment will touch individually on the two main parts of PCM, namely: sampling and quantization. The reason for taking this approach is primarily due to the fact that the target hardware already provides a CODEC which utilizes PCM to convert the signal from analog to digital. In this experiment you will modify the data generated by the existing CODEC and observe the consequences of the modifications on the output signal.

**All results are to be reported in the spaces provided in this outline. Make sure that the T.A. verifies every result you record.**

### 4.1 Sampling a Signal At The Wrong Sampling Rate

Since in every other experiment you started out trying to design the *right thing*, in this experiment you will be required to put in an extra effort and start by designing the *wrong thing*. It should come with no surprise that it is simpler than you think. Departing from a system working as it should, you will modify the sampling rate of the incoming signal to observe how **not** to sample it. This will be done with the use of a new block: the **downsample** block. Downsampling is also known as **decimation**. The textbook representation of decimation in block diagrams is a block containing an arrow pointing down followed by a number as the **decimation factor**. As the name implies, when you decimate or downsample a discrete signal by a factor, you *reduce* the sampling rate of the signal by that factor. This is to say that you will discard a number of samples in the process. For instance, if you have a 1KHz signal sampled at 48KHz, you will have 48 samples per period. When you decimate this signal by a factor of 2, you will discard every other sample and end up with 24 samples per period. This is equivalent to sampling the 1KHz signal at a 24KHz sampling rate.

Before you start, remember that:

- All blocks that you will use to build your system are in the `ECE416` blockset on `Simulink`.
- The input to an FFT-based scope must be buffered, with a buffer size that is an integer multiple of the FFT size (could be the same size). It makes no sense to attempt to perform an FFT on a single sample;
- A greater FFT length means a more accurate frequency domain representation of the signal (take 1024 points as a reasonable length). At this point you should be able to tell when the results are producing relevant information or insufficient information;
- The ADC block **must** be set for 1 sample per frame. The DSP target hardware will provide an error message otherwise.

Open a new model on `Simulink` and add a DSP Sine wave generator attached to one time scope and one FFT scope. Adjust your sine wave to be $2V_{pp}$ and $22KHz$, using a $48KHz$ sampling rate. Set yor Spectrum Scope to display frequencies between 0 and $f_s$. Make sure that this runs smooth. Now place the decimation block between your sine wave generator and the scopes. Set the decimation factor to be 2 and select "force single rate". Run your simulation.

*Be prepared to answer questions asked by the T.A.*

- *In the space below, sketch the output signal observed in the time domain and frequency domain after the decimation.*

- *Increase the decimation factor to 6. What is the new sampling rate? Sketch the output signal observed in the time domain and frequency domain. Is it appropriate to sample a 22KHz signal at this rate?*

Substitute the sine wave generator by the ADC block and the scopes by the DAC block. You should have a system that downsamples the input signal by a factor of 6 and sends it out. Download your model to the DSP target hardware, and input a $22KHz$, $2V_{pp}$ sine wave into it.

- *Use the* `Matlab` *program to display your output data in the time and frequency domain. Sketch the results below.*

## 4.2 Uniform Quantization And Number Of Bits

This part of the experiment will be done without simulation. You will build a system and download it to the DSP target hardware, and modify the number of bits utilized to pass the incoming data to the DAC. You have already verified in the first experiment of this course that if you build a "straight-through" system, the sinusoid obtained at the output is a faithful representation of the sinusoid you input to the system. This means that the quatization done by the CODEC is done in enough levels to provide for a very good representation of the incoming signal, as it was sampled at 48KHz. In this part of the experiment, you will limit the number of bits used in this quantization, and observe the results on the oscilloscope. As you have already reviewed, as you reduce the number of bits on the quantization, you increase the error introduced in the process by reducing the number of quantization levels available.

The technique you will utilize to modify the word length of the incoming signal is known as "masking", and you likely have already utilized in a Microprocessors course. It consists of performing a logical AND operation between the data word passed on to the DAC and a "mask" that you select. The CODEC utilized by your DSP target hardware operates with a 32bit word, representing a two-channel (stereo) signal. Each channel is represented by half of the word (16 bits). The daughtercard which provides you with the input and output channels is pre-set to operate with data words of 8 bits, allowing for 256 quantization levels for the input signal.

For example, if you are looking to select one channel on this 32-bit word, you should apply a mask defined by the hexadecimal number **0000FFFF**. Since what you are looking for is only 8 of those 16 bits, your mask to select the signal without modifying the number of relevant bits is **0000FF00**. Keep in mind, of course, that the data word differs for other types of hardware. The masks just shown are specific for your DSP target hardware.

You will find in your `ECE416` blockset three blocks that are to be used in this part of the experiment: the **channel selector for bit operation** block , the **channel recombiner after bit operation** block and the **bitwise logical operation** block. Build your system by linking the output of the ADC block to the input of the channel selector. One output of the channel selector is to be connected straight into one of the inputs of the channel recombiner. The second output of the channel selector will be fed into the bitwise logical operation block. The output of the bitwise logical operation block will be connected to the second input of the channel recombiner. Finally, the output of the recombiner will be connected to the DAC. Build your system and download it to the DSP target hardware.

With this configuration, you will have enabled one of the inputs and the two outputs of the DSP target hardware. Connect each of the outputs to one channel of the oscilloscope. On one channel you should see the input signal without any modifications, and on the second output you will see the modified version of the input signal. You can adjust your oscilloscope to overlap the two signals; it may provide you with a better picture for comparison.

While going through the steps below, you will be required to modify the mask on the bitwise logical operation block. Each time you modify the parameter on that block, you are to rebuild the model and download it to the DSP target hardware.

Using an input signal of $3.4V_{pp}$ and $80Hz$, follow the steps below. Make sure you have successfully used the mask which enables all 8 bits on the channel. That mask is **0000FF00**. Proceed after you make sure this mask works.

- *Set your mask to be* **00008000**. *Sketch your result and explain what is the mask selecting.*

- *Change your mask to reduce the data word to allow for 32 quantization levels. What is the mask to be utilized?*

- *Sketch the output produced if a 3 bit data word is utilized in the quantization of the input signal.*

## 5 Going The Extra Mile

If you have not yet looked at the C code generated by `Simulink`, maybe this is a good time to do so. Check, for instance, where you could have modified your mask without having to refer back to your model in `Simulink` and rebuild the entire model. If you decide to modify the mask within Code Composer Studio, do not forget to recompile your project (that is the ".pjt" file) after you make the modification.

Also, you can try to look into the individual blocks for channel selection and recombination. A number of bitwise and shift operations were done in order to select the data appropriately. These operations, as you probably already know, are very useful when dealing with signal processing. Try to understand what was done in the selection and recombination processes. Keep in mind that the need for all these operations is dictated by the hardware; different targets will demand that you manipulate the data differently.

## 6 Conclusion

In this experiment, you acquired a better understanding of Pulse Code Modulation by further probing into sampling and quantization. You verified the introduction of aliasing by sampling a signal at a rate lower than that dictated by the sampling theorem. You also verified the consequence of reducing the quantization levels by limiting the number of bits utilized to represent a signal. The concepts studied in this experiment are fundamental to a multitude of areas in engineering.

**Obs: The books cited in this outline contain relevant material for the student to better understand the topics pertaining to the experiment. The student should note that by reading the course textbook only, one should be able to successfully perform the experiment. This is to say that it is not necessary for the student to purchase all the references.**

## References

[1] B. P. Lathi, *Modern Digital and Analog Communication Systems*, 3rd Edition. New York: Oxford University Press, 1998.

[2] S. Haykin *Communication Systems*, 4th Edition. Toronto: John Wiley & Sons, Inc., 2001.

[3] Oppenheim, Willsky, with Young, *Signals and Systems*, 2nd Edition, Prentice Hall

[4] S. Orfanidis, *Introduction to Signal Processing*, Prentice Hall

[5] E. Ifeachor and B. Jervis, *Digital Signal Processing - A Practical Approach*, Addison Wesley