Experiment #2 – Pulse Code Modulation (PCM)

# Youssef Samwel

# [yo800238@ucf.edu](mailto:yo800238@ucf.edu)

# EEL4515 Fundamental of Digital Communications

Prof. Dr. George Atia - Section 0012

# 2/1/2024



# **Experiment Objective**

The objective is experiment is to introduce students to PCM modulation through MATLAB simulation and by pre-built circuit to view PCM waveforms on the oscilloscope. The lab also provides students with a method of modulating and demodulating PCM signals.

# **2.0 About Laboratory Day and Equipment List**

# The laboratory session took place on the Thursday section between 9:00am and 11:50am on February 1st, 2024. My lab partner was Isiah. The equipment for the is experiment is listed below,

1. MATLAB
2. Rohde & Schwarz RTM 3034 Oscilloscope
3. Module 296f board

# **3.0 Simulation**

In PCM modulations there is a discrete number of levels that we use to represent the signal. The level count is determined by the bits used to quantize the signal, for example, for N bits we get levels. In simulation we quantized a sine signal from 2 levels to 16 levels. In addition to quantization, we also computed the quantization error and SNR ratio. 



# **4.0 Implementation**

When the encoded bits are transmitted, how does the receiver know when the bit sequence starts and when it ends?

The prebuild module 296f boards implements synchronization by sending two “0” bits before the message and two “0” bits after the message, followed by eight “1” bits.

A diagram of a computer component

Description automatically generated

The modulation occurs by comparing the input signal with ramp voltage generated by 4-bit counter. Once the ramp voltage is higher than the input signal, the counter will latch its value to the shift register where it is transmitted. Demodulation occurs in a similar fashion.

A diagram of a computer component

Description automatically generated

The demodulation circuit works in reverse compared to modulation circuit. After the synchronization circuit, first it reads the digital PCM value with the counter and then using resistor ADC it outputs the recovered analog value.

# **4.5 Questions and Results**

The MATLAB simulation matched with the 296f module.

Display the ‘analogue input’ together with ‘analogue output’ on the scope, where ‘analogue output’ is the demodulated message. Is the input signal accurately reproduced? Explain any differences.

The reproduced signal does not fully match the input signal, this is due to quantization error; the recovered signal had the expected discrete levels which the input signal did not have. However, the reproduced signal could be improved by applying a low pass filter to reduce the stair-case effect. The low pass filter works because the rapid jumps have high frequency components and therefore after the filter the signal will be smoothed.

Press the ‘inhibit sync’ button on the decoder clock. What happens? Is synchronization necessary for PCM?

We did not have the chance to press the ‘inhibit sync’ button, but from my current knowledge, I would assume the modulator will be able to transmit at a higher rate at the cost of inaccurate recovering of the PCM signal. I would expect the bit stream to get out of sync therefore the demodulator would not read the binary value leading to the wrong level being generated.

Consider again the output signal from step (6) of the decoding portion in section 2.4. What could you do to improve the demodulated signal? What would that cost be? Use bandwidth, data storage, cost or any other relevant factor as your basis.

To improve the quality of the recovered signal, I would increase the quantization levels (increase counter bits). The cost of higher quantization will lead to a lower rate data of transfer if the pulses take the same amount of time. However, if the pulses duration was reduced to keep the same data rate, then the bandwidth would get wider because contraction of a signal in time domain will lead to dilation of the signal in frequency domain. In addition, storage requirements will larger because each sample will contain more information. The most expensive cost of increasing quantization levels is the cost of higher resolution ADC and DAC, these components at higher resolution cost much more (can be in the thousands of USD) and have much higher power consumption and are typically more sensitive to noise and PCB placement.

# **5.0 MATLAB Code**

Used to generate figure(s)

clear;

clc;

close all;

%% Simulation Parameters

Smax = 1;

Smin = -1;

Omega = 1;

%% Generate Signal

t = 0:0.1:10;

signal = sin(Omega.\*t);

%% Quanziation

f = figure;

f.Position = [0, 0, 1280, 720];

centerfig(f);

index = 1;

for n = 1:1:4

quantized\_signal = QuantizeSignal(signal, Smax, Smin, n);

subplot(2,2,index);

plot(t, signal);

hold on;

grid on;

pwr = ComputeSignalPower(signal);

error = ComputeQuantizationError(signal, quantized\_signal);

SNR = 10\*log10(pwr/error);

leg = sprintf("Quantization Error = %.2f", error);

leg = leg + " --- SNR: " + SNR;

qp = plot(t, quantized\_signal, 'DisplayName', leg);

title("Signal vs Quantized Signal using " + n + "-bits or " + (2^(n)) + " levels", 'FontSize', 14);

xlabel("time (sec) \rightarrow", 'FontSize', 13);

ylabel("Amplitude", 'FontSize', 13);

ylim([-1.5,1.5]);

legend(qp);

levels = 2^(n) - 1;

quant\_step = (Smax - Smin) / levels;

for level=0:1:levels

yline(Smin + (level \* quant\_step), '--', dec2bin(level, n), 'HandleVisibility','off');

end

index = index + 1;

end

function qs = QuantizeSignal(source, smax, smin, bits)

% Determine level step

levelCount = 2^(bits);

n = levelCount;

qs = source;

qs\_len = length(source);

for i = 1:1:qs\_len

a = qs(i);

b = a + smax;

c = (n - 1) \* (b / (2 \* smax));

d = round(c);

a\_quan = 2 \* smax \* d / ( (n - 1)) - smax;

a\_error = a - a\_quan;

qs(i) = a\_quan;

end

end

function error = ComputeQuantizationError(sourceSignal, quantizedSignal)

err = sourceSignal - quantizedSignal;

err = err .^ 2;

error = sum(err);

end

function pwr = ComputeSignalPower(sourceSignal)

pwr = sum(sourceSignal .\* sourceSignal);

end

# **6.0 Learned Objectives**

* PCM Modulation
* PCM Demodulation
* MATLAB Simulation
* Methods of PCM Modulation and Demodulation

# **7.0 Conclusion**

In conclusion, the Pulse Code Modulation (PCM) experiment effectively introduced students to the principles of PCM modulation and demodulation, combining MATLAB simulation with practical implementation on a module 296f board. The simulation phase illuminated the impact of quantization levels on signal accuracy, while the practical implementation demonstrated synchronization methods and the modulation-demodulation process. Despite the demodulated signal's divergence from the input due to quantization error, suggestions for improvement, such as employing a low-pass filter, were discussed. The experiment provided valuable insights into the trade-offs associated with quantization levels, contributing to a comprehensive understanding of PCM systems and their applications in digital communication. The accompanying MATLAB code served as a useful tool for signal analysis and visualization, enhancing the overall learning experience.