Measurement 93 (2016) 288–302

Contents lists available at ScienceDirect

Measurement

journal homepage: www.elsevier.com/locate/measurement

A low complexity digital unit for scalar linear interpolation and compression

Nikos Petrellis

Computer Science and Engineering Dept., TEI of Thessaly, Greece

ARTICLE INFO

Article history:
Received 14 July 2014
Received in revised form 17 March 2016
Accepted 5 July 2016
Available online 6 July 2016

Keywords:
Signal reconstruction
Analog digital conversion
SNR improvement

ABSTRACT

A linear interpolator with compressed output and its alternative version that supports error correction are presented in this paper. The interpolation devices can be driven by an Analog/Digital Converter (ADC) that accepts as input, low frequency signals, like sensor values. The proposed interpolation methods can correct ADC linearity errors and increase the dynamic resolution of an ADC or they can reconstruct a signal from fewer samples in non-uniform distance. If the input signal is partially exponential or logarithmic then, specific correction rules can be employed to achieve a lower signal quantization error. These rules are based on simple operations like shifts and comparisons and thus, they can be implemented using low complexity hardware. Spline, Linear, Quadratic interpolation, popular compression tools and reference signals are used to evaluate the experimental results. The proposed architecture is scalar since multiple interpolators can be connected in series. A 3-stage interpolator with 9-bit input and 12-bit typical output resolution was tested with sinusoidal input. The Signal to Noise and Distortion Ratio (SNDR) of the ADC that has been used was increased up to the double. The proposed method was also tested using a pressure sensor, an acoustic signal and image reconstruction leading to Mean Square Error (MSE) that is up to 10 times lower. The achieved compression ratio, ranges between 11% and 76%. The implementation complexity of a one stage interpolator with 9-bit input resolution requires 583 Logic Elements (LE) i.e., less than 3% of the LEs that exist in an Altera Cyclone III EP3C25N Field Programmable Gate Array (FPGA).

1. Introduction

Several interpolation methods have been studied for accurate signal reconstruction including the Linear (or first-order hold) and Spline interpolation in the time domain or more sophisticated methods like the interpolation in the frequency domain, Lagrange, and min-max interpolation. An implementation with low complexity hardware is feasible if linear interpolation in the time domain is used, since most of the other methods require more complicated operations like floating point multiplications, divisions, and matrix inversion. The use of a simple interpolator at the output of an ADC can lead to a system with lower overall complexity since an ADC with lower resolution/sampling speed can be employed.

The linearity and quantization errors of an ADC may significantly vary in different input signal frequencies. One way to correct such errors in real time is the use of a calibration method that is customized for a specific ADC architecture and can be embedded in the same chip. Another solution is to use a generic interpolation method in order to improve the linearity (resolution, slope, etc) of different ADC architectures, by post-processing their outputs. It can also be employed when the reconstruction of a signal from fewer samples is desired or for the correction of samples that have been distorted by noise. In the following paragraphs, several ADC quantization and linearity error correcting approaches are described. Although many of them are not related to the proposed interpolation method, they are used as a reference in order to compare their efficiency and complexity.

The approach presented in this paper is closer to the one described in [1] where several linearly interpolated samples are inserted between successive known samples. The typical resolution in [1] is increased from 8 to 12 bits by inserting between each pair of successive samples, 16 new interpolated values generated by a counter. The interpolator described in [1] has been fabricated in a 1×1 mm^2 CMOS integrated circuit.

The proposed interpolation method is post-processing the output values of an ADC. Several post-processing modules with different functionality have been proposed in the literature e.g., for the correction of Differential or Incremental Non-Linearity (DNL/INL) static errors. Such techniques are often based on the excitation of an ADC with DC levels or voltage ramps (best fit curves [2] or

E-mail address: npetrellis@teilar.gr

http://dx.doi.org/10.1016/j.measurement.2016.07.025
0263-2241/

Histograms [3]). More specifically, in the histogram approach, an M-bit ADC is excited with a ramp signal in order to get the initial 2^M outputs and then the ramp signal is shifted up to retrieve another set of 2^M outputs [4]. The differences between the corresponding values of these sets, or the Least Squares method is then used to estimate parameters like INL/DNL error [5].

The DNL linearity errors can be corrected by signed factors that are estimated with methods like the ones described in [4] or [5] and are stored in look up tables that have often large size. An accurate sawtooth signal generator is used in [6] to estimate the INL and extract error compensation coefficients of pipeline ADCs, in real time. The calibration circuit described in [6] is evaluated using reconfigurable hardware. The proposed interpolation method was also evaluated in reconfigurable hardware. Real time calibration of the DNL error correction coefficients is described in [7], too.

The basic concepts of the interpolation are described in [8]. Moreover, a detailed description of the cubic spline interpolation can also be found in [8]. Cubic spline interpolation can fit a 3rd order polynomial to 3 successive points (not necessarily in a uniform distance) but matrix inversion is required to solve a linear system of 3 expressions and 3 unknowns. Higher order polynomials like the quadratic used here for comparison, can also be used to match a higher number of successive points. The description of different interpolators like min-max and Lagrange can be found in [9] where lower complexity hardware is used.

More complicated interpolation techniques are presented in [10]. The block based and windowed block based methods [10] require matrix inversions and are tested for the reconstruction of finite length signals from samples with non-uniform distance. Multiplication and division operations are also required in a barycentric interpolation described in [11], that can be applied both to uniform and non-uniform or long intervals. In [12] the interpolation is also performed by a weighted summation of preceding and succeeding samples and the authors compare the two algorithms they have developed (Least Mean Lattice (LSL) and QR Decomposition LSL algorithm (QRD-LSL)). A digital error correction method formulated as a matrix problem is also employed in a time interleaved ADC architecture in [13].

The dynamic linearity errors are usually measured by Signal to Noise and Distortion Ratio (SNDR), Spurious Free Dynamic Range (SFDR) and Total Harmonic Distortion (THD). These parameters depend on the operating frequency of the ADC and are taken into consideration in approaches like the ones presented in [1,7,10,13].

The conversion of the ADC output to an analog signal through an ideal Digital to Analog Converter (DAC) is actually a zero-order interpolation (or zero-order hold). In the first-order interpolation, two successive ADC samples are assumed to be linearly connected. The approximation error is small if the distance between the successive samples is small. A non linear curvature between the two successive values cannot be approximated using a first-order hold. However, even the methods that can theoretically model this curvature like Cubic spline interpolation [8], may fail to match the original signal accurately.

The first interpolation scheme (Uncorrected Interpolator-UI) presented in this work, is based on a first-order interpolation in the time domain. It can be implemented with low cost hardware that does not require more complicated components than adders, comparators, multiplexors and counters. A second method (Corrected Interpolator-CI) attempts to reduce the quantization error of the interpolated values by taking into account the potential logarithmic or exponential curvature of the original signal. Although there cannot be a common correction method for all types of curves, a strategy that seems to reduce the quantization error in the most cases is adopted and tested. The error correcting factors in the CI scheme are also generated by circuits based on adders and comparators only, avoiding more complicated operations.

The proposed interpolation methods can significantly improve the quantization error of low resolution signals that are highly distorted by the digitization process. More specifically, the SNDR of a 2 MHz sinusoidal signal has been increased from 22 dB to approximately 40 dB (an improvement approximately equal to 100%) using three instances of the proposed interpolator module connected in a cascade mode. Signals with higher SNDR are improved by a smaller factor. The interpolators described in this paper are tested on sinusoidal and acoustic signals digitized by a low frequency commercial ADC and a higher frequency ADC developed by the authors [14]. Moreover, the developed interpolator is tested using the output of a pressure sensor. Although the present form of the interpolator is not optimized for image processing applications, it is also tested on grey images causing a smoothing effect and improving the MSE up to 10 times.

Besides the linearity improvement, the interpolator performs by default a real time compression in the case of sparse or low frequency signals. For example, the output of the pressure sensor was compressed to the 11% of its original size. In the compressed interpolator outputs the signal is represented in pairs of the form (*VAL*, *CNT*) where the value *VAL* of the signal appears *CNT* times. If no identical values appear in successive samples in the original signal then no compression is achieved. A similar compression method has been used in standards like LZ77/78 that are employed by the PNG, GIF format, etc. Another critical feature of the proposed interpolator is its scalability. Higher linearity can be achieved if multiple interpolators are connected in series (in real time environments) or if an interpolator is used recursively (for off line processing).

An interpolator with three stages of 9, 10 and 11 bit input resolution along with their corresponding decompression units require 3325 Logic Elements (LEs) or the 14% of the LEs of a Cyclone III EP3C25N FPGA.

The modeling of the UI interpolation method and the architecture of such an interpolator is described in Section 2. The quantization error correction rules that can be employed by the CI interpolator are discussed in Section 3. Finally, the implementation issues as well as the experimental results are presented in Section 4.

2. The UI interpolator architecture

The estimation of the intermediate points between two successive ADC output values (not necessarily in uniform intervals) with zero and first order hold is shown in Fig. 1. The dashed line denotes the original signal that is sampled by an ADC (solid line) and the output of the ADC is sampled at uniform intervals (vertical arrows) by the local interpolator clock. The sampling rate of the interpolator clock is higher than the sampling rate used by the ADC. The oversampling ratio R_{os} is defined as the ratio of the interpolator clock frequency to the ADC sampling rate. Using a higher interpolator clock frequency than the sampling rate of the ADC is feasible, since the interpolator is a digital circuit. It is more difficult to apply a higher sampling rate to an ADC unless it is based on a parallel (Flash) architecture, but the cost in this case is larger die area, lower resolution and higher power consumption. The sampling rate of the ADC may be reduced if a low cost interpolator like the UI described here, is used. In Fig. 1a, N_1 of the samples are quantized to the value V_1, N_2 of the samples are quantized to the value V_2, etc.

The UI interpolator replaces the last half of the samples with value V by the average of the last two successive sample values. For example, in Fig. 1a, the V_1 samples in the last $N_1/2$ interpolator clock periods are replaced by the value $(V_1 + V_2)/2$. If the original signal does not change linearly between the values V_1 and V_2 as