An FPGA-based Re-configurable 24-bit 96kHz Sigma-Delta Audio DAC

Ray C.C. Cheung[1], K.P. Pun[2], Steve C.L. Yuen[1], K.H. Tsoi[1] and Philip H.W. Leong[1]

[1] Department of Computer Science & Engineering [2] Department of Electronic Engineering

The Chinese University of Hong Kong, Shatin, Hong Kong

ABSTRACT

This paper presents a reconfigurable sigma-delta audio Digital-to-Analog Converter (DAC) which is suitable for embedded FPGA applications. The Sigma-Delta Modulator (SDM) design can be configured as a 3rd or 5th order SDM and allows different input word lengths. Different input sampling rates are also entertained by employing a programmable interpolator. The DAC accepts 16-/18-/20-/24-bit PCM data at sampling rates of 32/44.1/48/88.2/96 kHz for applications in CD, SACD and DVD audio.

1. INTRODUCTION

Field programmable gate arrays are able to offer advantages over traditional VLSI technology in terms of time to market, lower costs for small quantities and dramatically reduced development times. As Moore's law continues to improve device density, the trend is to integrate increasingly higher levels of functionality into FPGA designs. Many control and digital signal processing systems require data converters in order to provide analog outputs from a digital system. In such systems, an off-chip digital to analog converter (DAC) is normally employed.

Although FPGA technology might at first seem to not be suitable for the implementation of analog components such as a DAC, their architectures turn out to be very suitable for sigma-delta converters which are primarily digital. Having the flexibility to incorporate DACs into FPGA designs allow for higher levels of integration, reducing cost, board area and possibly power consumption. FPGAs also make an excellent prototyping environment for sigma-delta converter designs. In this paper, a flexible audio frequency DAC implemented using FPGA technology is presented. This design can be used as an intellectual property (IP) core which can be incorporated in FPGA based systems.

Surprisingly few FPGA-based DACs have been reported to date. Apart from a first order sigma-delta DAC with 6-10 bit accuracy reported by Logue [4], we are not aware of other FPGA-based DACs.

The rest of this paper is organized as follows. In Section 2, the architecture of the DAC is presented. Section 3 details implementation issues associated with our design. In Section 4, measured results are presented and conclusions are drawn in Section 5.

2. DAC ARCHITECTURE

2.1 System architecture

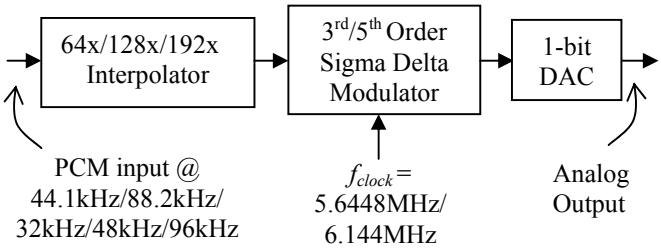

Figure 1: Block diagram of the audio DAC.

Figure 1 shows the system architecture of the audio DAC. It consists of three blocks, namely, the interpolator, the sigma-delta modulator and the 1-bit DAC.

The audio DAC accepts PCM input data at sampling rates of 32/44.1/48/88.2/96 kHz. The interpolation ratio of the interpolator can be configured to 64x, 128x, and 192x. For 44.1/88.2 kHz input signals, the interpolator gives the output data rate of 5.6448 MHz by setting the interpolation ratio as 128x/64x respectively. For 21/48/96 kHz input signals, the interpolator gives the output data rate of 6.144MHz by setting the interpolation ratio as 192x / 128x / 64x respectively.

The main function of the digital-to-analog conversion is performed by the sigma-delta modulator, which produces one-bit output and spectrally shapes the quantization noise to high frequencies. The modulator is clocked at 5.6448MHz or 6.144 MHz, depending on its input data rates as described in the previous paragraph. These two sampling frequencies of the modulator correspond to an over-sampling ratio (OSR) of about 128x with respect to the audio bandwidth of 20 kHz.

2.2 Programmable interpolator

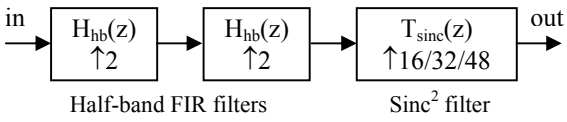

Figure 2: Interpolation filter.

The architecture of the 64x/128x/192x interpolation filter is shown in Figure 2. It is a multi-stage filter. The first two half-band filters are non-configurable. They increase the sampling rate of the signal by four times. The last stage is a programmable *sinc* filter to provide variant interpolation ratios.

The half-band filters are designed as 83rd order hardware-efficient FIR filters in a tapped cascaded interconnection of identical sub-filters[1], which requires no multipliers. An implementation of an 83rd-order FIR filter needs to perform only 124 additions at the input data rate. The *sinc*^{2} filter[2] has the transfer function of

$$T_{\sin c}(z) = \frac{1}{M^2}\left(\frac{1-z^{-M}}{1-z^{-1}}\right)^2, \qquad (1)$$

where $z = e^{j2\pi f/(Mf_{in})}$, f_{in} is the input sampling frequency, and M is the interpolation factor. The filter has a frequency response of sinc-shape, with notches at integer multiples of f_{in} to reject images. Here a sinc filter of second order is sufficient, because the high frequency images are not critical to the performance of the subsequent sigma-delta modulator.

The interpolation factor M of the *sinc* filter can be set as 16, 32 and 48 for the overall interpolation ratios of 64x, 128x, and 192x respectively. Figure 3 shows the block diagram of the *sinc*^{2} filter. The architecture of the filter is fixed. The programmable parts are the sampling frequency (f_{in}*M) of the last two integrators and the divisor M. For M=16 and 32, the division operation equals bit shifting. A fixed-point multiplier is required for M=48 only.

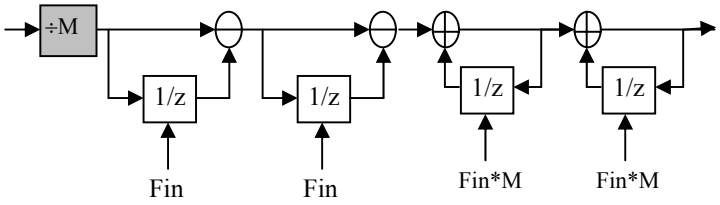

Figure 3: Block diagram of the *sinc*^{2} filter.

2.3 Re-configurable sigma-Delta Modulator

Besides variant input data rates, different input word lengths of 16-/18-/20-/24-bits are also accepted in the proposed audio DAC. One simple approach to do this is to use a sigma-delta modulator that meets the requirement of 24-bit accuracy for all the cases. Input data of less than 24-bit are then simply extended to 24-bit by padding zeros or ones. However, such a modulator is over-designed for low word lengths, and power will be wasted.

In our design, a re-configurable 5th/3rd order sigma-delta modulator is used instead. The architecture of the proposed modulator is depicted in Figure 4. It is a single loop modulator with all the zeros of the noise transfer function at DC. The coefficients a[i], b[i] and c[i] of the modulator are obtained using a Matlab design tool [3].

For 18-/20-/24-bit inputs, the modulator is configured as a 5th order one with all the shadowed blocks of Figure 4 active. For 16-bit inputs, all the shadowed blocks of Figure 4 are shutdown, the output of the third integrator is switched to the input of the quantizer, and the modulator becomes a 3rd order one.

Under the 3rd order configuration, the maximum signal-to-noise ratio (SNR) of the modulator within the audio band is about 96 dB, which is sufficient for the 16-bit data. Figure 5 shows the simulated output spectrum of the modulator excited by a 20kHz sinusoidal input.

Under the 5rd order configuration, the maximum

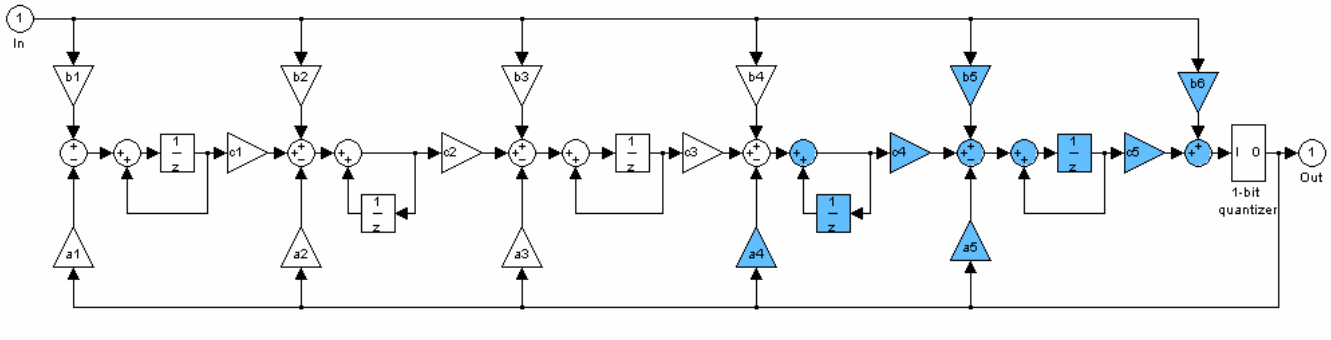

Figure 4. Architecture of the re-configurable 5th/3rd order sigma-delta modulator.

signal-to-noise ratio (SNR) of the modulator within the audio band is about 140 dB, which is sufficient for the 18-, 20- and 24-bit data. Figure 6 shows the simulated output spectrum of the modulator excited by a 20kHz sinusoidal input. It can be observed from the two figures that the noise level in the 5th order setting is about 35dB lower than that in the 3rd order setting.

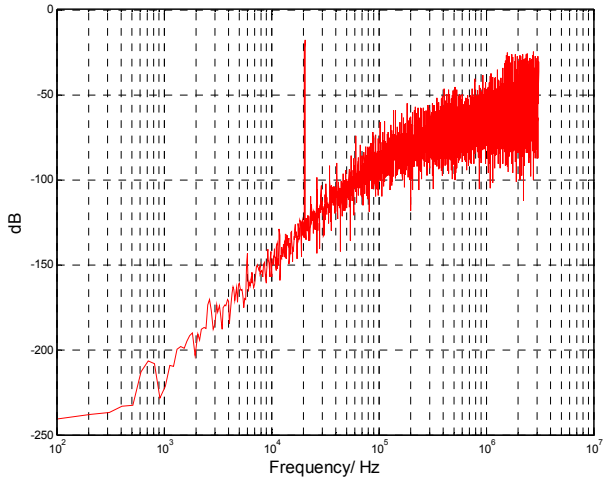

Figure 5: DAC output spectrum with the 3rd order setting.

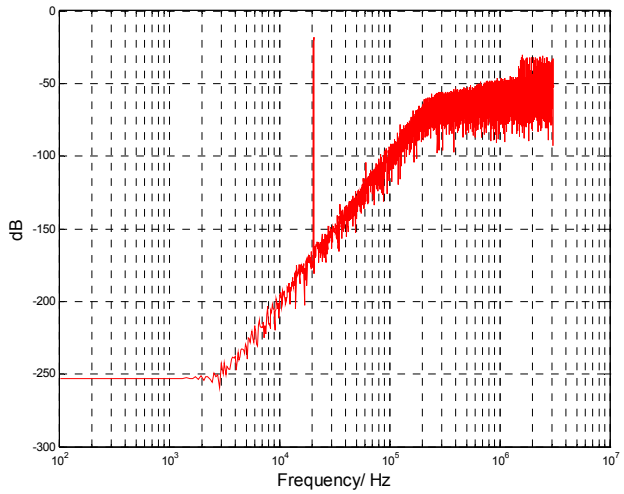

Figure 6: DAC output spectrum with the 5th order setting.

3. IMPLEMENTATION

In this section, the design flow and the rapid prototyping platform are presented. The software simulation is divided into two parts: the matlab simulation and the High-level description language simulation. The design flow provides a short turnaround design time for various SDM designs: different input data bit-width, different sampling rate, different oversampling clock rate and different SDM ordering. The design is suitable for reconfigurable platforms. We have implemented and verified our designs by using two different FPGA boards. They are from Celoxica and our research group and their details are described in later sections. We have used high precision logic analyzer to obtain and test the final output data.

3.1 Design Flow

There are two major stages in designing a reconfigurable Sigma-Delta Modulator (SDM) DAC: the prototyping stage and the running stage. In the prototyping stage, the input to our design is the specification of the Sigma-Delta Modulator in which it clearly states the input data rate, the input data size, the sampling frequency, the variable ordering and the internal SDM coefficients. The output from this prototyping stage is a workable SDM DAC. In the running stage, the input to the DAC is a set of digital PCM data which describes the input waveform. This DAC is able to shape the quantization noise to high frequency.

The overall design flow is depicted in Figure 7. We first obtain a correct design by performing extensive Matlab software simulation. Next, we model different orders of SDM designs by using hardware languages and this design is further generalized to be parameterized to certain objectives. The hardware design is also verified and simulated by using hardware compilers.

As shown in Figure 8, there are generally four steps in the design process: Specify, Design, Compile and Implement. The specify step is solely used for designer to preset the architecture of the DAC. The design step enables designer to turn the circuit components into description statements, to break large components into pieces and to reuse existing resources. The compile step

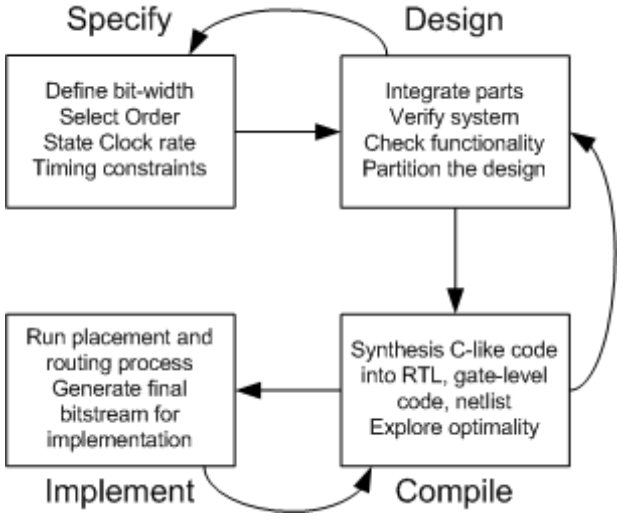

Figure 7. Design Process.

turns input high-level hardware description into Register-Transistor-Level (RTL) code and gate-level netlist. The area and speed optimization have been carried out in this step. The final step is the implementation step which generates the final bitstream for downloading onto the FPGA chip. Note that FPGA designs enable great flexibility for moving to any previous step in the design cycle.

3.1.1 Interpolator

The objective of an interpolator is to provide oversampling to the input of SDM. It means that the number of input is multiplied by a factor, the oversampling ratio. For instance, one period of the input PCM sine wave is sampled to 1000 points, the resulting wave would be very rough. An interpolator is actually a multi-stage filter which can be programmed and reconfigured in FPGAs. By using an interpolator, we can spot the intermediate points between every two input points. If the interpolator is a 64x interpolation filter, then the input data to the SDM is further increased to 64,000 points. As shown in Figure 8, the input wave "Vin" is distorted due to the insufficient sampling points. The two half-band FIR filters and Sinc filter generate the perfect output oversampled wave "Sinc".

We have implemented a sine wave generator using the C programming language for any bit-width and sampling frequency. This generator is used to produce the input static PCM data set. We have also integrated the interpolator onto this wave generator for adjusting the over-sampling ratio.

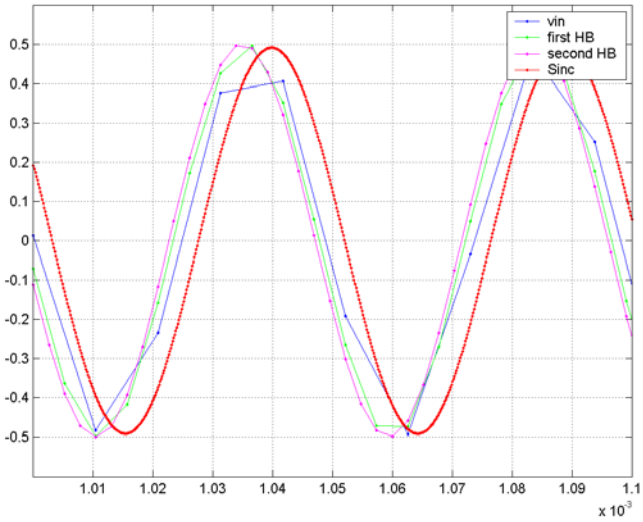

Figure 8. Interpolator Curve.

3.1.2 Sigma Delta Modulator

The reconfigurable SDM inputs a set of PCM data which describes a period of a 20kHz wave. It operates at the speed of the PCM input frequency times the interpolator ratio. It generate a bit-serial output data for a 1-bit DAC. The internal architectural of the reconfigurable SDM has been presented in the previous section. The regular pattern of the SDM module makes the design easily expandable and provides code reusing in our design. The SDM is a closed and loopback system that accumulates the noise to high frequency and we can use low-pass filter to remove all these accumulated noise from the input signal.

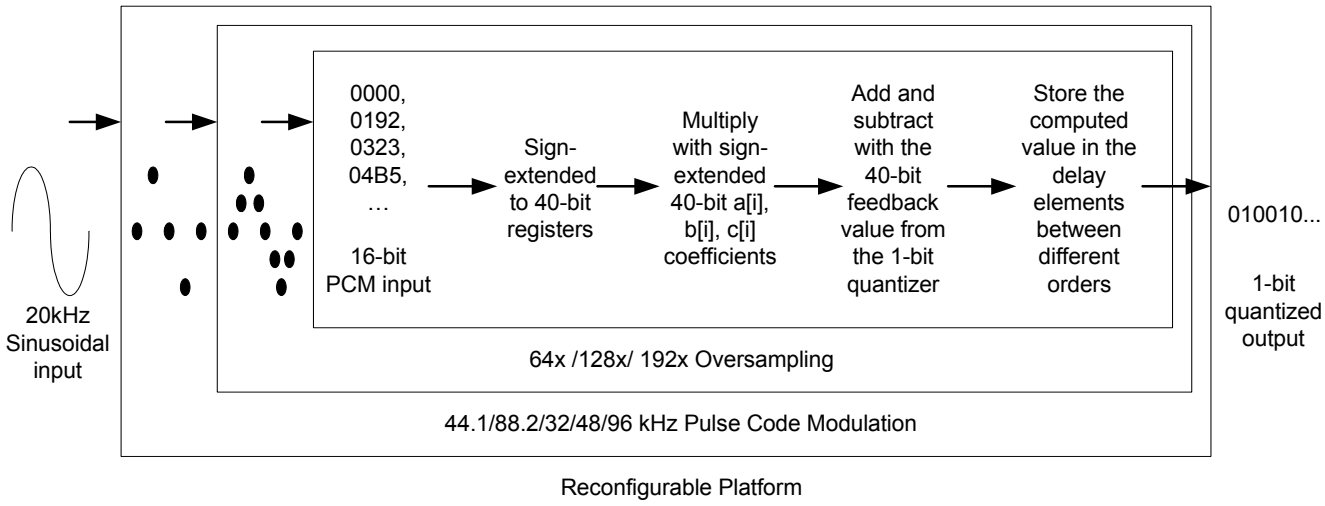

Figure 9. Sigma Delta Modulator Implementation.

```
/* Implementation of the 5th order SDM */
int 40 s1, s2, s3, s4, s5;
s5 = b[5] * extended_data - a[5] * feedback;
s4 = b[4] * extended_data - a[4] * feedback;
s3 = b[3] * extended_data - a[3] * feedback;
s2 = b[2] * extended_data - a[2] * feedback;
s1 = b[1] * extended_data - a[1] * feedback;
unit_delay[5] = unit_delay[5] + s5 + unit_delay[4] + s4 + unit_delay[3];
unit_delay[4] = unit_delay[4] + s4 + unit_delay[3];
unit_delay[3] = unit_delay[3] + s3 + unit_delay[2] + s2 + unit_delay[1];
unit_delay[2] = unit_delay[2] + s2 + unit_delay[1];
unit_delay[1] = unit_delay[1] + s1;
```

Figure 10. Partial implementation of the SDM in Handel-C.

Figure 9 shows the basic implementation of the SDM. The a[i], b[i] and c[i] coefficients are all floating-point numbers. However, the several large float-point multipliers which calculate the intermediate value between the input data and these coefficients would eventually used up all the resources of the FPGA chip. As a result, we extract the mantissa part of these coefficients and transform all the floating-point multiplication into fixed-point multiplication. We can achieve the same correct SDM result by using reduced logics. In the SDM, there are several large delay elements and adders. We can easily model them by using Hardware language. In our design, there are several user-defined values before compilation such as the ordering value, the bit-width.

Figure 10 and 11 show the pseudo code of the SDM implementation. There are basically two steps: the read data and the processing.

```
/* Implementation Flow for a 16-/24-bit SDM*/
1.  Specify the clockrate, order, bit-width
2.  Declare the bus I/O
3.  Pack every 2/3 bytes input data into RAM
4.  Stop until a period of data has been put into RAM

5.  Label l
6.    do
7.      Read data from RAM in every clock cycle
8.      Sign-extended the read data
9.      Pass into and calculate by the reconfigured SDM
10.     Output one-bit of data
11.   while (not the end of a period of data)
12. Goto l
```

Figure 11. The pseudo code of the implementation.

3.2 Prototyping Platform

This design can be implemented on any FPGA-based prototyping platform. We select two FPGA platforms for implementation. The first one is Celoxica RC200 development board as the testing platform which has embedded the Xilinx Virtex II XC2V1000-4FG456C FPGA chip. The proposed design has been coded with Handel-C 2.1 and synthesized by using Celoxica DK1.1. The second platform is the Pilchard [5] FPGA memory-interface board which is embedded with Xilinx XCV1000E. The FPGA chip is then configured as an SDM DAC and is used to calculate the SDM output bit. We use an Agilent 16702B logic analyzer to record and verify the output data.

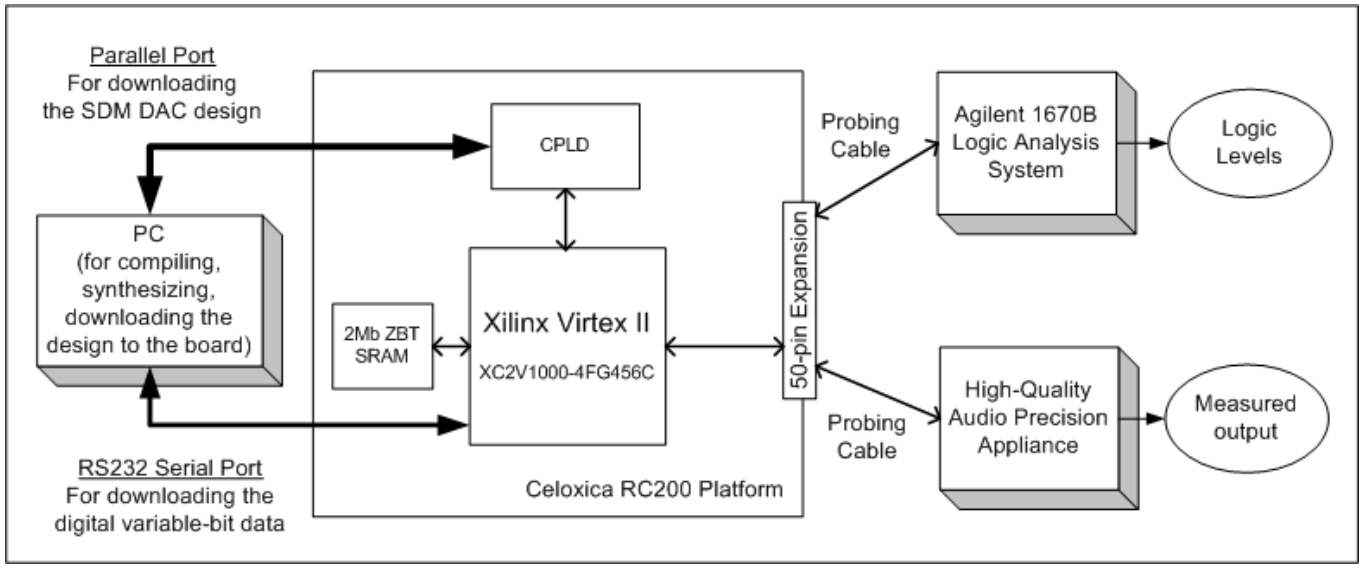

Figure 12. Prototyping Environment.

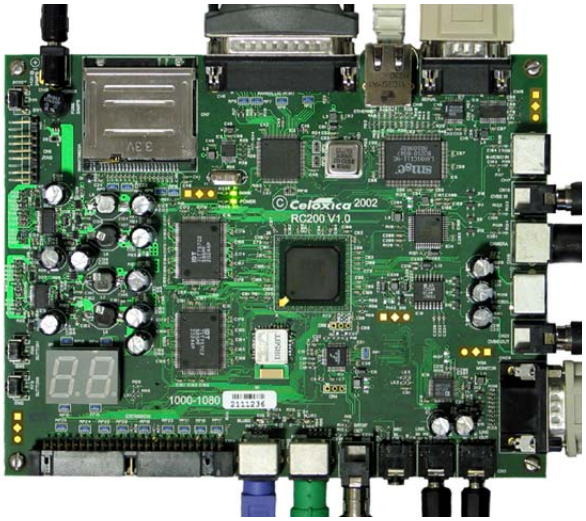

Figure 13. Photograph of the Celoxica RC200 board

3.2.1 RC200 Interface

The RC200 development board provides a wide selection for interfacing on different digital designs such as networking, blue-tooth, video, serial and parallel ports. The design and the photograph of the RC200 platform have been shown on Figure 12 and Figure 13 respectively. We have selected the RS232 serial port for transferring the input data onto the board. The interfacing between PC and the development board is made by the parallel and serial ports for bitstream and data transmissions. The download sampling data is stored on the on-board SRAM which provides interleaved Read/Write without wasting any turnaround cycles. Since the serial bus transmission can only allow 1-byte data per cycle, we need to break down the data into bytes at the PC host side. For example, we need to break 16-bit data into 2 bytes and reassemble it on the on-board SRAM before feeding into the FPGA for calculation. The calculated output signals are connected from the FPGA to the 50-pin expansion pins. The signals from the expansion pins are first probed and reported on the logic analyzer before using the high quality audio precision appliance.

3.2.2 Pilchard Interface

The Pilchard memory-interface board provides a fast data transfer between the host PC and the FPGA board. The sampled input data can be transferred to the FPGA board in a very high speed. In order to reduce the input data rate, the interpolator is again used for locating all the intermediate points. The photograph of Pilchard FPGA board is shown on Figure 14.

Figure 14. Photograph of the Pilchard card

4. RESULTS

4.1.1 Implementation

Here we show the result of the implementations in the table below. The hardware resources usage and the maximum working clock frequency have been put into the table. Figure 15 shows the placed and routed results of the 3rd order design.

Sigma-Delta Modulator Designs	Resource (Slices)	Timing (MHz)
3rd order 24-bit on Pilchard	1,721 / 3,072	23.523
5th order 24-bit on Pilchard	2,477 / 3,072	18.882
3rd order 24-bit on RC200	2,188 / 5,120	33.816
5th order 24-bit on RC200	3,167 / 5,120	27.485

4.1.2 Logic Analyzer

In order to debug the system in a noise free environment, we used an Agilent 16702B Logic analysis system to measure the output different bit-width and order settings. The state mode sampling method is chosen for synchronous sampling clocked by the FPGA board itself so that the exact output of the DAC can be captured by the logic analyzer. 1M of data thus acquired is then passed through a Hann window and an FFT used to display the result in the frequency domain. In figure 16, a simulation showing the SNR as a function of the input amplitude is given for the DAC implementation with 20kHz sine wave input and 24-bit input data input. For a sampling frequency of 96kHz with 64x oversampling. It can be seen that the maximum SNR is 139.8dB. The measured results for different configurations are shown in Figures 17 – 20. An Audio Precision System Two Cascade audio analyzer with -112dB THD+N for a 20kHz input will be used to further verify the system and results will be presented at the conference.

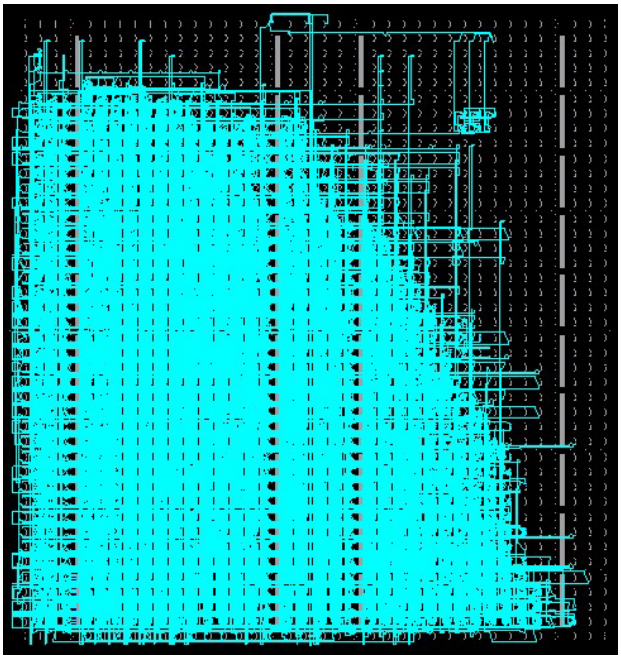

Figure 15. Place & Route Floorplan of the completed 5rd order design on Xilinx Virtex II XC2V1000, RC200

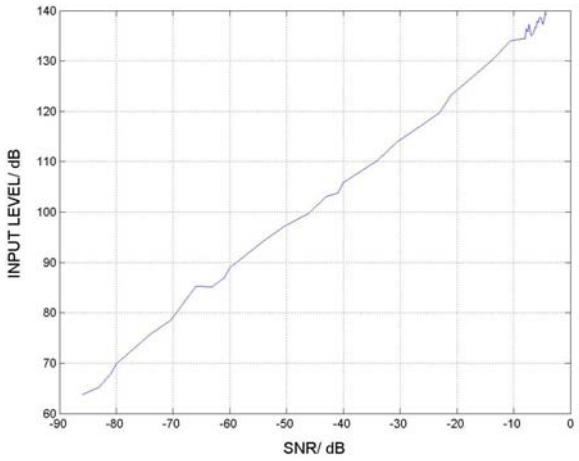

Figure 16. Simulation of the input level vs. SNR

The measured results show the noise shaping performed by the SDM. As expected, the higher order implementations provide a higher SNR. In our experiments, we adjust the input signal amplitude in order to achieve the highest SNR data. The input frequency was fixed at 20kHz for all results.

5. CONCLUDING REMARKS

In this paper, we have presented the design and implementation of a reconfigurable Sigma-Delta audio DAC on two different FPGA platforms: the RC200 and the Pilchard platforms. There are several parameters for designers to control such as the bit-width, the oversampling ratio, the operating clock rate and the order of the design. With different designs, we are able to achieve different SNR ratios for various audio applications. The maximum achievable SNR from our designs is around 170db. The reconfigurable platform is proved to be suitable for both digital designs and analog related devices such as the SDM in audio DACs.

ACKNOWLEDGEMENTS

This work was supported by the Department of Electronic Engineering and the Department of Computer Science & Engineering, The Chinese University of Hong Kong. We acknowledge the support of Celoxica and Xilinx.

REFERENCES

[1] T. Saramäki, “Design of FIR filters as a tapped cascaded interconnection of identical subfilters,”, *IEEE Transactions on Circuits and Systems*. Vol.34, pp.1011-1029, 1987.

[2] Steven R. Norsworthy and Ronald E. Crochiere, “Decimation and Interpolation for ΣΔ conversion”, in *Delta-Sigma Data Converters,*edited by S.R. Norsworthy, R. Schreier and G.C. Temes, IEEE Press, 1997.

[3] Richard Schreier, The Delta-Sigma Toolbox, http://www.mathworks.it/matlabcentral, File Exchange > Control and System Modeling > Control Design > delsig.

[4] R. Kress, A. Pyttel, and A. Sedlmeier, “FPGA-Based Prototyping for Product Definition”, Field-Programmable Logic and Applications (FPL), pp.78-86, 2000.

[5] P.H.W. Leong and M.P. Leong and O.Y.H. Cheung and T. Tung and C.M. Kwok and M.Y. Wong and K.H. Lee "Pilchard - A Reconfigurable Computing Platform with Memory Slot Interface", Proceedings of the IEEE Symposium on FCCM, 2001

[6] Celoxica. Inc. http://www.celoxica.com.

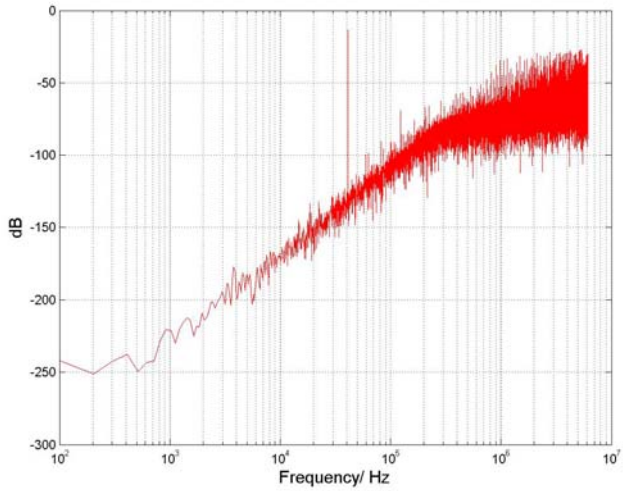

Figure 17: 3rd order, 24-bit SNR=98.2088dB, input=0.85, fin=20kHz, fs=96Khz, oversampling ratio=64x

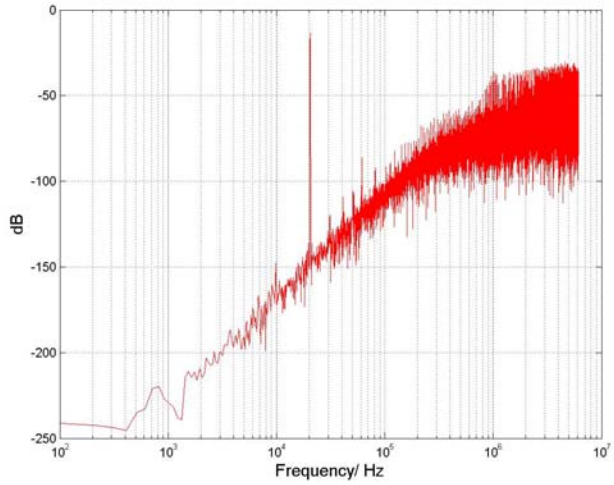

Figure 18: 3rd order, 24-bit, SNR=115.9667dB, input=0.85, fin=20kHz, fs=96Khz, oversampling ratio=128x

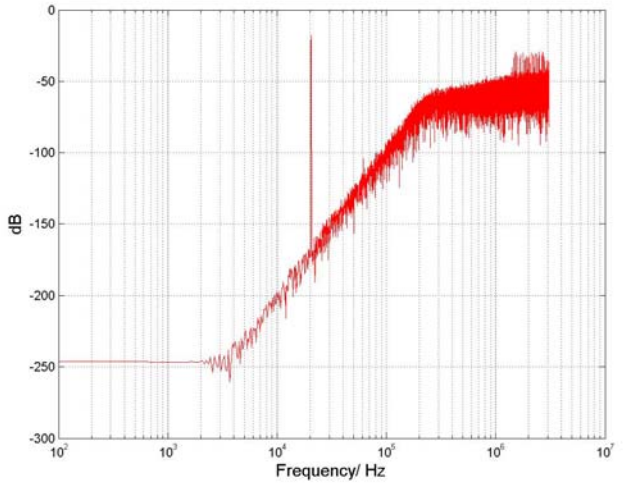

Figure 19: 5th order, 24-bit, SNR=139.2924dB, input=0.525, fin=20kHz, fs=96Khz, oversampling ratio=64x

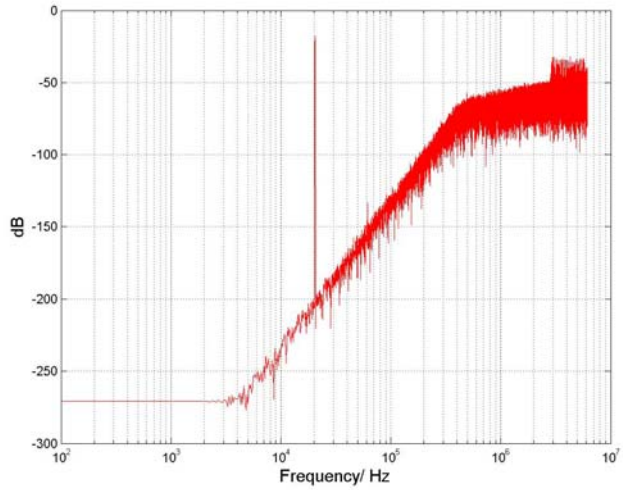

Figure 20: 5th order, 24-bit, SNR=171.8071dB, input=0.525, fin=20kHz, fs=96Khz, oversampling ratio=128x