A Sigma-Delta Digital Audio Power Amplifier — Design and FPGA Implementation

A.J Magrath, I.G. Clark and M.B. Sandler

Department of Electronic and Electrical Engineering
King's College London
Strand
London WC2R 2LS

Tel: +44 (0)171 873 2041
e-mail: ant@orion.eee.kcl.ac.uk

June 16, 1997

Abstract

We discuss the DSP design and implementation of a digital audio power amplifier using a FPGA. We concentrate on the sigma-delta modulator and associated bit-flipping architecture which is used to convert an oversampled PCM input signal into a 1-bit code suitable for controlling a power switch. We discuss details of FPGA architecture and the optimisations required for implementation.

1 Introduction

Digital power amplifiers (DPAs) are used to convert a PCM digital input signal to analogue directly at high power, without the use of intermediate analogue amplification. The block diagram of a typical DPA is shown in figure 1. A digital input signal is oversampled and converted to a one bit signal. The one bit signal controls a power switch comprising a MOSFET bridge, which reconstructs the one-bit signal at a higher voltage level. The resulting waveform is passed into an analogue passive low-pass filter which recontructs the analogue signal, whilst filtering the high frequency energy in the waveform. Applications include domestic and professional audio systems and portable equipment such as mobile telephony.

DPA systems which are currently under development utilise signal processing based upon either pulse-width modulation (see for example [4]) or sigma-delta modulation (SDM). The advantage of SDM over PWM is that the clock frequency used to control the pulse edges can be considerably reduced [2], resulting in an implementation with low clock frequencies and low power dissipation.

In this paper we concentrate on the design and implementation of a sigma-delta modulator suitable for power conversion. We first consider design and structural optimisations, then report on the FPGA implementation of the design.

2 Algorithmic Design and Structural Optimisations

2.1 Sigma-Delta Modulation and the Bit-flipping Algorithm

SDM systems are widely used in analogue-to-digital (A-D) and digital-to-analogue (D-A) converters to generate a single bit (two-level) code which represents the input signal in the low frequency portion of the spectrum. The modulator employs feedback around a one-bit quantizer to redistribute the quantization noise so that it is located at high frequencies outside the audio spectrum.

For DPA systems, the essential features of the SDM are that it should generate a one-bit code with high resolution in the baseband (> 16-bits for audio) and a low average pulse-repetition frequency (PRF). The baseband resolution is determined by the noise transfer function (refer to section 2.2) and the oversampling ratio (L). A low PRF is required to ensure that the power dissipation in the switching stage is kept to a minimum. SDMs generally produce an output code with a PRF which is dependent on the amplitude of the input signal amplitude [1], with the highest PRF produced at low amplitudes, with output patterns such as {1, -1, 1, -1...} or {1, 1, -1, -1, 1, 1, ...}. For oversampling ratios high enough to produce good baseband resolution (typically $L = 64$ for 16-bit resolution), the resulting PRF is too high for efficient conversion [1]. Furthermore the PRF is signal dependent and this can give rise to distortion and increased noise in the presence of a non-ideal output stage [2]. To overcome these problems we utilise the technique of *bit flipping* [3] to reduce and regulate the PRF. The concept of bit-flipping is illustrated in figures 2 and 3. The quantizer output is selectively inverted to reduce the transition rate in the output. Control logic is configured so that bit-flipping occurs when there is a transition between the previous and current quantizer output, and the PRF exceeds a target PRF F_t. It has been shown in [1] that the PRF control can be implemented as a counter which counts down on every sample and counts up N steps if a transition occurs between the current and previous output bits. A counter value greater than zero indicates the target rate is exceeded [1] and bit-flipping should occur. For oversampling ratio L and sampling rate F_s, the value of N is given by [1]:

$$N = \frac{LF_s}{2F_t} \tag{1}$$

For the present design, a target PRF of around $F_t = 350\ kHz$ is required (this has been found to yield low power dissipation in the output stage [4]), leading to a value $N = 4$ for parameters $L = 64$ and $f_s = 44.1\ kHz$.

The bit-flipping causes the baseband noise power to increase [3]; however this effect can be minimised by utilising an *alternation control*, which ensures that every $1 \rightarrow -1$ bit-flip is followed by a $-1 \rightarrow 1$ flip. This has the effect of shaping the error introduced by the bit-flipping so that it has a high-pass spectrum [5]. It has been shown in [1] that the alternation control can be implemented with a simple counter which counts the number of positive and negative bit-flips.

A condition can arise [1] where bit-flipping the current sample causes the transition rate in future samples to increase. As a consequence, the bit-flipping rate becomes excessively high and the stability of the modulator is jeopardised. To improve stability, the technique of *look-ahead* [2] is used. This predicts the influence of bit-flipping on the two next future quantizer output states and only allows bit-flipping if the number of transitions is minimised.

The overall structure of the bit-flipping sigma-delta modulator (BF-SDM) is shown in figure 5. The look-ahead block generates two outputs representing the next two quantizer states. These outputs, together with two outputs representing the current and previous quantizer states, are used in conjunction with the PRF and alternation controls to determine whether or not to flip the current quantizer output. For causality reasons [2], the bit-flipping must occur outside the loop, and compensation is made within the loop by modifying two internal loop filter states. This is implemented using combinational logic and two extra adders in the loop filter structure. Full details of the algorithm are omitted here due prevent repetition, and the reader is referred to [2] for more details.

The bit-flipper is implemented as a multiplexer, which outputs either the one-bit quantizer output, or its inverse, according to the state of the control input.

2.2 Loop filter Optimisations

The noise transfer function (NTF) defines the spectral shaping of the error introduced by the 1-bit quantizer and bit-flipper in the BF-SDM. For an input $X(z)$, loop filter $H(z)$ and combined quantizer and bit-flipping error $E(z)$, the output of the modulator is given by:

$$Y(z) = X(z)\frac{H(z)}{1+H(z)} + E(z)\frac{1}{1+H(z)} \tag{2}$$

The NTF is given by $NTF(z) = 1/(1 + H(z))$, and is designed as a high-pass filter to attenuate the error in the baseband.

Even with the alternation control and look-ahead, the baseband noise power in a BF-SDM is greater than a standard modulator [1]. To overcome this problem, a high order loop filter is required to reduce the level of quantization noise in the baseband. For the prototype design, a seventh order modulator is used at an oversampling ratio of 64. To reduce the hardware cost of this filter, the loop filter coefficients are chosen to have power-of-two values. This means that they can be implemented as shifters (i.e. data-bus alignment), therefore no physical multipliers are required. A cascade of resonator filter structure is used (figure 4), which has low sensitivity to coefficient quantization and requires a minimal number of coefficients.

2.2.1 Coefficient Optimisation

In the filter, the first integrator section implements the DC zero of the noise transfer function, and the next three integrator pairs implement the three complex conjugate zeros which are distributed across the baseband at locations detemined by the b_i coefficients. It can be shown that the noise transfer function of the filter is given by:

$$NTF(z) = \frac{(z-1)N_1N_2N_3}{(a_0 - 1 + z)N_1N_2N_3 + D_1N_2N_3 + D_2N_3 + D_3} \tag{3}$$

where

$$N_i(z) = (z-1)^2 + b_i \tag{4}$$
$$D_k(z) = a_{2k-1}(z-1) + a_{2k} \tag{5}$$

The complex zero locations on the z-plane (which are defined by the resonant frequencies of the structure) are given by:

$$z = 1 \pm j\sqrt{b_i} \tag{6}$$

Therefore to implement zeros at radian frequencies θ_i, we choose $b_i = tan^2\theta_i$. Optimal zero locations θ_i for minimal baseband noise are defined in [6], however it is not possible to simply quantize the resulting b_i to power-of two values, as this destroys the optimisation. Furthermore the zero locations also affect the poles (equation 3) therefore it is better to optimise the b_i and a_k coefficients together.

The a_k coefficients control the transition width of the filter and the high frequency quantization noise gain [7]. To ensure stability, the high frequency gain should be as low as possible [7]; however a low gain results in poles close to the baseband edge, which has the effect of increasing the baseband noise power.

To determine optimal coefficients for a_k and b_i, a bounded power-of-two coefficient search has been used (i.e every set of coefficients within certain bounds are evaluated). To evaluate the performance of each coefficient set, a two-stage procedure was used. In the first stage, candidate coefficient sets with baseband integral P_i below a certain target level were selected.

$$P_i = \frac{1}{\pi} \int_0^{\pi/L} |NTF(\theta)|^2 d\theta \tag{7}$$

The target level was chosen to maintain over 16-bit resolution in the baseband, using an approximate quantization error power of $\Delta^2/12$ [8] for a quantizer step size Δ. In practice, this error power is subject to considerable error due to the additional error introduced by the bit-flipping, and the nonlinearity of the bit-flipping and quantization processes. To deal with this problem, in the second stage the modulator was simulated at the maximum required input level with each of the candidate coefficient sets, and the SNR and stability measured directly. The modulator which exhibited the highest SNR and maintained stability was selected. The use of a two stage procedure considerably reduces the simulation time, in comparison to a full-simulation optimisation.

2.2.2 Integrator Structure and Wordlengths

The integrator structures in the filter of figure 4 are implemented as shown in figure 6. The integrators include multiplexers which are configured to reset the internal filter states if the internal signal levels become higher than a trigger level T. This configuration deals with instability, which can occur for higher order SDMs if the input level exceeds a certain threshold. The instability is characterised by high internal filter signal levels [9] and a failure of the modulator to code the input signal. The use of a local reset (rather than a global filter reset) reduces the routing overhead in the FPGA. The first two integrators I_1 and I_2 also include an extra adder which implements the look-ahead.

The number of bits required by the internal adders and delays depend on the resolution of the system and the number of overload bits that are required. Due to the feedback around the noise shaping filter, the quantization error introduced at each filter node is noise-shaped, with more severe noise shaping occurring at nodes closer to the quantizer. As a consequence, the filter wordlengths can reduce towards the quantizer without a severe increase in baseband noise power. The wordlengths were optimised heuristically using the Silage [10] bit-true simulation language, resulting in wordlengths ranging from 26-bits at the first integrator output (I_1) to 15-bits at the last integrator output (I_7). To prevent excessive internal signal levels, the integrator structure also incorporates shifters before each node which re-align the data-bus so that the maximum signal level produced during normal operating conditions is approximately constant throughout the filter. This prevents a wordlength build-up which can occur with high order cascade of resonator structures. It introduces a linear change in the filter transfer function which is compensated for by modifying the a_k and b_i coefficients.

3 Hardware Implementation and Optimisations

3.1 From SPDIF to 64Fs data

The interface between the CD player and the FPGA is implemented using standard components. The SPDIF serial data stream from the digital output of the CD player is decoded using a Crystal Semiconductors CS8412 to produce left-right serial PCM data, and then interpolated to $8F_s$ using a Burr Brown DF1700. The block diagram is shown in figure 7. For more detailed information the reader is refered to the manufacturers data books [11, 12].

The serial output from the DF1700 is fed to the FPGA where serial-to-parallel conversion is performed. This data is then interpolated eight times using a sample-and-hold and first order moving average filter (MAF) (figure 8). The transfer function of the MAF is:

$$H(z) = \frac{1}{8}\left(\frac{1-z^{-8}}{1-z^{-1}}\right) \tag{8}$$

The resulting data at $64F_s$, is then passed to the BF-SDM.

3.2 FPGA Architecture

The AT6000 series of FPGAs from ATMEL [13] lend themselves naturally to implementing complex DSP algorithms such as BF-SDMs. The FPGA is SRAM-based, is programmed at power up and is also reconfigurable on-the-fly. The device used in this application was the AT6010 which has a symmetrical grid of 80x80 cells. The grid is connected using a flexible bussing network of local and express busses (figure 9). Each cell can be configured to implement a vast range of combinatorial, register, tri-state, constant or routing based functions (figure 10). Local busses are used to pass information between cells in blocks of 8x8. If connections over a distance greater than 8 cells are required, then express busses are used. This combination of local communication, local and express busses makes the routing of complex systems such as this modulator possible.

3.3 Mono and Stereo Implementation

The structure of the FPGA is register-rich: each cell has a flip flop, and overall the AT6010 has 6400 registers. This allows a parallel data structure to be implemented, which is more straightforward to design than the bit serial structures used previously [14, 15]; and allows a much lower clock speed. Due to an intensive routing requirement, there was insufficient space to directly implement two mono modulators on one device, but a novel method requiring minimal space overheads was implemented, as follows. The incoming data is interleaved into alternate left, right samples, and then de-interleaved at the output. By replacing all single delays with double delays and operating the system at double its normal sampling frequency, the modulator effectively operates independently on the left and right data (figure 11). This technique is well suited to the FPGA architecture, since of all the components used in the modulator, the unit delay is among the smallest to implement on the FPGA, taking just one cell per bit. For $L = 64$, $F_s = 44.1\ kHz$, the FPGA clock rate is only $5.64\ MHz$.

3.4 FPGA Optimisation and Implementation

Some areas of the modulator produced a propagation delay greater than the cycle time at $128F_s$, therefore some optimisation was required. A VHDL implementation of the modulator was used in the hardware design and optimisation. This gave the opportunity to simulate the design in hardware form and to verify the hardware optimisations necessary for stereo implementation. From VHDL, the modulator design was converted to the Viewlogic schematic entry tool format and from here the ATMEL FPGA design tool FIGARO was able to extract the design and perform floor planning.

3.4.1 Hardware Optimisations

Several hardware optimisations were performed to reduce propagation delays in critical areas of the modulator. There were two main types of optimisation performed. The first optimisation reduced the propagation delay of the feedback circuit. The chain of adders in figure 4 gives rise to a propagation delay of $T_p = d(M + (n - 1))$, where d is the delay for a 1 bit adder, M is the number of bits in the (largest) first adder, and n is the number of data items to be added, giving $T_p = 32d$ for the structure shown ($M = 26$, $n = 7$). An alternative tree structure (figure 12) improves on this propagation delay by first adding the data in pairs, then adding these results in pairs. This is repeated until all data has been added. The propagation delay for the tree structure is $T_p = d(M + (\mathcal{R}\{log_2 n\} - 1)) = 28d$, where $\mathcal{R}$ represents the operation of rounding up to the next largest integer.

The second optimisation was used to reduce the propagation delay in the stereo implementation. By moving one of the two cascaded unit delays past a functional block, we are effectively pre-calculating each result in the alternate cycle so that it is ready at the start of the next cycle (refer to figure 13). This also has the advantage of reducing the hardware overhead of the double delays.

With these two types of optimisation in place, and some minor combinational synthesis of the modulator feedback, alternation and PRF controls, the propagation delay of the whole system was less than the $128F_s$ cycle time.

3.4.2 Floor Planning

It was apparent that the automatic place and route tools of FIGARO would not be capable of finding a suitable layout. To overcome this problem the serial to parallel converter and interpolation filter were manually routed to maximize silicon usage, releasing sufficient space to allow the routing tool to layout the main modulator automatically.

4 Performance

Bit-true simulations using the Silage language [10] and VHDL reveal that the modulator achieves over 18-bit resolution at an average PRF of 352.8 kHz. The baseband response of the modulator with a 1 kHz sinewave at in input level of 0.3 is shown in figure 14. The performance of the implemented modulator shows slight degradation due to analogue post-processing of the bit-stream. This is due to jitter in the pulse edges, mismatched rise and fall times in the output switch [2] and nonlinearity in the output filter.

5 Conclusion

This paper has reported on the design and implementation of a power DAC using sigma-delta modulation. The modulator uses the technique of bit-flipping, which lowers the pulse-repetition frequency of the bitstream. The control algorithms required to achieve stable operation and maintain a high SNR have been described. The modulator loop-filter has been designed using a two-stage power-of-two coefficient optimisation, and this has been followed by wordlength optimisation to minimise the adder and delay complexity, whilst maintaining 18-bits resolution. For the FPGA implementation, a parallel data structure has been used, resulting in a very low clock rate (5.64 MHz), and a novel interleaving technique has been used to implement the stereo modulator with minimal hardware overheads. Some of the optimisations required to minimise the propagation delay and alleviate routing difficulties have also been described.

6 Acknowledgements

The authors would like to acknowledge the support of EPSRC grant GR/H 78702, and are indebted to ATMEL for donating the hardware development board, FPGA devices and software. The authors would also like to express gratitude to Gordon Pocock of Insight Technology for his advise on FPGAs and FIGARO.

References

[1] A.J. Magrath. *Algorithms and Architectures for High Resolution Sigma-Delta Converters*. PhD thesis, King's College, University of London, October 1996.

[2] A.J. Magrath and M.B. Sandler. Digital Power Amplification using Sigma Delta Modulation and Bit Flipping. *Journal of the Audio Engineering Society*, June 1997.

[3] A.J. Magrath and M.B. Sandler. Power digital-to-analogue conversion using a sigma-delta modulator with controlled limit cycles. *Electronic Letters*, 31(4), February 1995.

[4] M.S. Pedersen. All digital power amplifier based on pulse width modulation. *Presented at the 96th Convention of the Audio Engineering Society, Amsterdam*, Preprint 3809, February 1994.

[5] A.J. Magrath and M.B. Sandler. Digital-domain Dithering of Sigma Delta Modulators using Bit-Flipping. Journal of the Audio Engineering Society, June 1997.

[6] R. Schreier. An Empirical Study of High-Order Single-Bit Delta-Sigma Modulators. *IEEE Transactions on Circuits and Systems*, CAS-40(8):461–466, August 1990.

[7] R. W. Adams, P. F. Ferguson, Jr., A. Ganesan, S. Vincelette, A. Volpe, and R. Libert. Theory and practical implementation of a fifth-order sigma-delta A/D converter. *Journal of the Audio Engineering Society*, 39:515–528, July-August 1991.

[8] K.C.H. Chao, S Nadeem, W.L. Lee and C.G. Sodini. A Higher Order Topology for Interpolative Modulators for Oversampling A/D Converters *IEEE Transactions on Circuits and Systems*, CAS-37(3):309–318, March 1990.

[9] S. Hein. On the stablility of sigma-delta modulators. *IEEE Transactions on Signal Processing*, SP-41(7):2322–2348, July 1993.

[10] L. Nachtergaele, I. Bolsens, and H. De Man. Specification and simulation front-end for hardware synthesis of digital signal processing applications. *International Journal in Computer Simulation*, Vol. 2, 1992.

[11] Crystal Semiconductor Corp. Data Book Vol 1. Analogue/Digital Conversion ICs. DADBS.2, 1992.

[12] Burr Brown Corp. IC Data Book. Data Conversion Products. LI-465, 1995.

[13] ATMEL corp. Configurable Logic Data Book. 0266E-8/95/50m, August 1995.

[14] L. Risbo. FPGA based 32 times oversampling 8th order sigma-delta audio DAC. *Presented at the 96th Convention of the Audio Engineering Society, Amsterdam*, Preprint 3808, February 1994.

[15] J. Isoaho, H. Tenhunen, J. Heikkila, and L. Lipasti. High resolution DAC design based on FPGAs. In *Proc. Oxford International Workshop on FPGA Logic and Applications*, pages 343–352, September 1991.

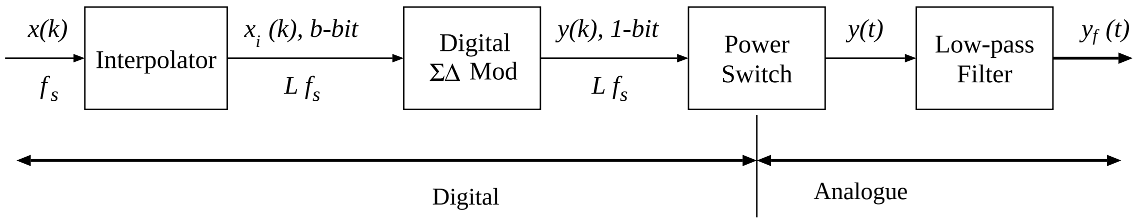


Figure 1: Sigma-Delta Modulator Power D-A Converter.

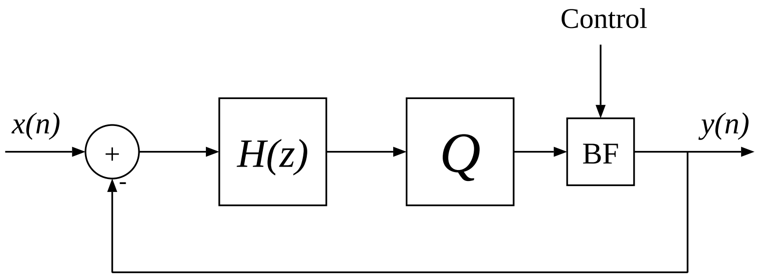


Figure 2: General Block Diagram of Bit-Flipping Sigma-Delta Modulator.

1 -1 1 -1 1 -1 1 1 -1 -1 1 -1

1 1 1 1 1 -1 -1 1 1 -1 -1 -1

Figure 3: Transition reduction using bit-flipping. The upper sequence is bit-flipped to produce the lower sequence.

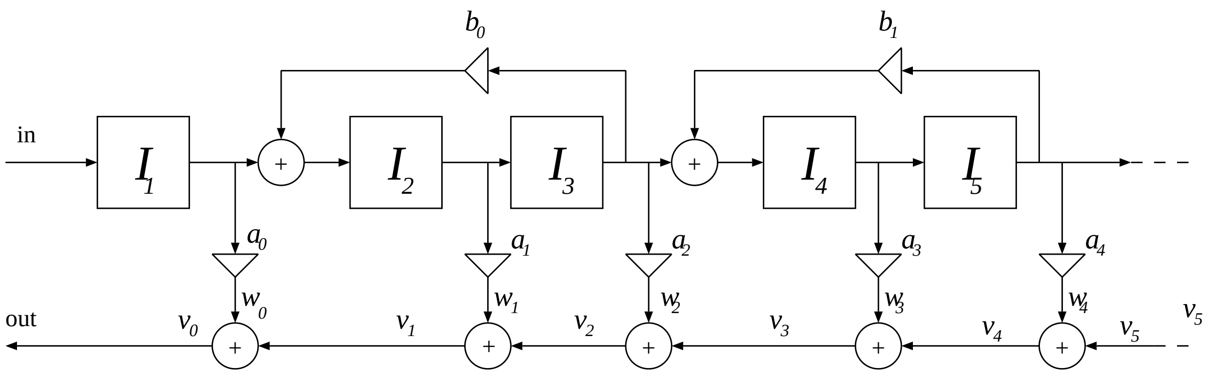


Figure 4: $H(z)$ Loop filter Structure. I = integrator.

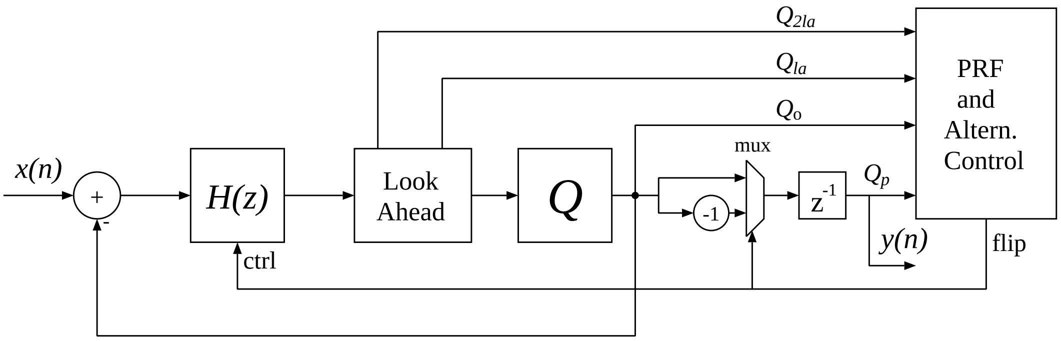


Figure 5: Complete Block Diagram of BF-SDM.

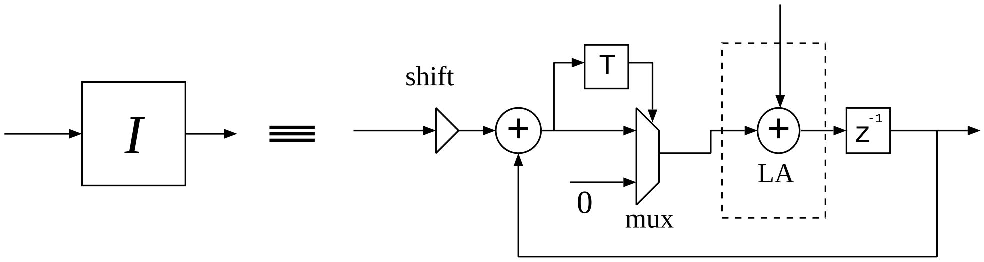


Figure 6: Integrator Structure. The look-ahead adder (LA) is present only in I_1 and I_2.

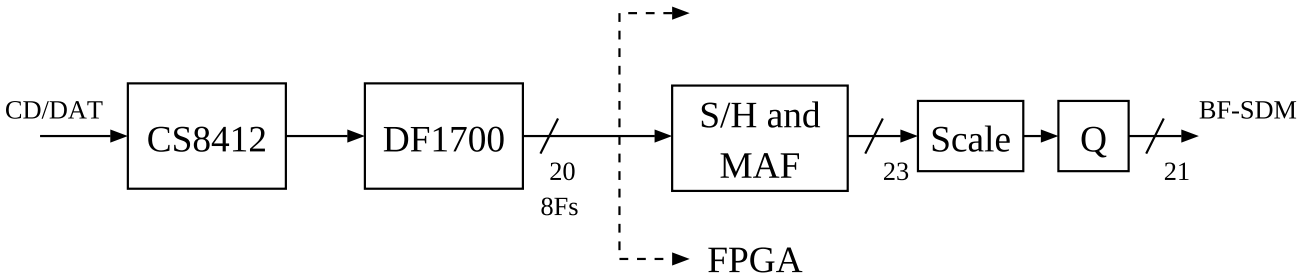


Figure 7: Block Diagram of Input section

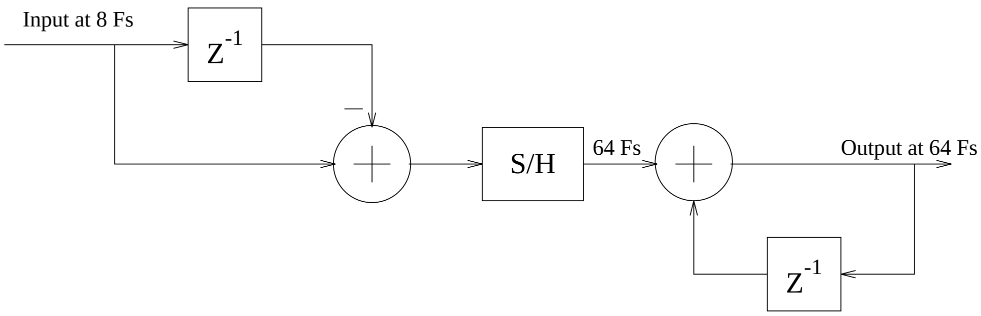


Figure 8: Moving average interpolation filter

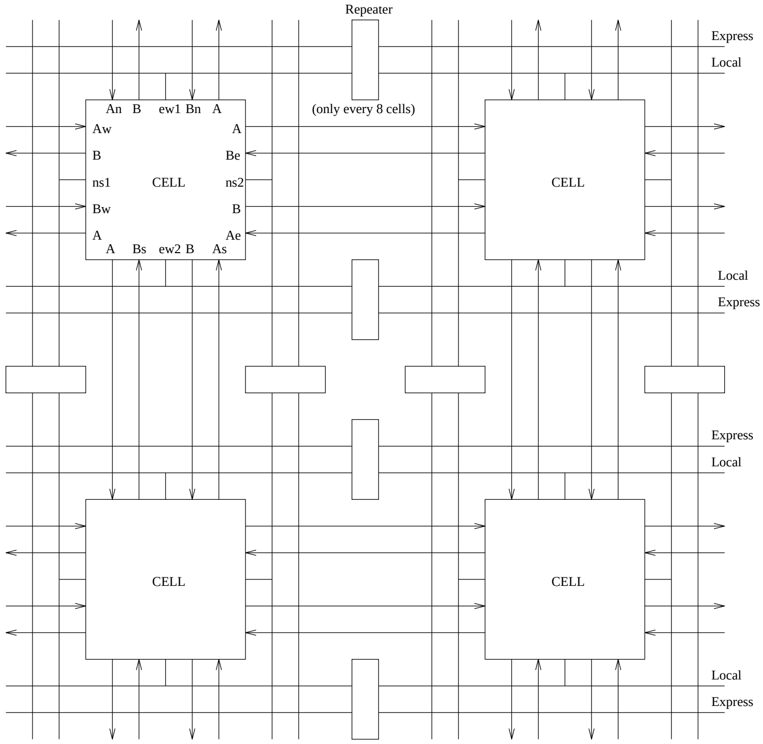


Figure 9: Cell Connections and Busses

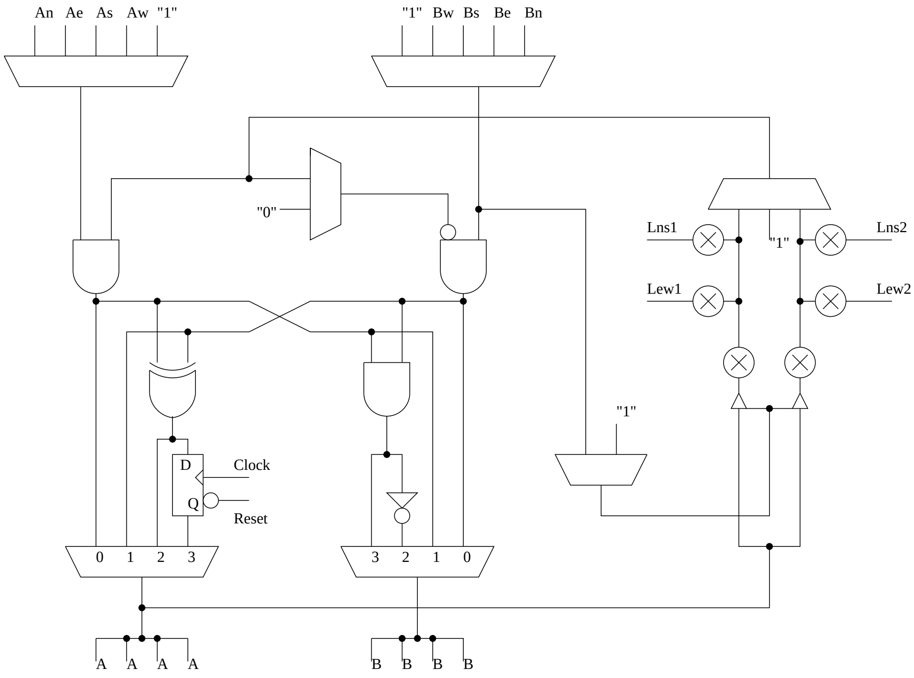


Figure 10: ATMEL FPGA cell layout

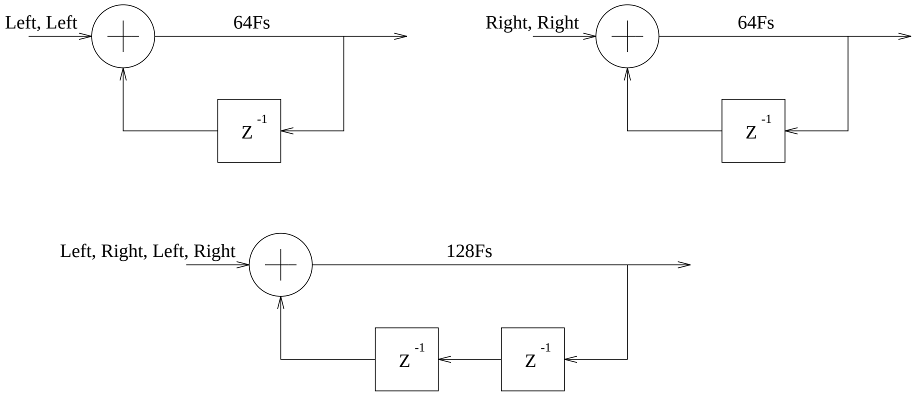


Figure 11: Interleaving Samples for Stereo Implementation (showing a simplified integrator)

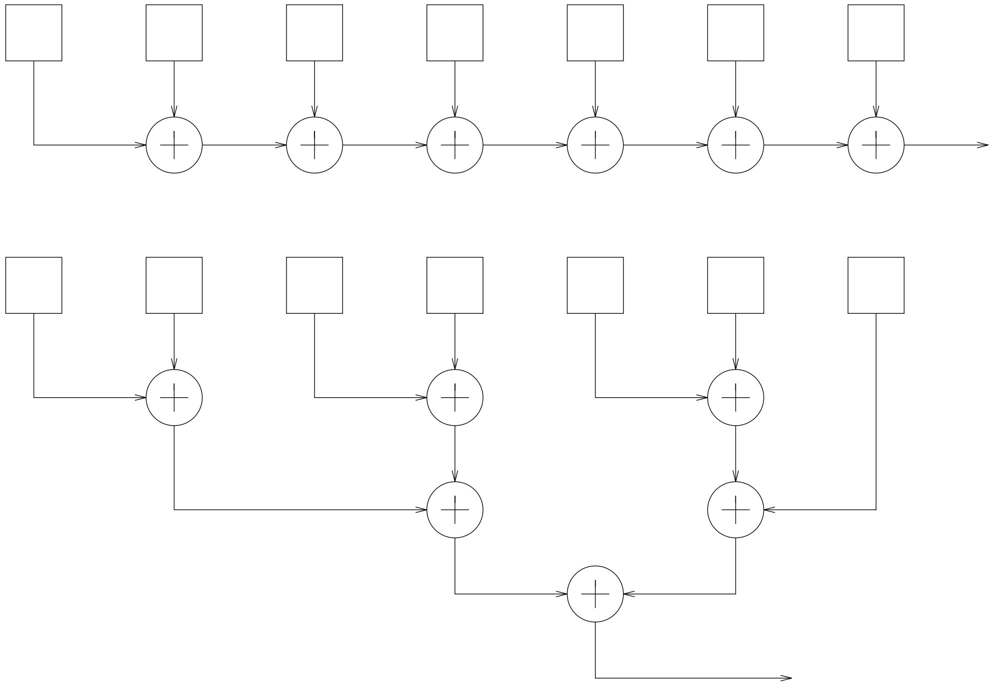
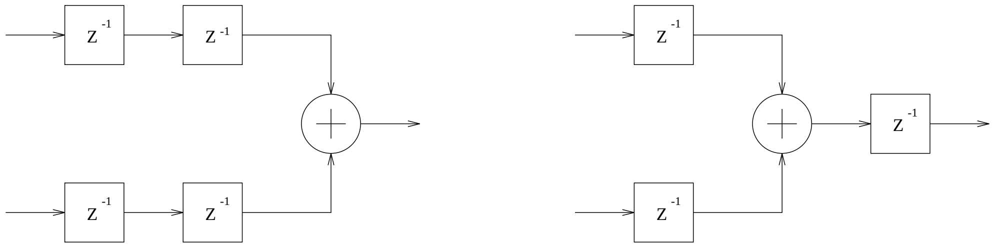


Figure 12: Conversion of cascaded adders to tree structure



(a)

(b)

Figure 13: (a) Cascaded delays before function block. (b) Pre-calculation in Previous Cycle.

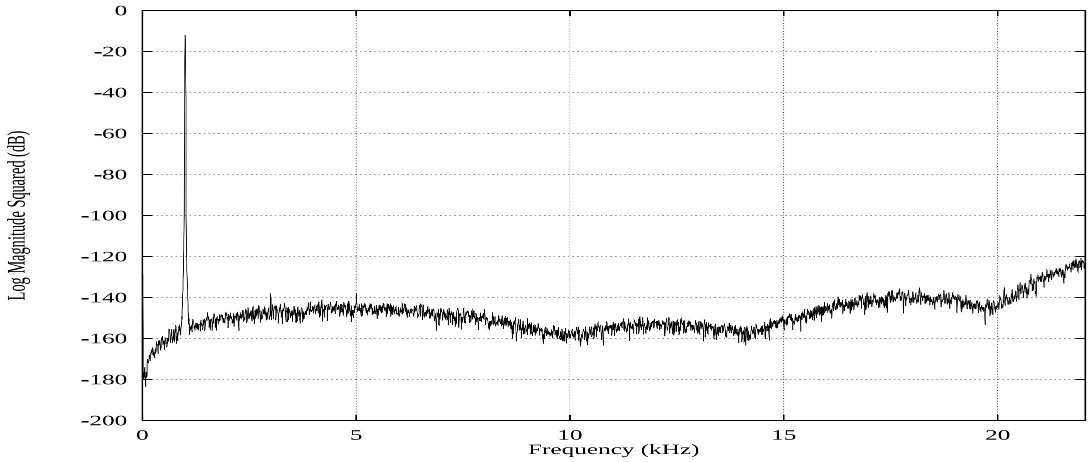


Figure 14: Baseband FFT of Bit-true Simulation