Digital Audio Power Amplifier

Nelson Mão de Ferro, Gonçalo Tavares, Moisés Piedade

Abstract—The implementation of click modulation is very computationally demanding. The purpose of this work is to validate this technique using one DSP and one FPGA.

The DSP makes all the click modulation computing. It computes the signal zeros by linear interpolation. The FPGA generates the binary PWM signal (with 9 bits of resolution) using the signal zeros calculated by the DSP.

Index Terms—Click Modulation, Amplitude Exponencial Modulation, Zero Finding, Infinite Clipping.

I. INTRODUCTION

THE purpose of the click modulation algorithm is sketched in Figure 1. The frequencies $f_{PWM}'$ and $f_{click}'$ are the switching frequency of the power stage. The purpose of using click modulation is to have a much smaller switching frequency.

Choosing the DSP was not a very easy task. Firstly a TMS320C6713 DSP from Texas Instruments was used. Despite its power for floating point computation, it was unable to compute the algorithm in real time, so the final choice was the TMS320C6416 @ 1GHz. With this DSP, all computing is done with 16 bit fixed point arithmetic.

To generate the binary signal in real time, a FPGA (Field-programmable gate array), SPARTAN3 from XILINX is used. The resulting binary signal is similar to PWM signal but with a different frequency spectrum as we can see in Figure 1.

The development boards used in this work are the DSK6416 for the DSP and the Spartan-3 Starter Kit for the FPGA.

II. IMPLEMENTATION

First the filters must be designed. They are designed in MATLAB using the FDAtool and quantization in Q15. According to some tests with the DSP the following coefficients numbers were found:

- 1024 for Hilbert Transform
- 64 for the PolyPhase LowPass filters
- 256 for the LowPass filter

Decreasing the number of coefficients, will lead to larger oscillations in the bandpass of the Hilbert Transform and filters and will decrease the attenuation of the LowPass filters.

For the power stage, two amplifiers TAS5261 for stereo were built and used. This power stage was built around the TAS5518-5261K2EVM from Texas Instruments. An analog audio input was included in this board.
Fig. 4. Magnitude response of low pass filter with LMS method, Q15 quantization.

The FPGA runs at approximately 180 MHz to generate the binary signal with 9 bit of resolution.

The power stage delivers about 150 Watt with a load of 8 Ω. A 5th order analog low pass filter was added to the end of the power stage substituting the original 2nd order filter. This new filter with so high order was necessary as we will see in section IV.

III. Algorithm codification

A. DSP

The key to use efficiently all the processing power of this DSP is to make sure the pipeline is well used and all functional units are used as much as possible in parallel using circular buffering. All that is only possible by programming the filters routine in assembly language.

The CODEC in this development board is used to sample the original analog signal. The sample rate is determined by the timer0 of the DSP which gives a sample rate of 44.138kHz. Just before AEM an over-sampling of 8 times is performed. Once again, this is synchronized with the same timer.

After the over-sampling, polyphase low pass filters are used. The oscillators are made with two simple tables with seven elements to give a frequency of 50.4kHz, and this is the fundamental frequency of the power stage commutation. Linear interpolation is used to compute the signal zeros localization. As this is calculated using over-sampling, the calculation is somewhat precise.

B. FPGA

The FPGA does the job of infinite clipping. Unfortunately at the over-sampling rate of the DSP, the resolution of the binary signal produced by the FPGA is only 9 bit because at this rate and resolution, the work frequency of the FPGA is already at 180MHz.

A PLL was implemented in the FPGA, three FIFO’s and a delay clock for the FIFO’s. And all this because there is jitter in the data from the DSP.

IV. Hardware

The power amplifier board was designed with the TAS5261 from Texas Instruments. The board has an S/PDIF and analog input. The first is through TAS5518 and the second trough PCM1802 and TAS5010, all from Texas Instruments.

As the main goal of this algorithm is to use low commutation frequency for the amplifier stage, the analog low pass filter had to be a 5th order. In this way the harmonics of the power stage do not disturb the audio signal. On the second harmonic of the switching frequency the filter has an attenuation of about 52dB.

As the power amplifier is a full bridge configuration (figure 5), the filter must be a differential filter as we can see in figure 6. The filter was designed with a Butterworth implementation so we can get a flat amplitude response as much as possible.

Fig. 5. Full bridge Class D amplifier.

Fig. 6. 5th order Butterworth differential low pass filter circuit.

An S/PDIF input board was designed (but not constructed) to bypass the CODEC from DSK6416 board. The CODEC has only 16 bit to quantize samples. With S/PDIF, 24 bit can be used which can increase the quality of the output of this implementation.

A simulated loudspeaker was made to test the amplifier which the schematic can be found at www.stereophile.com.
V. RESULTS

A. Amplifier frequency response

The results are somewhat disappointing. Neither the results from Analog/PCM/PWM nor the results from DSP/FPGA are good. But the first is better than the last. We can see from figures 9, 8, 11 and 10 the amplitude response from the analog input or by the DSP/FPGA is the same with the resistive load or with the simulated speaker. The phase response is different from both and is distorted when the input comes from DSP/FPGA.

B. Amplifier distortion

In the distortion the same behavior occurs. The analog input is not good enough and the click modulation implementation is worse than analog input results as we can see in figures 12, 13, 14 and 15.

VI. CONCLUSION

We can not get any real conclusions because as we saw on section V the algorithm implementation is not working properly. The distortion increases as we increase the input signal frequency.

The problem for those bad results was not found and all debug points had what was expected, specially in the DSP code.

At this point, it is not difficult to make this implementation work with some extra time effort.

Finally we have some ideas for future implementation for this algorithm to have better quality. The first thing is to increase the binary signal resolution in the FPGA. To do this we must do down sampling in the DSP, but as down sampling will make data loss and make the linear zero finding less accurate, we need another DSP to make cubic interpolation and this way we can have good zero guessing and increase the number of bits in the FPGA. To be more specific, the first DSP should make all the calculation until zero finding with digital input (e.g. SPDIF), the second DSP connected in series with the first can make
the cubic interpolation. At last the FPGA generates the binary signal with 11 bits against 9 bits if a down-sampling of 4 is used.

References