The VLSI design of the sub-band filterbank in MP3 decoding

Jia-Xin Liu and Li Luo

School of Electronic and Information Engineering, Beijing Jiaotong University
Beijing, China

Corresponding author's e-mail: 15120018@bjtu.edu.cn

Abstract. The sub-band filterbank is one of the most important modules which has the largest amount of calculation in MP3 decoding. In order to save CPU resources and integrate the sub-band filterbank part into MP3 IP core, the hardware circuit of the sub-band filterbank module is designed in this paper. A fast algorithm suit for hardware implementation is proposed and achieved on FPGA development board. The results show that the sub-band filterbank function is correct in the case of using very few registers and the amount of calculation and ROM resources are reduced greatly.

1. Introduction

MP3, proposed by MPEG, is a kind of audio format with lossy compression [1]. MP3 has been widely used so far because of its high compression ratio and sound quality [2]. The sub-band synthesis filterbank is the final step of MP3 decoding, but also the most critical step. The data after IMDCT transformation can be regarded as approximately spectrum representation of the time sample values. These data will completely transform back to the time domain after the sub-band synthesis filterbank. And the original PCM samples are obtained. The calculated amount of the sub-band filterbank is very large, accounting for more than 60% of the calculation in MP3 decoding [3].

Until now, the technology of MP3 decoding is pretty mature. Currently, general structure is achieving MP3 software decoding based on DSP or RISC processor. This method is very flexible and can support a variety of audio format decoding. However, it is easier to implement but cost higher if floating-point DSP is adopted. And it is more cost-effective but harder to achieve if fixed-point DSP is utilized [4]. There are also some parallel decoding methods proposed to decrease the decoding time [5], but a lot of resources are wasted because more than one CPU is needed. Comparatively speaking, using hardware to achieve is faster and has higher efficiency and lower power consumption. In conclusion, using hardware circuit to achieve is more suitable for the sub-band filterbank module, which has a large amount of calculation. And hardware implementation can save CPU resources, hence CPU can only control other modules. Besides, the sub-band filterbank module can be integrated into IP core, and reusable in other hardware design and SOC applications. It cannot be denied that it is not flexible if hardware method is used, but compared to the advantages, achieving the sub-band filterbank in hardware method is a domain worthy of research and development.

In this paper, a fast algorithm of the sub-band filterbank is introduced, which can reduce the amount of calculation and memory room of the module effectively. And the hardware implementation of the function is given. The module is simulated in Modelsim software, and the results are compared with the decoding results using C language. And the function is implemented on FPGA development board.

2. Another section of your paper

The process of the sub-band filterbank can be divided into three parts: matrix transformation, windowing and accumulation. The first step is the matrix transformation. This process is similar to the transformation of IMDCT. Its computational formula is shown below.

$$V_i = \sum_{k=0}^{31} X_k \cos\left(\frac{\pi}{64}(16+i)(2k+1)\right) \quad i = 0 \dots 63 \tag{1}$$

The parameter X_k represents 32 inputs taken from each sub-band, and 64 output values V_i are generated. Then the outputs are stored in a FIFO with depth of 1024. And then extract 512 sample values among them, written as U_i, the extraction formula is shown in (2) and (3).

$$U[64i+j] = V[128i+j] \tag{2}$$

$$U[64i+32+j] = V[128i+96+j] \quad i = 0 \dots 7, j = 0 \dots 31 \tag{3}$$

Then windowing transformed U_i, the formula is shown in (4). D[i] is window function, which is already given in MP3 decoding standard.

$$W[i] = U[i]D[i] \quad i = 0 \dots 511 \tag{4}$$

The last step is accumulation. The 512 W vectors after windowing can be arranged to 16 lines according to each line of 32 data. The original PCM data can be obtained by stacking together the 16 W vectors of each column. And finally, 32 PCM audio data are obtained in all. The accumulation formula is shown below.

$$PCM[j] = \sum_{i=0}^{15} W[j+32i] \quad j = 0 \dots 31 \tag{5}$$

3. Improvement of the algorithm

The calculated amount of the sub-band filterbank is very large, and mainly concentrated in the matrix transformation part. So in order to reduce the amount of calculation, this part should be taken into consideration first. There are many fast algorithms of the matrix transformation [6-7], but most of them use a lot of multiplications and divisions. Hardware implementation isn't suitable for these algorithms because of the restrictions on accuracy of the calculation and the characteristics of the circuits and etc. Besides, the multiply-accumulator structure can also be used [8-9], but this structure will cost large chip area, a lot of hardware resources are wasted and the power consumption is very high, so it is not suitable either. So in order to simplify the matrix transformation formula, the symmetry of cosine function should be taken full advantage of. First, (1) has symmetry, the cosine function in matrix transformation is written as $C_{i,k}$, so:

$$\begin{aligned} C_{32-i,k} &= \cos\left(\frac{\pi}{64}(48-i)(2k+1)\right) \\ &= \cos\left(\left(\frac{\pi}{64}i - \frac{3\pi}{4}\right)(2k+1)\right) \\ &= -\cos\left(\left(\frac{\pi}{64}i + \frac{\pi}{4}\right)(2k+1)\right) \\ &= -\cos\left(\frac{\pi}{64}(i+16)(2k+1)\right) \\ &= -C_{i,k} \quad i = 0 \sim 15 \end{aligned} \tag{6}$$

Similarly, when i varies from 33 to 47, $C_{96-i,k} = C_{i,k}$, when i equals to 16, $C_{i,k} = 0$, when i equals to 48, $C_{i,k}$=1. Then take the cosine function back into matrix transformation formula, and we can get:

$$\begin{aligned} V_{32-i} &= -V_i \quad i = 0\sim15 \\ V_{96-i} &= V_i \quad i = 33\sim47 \\ V_i &= 0 \quad i = 16 \\ V_i &= \sum_{k=0}^{31} X_k \quad i = 48 \end{aligned} \tag{7}$$

Hence, in the calculation of a sub-band, only 32 V[i] vectors need to be calculated. The other data can be obtained directly or negatively. And the calculated amount is reduced to half. In the remaining 32 V[i] vectors, continue to observe the symmetry of the cosine function in each data that needs to be calculated. When i is fixed, we can notice that:

$$\begin{aligned} C_{i,k} &= \cos\left(\frac{\pi}{64}(16+i)(2k+1)\right) \\ C_{i,31-k} &= \cos\left(\frac{\pi}{64}(16+i)(63-2k)\right) \\ &= \cos\left((16+i)\pi - \frac{\pi}{64}(16+i)(2k+1)\right) \end{aligned} \tag{8}$$

So, when i is odd, we can get $C_{i,k}$ =-$C_{i,31-k}$， $i = 0\sim15$, according to the symmetry of cosine function. And when i is even, $C_{i,k} = C_{i,31-k}$， $i = 0\sim15$. So, in all 32 inputs, the k^{th} and the 31-k^{th} inputs can be extracted together. First add or subtract according to the parity of i. Then multiply with cosine function. In summary, the matrix transformation formula in the sub-band filterbank can be written as (7) and (9):

$$\begin{aligned} V_i &= \sum_{k=0}^{15} (X_k + X_{31-k}) \cos\left(\frac{\pi}{64}(16+i)(2k+1)\right) \quad \text{i is even} \\ V_i &= \sum_{k=0}^{15} (X_k - X_{31-k}) \cos\left(\frac{\pi}{64}(16+i)(2k+1)\right) \end{aligned} \tag{9}$$

i is odd

In terms of calculated amount, if the original algorithm is used, 64*32=2048 times multiplication and 64*31=1984 times addition need to be done in the calculation of a sub-band. And if the improved algorithm is used, only 32*16=512 times multiplication and 32*(16+31)=1504 times addition need to be done. It is very convenient to implement by hardware while the calculated amount is reduced greatly. Besides, in terms of ROM resource consumption, if the original algorithm is used, 64*32=2048 cosine values should be stored. And only 32*16=512 cosine values need to be stored if the improved algorithm is used, saving 3/4 of storage space. And ROM resource consumption is reduced greatly.

4. Hardware design

4.1. Design of top module

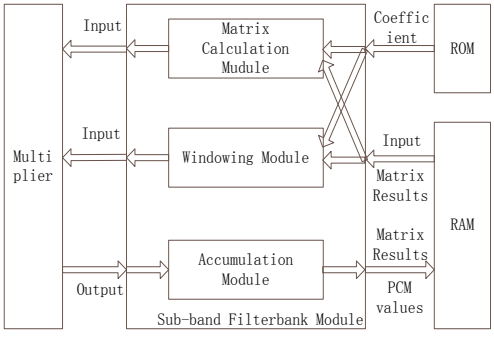


Figure 1. Design of top module of the sub-band filterbank

The design of top module of the sub-band filterbank is shown in Figure 2. In order to complete the function, a multiplier, a ROM, a RAM and a sub-band filterbank module is needed. Among them, input of the multiplier is 20-bit fixed-point number. The most significant bit is the sign bit. The remaining 19 bits represent the fractional part. Output of the multiplier is 40-bit number, the second highest bit is the sign bit and the subsequent 19-bit data represents the result of the multiplier. The simplified cosine function values and the window function values are stored in ROM. The decoding results of IMDCT are stored in RAM, for the use of matrix calculation. Meanwhile, 1024-long FIFO is also developed in RAM. The design of the sub-band filterbank module is divided into three parts. The matrix calculation module reads the data in RAM and the function values in ROM and then sends them to the multiplier. The results of the matrix calculation can be obtained by accumulating the outputs of the multiplier, and stored in the developed FIFO. The windowing module reads the results of the matrix calculation and window function values according to a specific law, and sends them to the multiplier. The PCM values are obtained after accumulating and then written to RAM.

4.2. Timing control of matrix transformation

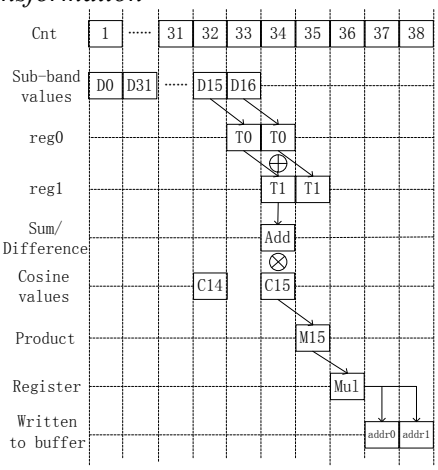


Figure 2. Timing control of matrix transformation

Timing control of matrix transformation is shown in Figure 3. When the counter value varies from 1 to 32, the addresses of k^{th} and 31-k^{th} input data are given separately and the data is read out in the next clock cycle. Then in the following two clock cycles, the two values are given to two registers T0 and T1. Using combinational logic, add or subtract the two values according to the parity of i. The sum or difference is written as Add. Add multiplied with cosine function value C15 extracted from ROM and a matrix transformation result M15 is obtained. Then in the next clock cycle, the value M15 is given to register Mul. If the bit after 20 valid bits equals to 1, add 1 to its valid bit. If the bit equals to 0, just take the valid bit. In the next two clock cycles, write Mul or its negated values into two corresponding addresses respectively. At this point, the calculation of a sub-band is completed, and a granule needs 18 sets of such calculation.

The matrix transformation module owns the largest calculated amount in the entire sub-band filterbank. It can be seen from the timing control diagram that, on the premise of ensuring the correct decoding results, the symmetry of cosine function is taken full advantage of, in the same clock cycles, the amount of multiplication and addition as well as read and write operations of ROM and RAM are greatly reduced. Two matrix transformation values are decoded at one time, so the decoding speed is improved and the decoding power consumption is reduced.
Some text.

4.3. Design of FIFO

After matrix calculation, the obtained 64 outputs need to be written to a 1024-long FIFO. Then extract the i^{th}(i=0-31) of the first 32 data and the last 32 data every 128 data, multiplied with window function values and accumulate, then PCM audio samples are obtained [10]. In order to save resources and

make sure that no more FIFO is called, 1024-long space is opened up in RAM, and it is used as a FIFO through controlling the addresses. The sketch map is shown in Figure 4.

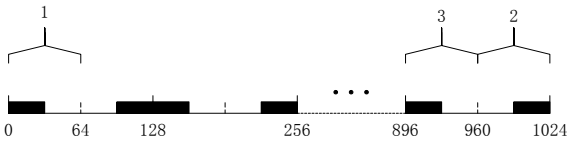


Figure 3. Design of FIFO

The first set of matrix transformation results is written to the address 1 shown in the figure above. First extract the i^{th} data and i^{th} window function value, then extract a data every 96,32,96,32… addresses and extract a window function value every 32 addresses, multiply and accumulate 16 sets of data and the PCM values are obtained. The second set of matrix transformation results is written to the address 2. From address 960, extract data, multiply and accumulate according to the law mentioned above. All the PCM data can be obtained so.
Some text.

4.4. Simulation results and comparison

After the hardware design of the sub-band filterbank part is completed, run simulation with Modelsim. The original file used for the simulation is a MP3 file of 2 seconds. After simulation, the obtained PCM audio values are written to a new file. The PCM audio data can be played by writing a file header of wav format in the front of the new file. After verification, the playing effect of PCM audio values is consistent with the original MP3 file. In order to show the results of simulation more intuitively, the software Cool Edit can be used to view the waveform of wav files. The waveform is shown in figure below.

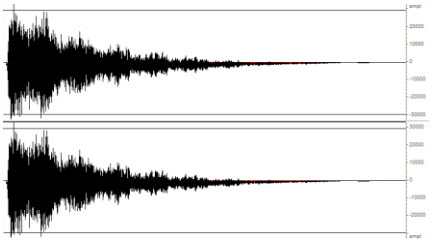


Figure 4. Waveform decoded by C language

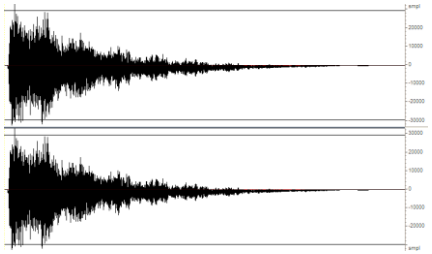
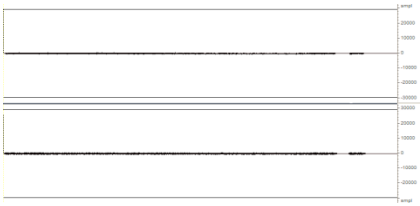


Figure 5. Waveform decoded by Verilog language



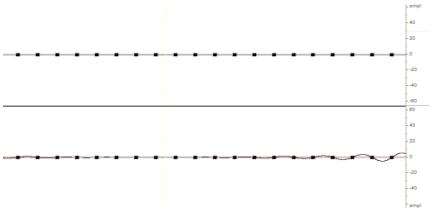


Figure 6. Waveform of the difference between two decoding results

Figure 6 shows the waveform of the results decoded by C language, the two lines are the waveforms of left and right channels. Figure 7 shows the waveforms of the two channels decoded by Verilog language. And figure 8 shows the difference between two decoding results. From the three figures above, we can see that the waveforms obtained in two different ways are basically the same, the peak is more than 30000. And the difference is very small, almost negligible. So the hardware design of the sub-band filterbank is correct.

Besides, the module is implemented on FPGA development board DE2-115 of ALTERA company. The results of analysis and synthesis and power analysis are shown in Figure 9.

Revision Name	Filterbank
Top-level Entity Name	Filterbank
Family	Cyclone IV E
Total logic elements	524
Total combinational functions	464
Dedicated logic registers	236
Total registers	236

	Fmax	Restricted Fmax	Clock Name	Note
1	173.82 MHz	173.82 MHz	Clk	

Revision Name	Filterbank
Top-level Entity Name	Filterbank
Family	Cyclone IV E
Device	EP4CE115F29C8
Power Models	Final
Total Thermal Power Dissipation	164.38 mW
Core Dynamic Thermal Power Dissipation	1.26 mW
Core Static Thermal Power Dissipation	98.60 mW
I/O Thermal Power Dissipation	64.51 mW

Figure 7. Analysis results in Quartus II

From the figure above, the sub-band filterbank module is implemented in the case of using only 524 logic units, saving a lot of resources. And the maximum clock frequency can reach 173.82MHz, completely meeting the timing sequence requirements of MP3 decoding. The input motivation of the module is the output results of the previous module IMDCT in MP3 decoding, the power consumption of the sub-band filterbank module is 164.38mW in the case of typical mode and clock frequency of 10MHz.

5. Conclusion

The algorithm of the sub-band filterbank in MP3 decoding is studied in this paper, and it is simplified by taking full advantage of the symmetry of cosine function. And the hardware design of the sub-band filterbank is given according to the characteristics of hardware circuit. The calculated amount and memory resources are reduced greatly on the premise of ensuring the decoding results are correct. Besides, the implementation of the function on development board DE2-115 of ALTERA company is given in this paper. The results show that, the function is realized correctly in the case of using very few registers.

6. Acknowledgment

This work was supported by National Natural Science Foundation of China (U1431119).

References

[1] S. Shlien, "Guide to MPEG-1 audio standard," in *IEEE Transactions on Broadcasting*, vol. 40, no. 4, pp. 206-218, Dec 1994.

[2] H. G. Musmann, "Genesis of the MP3 audio coding standard," in *IEEE Transactions on Consumer Electronics*, vol. 52, no. 3, pp. 1043-1049, Aug. 2006.
[3] B. P. Gangamamba, N. S. Murthy and P. Muralidhar, "Low power reconfigurable sub -band filter bank ASIC for MP3 decoder," 2008 International Conference on Electronic Design, Penang, 2008, pp. 1-5.
[4] Jing Chen and Heng-Ming Tai, "Real-time implementation of the MPEG-2 audio codec on a DSP," in *IEEE Transactions on Consumer Electronics*, vol. 44, no. 3, pp. 866-871, Aug 1998.
[5] W. Xue, S. Sun and X. Li, "Overlapping MP3 Multi-thread Parallel Decoding," *2015 3rd International Conference on Applied Computing and Information Technology/2nd International Conference on Computational Science and Intelligence*, Okayama, 2015, pp. 458-463.
[6] Byeong Gi Lee, "A new algorithm to compute the discrete cosine transform", *IEEE Transactions on acoustics, speech and signal processing*, 32(6), 1984.
[7] M. D. Valdes, M. J. Moure, J. Dieguez and S. Antelo, "Hardware solution of a polyphase filter bank for MP3 audio processing," 2008 IEEE International Symposium on Industrial Electronics, Cambridge, 2008, pp. 1225-1229.
[8] T. Geng, L. Waeijen, M. Peemen, H. Corporaal and Y. He, "MacSim: A MAC-Enabled High-Performance Low-Power SIMD Architecture," *2016 Euromicro Conference on Digital System Design (DSD)*, Limassol, 2016, pp. 160-167.
[9] D. De Caro, N. Petra, A. G. M. Strollo, F. Tessitore and E. Napoli, "Fixed-Width Multipliers and Multipliers-Accumulators With Min-Max Approximation Error," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 9, pp. 2375-2388, Sept. 2013.
[10] Y. Jhung and S. Park, "Architecture of dual mode audio filter for AC-3 and MPEG," in *IEEE Transactions on Consumer Electronics,* vol. 43, no. 3, pp. 575-585, Aug 1997.