A Specific Frame Blocking and Windowing Architecture for the Pre-processing of Speech Signals

Roger Pizzatto Nunes, José Gómez, Dante Barone, Sergio Bampi
rogerpn@inf.ufrgs.br

Universidade Federal do Rio Grande do Sul - Instituto de Informática
Cx. Postal 15064 - Av. Bento Gonçalves, 9500 - Campus do Vale - Bloco IV
Bairro Agronomia - Porto Alegre - RS - Brasil - 91501-970
Fone: 55-51-33167036 Fax: 55-51-33167308

Abstract

The present work, shows a hardware implementation of a specific architecture for the pre-processing of speech signals in an automatic speech recognition system. For this, a frame blocking and windowing processor were designed in FPGA.

1. Introduction

The pre-processing of speech signals has great importance, because it converts the speech waveform to some type of parametric representation. It represents the acoustic events in the speech signal in terms of a compact set of speech parameters. In figure 1, this process is shown in a general form.

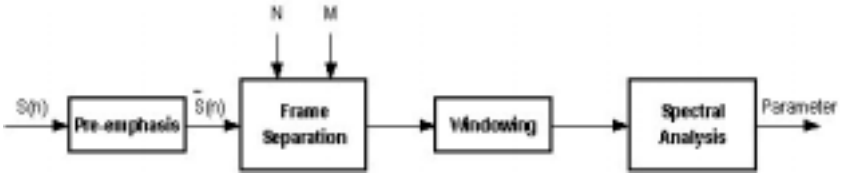

Figure 1 – Block diagram of pre-processing processor.

The digital speech signal *s(n),* is put through a low-order digital first order filter to make it less susceptible to precision effects. In the sequence, a frame separation and a windowing process is performed. Finally, a spectral analysis is applied to extract the parameters used in the system [RAB 93] [RAB 78] [GOM 01].

The main objective of this work is the design of a hardware architecture to perform the frame blocking and windowing of speech signals. In the next sections, such implementation is shown.

2. Frame Blocking and Windowing Algorithm

In all practical signal processing applications, it is necessary to work with short parts of the signal named *frames.* In speech signals, whose characteristics are non stationary, the duration of each frame correspond to a portion of the signal that can be assumed to be stationary [DEL 87].

The signal *s(n)* of figure1 is blocked into frames of *N* samples and each adjacent frames are separated by *M* samples. Thus, the first frame is formed by the first *N* samples of the speech signal. The second frame is *M* samples after the first, overlapping it by *N-M* samples.

In the same manner, the third frame begins *2M* samples after the second and overlaps it by *N-2M* samples. Generally, it is used that *M=M/3*. The group of *M* samples is denoted by S_i and the set of three consecutive S_i groups is named by F_j (frames). The process described above continues until all the speech samples are accounted (see figure 2).

In this work, *N* =252 and *M*=84 were used, for a frame duration of 23ms (11025 Hz frequency sample).

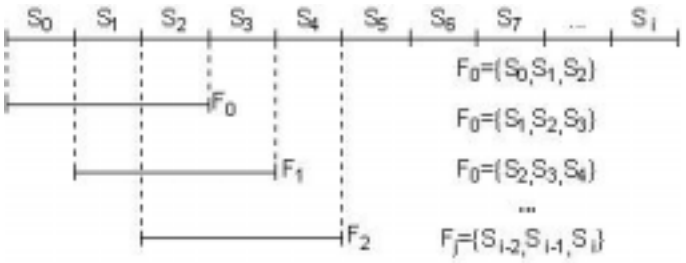

Figure 2 –The frame blocking algorithm.

The next step is to window each frame in order to minimize the signal discontinuities at the beginning and the end of each frame. It was used the Hamming window, that has the form:

$$w(n) = 0.54 - 0.46 \cdot \cos\left(\frac{2 \cdot \pi \cdot n}{N-1}\right) \quad (1)$$

where *w(n)* is the window value, *N* is the number of samples of each frame and $0 \le n \le N-1$.

3. FPGA Implementation

For hardware simplification, the frame blocking and windowing function were designed in the same circuit. In figure 3, the complete structure of this circuit is shown in a general form. It is composed by a RAM memory (that store the samples to compute the frames), a ROM memory and a multiplier (that implement the windowing function), a multiplexor (which select the inputs) and a register (that accumulates each windowed sample).

In the first moment, the *Mux_Mult* component select the speech signal, that is being stored in the memory, to apply the Hamming window and compute the first frame. Next time, this multiplexor selects the input from memory to calculate the remaining frames.

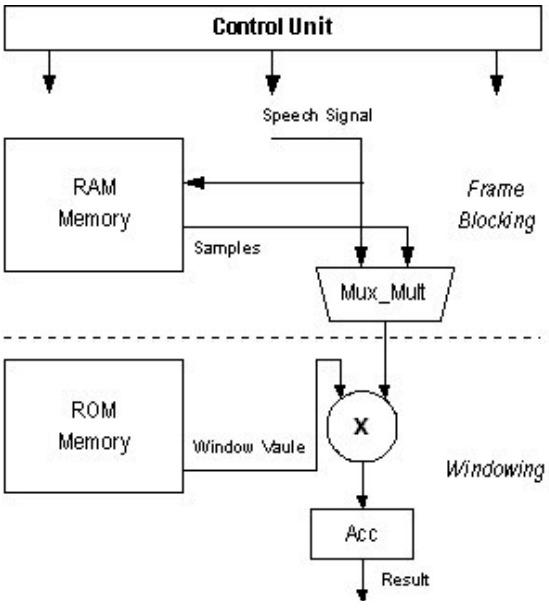

Fig. 3 - Frame blocking and windowing

Instead of windowing function implementation that is easy to see, the frame blocking algorithm was implemented by a set of reads and writes of specifics regions of the RAM memory. A dual-port RAM memory segmented in 3 equal parts *P0* (addresses from 0 to 83), *P1* (addresses from 84 to 167) and *P2* (addresses from 168 to 251) is used (see figure 4).

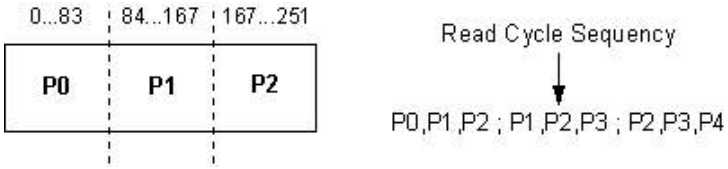

Figure 4 – Memory partition.

First, the group of samples S_0, S_1 and S_2 are stored in *P0, P1* and *P2* respectively and the frame F_0 is computed. Next, 28 samples of S_3 are written in *P0*. In the sequence, 28 samples of S_3 are saved in *P0* while S_1 is read from *P1*. Similarly, 28 samples of S_3 are saved in *P0* while S_2 is read from *P2*. Finally, 28 samples of S_4 begin to be stored in *P1* while S_3 is completely read from partition *P0,* finishing the composition of frame F_0 (see figure5).

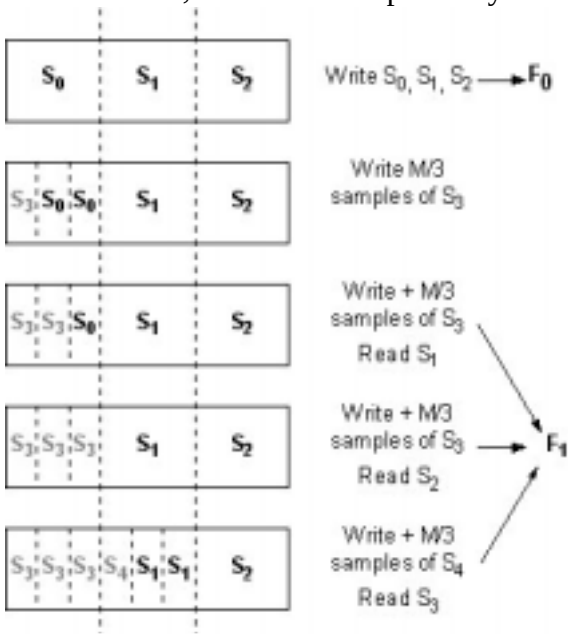

Fig. 5 - Framing block algorithm in hardware.

In figure 6, the architecture implemented for this algorithm is shown more specifically. The control unit distributes the control signals for all the components. The *Up_Counter8* indexes the written addresses for the RAM and *Up_Counter8pl* indexes the correspondent read addresses for the RAM at the same time. The initial address value in the *Up_Counter8pl* is loaded from the output of *Mem_Index* registers. The *Updown_Counter8* selects the value of *w(n)* from the ROM memory.

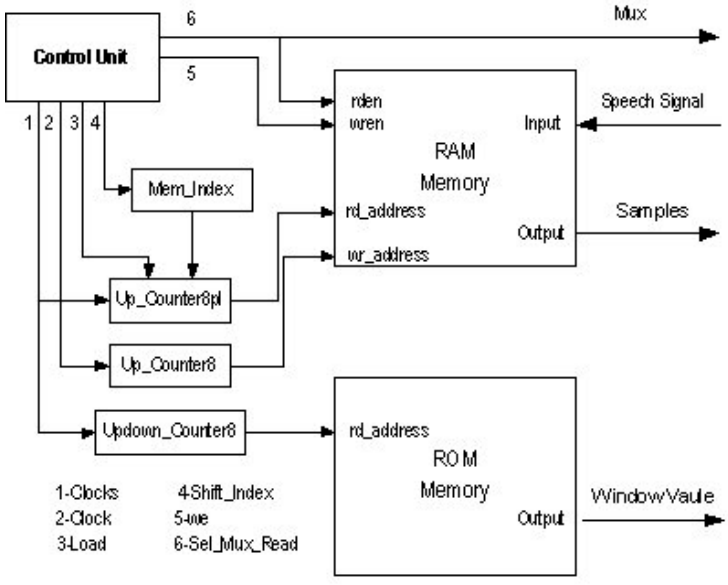

Figure 6 - Detailed hardware architecture for frame blocking.

The component *Mem_Index* is formed by 3 registers connected in cascade (figure 7 specify this structure). In the Reset state, the addresses of each partition are loaded in each correspondent register to satisfy the read partition cycle of figure4.

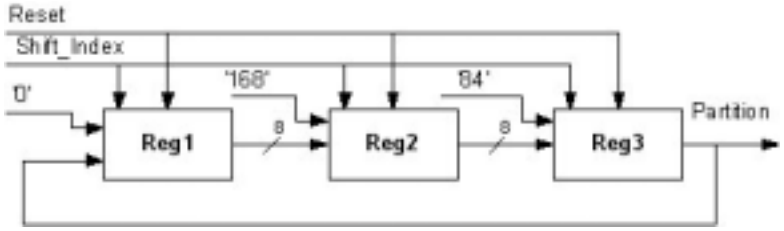

Figure 7 - *Mem_Index* component.

For numeric representation, it was used fixed point, with 16 bits precision All the system was tested by comparison of results from Maxplus II and Matlab tools. The results of three different implementations are shown in table 1. The first uses three single memories. The second uses one dual-port memory. The last one, is a simplified version of the second.

Table 1- Results of hardware implementation.

Circuit	*Device*	*Input Pins*	*Output Pins*	*Memory Bits (%)*	*Memory Utilized (%)*	*LC's*	*LC's (% utilized)*	*Clock Frequency (MHz)*
FBW_RTL0	*EPF10K30ETC144-1*	13	20	3402	13	394	22	92.59
FBW_RTL1	*EPF10K30ETC144-1*	13	20	3402	13	348	20	94.33
FBW_RTL2	*EPF10K30ETC144-1*	13	20	3042	13	325	18	96.15

4. Conclusion

In this work, an architecture for frame blocking and windowing is shown. Although the choice of which parameters to use is dictated by other considerations, the way that it is computed is based strictly on signal processing methods.

The goal is to use this circuit for the pre-processing of voice signals, in an automatic speech recognition system.

5. References

[DEL 87] DELLER, J. R. JR.; PROAKIS J. G.; HANSEN J. H. L. **"Discrete-Time Processing of Speech Signals"**, New Jersey: Prentice-Hall,1987. 908p..

[RAB 93] RABINER, L. R.; JUANG, B. **"Fundamentals of Speech Recognition"**, New Jersey: Prentice-Hall, 1993. 507p.

[RAB 78] RABINER, L. R.; SCHAFER, R. W. **"Digital Processing of Speech Signals"**, New Jersey: Prentice-Hall, 1978. 512p.

[GOM 01] GOMEZ, José Cipriano; NUNES, Roger Pizzatto; BARONE, Dante; BAMPI, Sergio. **"Arquitetura Específica de Pré-processamento com Extração de Parâmetros Mel-cepstrais para um Sistema de Reconhecimento Automático de Voz"**, VII Workshop Iberchip, Março 2001, Montevideu, pág.62.