DWT ALGORITHM SIMPLIFIED WITH NOBLE IDENTITY AND IMPLEMENTED IN THE F28379D DSP

1st Eugênio Pozzobon
Digital Processing Class (PPGEE)
Federal University of Santa Maria (UFSM)
Santa Maria, RS, Brazil
eugenio.pozzobon@acad.ufsm.br

***Abstract*—The Wavelet Decomposition is commonly used for signal processing techinque. The calculation process is simplified with a filter bank, which can be often hard to some microcontrollers. To solve this problem, the Noble Identity can be used to simplify the problem, looking for a faster implementation. In this paper, the F28379D have been used to compute the wavelet for all the 24 analog channels, as required by the aplication context. The results shown that the wavelet can be implemented with the Ideentity, reducing the total CPU time for the technique.**

***Index Terms*—DWT, rbio3.1, F28379D**

I. Introduction

There are many signal processing tools useful for spectrum analysis. The most common and widely used is the Fast Fourier Transform., which processes time series data into its respective frequency components. The FFT has been used in time series analysis, image processing, and also image compression [1].

However, in a large time series data, the FFT lose the time-domain information, and another techiniques must be used. One approach that aplies the FFT is the Short Time Fourier Transform, that split the time-series data into several small size time-series data. Then, the FFT is processed for each one of the small part of the large signal [2]. This last tool is used, for example, in audio processing by softwares like Adobe Audition.

In addition, we have the Discrete Wavelet Transform (DWT), which employs a filter bank to break down a time-series signal into multiple time-series signals, each representing a specific portion of the original signal's spectrum. The DWT is extensively utilized in the analysis of biomedical signals, including electrocardiograms (ECG), electroencephalograms (EEG), and blood pressure signals. Similar to the Fast Fourier Transform (FFT) employed in JPEG image compression, the DWT serves as a compression tool within the JPEG2000 format [3]

II. Aplication Details

The subject of this paper is to present a technique to compute the DWT with the F28379D, a Texas Instrument Digital Signal Processor(DSP). The target application will run a decision-making algorithm based on features processed with the DWT. This algorithm will run individually for all the 24 analog channels of the DSP, making the routine deadlines short. Each analog channel will have a 250kHz sampling frequency.

III. Discrete Wavelet Decomposition

The discrete wavelet transform (DWT) is a technique for decomposing a signal to capture both high and low-frequency components. It is a linear transformation that operates on a finite-length signal, and it can be implemented using a filter bank, as shown in the Figure 1, where the downarrow represents a downsampling operation, with j represents decomposition level and, $a_0 = x(n)$, H_1 is the highpass filter, H_0 is the lowpass filter, while N is the number of coefficients of the filters. Given the convolution operation by (1), the filter bank can be implemented accourding with (2) and (3), which the filter inpulse response convolutes the input signal $x(n)$ while performing the downsampling with a factor of two [4].

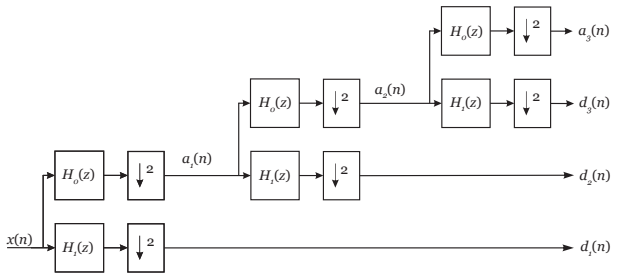

Fig. 1. Wavelet Filter Bank.

$$y[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k] \tag{1}$$

$$a_j[n] = \sum_{k=0}^{N-1} h_0[k]a_{j-1}[2n-1-k] \tag{2}$$

$$d_j[n] = \sum_{k=0}^{N-1} h_1[k] a_{j-1}[2n-1-k] \tag{3}$$

A. FIR Filters

Finite Impulse Response (FIR) filters are described by the equation 4, where $x[n]$ is the input signal, $y[n]$ is the output signal.

$$y[n] = \sum_{k=0}^{M-1} b_k x[n-k] \tag{4}$$

Or, in the Z domain, by the equation 5 [5].

$$Y(z) = \frac{\sum_{k=0}^{M-1} b_k z^{-k} X(z)}{z^{-M}} \tag{5}$$

B. Noble Identity

When working with filter banks, it is often useful to combine the filters into one filter, in order to reduce the number of required operations. In the case of wavelets filters, when there are just few decomposition levels required for the application, it is interesting to combine the filters into one single filter.

Normally, when there are two cascaded filters, the response of the combined filter is given by the multiplication of the two filter responses in the z-domain [6]. However, in the wavelet filter bank, there is a downsampling operation between the two filters. So, to combine both, the Noble identity can be usefull [7], see Figure 2.

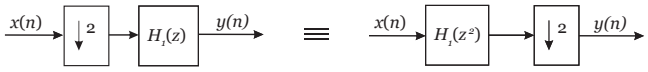

Fig. 2. Noble identity for commuting downsampler and the sparse transfer function [6].

Therefore, the filter bank can be implemented with a single filter per decomposition, as shown in the Figure 3. Then the transfer functions can be combined as in (6), where $G(z)$ is the transfer funcion of the filter and $G'(z)$ is the transfer function of a second filter. This is equivalent to perform a convolution between both filters coefficients.

$$G(z)G'(z) = \sum_{n=0}^{N-1} g[n] z^{-n} \sum_{n=0}^{N-1} g'[n] z^{-n} \tag{6}$$

One way to verify if this is correct, is comparing the results of this aproach with the results optained with the filterbank. In Figure 4 there is the filter implementation without downsampling and the signal decomposed by the PyWavelet library, which uses the downsampling [8].

Although this filter is been aplied in a 4096 points signal, in the DSP this filter must be used in a shorter part of the signal. This can lead with padding problems. So, for now, the padding is not been used, and the result of the convolution in the DSP is been truncated for each 256 points filtered, as shown in the Figure 5.

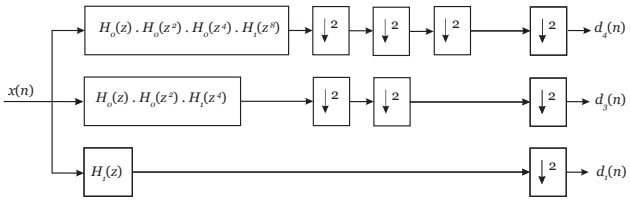

Fig. 3. Wavelet Filter Bank with Noble identity [7].

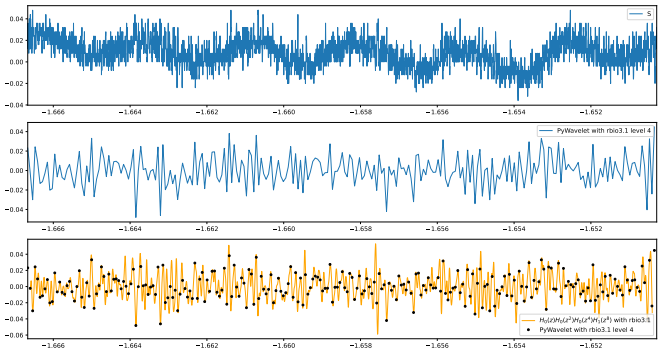

Fig. 4. Wavelet with Filter Bank implementation and with Noble identity.

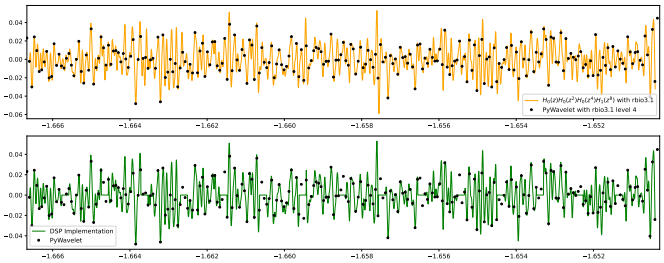

Fig. 5. Wavelet Filter Bank with Polyphase Decomposition.

C. Selecting the Mother Wavelet

For this aplication, the rbio3.1 was selected as the mother wavelet, which is a non symetric biorthogonal wavelet with three vanishing moments, as shown the Figure 6.

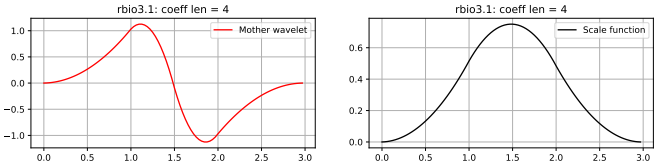

Fig. 6. Rbio3.1 wavelet function.

The frequency response of the Aproximation and Detail is given in Figure 7. This mother function was selected because it has the lowest side lobes on the filter response, when compared to other mother wavelet functions. Also, the rbio3.3 and 3.5 can be used for the proposed algorithm in its context. However, when performing the noble identity, the order of the filter

grows as the decomposition level desired is higher. Therefore, it is important to use the one that has the lowest order for decomposition level 1.

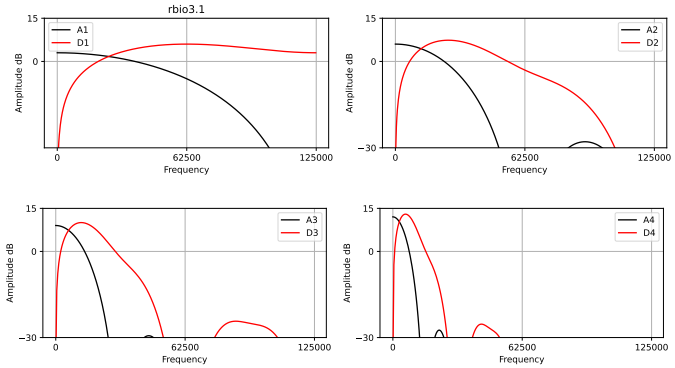

Fig. 7. Rbio3.1 frequence response

D. Analyzed Features

Given the decomposition levels of the rbio3.1 , the detail level 4, 3 and 1 were used. Then, the power of each decomposition level was calculated, where $x[n]$ is the input signal, and N_p is the length of the signal. For calculate the power of $d_4(n)$, $N_p = 16$, for $d_3(n)$, $N_p = 32$ and for $d_1(n)$, $N_p = 128$, considering that the batch size is 256 samples.

$$P = \frac{1}{N_p} \sum_{n=0}^{N_p-1} |x[n]|^2 = \frac{\boldsymbol{x} \cdot \boldsymbol{x}}{N_p} \tag{7}$$

E. Decision Making

The decision-making algorithm can be implemented using either a heuristic-based method or an Artificial Intelligence (AI) technique. In either case, the feature extraction routine in the Digital Signal Processing (DSP) system must be sufficiently fast to allow enough time for the decision boundary.

To meet this requirement, numerous approaches have been tested and evaluated. The most efficient methodology can execute within a time frame as low as 10 microseconds (10us), ensuring a swift feature extraction process, enabling timely implementation of the decision boundary.

By optimizing the feature extraction routine and reducing processing time, the overall algorithm can achieve faster decision-making, allowing for real-time or near real-time analysis and response.

IV. F28379D Implementation

The C code implementation was developed on the Code Composer Studio. The target DSP is the TMS320F28379D, which is a 32-bit floating point CPU developed by Texas Instruments. It has a Floating Point Unit and a Trigonometric Math Unit. This DSP has also 2 CPUs, each one have also a Control Law Accelerator (CLA) [9]. Then the workaload used to process 24 channels can be split by the two CPUs, each one with 12 channels.

The CLA is a specialized co-processor present in the Texas Instruments C2000 family of microcontrollers. It operates in parallel with the C28x CPU, providing additional computational power for real-time control applications. Also, has direct access to on-chip peripherals, enabling it to interact with the system's input/output (I/O) devices, timers, and other peripherals. However, the CLA has a limited ammount of memory, which is shared with each CPU, in the Local Shared(LS) block [10].

Due this limitations, the CLA is used for this aplication to adquire the time-series data and store it in a batch buffer. Also, as the 12 bits Analog to Digital Converter (ADC) reads the data with values between 0 and 4096, the CLA makes the conversion of the readen values to float32 and transform it to the real values used by the application.

Then, when the batch buffer is loaded, each CPU perform a copy to the General-porpouse Scrachpad space of RAM and start to perform the DWT computation for each window. This process timing diagram is shown in Figure 8, that exemplifies the execution time of the algorithm for all channels.

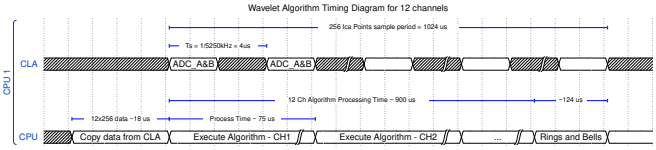

Fig. 8. Timing diagram of the algorithm.

A. FPU Optimization

The only way to compute the DWT faster with the F28379D DSP is by using the FPU. For this, Texas Instruments hands a special library with ready-to-use functions looking for float array operations, complex numbers, FFT, and FIR filters [11]. With it, there are three ways to compute the DWT:

1) FIR Filter Function
2) FFT Function
3) Array Functions

The FIR Filter creates its batch window, performing the filter dynamically. According to the TI documentation, this approach has the highest computational cost, which is insufficient for this application. On the other hand, the FFT can be used to compute the convolution in the frequency domain. However, to have accuracy with this computation process, the FFT must be implemented with the Highest Possible number of points, which is not possible in this application, as the window limitation is 256 points due to the lack of memory. Finally, the array operations can perform the convolution faster. Also, there are two approaches for it: one with the array multiplication function, and another that computes the dot product.

V. Implementation Results

Then, the two alternatives with array functions were combined to compute the DWT, with and without the noble identity. As Table I shows, the Noble Identity is required for

TABLE I
DWT IMPLEMENTARION TIMING.

DWT Signals	Normal	With Noble
a1, d1	37us	37us
a2, d2	19us	-
a3, d3	9us	16us
a4, d4	5us	8us
d1, d3, d4	116us	61us

this project, as it makes the decomposition levels 1, 3, and 4 direct accessible, reducing total execution time. The normal column of the table refers to compute the given signals having the previous one already calculated. In the last row, is the total value, considering all the middle-steps signals that must be calculated in the case of the normal algorithm.

VI. CONCLUSION

The algorithm requires a total execution time per channel of 75us using the rbio3.1 filters. That is enough to perform it for all the 12 channels per CPU under 1024us. However, the proposed algorithm was tested with one channel connected to the target environment, although all channels were processed. Therefore, the DSP still needs further tests to ensure that all processes are being respected in the routine.

VII. REFERENCES

REFERENCES

[1] M. H. Riza Alvy Syafi'i, E. Prasetyono, M. K. Khafidli, D. O. Anggriawan, and A. Tjahjono, "Real time series dc arc fault detection based on fast fourier transform," in *2018 International Electronics Symposium on Engineering Technology and Applications (IES-ETA)*, 2018, pp. 25–30.

[2] R. Balamurugan, F. Al-Janahi, O. Bouhali, S. Shukri, K. Abdulmawjood, and R. S. Balog, "Fourier transform and short-time fourier transform decomposition for photovoltaic arc fault detection," in *2020 47th IEEE Photovoltaic Specialists Conference (PVSC)*, 2020, pp. 2737–2742.

[3] A. Graps, "An introduction to wavelets," *IEEE computational science and engineering*, vol. 2, no. 2, pp. 50–61, 1995.

[4] Z. Wang, S. McConnell, R. S. Balog, and J. Johnson, "Arc fault signal detection - fourier transformation vs. wavelet decomposition techniques using synthesized data," in *2014 IEEE 40th Photovoltaic Specialist Conference (PVSC)*, 2014, pp. 3239–3244.

[5] A. V. Oppenheim, *Discrete-time signal processing*. Pearson Education India, 1999.

[6] P. P. Vaidyanathan, *Multirate systems and filter banks*. Pearson Education India, 2006.

[7] C.-C. Cheng, C.-T. Huang, C.-Y. Chen, C.-J. Lian, and L.-G. Chen, "On-chip memory optimization scheme for vlsi implementation of line-based two-dimentional discrete wavelet transform," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 17, pp. 814 – 822, 08 2007.

[8] G. R. Lee, R. Gommers, F. Waselewski, K. Wohlfahrt, and A. Leary, "Pywavelets: A python package for wavelet analysis," *Journal of Open Source Software*, vol. 4, no. 36, p. 1237, 2019.

[9] Texas Instruments, *TMS320F2837xD Dual-Core Delfino Microcontrollers*, Texas Instruments, 2021. [Online]. Available: https://www.ti.com/lit/ds/symlink/tms320f28375d.pdf

[10] ——, *TMS320F2837xD Dual-Core Microcontrollers Technical Reference Manual*, Texas Instruments, 2019.

[11] ——, *TMS320C28x FPU Primer*, Texas Instruments, 2020.