Philippine Engineering Journal (1999) XX (2): 6-17

DIGITAL SIGNAL CONDITIONING OF BIOELECTRIC SIGNALS: A STUDY OF DIGITAL FILTER IMPLEMENTATION

S.V. Rodriguez and R.C. L. Guevara, Ph.D.
Instrumentation, Robotics and Control Laboratory
Digital Signal Processing Laboratory
Electrical and Electronics Engineering Department
College of Engineering
University of the Philippines
Diliman, Quezon City Philippines

ABSTRACT

This paper presents possible ways of implementing digital Finite Impulse Response (FIR) filters in microprocessors to perform signal conditioning on biomedical electropotential signals, specifically electromyogram (EMG) and electrocardiogram (ECG). Digital filters are useful for these applications because of the flexibility offered by these filters as compared to their analog counterparts. This study focuses on the FIR type of digital filters because of their potential to minimize the distortion introduced to a signal. Different FIR filter design implementations are presented in order to come up with a comparative performance of the different implementations. The filters will be compared based on performance, accuracy, cost and complexity. Requirements and recommendations will be made describing various alternatives in digital filter implementation.

I. Introduction

One important field of biomedicine is the extraction of information from the human body. Different sources are available in extracting the desired information, and one of the most commonly used is the measurement of the body's bioelectric signals. Electrocardiogram (ECG), Electroencephalogram (EEG) and Electromyogram (EMG) all rely on the fact that the heart, brain and muscles, respectively, generate electrical impulses while performing their function. These bioelectric signals typically contain frequency components ranging from a few hertz to a few thousand hertz and are typically very low-amplitude signals.

In order to make these signals useful, some form of processing or signal conditioning is often required, sometimes because the desired signal is corrupted by unwanted noise, and sometimes to transform the available data into a form that more easily conveys the desired information. In electrocardiogram for example, the standard clinical bandwidth for a 12-lead clinical ECG is from 0.05 to 100Hz. The bandwidth for monitoring systems typically use 0.5 to 50Hz and cardiotachometers typically need frequencies of about 17Hz[1].

Digital filters are useful for these applications because of the flexibility they offer as compared to their digital counterparts. For example, one important advantage of digital filters over analog filters is their relative immunity to drift in component values caused by aging and varying environmental conditions, which is particularly essential in bioelectric applications. The FIR subtype of digital filters are used exclusively in this study to avoid the phase distortion inherent in other kinds of digital filters. Some simple applications of digital filtering in biomedicine applications have been studied by [2], [3], [4] and [5].

The design of digital filters have been extensively studied by [6], [7], [8], [9] and [10]. The disadvantage of digital filtering is the inherent errors they introduce due to their nature which include errors due to filter parameter quantization, input signal quantization, errors due to overflow and underflow conditions, and errors due to rounding and truncation. This paper will explore the advantages and disadvantages of different digital filters as implemented in microprocessors with varying levels of complexity.

As the word-length of a microprocessors decreases, finite word-length effects like quantization errors increases. This paper will study the actual performance of 8-bit non-DSP, 16-bit non-DSP, 16-bit DSP, and floating point processors. The Zilog Z86E40 non-DSP microcontroller is used to implement 8-bit filtering. The Motorola 68000 non-DSP microprocessor is used to implement 16-bit filtering. The TMS320C50 DSP processor is used to implement 16-bit filtering to compare the performance of DSP and non-DSP implementations. A personal computer is used to implement floating point filtering. Implementing digital filters with processors with larger word lengths will result in greater accuracy and better filtering performance but at the cost of more expensive hardware while conversely, implementing digital filtering with smaller word-length processors will result in cost savings at the expense of accuracy and performance.

This paper will demonstrate that with carefully chosen filters, digital FIR filters can be implemented properly with processors ranging from the low-end 8-bit microprocessors to the high-end floating processors. Recommendations are then made for various applications discussing the various tradeoffs involved.

II. Materials and Methods

The methodology to be followed is shown n the flowchart in Figure 1.

The software simulation involves the use of two different software - Matlab 5.3 and custom Turbo C programs. The simulation is controlled wholly by a Matlab script which expects an external program to perform the digital filtering process. Although Matlab is more than capable of doing the filtering process by itself, the C program can model the actual hardware implementation subtleties more accurately.

The hardware implementation and verification involve the use of different microprocessors representing the types under study. Different hardware systems were implemented to perform the required digital filtering. The system block diagram of the hardware is shown in Figure 2. In summary, three types of filters will be designed. These are the 8-bit fixed-point, 16-bit fixed-point and the 32-bit, single precision, floating point filters. Each of these filters will be implemented on four microprocessors, if possible, and the results will be gathered in three different sets - the 8-bit, the 16-bit and the floating point results. Filter comparisons will first be done on results within the same set. After the trend is established, inter-set comparisons will then be done. In this way, we can accurately compare the different results and explain the reasons for their different performance.

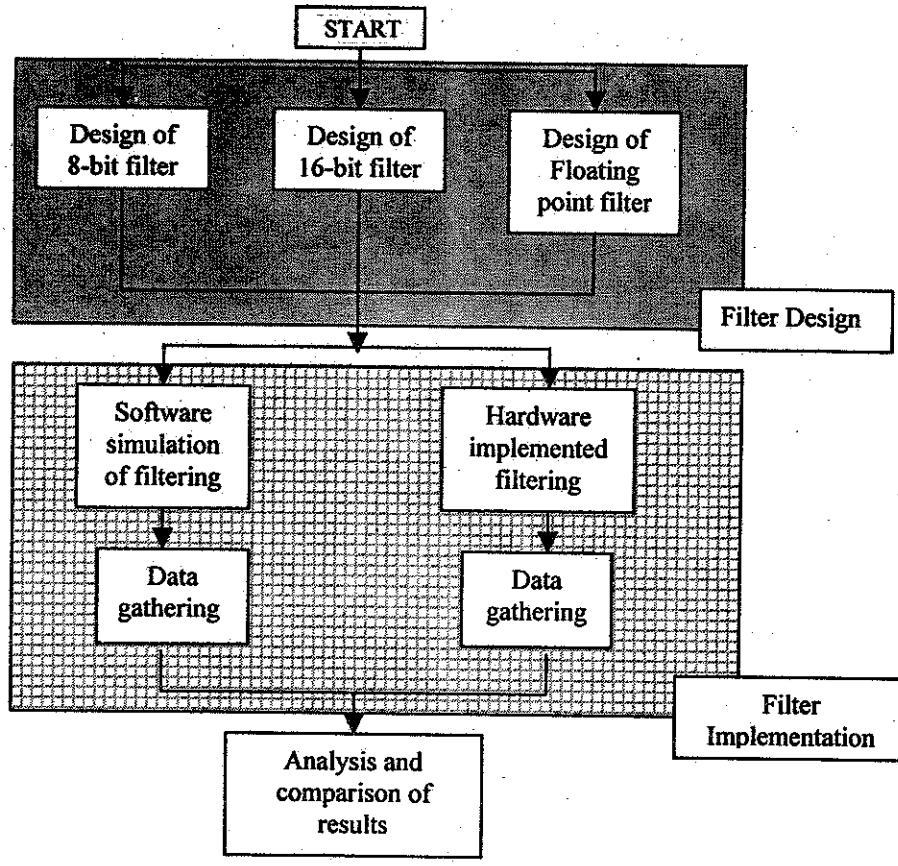


Figure 1. Flowchart of Procedure

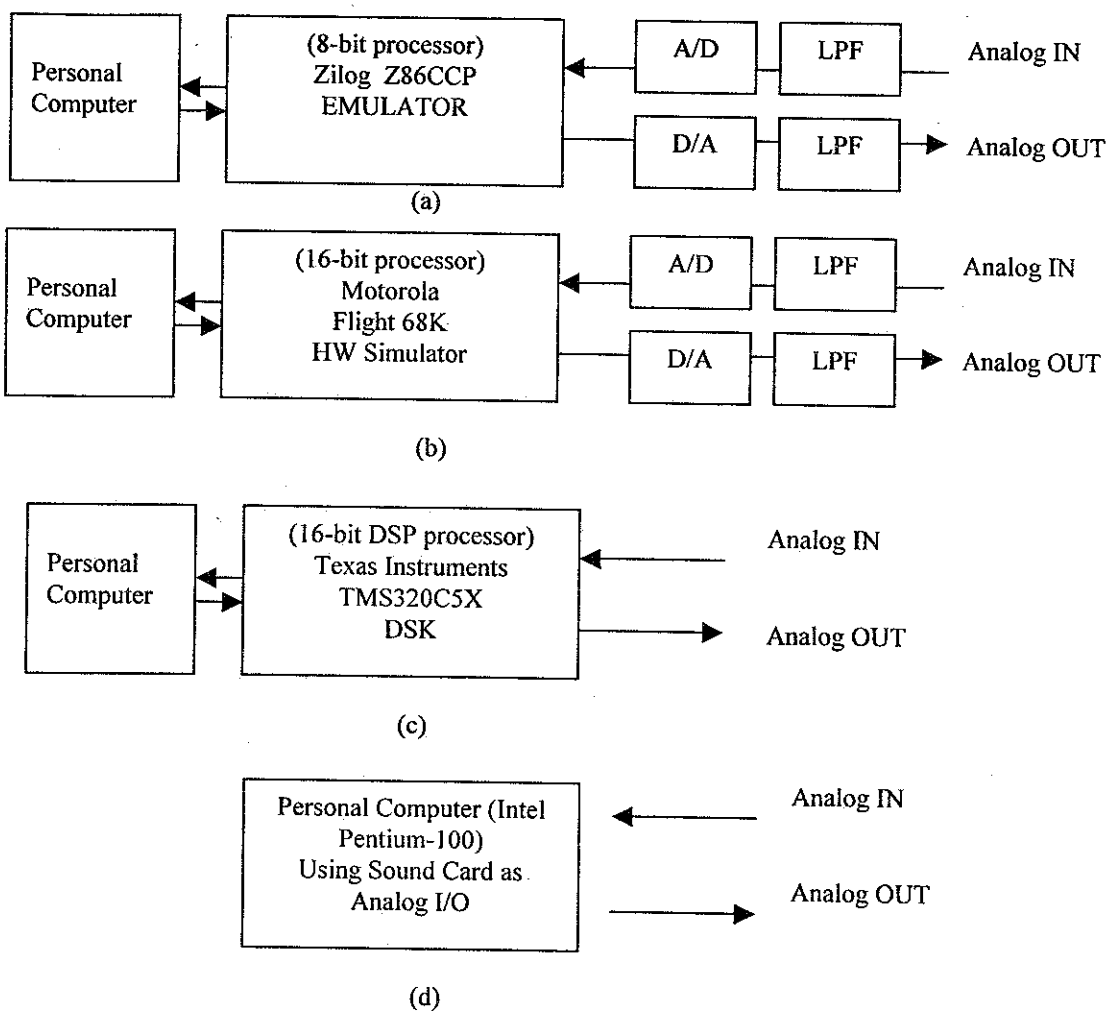


Figure 2 Equipment Setup - (a) setup for 8-bit filtering (b) setup for 16-bit non-DSP filtering (c) setup for 16-bit DSP filtering (d) setup for single precision floating point filtering

III. Filter Implementation

3.1 Introduction to filter design and specification of filters

For consistency, all filters used in the implementations are based on the ideal filter response shown in Figure 3.

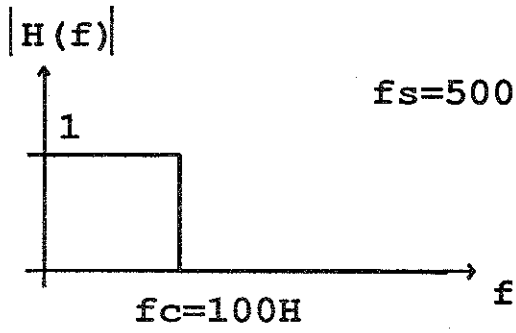


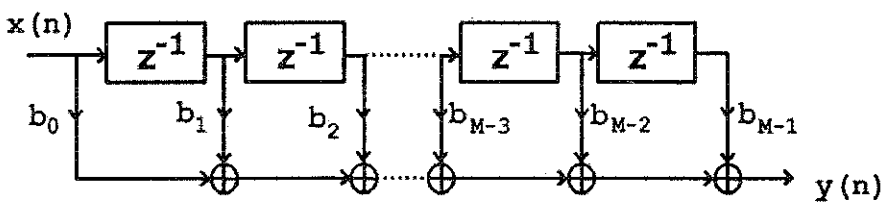
Figure 3. Desired Ideal Filter Frequency Response

The ideal frequency response shows a unity gain from DC up to 100Hz and infinite attenuation from 100Hz up to half the sampling rate of 500Hz. The given filter response is designed to pass frequencies in an electrocardiogram signal relevant for 12-lead clinical ECG.

Designing high-order filters will result in better performance such that the response approaches the ideal response. Low-order filters provide a poorer approximation of the desired response. Of course, higher-order filters require more processing and computing power.

The FIR filter topology to be considered in this study is the direct convolution type FIR filter (see Figure 4).

(a)



(b)

Figure 4. (a) Block diagram for an M-length, (M-1)th order FIR filter

3.2 FIR filtering

3.2.1 8-bit Fixed-point implementation

From the given lowpass filter cutoff requirements, different order FIR filters are generated using built-in Matlab functions. The generated filter coefficients are then represented in 8-bit 2's complement representation. The quantized 8-bit coefficients are then given to the different filter setups, both simulation and actual.

Table 1
List of 8-bit FIR filters to be Implemented by the Four Hardware Setups.

Microprocessor	8-bit fixed-point Filter Order				
	1	3	5	7	9
8-bit (Z86E40)	√	√	√		
16-bit non-DSP (68000)	√	√	√	√	√
16-bit DSP (TMS320C5X)	√	√	√	√	√
Floating point (Intel Pentium-100)	√	√	√	√	√

The results are recorded and some plots are shown in figure 5 in section IV.

3.2.2 16-bit Fixed-point implementation

The procedure in 3.2.1 is repeated with the exception that filter coefficients are represented in 16-bit 2's complement representation.

Table 2 shows the order of filters that were given to each setup. Note that the 8-bit processor wasn't used for 16-bit filtering. Theoretically, this is still possible by writing code to "emulate" 16-bit operation but the processing power of most 8-bit processors is not enough to do emulation and filtering at the same time.

Results are recorded and some plots are shown in figure 6 in section IV.

Table 2
List of 16-bit FIR Filters to be Implemented by the Four Hardware Setups.

Microprocessor	16-bit fixed-point Filter Order				
	6	8	10	20	40
8-bit (Z86E40)					
16-bit non-DSP (68000)	√	√	√		
16-bit DSP (TMS320C5X)	√	√	√	√	√
Floating point (Intel Pentium-100)	√	√	√	√	√

3.3.3 Floating point implementation

Again, 3.2.1 is repeated but this time, coefficients are represented using single precision IEEE floating point format. Obviously, only the floating point processor can do the filtering, although we could have programmed the other processors to emulate the floating point representation.

Again, Table 3 shows the order of filters that were given to each setup. As before, the 8-bit and 16-bit processors were not used because they don't naturally support floating-point filtering. It is important to note that the 16-bit DSP processor is powerful enough to perform floating-point operation and low-order filtering. This fact is useful when the need is present for both floating-point precision and lower-cost of a 16-bit point DSP processor as compared to a floating-point processor.

Table 3
List of Floating-Point Filters to be Implemented by the Four Hardware Setups.

Microprocessor	32-bit floating point Filter Order					
	6	10	20	40	80	100
8-bit (Z86E40)						
16-bit non-DSP (68000)						
16-bit DSP (TMS320C5X)						
Floating point (Intel Pentium-100)	√	√	√	√	√	√

Results are recorded and some plots are shown in figure 7 in section IV.

IV. Filtering Results

4.1 FIR Filtering Results

Figures 5 to 7 shows some of the results obtained from the filtering process. Plots on the left column result from software simulation, while plots on the right column result from the hardware setups. Figure 5 shows the result from the 8-bit fixed point filtering. For the software column, the first waveform shows the result from a simulated 8-bit processor, the second waveform shows the result from a simulated 16-bit processor and the third waveform shows the results from a simulated floating point processor. For the hardware column, the first waveform is from the 8-bit setup, the second from the 16-bit non-DSP setup, the third from the 16-bit DSP setup and the last from the floating point hardware setup. Figure 6 shows the result from the 16-bit fixed-point filtering, and figure 7 shows the results from the single-precision floating point filtering.

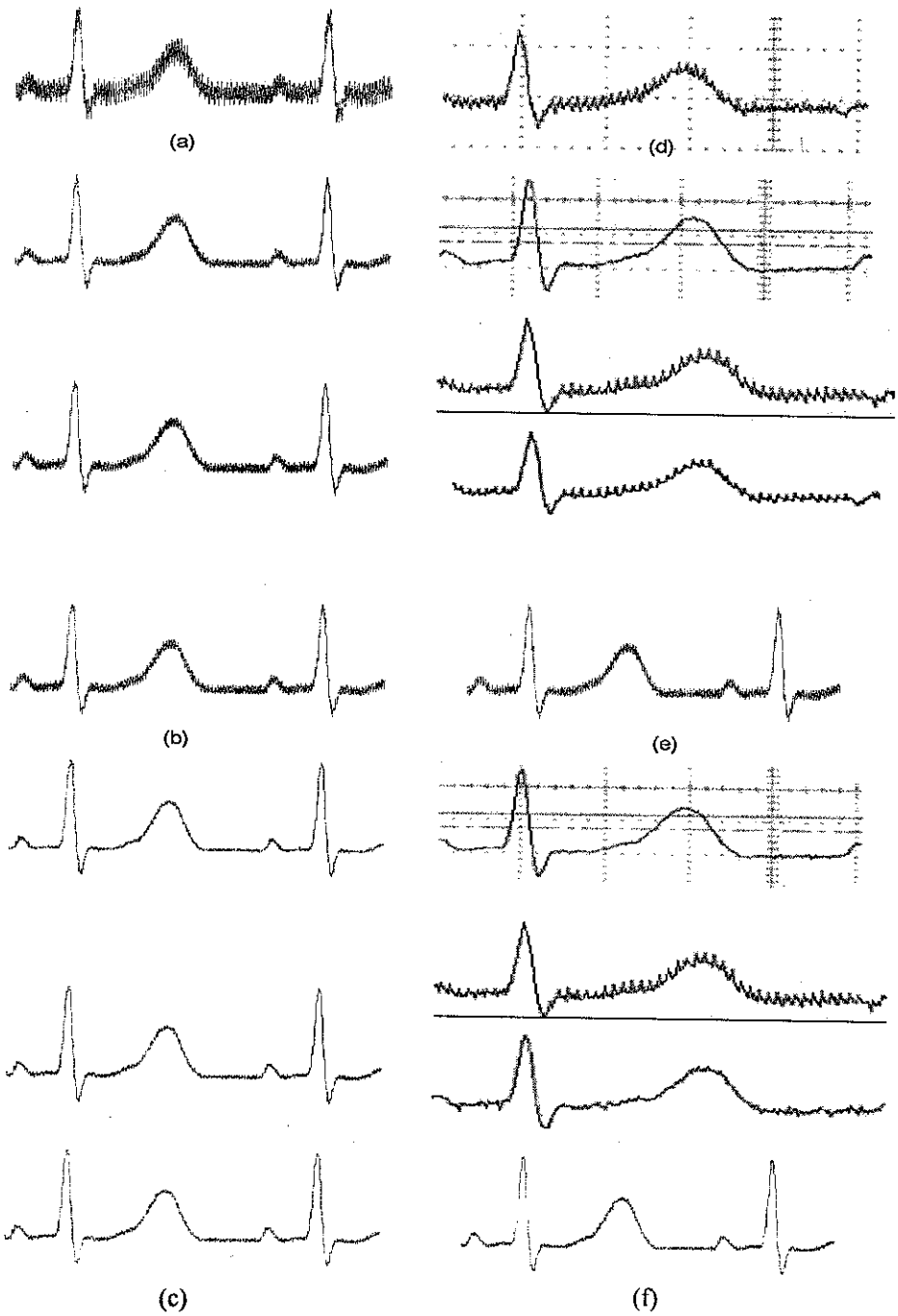


Figure 5. (a) Input to the software simulation (b) 1st order 8-bit filter results using simulation (c) 3rd order 8-bit filter results using simulation (d) Input to the hardware setups (e) 1st order 8-bit filter results using the four different hardware setups (f) 3rd order 8-bit filter results using the hardware setups.

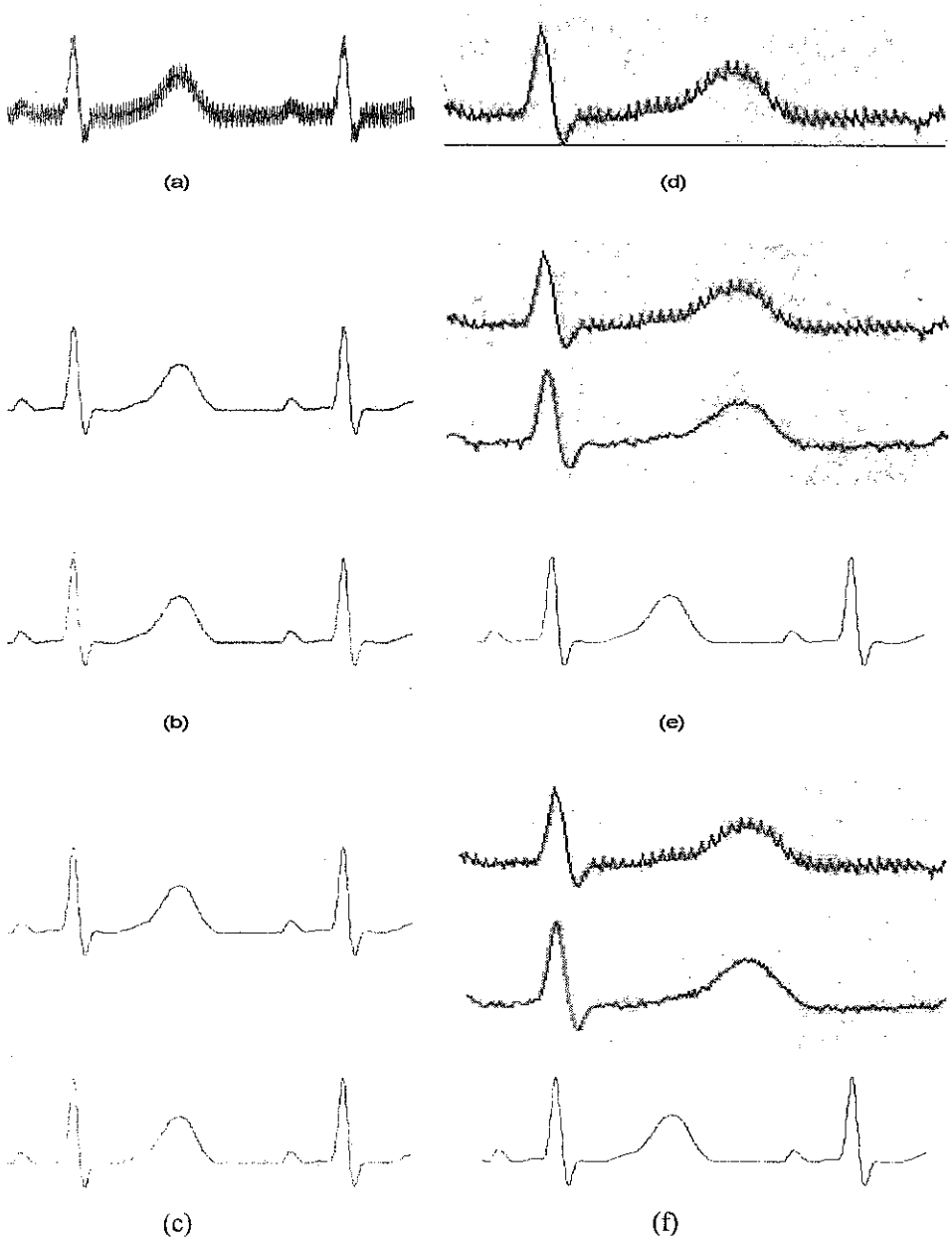


Figure 6. (a) Input to the software simulation (b) 6^{th} order 16-bit filter results from simulation. Top-16bit simulation, bottom-floating point simulation (c) 10^{th} order 16-bit filter results from simulation (d) Input to the hardware setups (e) 6^{th} order 16-bit filter results from hardware. Top-68000 hardware, middle-TMS320C5X hardware, bottom - Intel Pentium (f) 10^{th} order 16-bit filter results from hardware (order same as e)

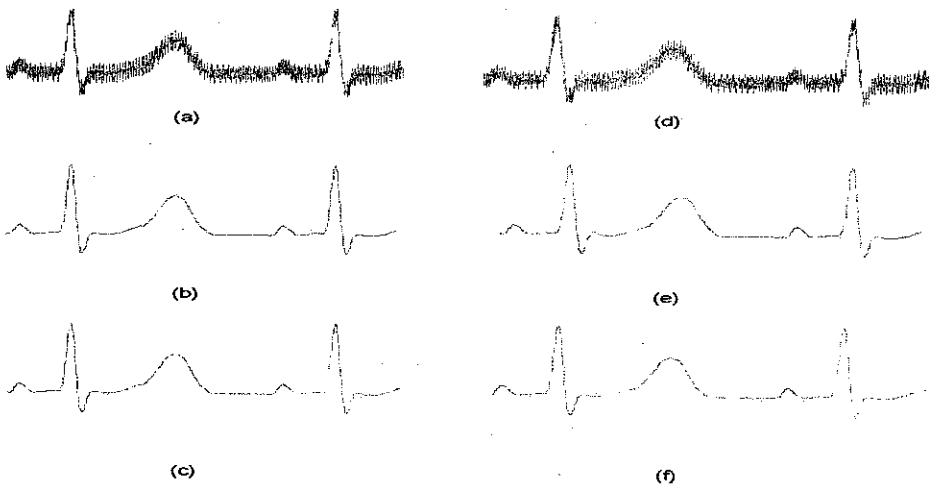


Figure 7. (a) Input to the software simulation (b) 10^{th} order floating point filter results from simulation (c) 20^{th} order floating point filter results from simulation (d) Input to the hardware (e) 10^{th} order floating point filter results from hardware (f) 20^{th} order floating point results from hardware.

V. Discussion of Results

5.1 Discussion of Criteria

The criteria used in this paper for evaluating the filters are the visual quality of the output and the complexity of the processor required to achieve it. A follow-up paper will be dedicated to the detailed numerical analysis of the results.

5.2 Discussion of FIR Results

Figures 5 to 7 show some of the results from the various filters used. The left columns show the results achieved through software simulation of the filtering and the right columns show the results achieved using the four hardware setups. The topmost waveforms in each figure show the input waveform to the filter and the succeeding waveforms show the results from each setup.

From the software simulation results, it can easily be seen that the filters performed as designed. There are no significant visual difference between the 8-bit, 16-bit and floating point filters. This means that for this application, the performance of 8-bit filtering is almost the same as the more accurate floating point filtering. As expected, increasing the order of the filter results in larger attenuation of the undesired signals, resulting in a higher signal-to-noise ratio. An analysis of Figure 7 shows that designing increasingly higher order filters result in no significant performance gain, which simply means that beyond a certain point, higher order filters are undesirable because the hardware they will require will result in a more expensive setup without a corresponding increase in quality level.

The results from the actual implementation followed the software simulation results except for the 16-bit filtering. The 8-bit filtering using the 8-bit microprocessor worked as expected, along with floating point filtering using the personal computer. Problems were encountered though, with the 16-bit filters, especially with the 16-bit fixed-point filter using the Motorola 68000 hardware setup.

As can be seen from Figure 5 and 6, the results from the 68000 setups were distorted by noise glitches. These glitches were found to be caused by an error in the setup, and not an error introduced by the filtering. The 16-bit results from the 68000 setup therefore, was erroneous.

The floating point results followed the simulation results which is as it should be since this is the most reliable and accurate setup, but obviously, the most expensive.

VI. Conclusions and Recommendations

This paper presented a study of the actual performance of different digital hardware in performing digital filtering of bioelectrical signals. Different hardware setups were implemented to study 8-bit non-DSP, 16-bit non-DSP, 16-bit DSP, and single precision floating point performance.

With the exception of the 16-bit non-DSP setup, which was faulty, all the setups performed as expected, with the floating point processor providing the best filter performance, both in filter complexity (higher order) and accuracy.

Using visual inspection of the output signals, it was found out that 8-bit filtering provided essentially the same performance as the floating point processor, but using a much less expensive system.

Digital filtering can be performed by processors with varying complexity, even a low-end 8-bit microprocessor. This fact can be exploited by building systems, requiring only simple filtering, which are based on cheap, readily-available low-end 8-bit microprocessors. As the requirement of the application increases, higher performance microprocessors are then considered. Lastly, digital filters offer significant advantages over their analog counterparts but if implemented poorly, these filters will cause more problems than they will solve.

VII. References

1. B.W. Bomar, L.M. Smith, R.D. Joseph, "Roundoff noise analysis of state space filters implemented on floating point digital signal processors", IEEE Trans on ckts and systems, Nov. (1997).
2. D. Bhattacharya, A. Antoniou, "Design of equiripple FIR filters using a feedback neural network", IEEE Trans on Ckts and Systems II, April (1998).
3. Filter with transient suppression," IEEE Trans. Biomed. Engg., vol. 42, pp. 1128-1132, Nov. (1995).
4. Hubta, J.G. and Webster, J.G., "60 Hz interference in electrocardiogram," IEEE Trans. Biomed. Engg., vol. BME-20, pp.91-103, Mar. (1973).
5. I.W. Selesnick and C.S. Burrus. "Maximally flat low-pass FIR filters with reduced delay," IEEE Transactions on Circuits and Systems II - Analog and Digital SP, Jan. (1998), vol 45, pp. 53-68.
6. J.G. Proakis, D.G. Manolakis, "Digital Signal Processing, 3rd Edition," Prentice Hall, (2000).
7. Parsa. U, Parker P.A., "Multireference Adaptive Noise Cancellation Applied to Somatosensory Evoked Potentials", IEEE Trans on Biomedical Engineering. August (1994).

8. S.C. Pei, C.C. Tseng, "IIR Multiple Notch Filter Design based on allpass filter", IEEE Trans on Ckts and Systems II, Feb. (1997).
9. Tompkins, W.J. (ed), "Biomedical Digital Signal Processing", Ch.2, Prentice Hall (1993).
10. Van Alste J.A. and Schilder T.S., "Removal of based-line wander and power-line interference from the ECG by an efficient FIR filter with reduced number of taps," IEEE Trans. Biomed. Engg., vol. BME-32, pp. 1052-1060, Dec. (1985).