I²MTC 2008 – IEEE International Instrumentation and
Measurement Technology Conference
Victoria, Vancouver Island, Canada, May 12–15, 2008

Sensor Signal Processing for Autonomous Wireless Sensors

S. Bicelli[1], A.Depari[1], A.Flammini[1], D.Marioli[1], M. Serpelloni[1], E. Sisinni[1], A.Taroni[2]
[1]Dept. of Electronics for Automation, University of Brescia, Brescia, Italy
[2]Carlo Cattaneo University, Castellanza, Varese, Italy
Tel: +39 030 3715899, Fax: +39 030 380014, email: alessandro.depari@ing.unibs.it

Abstract *– Wireless Sensor Networks are nowadays a reality. However, a wireless node must be a truly autonomous system, i.e. it must embed its own power source, generally a battery pack. For this reason, it is of main concern to limit every source of wasted power. In particular, nodes exploit low duty-cycle strategies to increase their life, spending most of their time in low power mode, turning off all devices within the system except a wakeup oscillator. However, in this way some time must be spent waiting for circuitries settling time, especially if a high accuracy measurement is required. In this paper an efficient implementation of a filtering algorithm that is able to reduce this time is discussed. It is based on a hybrid combination of median and mean filter, joining advantages of both linear and non-linear filtering. A well tailored implementation has been experimentally tested and discussed.*

Keywords – wireless sensor, low-power, sensor signal processing.

I. INTRODUCTION

Wireless sensors use is growing in many application fields, as home automation, environmental monitoring and process control [1]. Wireless transceiver cost is quickly decreasing and emerging technologies (e.g. IEEE802.15.4) allow a power reduction in order to obtain autonomous and mobile sensors [2]. Nowadays batteries are the most common way to realize autonomous sensors and several research activities concern with an efficient use of batteries [3]. On the other hand, many methods of power harvesting (solar cell, electromagnetic fields, piezoelectric generators, thermopiles...) are growing [4]; they can be used to recharge batteries or to replace batteries. In both cases low-power techniques must be adopted to save the small amount of available power. In most of applications, the wireless sensor periodically wakes up, measures the interesting quantity and, if necessary, transmits information. In the case of humidity or temperature sensor, information transmission rarely occurs, because humidity or temperature variation is very slow. In most cases, the measure is repeated every second while data transmission takes place less than one time every minute. For this reason, in order to save further power, the sensing element and the interface circuits are powered just before the measure and then they are turned off as soon as the measure is over. Both sensing element and electronic interface circuits require a certain time to furnish a stable and accurate measurement.

In the following, a temperature/humidity sensor is taken into account, but the proposed approach can be used with other type of sensors. Typical temperature or humidity sensors used in low-cost applications are resistive or capacitive sensors. In the first case, the sensor can be inserted in a voltage divider and the output voltage is acquired by means of an analog-to-digital converter. In the second case, the sensor is used in an oscillating circuit, e.g. based on the 555 multivibrator device, and a microcontroller estimate the output signal period by exploiting its timing unit. In both cases, the circuits must be operative for a short time, so that the power consumption could be limited. On the other hand, the use of filters is highly advisable, in order to improve the signal-to-noise ratio. However, low-pass filters applied to signals and power supply cause a long measure settling time. As a consequence, an enlargement of the operative time of the sensor, microcontroller and interface circuits is produced which determines an increment of the power consumption.

A simple and largely employed solution is waiting for a fixed time to perform the measure after the power supply has been applied to the sensors and the circuits. This fixed time value is established in the project stage by means of a-priori considerations about the system. However, this solution is neither flexible nor efficient, because the time value is usually oversized. This is due to the circuit output settling time which is not usually specified in the component data sheets and, furthermore, it strictly depends on the temperature, power supply and load conditions. In addition, even for the same component, it can vary among different manufacturers, especially if the component is not specifically developed for low power applications.

For these reasons, usually the output signal from the sensor is continuously measured until it reaches a stable value. This operation is not always simple, because of the noise overlapped to the signal, which makes the time to get a stable value longer. From a different point of view, this is a typical pattern recognition problem, made up of a pre-processing, feature extraction and detection phase. Feature extraction can be performed by means of edge detection (analyzing the gradient of the monitored signal), while for the detection a simple threshold rule can be used. However, these steps works well only in absence of noise, that must be eliminated by the pre-processing stage. In conclusion, the use of a suitable filter is highly recommended; it should be effective towards different noise typologies, but it does not have to increase too much the measure time. In addition, it should be fast and easy to implement in order to not increase microcontroller cost. In case of signal affected by Gaussian noise overlapped with impulsive noise, as it happens for

instance on audio or echocardiographic signal analysis, literature shows a large use of filters based on a combination of median, weighted median and linear filters for the sensor output smoothing [5]. For example, median hybrid filters suggested by Heinonen, and Neuvo[6,7], running medians with robust regression, analized by Davies et al [8], Qα-method-based filters proposed by Croux and Rousseeuw [9,10]. For audio signal elaboration and image processing, many realizations of efficient median filters have been proposed [11-14], even if usually in this field fast and powerful processors or DSPs are employed. Particularly interesting are the so called trimmed or winsorized mean filter, better described in the next section [15]. Generally, such filters are well designed for the signal of interest, but they are not easy to implement and need a lot of computational resources, requiring the microcontroller to be operative for a long time.

The goal of this work is the development of a simple filter easy to implement and suitable to the measurement stabilization detection.

II. THE PROPOSED APPROACH

As we said, the combination of a median and a mean filter is suitable for the application, but the implementation must be carefully tailored to be effective without requiring a lot of computational effort. The following analysis of application and noise is needed to better characterize filter requirements. The signal coming out from the sensor can be approximated as the summation of three terms: the first one represents the useful information, the second one models the broadband Additive White Gaussian Noise (AWGN) and the last one models impulsive noise. As well known, broadband noise can be effectively removed, or at least attenuated, by means of low-pass FIR filter (Finite Impulse Response filter, as the moving average), while impulsive noise can be removed using non linear filtering as median filters. Obviously, the span of the filter must be chosen in order to prevent underlying signal distortion and it is regulated by a trade off. A filter with too many taps requires a high computational effort and has a high latency (no suitable for fast sensors as resistive ones). On the contrary, AWGN and glitch immunity cannot be ensured with too few taps. Considering that sampling time of resistive sensors could be in the range [10,100] μs and in the range [0.1,1.0] ms for capacitive sensors, the overall elaboration time must be on the order of 100 μs per sample. As an example, in Figure 1 it is shown the output of a capacitive sensor used together with a TLC555 oscillator; the measurement readout is the signal period (time elapsed between two successive rising edges, on the order of 100μs). The jitter, due to switching broadband noise, makes measurement T1, T4 and T5 different; on the contrary, the glitch makes T2 and T3 outlier data that must be discarded.

As regards AWGN suppression, supposing to implement a linear phase FIR filter (i.e. the group delay is constant at all frequencies), an N-taps filter has a latency equal to: $Ts(N-1)/2$ [s], where Ts is the sampling period. Probably, the best choice is a moving average filter that provides the fastest step response for a given noise reduction.

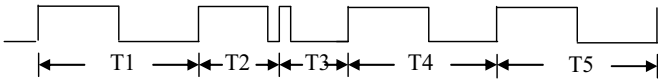

Figure 1. Output of a capacitive sensor used with TLC555 affected by an outlier.

Referring to impulsive noise, how many spikes can be effectively rejected? To answer this question it is usually adopted the estimator breakdown point; it is defined as the largest fraction of input data that can be replaced by arbitrarily large values without driving the estimator output error to infinity. The breakdown point of a median of $N=2K+1$ data points is K/N, i.e. at least halve of the sample needs to be replaced to completely destroy the filter output. On the contrary, the mean filter has a breakdown point equal to zero, regardless the filter length.

Aim of this work is to explore efficient implementations of a median filter able to reject at least two outliers, in conjunction with a moving average filter with an overall delay on the order of twice the sampling time. In other words, the median filter must have a minimum length of five and the moving average filter must be a four taps filter providing a new output value every four input samples still halving the noise. However, even if the moving average is an exceptionally good smoothing filter (the action in the time domain), it is an exceptionally bad low-pass filter (the action in the frequency domain). In addition, the non overlapping windowing of median filter leads to a decimation in time, worsening spectral properties (Figure 2a); in fact the cutoff frequency is equal to $(Ts \cdot N \cdot M)^{-1}$, where Ts is the sampling period, N is the median filter length and M the moving average length, respectively. In order to overcome this limit, it is possible to use overlapping windows shifted of one point if an odd length is chosen or two points if an even length is adopted. For instance, an overlapped 5-taps median filter followed by a 4-taps moving average filter (Figure 2b) has an absolute better noise bandwidth with respect to the non overlapped one (Figure 2a). A 6-taps median filter, that updates its output, i.e. the average of the two central points, every two input samples and it is followed by a 2-taps moving average filter (Figure 2c) has about the same time properties than the previous one (Figure 2b). In fact both solutions are able to filter out two outlier data (glitches with duration equal to two samples) and show the same behavior, with respect to a step and a monotonic input, than a 4-taps moving average. Obviously, non overlapping strategy does not satisfy our requirements; as regards overlapping implementations, it can be shown that an even-length median followed by a 2-taps average filter (Figure 2c) has better frequency rejection properties than an odd-length median (Figure 2b), due to different interleaving of linear and non-linear filtering actions.

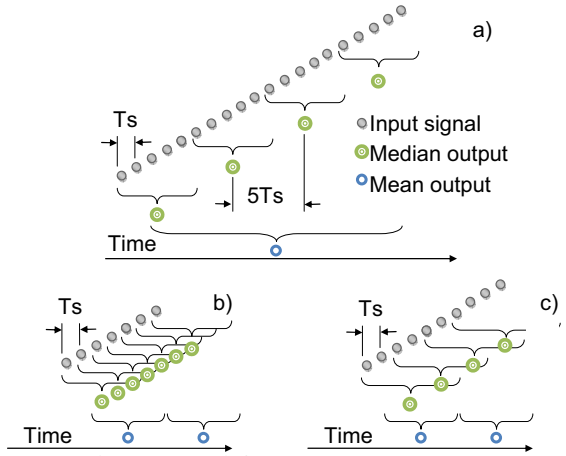

Figure 2. Comparison of a) traditional non-overlapping approach and b,c) modified overlapped hybrid median-mean filter.

A further improvement can obtained using the so called truncated, trimmed or winsorized mean [15]. In an ordered set X_i the placement of the sample is referred as the rank. Thus, in a set of cardinality N=2K+1 the median has rank = K and the Truncated Mean Filter –TMF– is given by the average of all the samples having rank [K+1-Q, K+1+Q], where Q is a constant value fixed *a-priori*. More formally:

$$TMF_{ODD}(X_1,...,X_N;Q)=\frac{\sum_{i=1}^{N} a_i X_i}{\sum_{i=1}^{N} a_i} \qquad (1)$$

$$a_i=\begin{cases}1, \text{if } K+1-Q \le \text{rank}(X_i) \le K+1+Q \\ 0, \text{otherwise}\end{cases}$$

If we considered an even length N=2K, the previous equation must be slightly modified since the median is defined as the average between samples of rank K and K+1 and (1) becomes:

$$TMF_{EVEN}(X_1,...,X_N;Q)=\frac{\sum_{i=1}^{N} a_i X_i}{\sum_{i=1}^{N} a_i} \qquad (2)$$

$$a_i=\begin{cases}1, \text{if } K-Q \le \text{rank}(X_i) \le K+1+Q \\ 0, \text{otherwise}\end{cases}$$

Following this reasoning, an 8-taps modified median filter, that is updated every four input samples and furnishes the average value of the four central points, exhibits an even steeper frequency response still preserving a good behavior with respect to step impulse, monotonic signals and glitches.

III. FILTER IMPLEMENTATION

In the following a new implementation of an 8-taps modified median filter will be described; an experimental evaluation will be carried out with real wireless sensors. As a reference, the above described 6-taps filter will be also tested (see Figure 2c). In Figure 3 the flow chart of the reference 6-taps filter is shown. We start from a 6-elements input vector; each element is a couple {value, time} representing the sample value and sampling instant, with 1≤time≤6. Neglecting the initialization phase, a list structure has been implemented and element {value, time} are updated in the vector (READ x_i and REPLACE x_{i-6}) according to the bubble sort algorithm. As the vector is already ordered, the bubble sort for the new entry is a very efficient method (SORT vect). Every two updates, the median filter furnishes the average of the two central points. The overall system output is the average of two consecutive outputs of the median filter.

In Figure 4 the flow chart of the proposed 8-taps filter is shown. It must be noticed that this implementation is different from the previous one that exploits the bubble sort algorithm; the proposed algorithm is focused on the discard of extremes, that are labeled *m1* (minimum value), *m2*(second minimum), *M2* (second maximum), *M2* (maximum value) (with $m1 \le m2 \le M2 \le M1$).

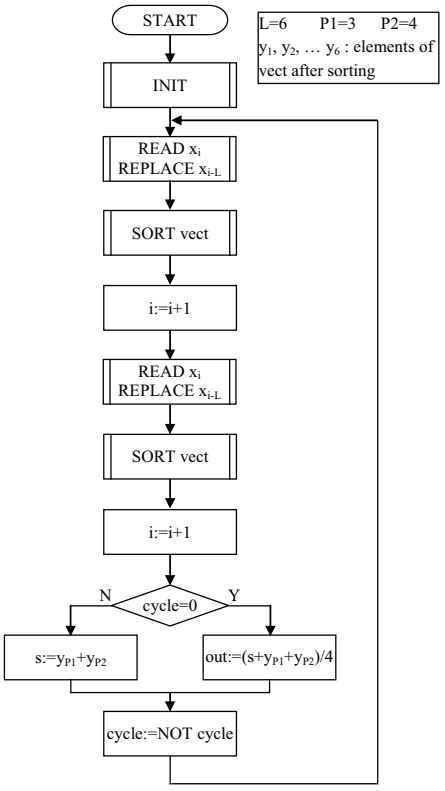

Figure 3. Flow-chart of the reference 6-taps filter.

The basic idea is to partition the data array; starting from an initial sorted array, every new element is forced in the outliers group or in the correct samples one. This is the fastest

way to proceed if no *a-priori* knowledge of the signal is available [16]. In fact, in this application we want to select outliers element, not to sort them, i.e. we can attend only the subset that contains them. After that, we can perform averaging of correct samples.

First three samples are sorted by a simple algorithm which initializes *m1*, *m2*, *M1* and *M2* values. The remaining 5 samples are checked to replace *m1*, *m2*, *M1* and *M2*. At the end of this process, the output result coincides with the mean value of the four central values (summation of all elements without *m1*, *m2*, *M1* and *M2*).

Thanks to this approach it is possible to decrease the computational effort leading to almost equal time for both best and worst case. The flow chart in Figure 4 is a simplified one. In fact, samples are processed while they are acquired in a continuous and more efficient fashion.

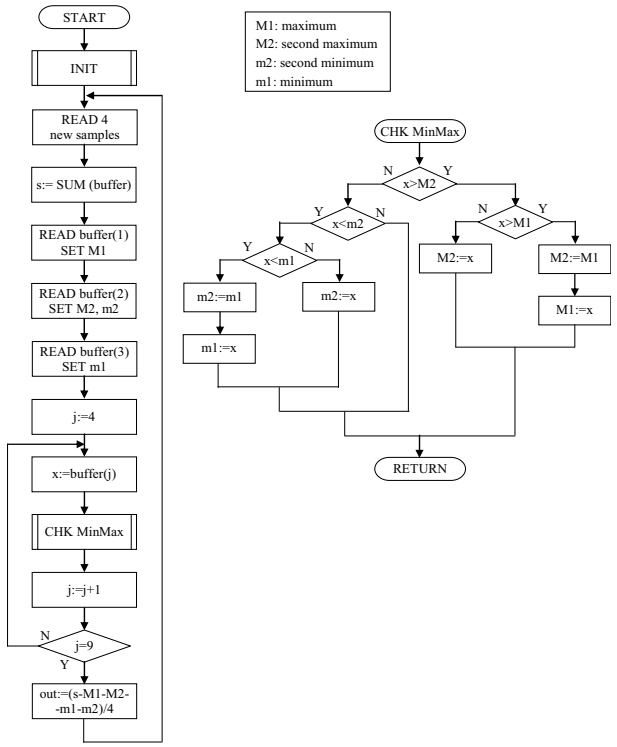

Figure 4. Flow-chart of the proposed 8-taps filter.

IV. RESULTS

Two wireless prototypes have been developed to test the filters performance: a temperature sensor (PT100) with an interface circuit that provides an output voltage; a capacitive humidity sensor with a frequency-coded output signal. Both prototypes are supposed to be battery powered, to measure every 100 ms and to transmit information every minute. In order to optimize the battery power (two AA NiMH rechargeable batteries -2400mAh-), a step-up DCDC (Texas TPS61070) interface is necessary between the power source and the circuits.

Battery life L [hours] of a wireless sensor depends on the battery capacity C [Ah], the power supply efficiency η, the power supply output voltage gain Kv and the total mean current consumption Icc,mean [A] according to L = (η·C)/(Kv·Icc,mean). Normally the device wakes-up every Tm seconds, takes about Ta (active phase) to start-up and measure quantities, transmits and receives information by the Radio Frequency (RF) link every T seconds (T>Tm) taking time T_{RF}. $I_{cc,mean}$ is shown in equation (3) where I_a, I_{RF} and I_{sleep} are the whole circuit current consumptions in the active, RF and sleeping phases respectively.

$$I_{cc,mean} = I_a \frac{T_a}{T_m} + I_{RF} \frac{T_{RF}}{T} + I_{sleep} \frac{T_m \cdot T - T_m \cdot T_{RF} - T \cdot T_a}{T \cdot T_m} \approx$$
$$\approx I_{sleep} + (I_a - I_{sleep}) \frac{T_a}{T_m} \qquad (3)$$

As in this case $T_{RF} << T$, the current consumption $I_{cc,mean}$ depends mainly on T_a. So, as we said, T_a must sufficiently high to guarantee a stable measurement, but not too high to save power.

Both the realized prototypes use low power microcontroller (Freescale MC9S08GT60A) and low power IEEE802.15.4 transceiver (Freescale MC13192). The conditioning circuits must be simple in order to abate power consumption and it should be possible to virtually turn off this circuitry without affecting transient response.

The first device is a wireless resistance temperature detector that uses a PT100 as the sensible element. Figure 5 shows the conditioning circuitry, while eq. (4) shows its input/output relation.

$$Vout = R_{PT100} \cdot \frac{VRef}{R3} \left(1 + \frac{R7}{R6}\right) mA = R_{PT100} \cdot 6mA \qquad (4)$$

The sensor is driven by constant current to reduce the energy lost in the resistance of the wires. The current generator circuit, composed by both op amps U1 and U2, excites the sensor. An operational amplifier, A4, is used to zero wire resistance error. A fourth amplifier (U3) is used to amplify the signal and filter possible alias interferences and wideband noise. The 10-bit converter of the MC9S08GT60A converts the voltage across the RTD to digital code. Every amplifier is provided of shutdown pin to enter in the low power mode (Texas TLV2455). The output of the conditioning circuit is depicted in Fig. 9b; sampling time is Ts=14μs and cutoff frequency of the second order Butterworth low pass filter is about 5kHz.

The second wireless device uses a capacitive sensor (Humirel HS1100) to measure relative humidity. Signal conditioning circuit (Figure 6) converts capacitance variations into a frequency coded signal IC according to Eq. (5).

$$f = \frac{1}{C@\%RH \cdot (R5 + 2 \cdot R6) \cdot \ln 2} \qquad (5)$$

The variable capacitor U1 is connected to the TRIG and THRES pin of a timer 555. Pin 7 is used as a short circuit pin for resistor R5. The sensor capacitance is charged through R6 and R5 to the threshold voltage (approximately 0.67Vcc) and discharged through R6 only to the trigger level (approximately 0.33Vcc) since R6 is shorten to ground by pin 7. To provide an output duty cycle close to 50%, R5 should be very low compared to R6. the frequency output can be computed as depicted in Figure 1b. At RH=55%, the sensor has a nominal capacity C=180pF, so the conditioning circuitry gives a frequency of 7483Hz (microcontroller input capture has a timing unit with 125ns of resolution). The timer TLC555 lacks of a shutdown pin, so it is powered by a microcontroller output port (P2).

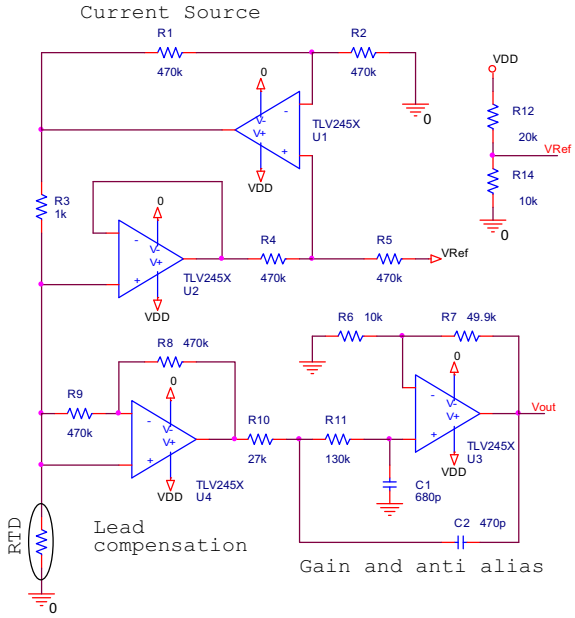

Figure 5. Interface circuit of the PT100.

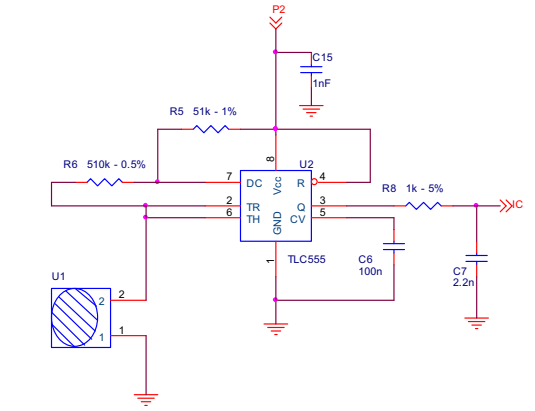

Figure 6. Interface circuit of the capacitive sensor.

In Figure 7 it is shown an example of the output of both circuits (PT100 signal corresponds to Vout in Figure 5, HR signal corresponds to IC in Figure 6); the image has been grabbed with an Agilent MSO6054A digital scope. It is clearly visible that both signals have a start up transient that must be identified and filtered out in order to obtain a correct measurement.

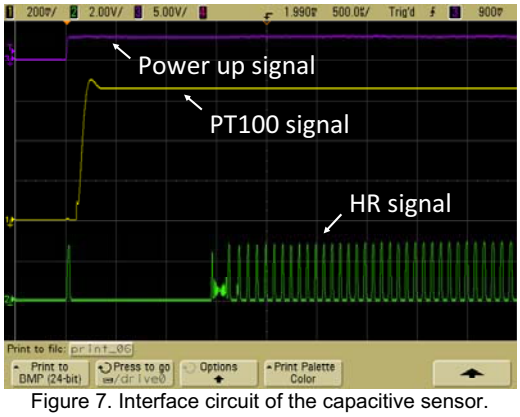

Figure 7. Interface circuit of the capacitive sensor.

To estimate filters performance, the two algorithms previously described (6 tap filter and the proposed 8 tap one) have been implemented in C language and tested with the best and the worst signal trends. No compiler optimizations have been set in order to be more hardware implementation independent. However, it must be taken into account that the adopted microcontroller has an 8-bit wide bus and stored data are 16-bit wide. For both filters, the best signal shape is a steady trend, because it requires the lowest operations number to evaluate and insert the new sample; conversely, the worst signal is a monotonic signal (e.g. decreasing ramp). As previously stated, both filters update their outputs every four input signal samples; in order to make a comparison, Table 1 shows the number of cycles required to process a new output sample for both solutions. It should be noticed how the 8 tap filter is almost independent from signal shape and offers a better mean case.

8 TAP FILTER		**6 TAP FILTER**	
Best case, signal = constant values			
N. cycles	Time [µs] (1cycle=125ns)	N. cycles	Time [µs] (1cycle=125ns)
2046	256	1985	248
Worst case, signal = decreasing ramp			
N. cycles	Time [µs] (1cycle=125ns)	N. cycles	Time[µs] (1cycle=125ns)
2180	272	5895	737

Table 1. Filters performance

Figure 8 portraits the capacitive sensor output with the 8 tap filter. On the y axes it is shown the number of ticks (one every 125ns) and on the x axes is reported the acquisition number. The filter gives a valid measure (within 0.1% of full scale) after the 44th sample (≈5.9ms after the startup). The presence of noise and glitches does not affect this result. It is evident that a simple algorithm that estimates the derivative

of the signal can be effectively used after the filter in order to detect the transient resolution and preserve batteries.

In this case the 8 tap filter takes 23981 cycles (3.0ms) of computing time while, on the same signal, the 6 tap filter takes 52749 cycles (6.6ms). This means that with the proposed approach the estimation of transient conclusion can be performed in real time, while signal is sampled. On the contrary, with the traditional approach, even considering that the first sample is available after 1.5ms (see Figure 7) and supposing to parallelize the acquisition and the processing phases, we have to wait an additional interval of about 1.5ms. Considering that the microcontroller and the electronic conditioning circuit has an average consumption in the active state of 7 mA, this means that we should waste about 10μC per measurement.

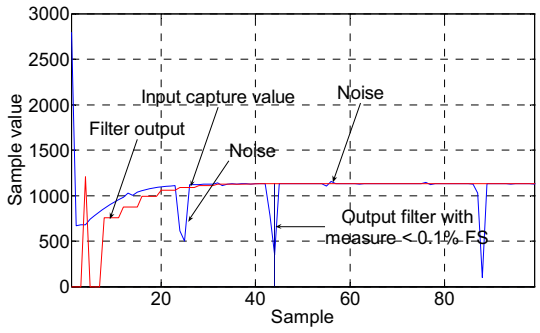

Figure 8. Filters output with noisy frequency signal.

Figure 9 shows the acquisition data from the resistive sensor. The filter produce the first valid measure after the 24th sample, that is after 336μs. Obviously, now the computation cannot be performed in real-time. A greater error threshold (0.5%FS) has been set to take in account the 10-bit of resolution of the ADC converter. Also in this case the 6 tap filter is less performing than the 8 tap filter; the first takes 26942 (3.4ms) cycles before giving the first correct value, while the latter one takes 13058 cycles (1.6ms) with a saving charge of about 13μC. As before, no optimization has been used.

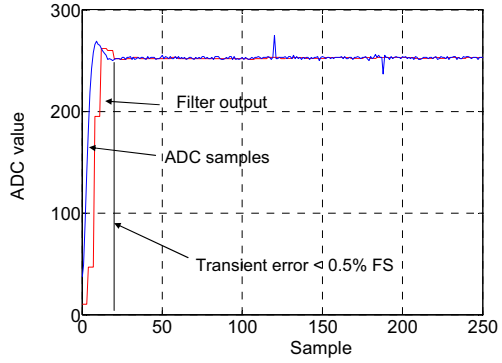

Figure 9. Filters output with noisy voltage signal.

V. CONCLUSIONS

In this paper an efficient implementation of a hybrid median-mean filter has been described and tested. It has been purposely designed to faster wakeup time of an autonomous wireless node. In this kind of applications, nodes spent most of their time in low power mode turning off all devices within the node except a low power oscillator. Obviously, this leads to a transient that must be discarded in order to obtain a good quality readout. The developed filter is able to preserve step response still rejecting broadband noise. Some experimental results have been described, showing filter performance and that the proposed implementation greatly decreases computational time with respect to a traditional approach.

REFERENCES

[1] P. Baronti, et al. Computer Communications 30 (2007) pp. 1655–1695.

[2] J. A. Gutiérrez, E. H. Callaway Jr., R. L. Barrett Jr., "Low-Rate Wireless with IEEE 802.15.4", IEEE Standards Wireless Networks Series, 2004, pp. 3-12.

[3] A. Jossen, Journal of Power Sources 154 (2006), pp. 530–538.

[4] N.G. Stephen, Journal of Sound and Vibration 293 (2006) pp. 409–425

[5] U. Gather, R. Fried, "Methods and Algorithms for Robust Filtering", COMPSTAT 2004: Proceedings in Computational Statistics, pp. 159-170, 2004.

[6] P. Heinonen, Y. Neuvo, "FIR-median hybrid filters" IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 35, pp. 832–838, 1987.

[7] P. Heinonen, Y Neuvo, "FIR-median hybrid filters with predictive FIR substructures", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 36, pp. 892–899, 1988.

[8] P.L. Davies, R. Fried, U. Gather, "Robust signal extraction for on-line monitoring data", J. Statistical Planning and Inference, Vol. 122, pp. 65–78, 2004.

[9] C. Croux, P.J. Rousseeuw, "Time-efficient algorithms for two highly robust estimators of scale", COMPSTAT 1992, Physica-Verlag, Heidelberg, pp. 411–428, 1992.

[10] P.J. Rousseuw, C. Croux, "Alternatives to the median absolute deviation", J. American Statistical Association, Vol 88, pp. 1273–1283, 1993.

[11] O. Vainio, Y. Neuvo, S.E. Butner, "A signal processor for median-based algorithms", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 37, pp. 1406-1414, 1989.

[12] C. Chandra, M.S. Moore, S.K.Mitra, "An efficient method for the removal of impulse noise from speechand audio signals", Proceedings of the 1998 IEEE International Symposium on Circuits and Systems, ISCAS 1998, Vol. 4, pp. 206-208, 31 May-3 Jun 1998.

[13] P.J.S.G. Ferreira, "Sorting continuous-time signals: analog median and median-type filters", IEEE Transactions on Signal Processing, Vol. 49, pp. 2734-2744, 2001.

[14] I. Kauppinen, "Methods for detecting impulsive noise in speech and audio signals", 14th International Conference on Digital Signal Processing, DSP 2002, Vol. 2, pp. 967-970, 1-3 July 2002.

[15] J. Astola, P. Kuosmanen, "Fundamentals of Nonlinear Digital Filtering", CRC Press, 1997, ISBN 0849325706.

[16] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling "Numerical Recipes in C: The Art of Scientific Computing" , Cambridge University Press, 1992, ISBN 0521431085