A new multi-channel real-time digital signal processing platform for acoustic signal processing and sensor response emulation

Patrick Gaydecki
School of Electrical and Electronic Engineering, University of Manchester, PO Box 88, Manchester M60 1QD, United Kingdom
E-mail: patrick.gaydecki@manchester.ac.uk

Abstract. In recent years, the DSP group at the University of Manchester has developed a range of DSP platforms for real-time filtering and processing of acoustic signals. A next generation system has now been designed, which incorporates a processor operating at 0.55 Giga MMACS, six input and eight output analogue channels, digital input/output in the form of S/PDIF and a USB interface. The software allows the user, with no knowledge of filter theory or programming, to design and run standard or completely arbitrary FIR, IIR and adaptive filters. Processing tasks may be described and linked using the graphical icon based interface. In addition, the system has the capability to emulate in real-time linear system behaviour such as sensors, instrument bodies, string vibrations, resonant spaces and electrical networks. Tests have confirmed a high degree of fidelity between the behaviour of the physical system and its digitally emulated counterpart. In addition to the supplied software, the user may also program the system using a variety of commercial packages via the JTAG interface.

1. Introduction
In recent years, the DSP group at the University of Manchester has developed a range of DSP platforms for real-time filtering and processing of music and audio bandwidth signals. A next generation system has now been designed, which incorporates a processor operating at 0.55 Giga MMACS, six input and eight output analogue channels, digital audio input/output in the form of S/PDIF and a USB interface. The software allows the user, with no knowledge of filter theory or programming, to design and run standard or completely arbitrary finite impulse response (FIR), infinite impulse response (IIR) and adaptive filters. Processing tasks may be described and linked using the graphical icon based interface. In addition, the system has the capability to emulate in real-time linear system behaviour such as sensor systems, instrument bodies, string vibrations, resonant spaces and electrical networks. Tests have confirmed a high degree of fidelity between the behaviour of the physical system and its digitally emulated counterpart.

In previous publications, the design was described of real-time DSP systems for the automatic design of FIR, IIR and adaptive filters, and for emulating in real time the behaviour of arbitrary linear systems, such as analogue circuits, sensor responses or acoustic instrument bodies [1, 2]. In this paper, several key advances are detailed that relate to all aspects of the system, including the hardware module, the high-level user interface and the real-time firmware. Of particular importance is the ability of the new system to accommodate analogue network responses, phase shift blocks and impulse or...
frequency responses of physical acoustic systems. The high level interface allows the user to specify signal processing operations in any desired order. This is achieved by dragging icons into a graphical design area, and linking them to appropriate input and output streams.

2. Signal processing repertoire

A signal processing operation is specified by locating an appropriate icon within the graphical design area. Broadly, the operations fall into two categories: simple processing functions involving a single scalar quantity applied to a signal, and more complex vector processing operations including filtering and transforms. The simple signal processing repertoire includes multiplication by a constant (gain adjustment), addition of signals, multiplication of signals (modulation) and time delaying. Vector operations include FIR and IIR filtering, adaptive filtering, real-time Fourier and Hilbert transforms, quadrature signal processing and IIR to FIR translation.

2.1. Basic linear filter theory and algorithmic implementation

The (linear) process of filtering in time \( t \) is encapsulated in the convolution integral

\[
y(t) = \int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau
\]

where \( y(t) \) is the output (filtered) signal, \( x(t) \) is the incoming signal, \( \tau \) is the time-shift operator and \( h(\tau) \) is the impulse response of the filter [3]. In discrete space, this equation may be implemented using either an FIR or IIR solution.

2.1.1. FIR design using the frequency sampling method. In the case of the FIR filter, the infinite response is truncated, which yields an expression of the form

\[
y[n] = \sum_{k=0}^{M-1} h[k]x[n-k]
\]  

with the z-transform of the impulse response, i.e. the transfer function \( H(z) \), being given by

\[
H(z) = \frac{Y(z)}{X(z)} = \sum_{k=0}^{\infty} h[k]z^{-k}
\]

The basic design problem is therefore reduced to finding the appropriate impulse response \( h[k] \) which yields the desired frequency response. The system here exploits the frequency sampling method, which proceeds as follows:

- The frequency response of the filter is specified in the Fourier-domain. We stress that this may, if desired, be an ideal brick-wall type, but it need not be so. It could equally be any arbitrary shape. For a linear phase filter, the frequency response is determined by setting the real terms to their intended values, and leaving the imaginary terms as zero.
- The inverse Fourier transform is taken of the designated frequency response, which generates a time-domain function. This is usually done with an inverse FFT. Because of the mechanics of its operation, this will result initially in a time domain signal which is not centred, i.e. the right-hand part of the signal will start at \( t = 0 \), and the left-hand part will extend backwards from the final value. Hence manipulation is necessary to centre the impulse response.
The ends of the impulse response must now be tapered to zero using a suitable window function, such as a Hanning type. Application of the window minimises ripples in the pass and stop bands but it also increases the width of the transition zone. For a simple linear phase pass-band filter, the frequency sampling method is encapsulated by the expression

\[ h[n] = f_w[n] F^{-1} \{ H[k] \} \]

where \( f_w[n] \) denotes the window function. For a filter with an arbitrary frequency response, the method is adapted so that

\[ h[n] = f_w[n] F^{-1} \{ H[k] \} \]

where \( f_w[n] \) denotes the window function. For a filter with an arbitrary frequency response, the method is adapted so that

\[ h[n] = f_w[n] F^{-1} \{ H[k] \} \]

2.1.2. IIR design using the bilinear z-transform and pole-zero placement. In contrast, IIR filters rely on recurrence formulae, where the output signal is given by

\[ y[n] = \sum_{k=0}^{N} a[k] x[n-k] - \sum_{k=1}^{M} b[k] y[n-k] \]

In this case therefore, the transfer function is

\[ H(z) = \frac{a[0] + a[1] z^{-1} + \ldots + a[m] z^{-m}}{1 + b[1] z^{-1} + \ldots + b[n] z^{-n}} = \frac{\sum_{m=0}^{M} a[m] z^{-m}}{1 + \sum_{n=1}^{N} b[n] z^{-n}} \]

The system allows the user to design filters using either the bilinear z-transform (BZT) method or the pole-zero placement method. The former technique is implemented by a module of the software containing a library of pre-defined passive analogue arrangements, based on capacitors, resistors and inductors, and whose Laplace descriptions are transposed by the software into the z-plane. The system allows the user to express digital filters based on passive equivalents, or on active equivalents such as Butterworth and Chebyshev low pass, high pass, band pass or band stop designs.

The pole zero placement method is, in contrast, an essentially interactive method, whereby the designer positions poles and zeros on the unit circle to effect a desired response. The software allows the user to locate as many as one hundred complex conjugate pairs of poles and zeros, i.e. in essence facilitating the realization of arbitrary IIR frequency response.

2.1.3. Inverse filtering. The output \( y(t) \) of any linear system may be modelled by considering the impulse response \( h(t) \), together with the input signal \( x(t) \). In such a system, the output is convolution operation between the input signal and the impulse response, i.e.

\[ y(t) = x(t) * h(t) \]
The presence of noise is a significant factor in determining the efficacy of the inverse filtering procedure; if, as is often the case, the noise is independent of the signal, then Equation (8) becomes

\[ y(t) = x(t) * h(t) + s(t) \]  \hspace{1cm} (9)

Inspection of Equations (8) and (9) shows that to conduct deconvolution in the time domain, the impulse response of the inverse filter must be pre-computed, and then applied to the output signal using the normal time convolution equation. Typically,

\[
\tilde{h}(t) = F^{-1}\left[\frac{1}{H(\omega) + s}\right]
\]  \hspace{1cm} (10)

Hence

\[ x(t) = y(t) * \tilde{h}(t) \]  \hspace{1cm} (11)

This approach is adopted by the present system and has proven very useful for both audio reconstruction and loudspeaker equalization.

2.1.4. Adaptive filtering. The real-time system incorporates an adaptive filter that is expressed as a least mean square (LMS). This is a successive-approximation technique that obtains the optimal filter coefficients required for the minimisation of the error, E. It is implemented as follows. Initially, the coefficients of the filter will be any arbitrary value; by convention, they are usually set to zero. Each new input signal value \(x[n]\) generates an output signal value \(y[n]\), from which is generated the error signal value, \(e[n]\), by subtracting it from an ideal signal value, \(d[n]\). The value of \(h[k]\) is then modified by the expression

\[ h_{\text{new}}[k] = h_{\text{old}}[k] + \Delta e[n] x[n-k] \]  \hspace{1cm} (12)

In other words, the new value of \(h[k]\) is calculated by adding to its present value a fraction of the error signal \(\Delta e[n]\), multiplied by the input signal value \(x[n-k]\), where \(x[n-k]\) is the input signal located at the \(k^{th}\) tap of the filter at time \(n\). For reasons of stability, it is important that only a fraction of \(e[n]\) is added. If the rate of convergence is too rapid, the algorithm will over-shoot and become unstable. The LMS algorithm is not the only method employed in the implementation of adaptive filters; neither is it the fastest to converge to the optimum value for a given filter coefficient. However, the LMS method is readily programmed for real-time environments and very stable.

Figure 1. True adaptive filter
The true adaptive filter, employed here and shown in Figure 1, assumes that the source of the noise is available as an input to the system. Figure 2 shows typical input and output signal traces (after convergence) with a peak-to-peak input signal-to-noise ratio of 1:100.

![Input and Output Signals](image)

**Figure 2.** Input (a) and output (b) signals from a true adaptive filter

With this DSP system, the principal application of the adaptive filter is for broadband noise cancellation, in which the bandwidth of the noise encroaches on that of the signal.

![DSP System](image)

**Figure 3.** The real time DSP system; (a) schematic; (b) circuit board

### 3. System Hardware and Software Architecture

The instrumentation described here represents an evolution of the *Signal Wizard 2.5* system, which is a dual-channel 24-bit device operating at 100 MMACS. Audio-bandwidth processing, involving in particular real-time sensor signal processing, sensor response emulation, musical instrument emulation and ambient sound field characterisation, often requires inputs from multiple sources. Hence for many applications, a powerful core is required to operate on the multi-channel data stream. To this end, a new system has been developed, based around a Freescale DSP56321 [4], operating at a maximum speed of 550 MMACS. The improvement in speed has been made possible primarily by the incorporation into the device of an enhanced filter coprocessor (EFCOP). The system also incorporates flash memory, to hold both the operating system and the coefficients of filters designed using the high level software interface.
The dual channel codec has now been replaced by a CS42438 device, which supports six input channels, eight output channels and has a maximum sample frequency of 192 kHz. Additionally, digital signal data inputs and outputs are provided in the form of an SPDIF interface unit. This obviates the need for analogue to digital and digital to analogue conversion, thereby precluding the introduction of noise in the signal transmission stages.

The system communicates with the host computer via a high-speed USB link, improving data transmission rates for real-time spectrum analysis and signal capture. A further critical refinement of this system is the incorporation of a JTAG \([\times]\) interface; this allows users to develop their own software, using 3rd party development tools such as the Freescale assembler and hardware debugger \([5]\), or the Tasking C compiler \([6]\). A schematic of the new hardware is shown in Figure 3, together with a photograph of the board.

The new hardware is accompanied by a radically different user interface, which allows for much greater flexibility with regard to the sequence of processing operations that may be conducted in real time. As shown by Figure 4, a sequence of algorithms is built up using the graphical design window, into which the operator places icons that represent the functions that are to be executed. These icons are linked by lines that represent the signal paths, and may be connected to the input channels, other icons and eventually to the output channels in any desired sequence.

![Figure 4. User interface of the design and control software](image)

4. Discussion
At the time of writing, the new DSP system is still undergoing performance evaluation using a number of standard processing benchmarks. Clearly, the advantage of the new methodology is its flexibility and its ability to replicate the behaviour of complex linear systems. It will undoubtedly have numerous applications in the field of audio-bandwidth signal processing, and indeed in the discipline of musical acoustics. Although there is a certain amount of processing overhead that is placed upon a system with such versatility, this small limitation is more than adequately compensated by the significantly improved performance of the new processor and the ease of use of the high level graphical interface.
5. Concluding remarks
A versatile, simple to use and powerful real-time DSP system has been developed incorporating a high-speed processing board and graphical Windows-based software that interfaces seamlessly with the hardware. This software allows a designer to specify an arbitrary series of algorithms with minimum effort. Once designed, the sequence of operations may be downloaded as instructions to the hardware and executed on demand.

References
[1] Gaydecki P Woodhouse J and Cross 2002 Advances in audio signal enhancement and stringed instrument emulation using real-time DSP. Proc. Forum Acusticum Sevilla 2002 (on CD; ISBN: 84-87985-06-8)
[2] Gaydecki P and Fernandes B 2003 An advanced real-time DSP system for linear systems emulation, with special emphasis on network and acoustic response characterisation Measurement Science and Technology 14 1944-1954
[3] Ifeachor E C and Jervis B W 2001 Digital signal processing: a practical approach Prentice Hall (2001).
[4] DSP56321 Reference manual Freescale Semiconductor Literature Distribution Centre, P.O. Box 5405, Denver, Colorado 80217 (2005)
[5] DSP56321EVM user’s manual Freescale Semiconductor Literature Distribution Centre, P.O. Box 5405, Denver, Colorado 80217 (2005)
[6] Motorola DSP56xxx software development toolset 2003 Altium Inc, 17140 Bernardo Center Drive, Suite 100, San Diego, CA 92128