IP Protection of DSP Algorithms for System on Chip Implementation

Roy Chapman and Tariq S. Durrani, *Fellow, IEEE*

***Abstract*—Silicon technology has now advanced to the point that there is a serious mismatch in the time taken to design advanced silicon-based systems and the time to market for any new product or product derivative. To obviate this delay, a new paradigm is emerging based on intellectual property (IP) exchange, where designers and differing companies share subsystems (virtual cores) between themselves to reduce design time to acceptable levels. To this end, over 150 companies including all the major players formed the Virtual Socket Interface Alliance in March 1997. The protection of IP has become a serious issue as intercompany subsystem design exchange becomes more commonplace.**

This paper presents new techniques to protect the IP of virtual cores that implement digital signal processing (DSP) algorithms. The approach involves embedding codewords into the design of fundamental signal processing algorithms such as digital filters and the DFT in such a way that proof of authorship can be retained, and, if required, easily identified. The techniques discussed can be adapted to protect other fundamental DSP algorithms such as convolution and correlation.

The protection of IP via watermarking techniques is increasingly being applied at all levels of design. It is particularly advantageous if such techniques are applied at the highest abstraction levels in the design flow, and if such techniques are applied at basic algorithm level, they become very difficult to detect at lower levels of system design.

***Index Terms*—DSP algorithms, intellectual property, system on chip, VLSI design, watermarking.**

I. Introduction

THE PACE of change and growth in silicon technology is enormous. In a keynote address over 30 years [2] after Moore had first introduced his famous law that stated that the number of devices on a silicon chip doubles every year, he explained that the same phenomena is still true, although it is now taking between 18 months and two years for device count to double. By 1997, the world's semiconductor industry was producing 10^{17} transistors annually, and the device density is now so great that this has ushered in the era of systems on chip (SoC) in which complete electronic systems can be fabricated on one or two chips.

As device density has increased with time , the design time has also increased, and the design cost is now a major problem. There are additional time-to-market issues that are forcing silicon chip designers to rethink the whole design process. Over the three-year period 1997 to 1999 [1], silicon complexity has increased from (200–500 K gates) to (4–6 M gates) and the design cycle for a silicon system has been reduced from approximately 15 to 9 mo, while a chip derivative cycle has reduced from approximately 7 to 3 mo. The 1998 update to the SIA Roadmap now predicts leading edge designs with 200 million transistors as a possibility by 2005.

One of the principal reasons why both original chip design time and chip derivative design time must be reduced is because the application areas for silicon chips is becoming more dominated by consumer products. The time to market for consumer products is dominated by fixed events in the consumer year such as Christmas, and consumer fashion dictates that product derivatives have to be available in a short time scale. A few weeks delay in the design of a silicon chip can have disastrous effects in such volatile markets. The next generation of chips must be more complex but be designed in a shorter time. The design productivity gap in silicon system design is not only real but is rapidly expanding.

The ever-increasing complexity of silicon functionality can only be exploited if there is a paradigm shift in the design methodology used in silicon system design toward reuse-based design. Designers will increasingly employ reusable cores, often from external sources, and interconnect them on a chip, in a fashion similar to the way electronic subsystems from differing companies are now assembled into complete systems on a printed circuit board [3]. Third-party IP vendors are already appearing [e.g., MIPS and advanced RISC machines (ARM's)], while organizations such as the DSP Group are solely concerned with developing and licensing IP for system level IC's, such as the OakDSPCore. This is a 16-bit general purpose low-power, low-voltage, and high-speed digital signal processing (DSP) core designed for speech/audio processing, telecommunications, digital cellular, and embedded control applications. Designers are used to incorporating a reusable core such as a memory layout within a company, but the concept of using a reusable core from an external source has important issues related to IP protection, testability, etc.

An ever-increasing move toward this reuse-based design philosophy implies a growing need for the development of a systematic methodology to protect the intellectual property of differing designers and companies. Unless this is achieved, cooperation between companies will not be possible, and the potential of future silicon fabrication technology cannot be fully exploited. There have been several attempts to facilitate this. Products are now available that allow designers to interchange VHDL or Verilog code models for a reusable core via commercial software packages. One example is the Verilog model com-

Manuscript received July 16, 1999; revised September 6, 1999. The associate editor coordinating the review of this paper and approving it for publication was Editor-in-Chief José M. F. Moura.

The authors are with the Electronic and Electrical Engineering Department, University of Strathclyde, Glasgow, U.K. (e-mail: r.chapman@eee.strath.ac.uk; durrani@strath.ac.uk).

Publisher Item Identifier S 1053-587X(00)01556-7.

piler (VMC) from Synopsis, which claims to create secure simulation models by compiling a Verilog-HDL source code model into a binary object model that cannot be deciphered or reverse engineered. These packages allow the potential user to alter design parameters such as word size, clock rates, etc., and simulate the performance of the reusable core within the total design but hide the vendor's VHDL or Verilog code from the customer.

The techniques outlined above are based on attempts to prevent an unscrupulous user from misusing third-party IP by some form of software guard. This paper examines IP protection from a differing viewpoint and proposes steps that should be taken at the reusable core design stage to personalize the design in such a way that proof of ownership can be authenticated in any legal dispute over IP violation.

More specifically, this paper addresses the problem of protecting basic DSP cores that may be integrated into a multiplicity of systems. Fundamentally, this paper is concerned with the protection of the design of systems that implement classical DSP algorithms by using additional constraints in the algorithm design process to encode an authorship signature, which will be difficult to detect. Such a technique has been proposed in [4], where it is shown that additional synthesis constraints could be applied throughout the whole design process, from algorithm, through system and behavioral synthesis, to logic synthesis and physical design. An example is given in [7] of applying additional constraints to a digital filter specification to make an individual designer's filter specification difficult to replicate. This paper tackles the problem of protecting the algorithmic intellectual property of DSP cores that contain filters and FFT building blocks and addresses practical issues of implementation not covered in [7]. The techniques proposed can also be used for other fundamental DSP algorithms such as convolution and correlation.

In the era of design reuse, it will be necessary to protect intellectual property at all levels of design abstraction. However, it is particularly attractive at the highest algorithmic level because whatever may be changed by an unauthorized user at either behavioral, logic, and physical synthesis level, the proof of authorship has not been compromised. IP protection has been investigated at the behavioral [11] and physical synthesis levels [8]. IP protection techniques have been applied to systems implemented on programmable hardware [9], [10].

Watermarking techniques [5], [6] have been proposed for different data modalities such as image, video, voice, and text. These techniques have tended to alter a substantial number of components in the watermarked objects and rely on the inability of human eyes and ears to detect the changes made to images and audio. This is permissible in multimedia applications but is unacceptable in DSP, where functionality and correctness of design is important.

Section II considers how designs containing digital filters can be protected. The case of IIR filter techniques to embed covert channels into the design to enable design authors to uniquely observe channel information is presented. A more analytical approach based on constrained minimization is also discussed in Section II for the design protection of FIR filters. With minor modifications, the covert channel approach could be used for FIR filter protection and vice versa.

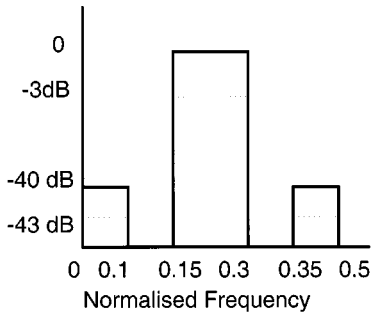


Fig. 1. Specification of passband and stopbands for bandpass digital filter design.

Section III discusses how system designs that utilize DSP algorithms containing window functions may be protected using watermarking techniques on the windowing functions, and Section IV discusses the similarities and differences of IP protection in virtual cores and multimedia data modalities. The overall conclusions are presented in Section V.

II. Digital Filter Designs

It has been proposed [7] that the use of covert channels could be used to protect the IP of DSP designs at an algorithmic level. The key concept was that a covert channel could be embedded into a design in such a way that only the design authors can observe and interpret information obtained through the channel. While [7] suggests a possible methodology, the present paper extends these ideas and addresses issues of practical filter implementation based on a rigorous mathematical analysis of the algorithm design process.

A. IIR Filter Design Protection: An Illustration

Initially, the design of IIR digital filters will be considered. It is assumed that a standard design package such as the signal processing toolbox in MATLAB is to be used to design the filter to a given specification. The challenge is to be able to prove that the design obtained using a common software package given a standard specification is unique.

As a preamble to the formal design process developed in this paper, an illustrative example is considered here to explain the proposed approach. Consider designing a bandpass filter with the specification shown in Fig. 1. The filter must have a maximum 3-dB ripple in the passband that is between normalized frequencies 0.15 and 0.3 Hz. The stopbands between 0–0.1 Hz and 0.35–0.5 Hz must have an attenuation of at least 40 dB. This filter is to be implemented by a transfer function of the form

$$H(z) = \frac{b_1 + b_2 z^{-1} + b_3 z^{-2} + \cdots b_N z^{N-1}}{1 + a_2 z^{-1} + a_3 z^{-2} + \cdots a_N z^{N-1}}. \qquad (1)$$

If MATLAB is used to design this digital filter, then a 16th-order Butterworth filter with normalized passband frequencies at 0.1476 and 0.3028 Hz is found to meet this specification. The coefficients of this filter are given in Table I, and the magnitude response is shown in Fig. 2. It is important to note that anyone using a classic design technique implemented in a commercial software package such as MATLAB will arrive at exactly the

TABLE I
COMPARISON OF BUTTERWORTH AND CODED YULE WALKER FILTER COEFFICIENTS

Butterworth Vector b	Butterworth Vector a	Yule Walker Vector b	Yule Walker Vector a
0.0004	1.0	0.0456	1.0
0.0	-1.9335	-0.0228	-0.9507
-0.0036	4.6765	0.0105	3.0222
0.0	-6.2011	-0.0323	-2.32
0.0126	9.1576	0.0756	4.706
0.0	-9.3027	0.0061	-2.3978
-0.0251	10.0491	-0.0110	3.9432
0.0	-8.0482	-0.0319	-0.8968
0.0314	6.7680	0.0232	1.78
0.0	-4.2761	0.0281	0.5517
-0.0251	2.8424	-0.0021	0.226
0.0	-1.3776	-0.0156	0.7367
0.0126	0.7174	-0.0028	-0.125
0.0	-0.2469	0.007	0.3158
-0.0036	0.0970	0.0029	-0.0372
0.0	-0.0188	-0.0002	0.049
0.0004	0.0051	-0.0023	0.0015

same filter coefficients, and it will not be possible to distinguish one design from another.

To protect the design IP, the aim is therefore to do the following.

- Make the filter design more unique to the individual.
- Any subsequent filter must have the same complexity as the standard filter (i.e., 16th order) to ensure that there is no additional implementation overhead.
- The same standard commercial design software must be used.
- Any new filter must meet the original specifications.

To achieve these objectives, the filter stopbands and passband were subdivided into seven regions with a bandwidth of 0.05 Hz. It is then possible to modify the filter specification for IP protection by embedding a 7-bit code word into the seven regions according to a prescribed procedure as given below. In the example shown in Table II, the ASCI character "C," which is equivalent to (1 000 011), is used as the code.

The code is then used to alter the original filter specification. In this example, a possible 3-dB ripple has been superimposed in the stopband. If the code in the filter band is "1," then 1 dB is subtracted from the upper band edge (i.e., −1 dB in the passband or −41 dB in the stopband). Similarly, if the code is "0," then 1 dB is added to the lower band edge.

Any filter that meets this modified specification must also satisfy the original one. An obvious problem is that standard commercial Butterworth filter design packages cannot easily cope with this more piecemeal specification. However, specifications typified by those in Table II may be used in more direct optimization routines. If the specification from Table II is used as input to the Yule-Walker filter design routines in MATLAB, the coefficients reproduced in Table I result. Fig. 2 illustrates the magnitude response of the standard Butterworth and Yule-Walker filters.

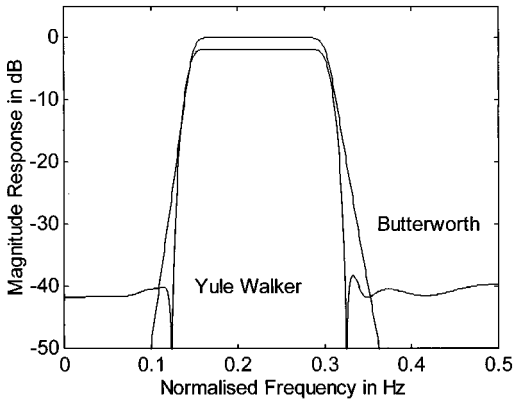


Fig. 2. Comparison between Butterworth and coded Yule-Walker filter magnitude responses. The solid line shows the Butterworth response, and the dashed line shows the Yule–Walker-based design.

TABLE II
CODED VARIATION TO FILTER SPECIFICATION IN FIG. 1

Band	Code	Magnitude Specification
0 - 0.05 Hz	1	-41 dB
0.05 – 0.1 Hz	0	-42 dB
0.15 – 0.2 Hz	0	-2 dB
0.2 – 0.25 Hz	0	-2 dB
0.25 – 0.3 Hz	0	-2 dB
0.35 – 0.4 Hz	1	-41 dB
0.4 – 0.45 Hz	1	-41 dB

Both of the filter designs in Fig. 2 meet the original specification and both filters are of the same complexity. Both have been designed using standard commercial software. However the Yule-Walker design has an embedded code word within it known only to the designer. By the time the filter has been implemented in hardware and/or software, it will become increasingly difficult for any unauthorized user of the IP to determine whether any algorithmic IP protection has been incorporated or not.

One obvious criticism of the watermarking example given above is that in many applications a Butterworth (or some other standard design procedure) may be part of the original specification. In these circumstances, it is not possible to introduce simple magnitude scaling as illustrated earlier without writing nonstandard CAD software. Even if this was done, the order and, thereby, complexity of the overall filter would have to increase. Coded Butterworth filters can be designed if the band edges as well as the passband ripple and stopband attenuation are included in the coding. This is illustrated in Fig. 3.

A 6-bit codeword can be used that incorporates the passband attenuation (D), the stopband attenuation (A), and the band-edges (B, C, E, and F). If a bit in the codeword is "0," then the original specification value is used. If a bit equals "1," then that particular specification value is made slightly more demanding. If all possible codewords are to be used, then it is necessary to

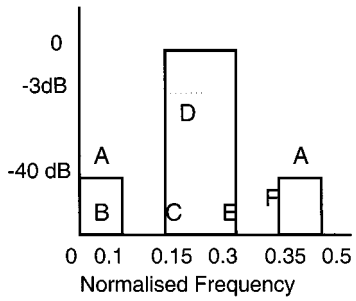


Fig. 3. Code parameters for Butterworth bandpass filter.

TABLE III
VARIATION IN A, B, C, D, E, F TO MODIFY SPECIFICATION IN FIG. 3, WHEN CODE BIT EQUALS "1"

A	B	C	D	E	F
-0.1dB	+0.00125Hz	-0.00125Hz	+0.1dB	+0.00125Hz	-0.00125Hz

ensure that the word (111 111) does not tighten the specification to the point that a higher order filter is required than needed for the original specification.

As an illustration, consider the bandpass filter specification given in Fig. 3, and assume that the specifications for attenuation levels and band edges are tightened in the manner shown in Table III if the relevant code bit equals a 1.

If a code word of decimal 25 was used (equivalent to 011 001), then the bandpass filter specification becomes

- stopband edges 0.101 25 and 0.348 75 Hz;
- passband edges 0.148 75 and 0.3 Hz;

while the passband and stopband attenuation specification remains unaltered.

Since the code word (decimal 25) has subtly altered the passband and stopband edges from those illustrated in Fig. 1, then a 16th-order Butterworth filter with passband frequencies of 0.1474 and 0.300 82 Hz is required to meet this modified specification. When such a filter is subsequently designed, the filter coefficients are given in Table IV. It is seen that these coefficients are slightly different from the filter coefficients that meet the original specification. The magnitude responses of both the standard Butterworth filter and the coded Butterworth filter are shown in Fig. 4. It is seen that for most practical purposes, the coded Butterworth response would be deemed acceptable.

In practice, a 16th-order filter would probably be implemented as eight second-order filters in cascade. Assuming that the poles of the original Butterworth bandpass filter at 0.5541 $\pm j0.7444$ were grouped with the zeros at $-1.0267 \pm j0.0114$, it would produce a second-order section with the following coefficients:

$$H_i(z) = \frac{1 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} = \frac{1 + 2.0534 z^{-1} + 1.0542 z^{-2}}{1 - 1.1082 z^{-1} + 0.8612 z^{-2}}.$$

The corresponding poles and zeros for the coded design are at $0.5553 \pm j0.7440$ and $-1.0356 \pm j0.0151$, respectively, and this

TABLE IV
VALUES OF STANDARD BUTTERWORTH AND CODED BUTTERWORTH COEFFICIENTS FOR 16TH-ORDER BANDPASS FILTER DESIGN

Original vector b	Coded Vector b	Original vector a	Coded vector a
0.0004	0.0004	1.0	1.0
0.0	0.0	-1.9341	-1.9919
-0.0036	-0.0034	4.6765	4.8098
0.0	0.0	-6.2019	-6.4929
0.0126	0.012	9.1569	9.5605
0.0	0.0	-9.3026	-9.8364
-0.0251	-0.0241	10.0472	10.5902
0.0	0.0	-8.0470	-8.5669
0.0314	0.0301	6.7658	7.1708
0.0	0.0	-4.2750	-4.5693
-0.0251	-0.0241	2.8411	3.0179
0.0	0.0	-1.3770	-1.4742
0.0126	0.012	0.7717	0.7606
0.0	0.0	-0.2468	-0.2640
-0.0036	-0.0034	0.0969	0.1022
0.0	0.0	-0.0188	-0.02
0.0004	0.004	0.0051	0.0053

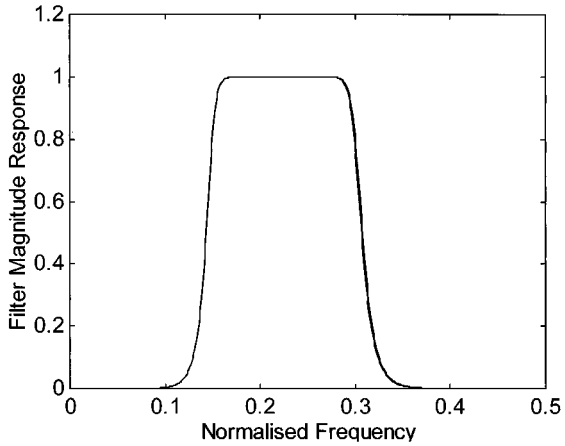


Fig. 4. Comparison between 16th-order Butterworth filter response and coded Butterworth filter response. The solid line represents standard response, and the dotted line represents coded Butterworth response.

produces the following equivalent second-order section:

$$H_i(z) = \frac{1 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} = \frac{1 + 2.0706 z^{-1} + 1.0721 z^{-2}}{1 - 1.1106 z^{-1} + 0.8619 z^{-2}}.$$

If the coefficients of both these second-order sections were coded in 8-bit 2's complement format with a quantization step of 0.017 (which allows the maximum filter coefficient to be coded with 7 bits) and rounding was used, then the stored coefficients in this implementation are given in Table V. The denominator coefficients are identical, and the numerator coefficients vary by 1 bit.

Some of the 2's complement coefficients for all eight of the coded filters second order sections would differ from those of the original filter by at most one bit, but these discrepancies are absolutely predictable providing one knows the coding scheme. Anyone misusing an overall design would only have access to the stored coefficient data and probably the original filter specification. In any dispute they would have difficulty in demonstrating how they had derived the stored coefficient data. Obviously unauthorized users of the IP could derive their own filter, but they are already faced with carefully examining the behav-

TABLE V
2'S COMPLEMENT FILTER COEFFICIENTS FOR ONE SECOND-ORDER SECTION OF THE 16TH-ORDER BUTTERWORTH AND CODED BUTTERWORTH BANDPASS FILTER

	b_1	b_2	a_1	a_2
Standard	01111001	00111110	11000001	00110011
Coded	01111010	00111111	11000001	00110011

ioral and physical design of any reusable core to ascertain what IP protection measures may have been implemented. If they are now faced with checking the fundamental algorithms as well, then it is becoming too costly and problematic to contemplate reusable core misuse.

B. FIR Filter Design Protection—An Analytic Approach

A variation to the above approach will be taken to protecting the design of FIR filters. The standard window based technique will be modified to represent it as a constrained optimization problem in which the constraints contain the essential coding to protect the design.

Consider a $(2N+1)$th-order zero-phase FIR filter of the form

$$H\left(e^{j\omega}\right) = \sum_{k=-N}^{N} a_k e^{-j\omega k} = \boldsymbol{A}^T \boldsymbol{W} \tag{2}$$

where

$$\boldsymbol{A}^T = [\,a_{-N} \quad a_{-(N-1)} \quad \cdots \quad a_0 \quad \cdots \quad a_{N-1} \quad a_N\,]$$
$$\boldsymbol{W}^T = [\,e^{j\omega N} \quad e^{j\omega(N-1)} \quad \cdots \quad 1 \quad \cdots \quad e^{-j\omega(N-1)} \quad e^{-j\omega N}\,].$$

It is assumed that the sampling period T has been normalized to unity. The designed filter can be made causal by introducing a linear phase shift at the implementation stage. The coefficient vector $\boldsymbol{A}$ is obtained by minimizing, in a squared error sense, a cost function that is the area between some ideal frequency response $H_i(e^{j\omega})$ and $H(e^{j\omega})$ over one period, as given by

$$J = \int_{-\pi}^{\pi} \left(H_i\left(e^{j\omega}\right) - H\left(e^{j\omega}\right)\right)^2 d\omega = \int_{-\pi}^{\pi} \left(H_i\left(e^{j\omega}\right) - \boldsymbol{A}^T\boldsymbol{W}\right)^2 d\omega. \tag{3}$$

This leads to the classic result for the filter coefficients using the window method as

$$\boldsymbol{A}_{opt} = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_i\left(e^{j\omega}\right)\boldsymbol{W}\, d\omega. \tag{4}$$

In practice, the filter coefficients given by (4) are usually modified by using a nonrectangular window so that the oscillations in the magnitude response due to Gibb's phenomena may be reduced.

The well-established FIR filter design based on the windowing method is automated in many DSP packages such as MATLAB. For a given specification, all designers would obtain the same filter coefficients, and if the resulting filter was incorporated into a reusable core, it would never be possible to ascertain design IP ownership by examining the filter coefficients. We propose a technique that modifies the results achieved by the windowing method that ensures protection of IP.

Assume the magnitude response is constrained to be equal to some predetermined values at P points in the frequency range of the filter chosen by the designer. Obviously, in practice, these magnitude values will be very close to those of the unconstrained filter at the chosen frequency points to ensure that the constrained filter still meets the original specification. Mathematically, the set of P constraints are

$$H\left(e^{j\omega_p}\right) = g_p \qquad p = 1, 2, \cdots, P.$$

This may be restated as

$$\boldsymbol{E}\boldsymbol{A} = \boldsymbol{g} \tag{5}$$

where

$$\boldsymbol{E} = \begin{bmatrix} e^{j\omega_1} & \cdots & 1 & \cdots & e^{-j\omega_1} \\ \vdots & & & & \vdots \\ \vdots & & & & \vdots \\ e^{j\omega_P} & \cdots & 1 & \cdots & e^{-j\omega_P} \end{bmatrix}$$

and

$$\boldsymbol{g}^T = [\,g_1 \quad g_2 \quad \cdots \quad g_P\,].$$

The constrained cost function to be minimized is

$$J = \int_{-\pi}^{\pi} \left(H_i\left(e^{j\omega}\right) - H\left(e^{j\omega}\right)\right)^2 d\omega + \sum_{p=1}^{P} \lambda_p \left(H\left(e^{\omega_p}\right) - g_p\right) \tag{6}$$

where λ_p are Lagrangian multipliers.

Substituting (2) into (6)

$$J = \int_{-\pi}^{\pi} \left(H_i\left(e^{j\omega}\right) - \boldsymbol{A}^T\boldsymbol{W}\right)^2 d\omega + \sum_{p=1}^{P} \lambda_p \left(\boldsymbol{A}^T\boldsymbol{W}(\omega_p) - g_p\right) \tag{7}$$

where

$$\boldsymbol{W}(\omega_p)^T = [\,e^{j\omega_p N} \quad e^{j\omega_p(N-1)} \quad \cdots \quad 1 \quad \cdots \quad e^{-j\omega_p(N-1)} \quad e^{-j\omega_p N}\,].$$

Differentiating (7) with respect to $\boldsymbol{A}$ and equating the derivative to zero produces

$$\boldsymbol{A}_c = \left(\int_{-\pi}^{\pi} \boldsymbol{W}\boldsymbol{W}^T d\omega\right)^{-1} \cdot \left(\int_{-\pi}^{\pi} H_i\left(e^{j\omega}\right)\boldsymbol{W}\, d\omega + \tfrac{1}{2}\sum_{p=1}^{P} \lambda_p \boldsymbol{W}(\omega_p)\right). \tag{8}$$

The new constrained filter coefficients $\boldsymbol{A}_c$ become

$$\boldsymbol{A}_c = \boldsymbol{A}_{opt} + \frac{1}{4\pi}\sum_{p=1}^{P} \lambda_p \boldsymbol{W}(\omega_p) = \boldsymbol{A}_{opt} + \frac{1}{4\pi}\boldsymbol{E}^T\lambda \tag{9}$$

where

$$\lambda^T = (\,\lambda_1 \quad \lambda_2 \quad \cdots \quad \lambda_P\,).$$

Substituting (9) into the constraint (5) produces

$$\boldsymbol{g} = \boldsymbol{E}\boldsymbol{A}_{opt} + \frac{1}{4\pi}\boldsymbol{E}\boldsymbol{E}^T\lambda$$

which can be manipulated to yield

$$\lambda = 4\pi\left(\boldsymbol{E}\boldsymbol{E}^T\right)^{-1}\boldsymbol{g} - 4\pi\left(\boldsymbol{E}\boldsymbol{E}^T\right)^{-1}\boldsymbol{E}\boldsymbol{A}_{opt}. \tag{10}$$

Substituting (10) into (9) results in an expression for the modified filter coefficient $\boldsymbol{A}_c$ in terms of the classic FIR rectangular

TABLE VI
COEFFICIENTS OF THE STANDARD FIR FILTER AND THE CODED FIR FILTER OBTAINED VIA CONSTRAINED MINIMIZATION

Standard Filter Coefficients	Coded Filter Coefficients
-0.0126	-0.0123
0.0111	0.0094
-0.0168	-0.0218
0.0556	0.0554
0.1069	0.1030
-0.0831	-0.0815
-0.2138	-0.2107
0.0395	0.0423
0.2618	0.2690
0.0395	0.0423
-0.2138	-0.2107
-0.0831	-0.0815
0.1069	0.1030
0.0556	0.0554
-0.0168	-0.0218
0.0111	0.0094
-0.0126	-0.0123

window coefficients and the designer-imposed magnitude constraints

$$\boldsymbol{A}_c = \left(\boldsymbol{I} - \boldsymbol{E}^T\left(\boldsymbol{E}\boldsymbol{E}^T\right)^{-1}\boldsymbol{E}\right)\boldsymbol{A}_{opt} + \boldsymbol{E}^T\left(\boldsymbol{E}\boldsymbol{E}^T\right)^{-1}\boldsymbol{g}. \quad (11)$$

Consider the design of a 16th-order (17 coefficient) bandpass filter with passband edges at normalized frequencies 0.15 and 0.3 Hz. The coefficients for the FIR filter obtained using the classical window method, using a rectangular window function, are given in Table VI. To utilize the analysis derived above, the filter magnitude was initially calculated at the four frequencies in the filter stopband shown in Table VII.

These magnitude values can be slightly altered without compromising the overall filter behavior significantly. The value of $\boldsymbol{g}$ and the frequency values required to determine $\boldsymbol{E}$ in (11) were obtained by representing the magnitude response values in Table VII to three significant figures.

These coefficient values for the constrained optimization procedure expressed by (11) are given in Table VI, and the magnitude responses of both the classical window design and the coded design are compared in Fig. 5. The ripple structure in the stopbands is slightly altered, as one would expect since the constraints were all imposed in the stopbands, but the passband responses are virtually identical. The Gibb's phenomena in both filter responses in Fig. 5 could be improved if the coefficients given in Table VI were modified using any of the nonrectangular window functions [14].

A variation to this example is to determine the exact frequencies where the nulls occur in the magnitude response of the classical filter. This is easily obtained from the zeros of the transfer function located on the unit circle in the z domain. The frequency vector $\boldsymbol{W}$ in (6) can be set to a slight variation of these null frequencies, and the vector $\boldsymbol{g}$ in (4) contains only zeros. Once again, the stopband characteristics are subtly altered from those of the classic design, but the passband responses are virtually identical.

The coefficient modification procedure outlined above is only mathematically valid for FIR filters designed using the window method. However, the technique has been used for filters in which the coefficient vector $\boldsymbol{A}_{opt}$ in (11) has been obtained from FIR filters designed using the Parks and McClelland equiripple method [12] and from a least squares minimization technique [13]. In these instances, acceptable results were obtained, and therefore, the coefficient coding defined by (11) is more general than the analysis initially implies.

TABLE VII
VALUES OF THE STANDARD FIR FILTER EVALUATED AT FOUR NORMALIZED FREQUENCIES IN THE FILTER STOPBAND, AND THE COEFFICIENT VALUES OF THE VECTOR $\boldsymbol{g}$ USED IN THE CODED FIR FILTER DESIGN

Normalised Frequency	Magnitude Response	Vector g
0.05	0.0227	0.023
0.1	0.0266	0.027
0.4	0.0210	0.021
0.45	0.0484	0.048

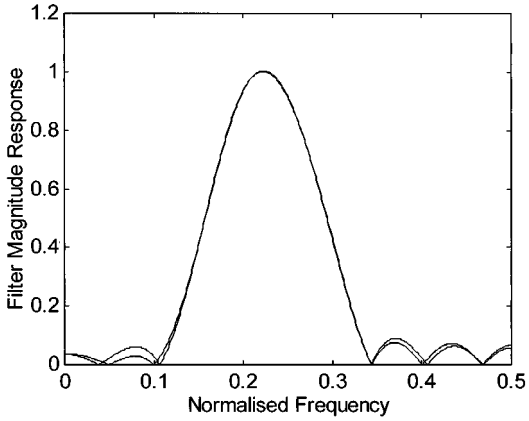


Fig. 5. Magnitude response of standard FIR filter and coded FIR filter. The solid line represents the standard filter, and the dashed line represents coded constraint-based design.

III. WINDOWING FUNCTION WATERMARKING

Window functions, which are used in FFT and filtering applications, are expressed as simple mathematical equations, and it may be thought that it would not be possible to detect from a window function the authenticity of a particular design. However, since window functions tend to be rather long, small variations can be introduced into them without significantly compromising the overall performance of the total algorithm. This will now be considered in more detail.

One of the significant features of window functions of length N is that they are symmetrical about their center term $N/2$, and it is imperative in any watermarking procedure that this symmetry is maintained. Starting with a classical window function

$$w(n) \qquad 1 \le n \le N \quad (12)$$

the first stage is to add a small amount of noise to the original window function

$$w_m(n) = w(n) + \alpha\sigma(n) \qquad 1 \le n \le N \quad (13)$$

where $\sigma(n)$ is a zero mean random sequence symmetrical about its center term $N/2$, and α is the small scalar term. The reason for adding noise to the original window function is to disguise the subsequent modification of the window function by some code word known only to the system designer.

Assume the designer has a code word $c(p)$ of length P and inserts it into the modified window sequence starting at posi-

TABLE VIII
COMPARISON BETWEEN COEFFICIENTS 25–28 FOR STANDARD HAMMING WINDOW AND CODED HAMMING WINDOW

Hamming	0.1581	0.1645	0.1712	0.1781
Coded	0.1684	0.1748	0.1751	0.1884

tion i. To ensure symmetry about the center term, the following equation must be satisfied:

$$w_c(n) = \begin{cases} w_m(n) & 1 \le n \le i-1 \\ w_m(n) + \beta c(n-i+1) & i \le n \le i+P-1 \\ w_m(n) & i+P \le n \le N/2 \end{cases}$$
$$w_c(n) = w_c(N+1-n) \qquad n = 1, 2, \cdots, N/2 \qquad (14)$$

where β is a small scalar value.

As an example of this technique, consider a 256-element Hamming window function. Noise was added to this window using a value for α in (13) of 10^{-3}. The code word used was "signal processing at Strathclyde is excellent." The binary representation of these ASCI characters was added to the modified window function starting at position 10, using a value for β in (14) of 10^{-4}. There are no particular reasons for these numerical choices for α and β, other than they produce a unique window function that compares favorably with the standard function.

The difference between the true Hamming window and the coded Hamming window for sample values 25–28 is shown in Table VIII. There is obviously a difference between the Hamming and coded window functions, but the performance of each window is very similar.

Fig. 6 shows the effect of using both these window functions on 256 samples of a sequence consisting of two sinusoids with normalized frequencies of 0.15 and 0.2 Hz. The average mean square error between the two windowed sequences in Fig. 6 is $3.69 * 10^{-3}$. The magnitude of the DFT for both these windowed sequences is shown in Fig. 7. The average mean square error between the magnitude response values in Fig. 7 is 0.0074.The experiment was repeated for 100 random input sequences. The overall average mean square error in the windowed data sequences was $1.16 * 10^{-5}$, the overall mean square average error in the DFT magnitude values was 0.0016, and the overall average mean square error in the DFT phase was 0.068 rad^2.

Only the original system designer knows the profile of the noise modified-window function $w_m(n)$, and therefore, no unauthorized user can determine the code word embedded in the coded window function.

IV. IP PROTECTION OF DSP DESIGNS

It is only with the introduction of SoC, which is a completely new design paradigm, that IP protection has become a major issue in chip design. While watermarking techniques have been used in other areas such as image and audio protection for some time [5], [6], [15], it is interesting to note the similarities and differences between these two application areas. The first point to note is that in image processing, IP is being protected in very unstructured environments such as the Internet. Reusable cores will only be interchanged between companies if the interchange environment is more regulated and structured. Several commercial companies have been set up to facilitate the interchange of SoC IP, including

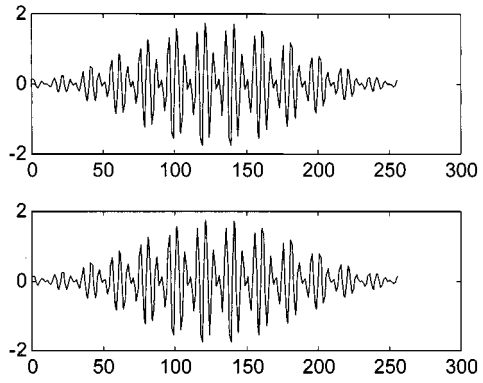


Fig. 6. Comparison of data windowed by standard Hamming window and coded Hamming window. The upper trace shows results for standard window, and the lower trace shows results for coded window.

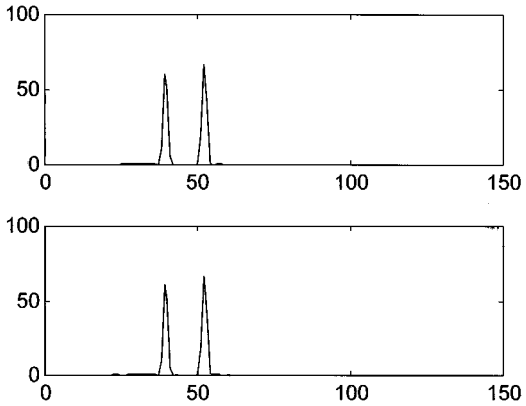


Fig. 7. Magnitude response of DFT of the data shown in Fig. 6. The upper trace shows the results obtained using the Hamming window, and the lower trace shows the results obtained for the coded Hamming window.

virtual component exchange (VCX) in Livingston, U.K. [18], and rapid application specific intellectual property developers (RAPID) in Campbell, CA, USA [19]. Both companies have recently set up a joint task force to share ideas and expertise.

Most watermarking procedures within image processing have concentrated on the robustness of the techniques presented, and Memon *et al.* [16] have correctly noted that it is far more important to concentrate on how legally effective the technique is in IP protection. They have shown that within image processing, it is possible for the unscrupulous to create a bogus "original" image and claim to have inserted a watermark that not only appears in their protected version but in the true copyright owner's watermarked versions as well. In such circumstances, the legal copyright owner cannot be legally verified by many of the watermarking techniques that have been proposed.

The application described in this paper is rather different. It is not the IP of the algorithms to which the watermarking techniques discussed in this paper have been applied that is being protected, since they are all standard well-documented techniques, but the chip design that implements these algorithms as part of a more complex system. It should be noted that IP protection techniques can be applied at all the design abstraction levels of the chip, such as basic cell implementations, logic structures, and register transfer level descriptions of various subcomponents of the overall design. The thrust of this paper is that as well as applying IP protection at the physical and behavior descriptions

of the IP core; it is also possible to apply IP protection within the fundamental computational algorithms that are being implemented. This would ensure that it would become extremely time consuming and costly to attempt to detect and circumvent the IP protection mechanism throughout the whole design flow.

In practice, the IP protection mechanisms used throughout the complete design flow from algorithms to cell layout would be registered with a third party such as VCX before any exchange of IP was contemplated. In this way, it is hoped that IP exchange within the electronic chip design fraternity can be eased.

Notwithstanding the comments above on the differences between IP protection applied to multimedia data and virtual cores for SoC, it is fair to say that IP protection within chip design is in its infancy, and the methodology proposed by Memon *et al.* [16] for images is well worth noting, in which the effectiveness as well as the implementation technique for any IP protection technique be considered jointly.

V. Conclusions and Applications

This paper has considered techniques to protect intellectual property of systems that incorporate standard DSP algorithms. It has been shown that for two of the most fundamental algorithms (a digital filter and the calculation of the DFT of a sequence), it is possible to embed a unique code word into the algorithm to enable proof of system design to be verified. In the predicted era of system on chip, when systems may be built on silicon using virtual cores from a multiplicity of sources, it will become increasingly important to protect intellectual property. This paper has illustrated that IP protection mechanisms can be employed at the highest abstraction level in the design process where such techniques provide maximum benefit.

The thrust of this paper has been to investigate the possibility of introducing some form of IP protection at one of the highest abstraction levels in the overall design process. There are many other issues facing a chip design team. One problem is the complexity of the algorithms to be implemented. This may be particularly important in low power designs. The protected algorithms in this paper have identical complexity to their nonprotected classical counterparts if complexity is defined as the number of arithmetic operations or memory locations used. A recent publication has suggested [17] that additional constraints could be included in the design of FIR filters to obtain filter coefficients that are optimized for low power realizations. A fruitful area of research may be to attempt to use a constrained optimization approach to design filters incorporating some form of IP protection and low power realizability.

References

[1] Virtual Socket Interface Alliance, "Architecture Document, Version 1.0,", http://www.vsi.org, Mar. 1997.

[2] Intel Corp., http://www.intel.com/pressroom/archive/speeches/GEM93097.HTM#IntelTop.

[3] S. Glaser, "IP fuels a transformation of culture, companies and cooperation," *Electron. Des.*, pp. 278–305, Jan. 12, 1998.

[4] B. Salefski, G. Martin, S. J. Krolikoski, and F. R. Schirrmeister, "Reuse driven methods can help designers optimize systems," *Electron. Des.*, pp. 82–86, June 22, 1998.

[5] L. Boney, A. H. Tewfik, and K. N. Hamdy, "Digital watermarks for audio signals," in *Proc. IEEE Int. Conf. Multimedia Comput. Syst.*, 1996, pp. 473–480.

[6] F. Hartung and B. Girod, "Digital watermarking of raw and compressed video," *SPIE*, vol. 2952, pp. 205–213, 1996.

[7] I. Hong and M. Potkonjak, "Techniques for intellectual property protection of DSP designs," in *Proc IEEE Int. Conf. Acoust., Speech, Signal Process.*, Seattle, WA, May 1998.

[8] A. B. Kahng, S. Mantik, I. L. Marcov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe, "Robust IP watermarking methodologies for physical design," in *Proc. ACM/IEEE Des. Automat. Conf.*, 1998.

[9] J. Lach, W. H. Mangione-Smith, and M. Potkonjak, "Fingerprinting digital circuits on programmable hardware," in *Proc. Workshop Inform. Hiding*, 1998.

[10] ——, "FPGA fingerprinting techniques for protecting intellectual property," in *Proc. CICC*, 1998.

[11] A. B. Kahng, J. Lach, W. H. Magione-Smith, S. Mantik, I. L. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe, "Watermarking techniques for intellectual property protection," in *Proc. 35th ACM/IEEE DAC Des. Automat. Conf.*, San Francisco, CA, June 1988, pp. 776–781.

[12] L. R. Rabiner, J. H. McClellan, and T. W. Parks, "FIR digital filter design techniques using weighted Chebyshev approximation," *Proc. IEEE*, vol. 63, 1975.

[13] T. W. Parks and C. S. Burrus, *Digital Filter Design*. New York: Wiley, 1987, pp. 54–83.

[14] B. Mulgrew, P. M. Grant, and J. S. Thompson, *Digital Signal Processing, Concepts and Applications*. New York: Macmillan, 1999.

[15] L. Cox, J. Kilian, T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," NEC Res. Inst., Tech. Rep. 95-10, 1995.

[16] S. Cracer, N. Memon, Y. Boon-Lock, and M. M. Yeung, "Resolving rightful ownership with invisible watermarking techniques: Limitations, attacks, and implications," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 573–586, May 1998.

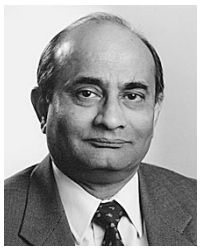
[17] M. Mehendale, S. D. Stierleckar, and G. Venkapesh, "Low power realization of FIR filters on programmable DSPs," *IEEE Trans. VLSI Syst.*, vol. 6, pp. 546–553, Dec. 1998.

[18] Virtual Component Exchange, http://www.vcx.org.

[19] Reusable Application-Specific Intellectual Property Developers, http://rapid.org.

Roy Chapman received the B.Sc. and M.Sc. degrees from the University of Newcastle upon Tyne, Newcastle upon Tyne, U.K.

He has been with the Signal Processing Division, Electronic and Electrical Engineering Department, University of Strathclyde, Glasgow, U.K., since 1981. He currently holds the post of Senior Lecturer. He worked at AEI Electronic Apparatus Division Lincoln, U.K., English Electric Valve Company, Lincoln and Chelmsford, U.K., the University of Newcastle upon Tyne and the University of Paisley, Paisley, U.K., before joining Strathclyde. His current research interests include filter design and implementation, higher order statistics, sonar signal processing, and implementation of signal processing algorithms in VLSI.



Tariq S. Durrani (F'89) received the B.Eng. degree from the University of Engineering and Technology, Dhaka, Bangladesh, in 1965, and the M.Sc. and Ph.D degrees in 1967 and 1970, respectively, from the University of Southampton, Southampton U.K.

After postdoctoral research at Southampton, he joined the University of Strathcylde, Glasgow, U.K., as a Lecturer in 1976 and was appointed Professor of Signal Processing in 1982. He was Head of the Department of Electronic and Electrical Engineering from 1986 to 1990 and Deputy Principal for Information Technology from 1990 to 1991. For the past 25 years, he has worked on and supervised some 60 projects, sponsored by the EPSRC, U.K. Department of Trade and Industry, Ministry of Defense, the U.S. Navy, EEC-ESPRIT, EU BRITE-EURAM, RACE, and by several industrial organizations. He has supervised more than 30 Ph.D. students. His research interests are in the areas of adaptive and nonlinear signal processing, image processing and multimedia systems, and technology management. He has published more than 270 papers and coauthored/coedited six books.

Prof. Durrani was the 1994–1995 President of the IEEE Signal Processing Society, the 1996–1997 Chair of the IEEE Periodicals Council, and the 1998–1999 Chair of the IEEE Periodicals Review Committee. He was Chairman of the IEE Professional Group E5 on Image Processing from 1983 to 1987 and Professional Group E4 on Signal Processing from 1987 to 1990. He is a Fellow of the Royal Academy of Engineering, the IEE, and the Royal Society of Edinburgh.