VLSI Implementation of an AMDF Pitch Detector

T. D. Smith[2], F. Gittel[1, 2], A. Th. Schwarzbacher[2], E. Hilt[1] and J. T. Timoney[3]

[1]Deutsche Telekom University of Applied Sciences, Leipzig, Germany.
E-mail: falkogittel@web.de
hilt.e@berlin.de

[2] Dublin Institute of Technology School of Electronic and Communications Engineering, Dublin 6, Ireland.
E-mail: Tony.Smith@dit.ie
Andreas.Schwarzbacher@dit.ie

[3]National University of Ireland, Department of Computer Science, Maynooth, Ireland.
Email :jtimoney@cs.may.ie

Abstract -- **Pitch detectors are used in a variety of speech processing applications such as speech recognition systems where the pitch of the speaker is used as one parameter for identification purposes. Furthermore, pitch detectors are also used with adaptive filters to achieve high quality adaptive noise cancellation of speech signals [1]. In voice conversion systems, pitch detection is an essential step since the pitch of the modified signal is altered to model the target voice [2].**

This paper describes a VLSI implementation of the computationally efficient and accurate pitch detection algorithm known as the Average Magnitude Difference Function (AMDF). The superior speed of a hardware pitch detector is essential particularly for use in real-time signal processing devices such as mobile phones.

Keywords: **Pitch Detection, AMDF, VLSI, CMOS Design**

I INTRODUCTION

Pitch detection is an important step in many digital speech processing systems. Various algorithms exist for this purpose but for real-time VLSI implementation, the most computationally efficient algorithms must be chosen. This paper describes the VLSI design of such a pitch detection algorithm.

The pitch detector developed will be incorporated into an adaptive noise canceller in conjunction with an adaptive filter where the pitch period will be used as a delay step before the adaptive filter and acts effectively as an enable to update the filter coefficients. The speech delayed by one pitch period is highly correlated with the original speech and thus the adaptive filter can remove the noise components of the speech signal [1]. For potential implementation into a voice conversion system, the pitch detector can be used as a speaker recognition step or alternatively the pitch may be shifted to change the speech characteristics of a speaker.

a) Pitch

When generating speech sounds the lungs act as an air reservoir and bellows, pushing air between ligaments called the vocal cords which forces them to vibrate by opening and closing. The area between the vocal cords is known as the glottis. The rate at which this vibration occurs is called the pitch of the resulting speech signal. The pitch is often referred to as the fundamental frequency, F_0. The pitch period, T, is related to the fundamental frequency by $T=1/F_0$. While the vocal cords are tensed the air passing through them causes the cords to vibrate at a higher pitch. When relaxed, the vocal cords vibrate slower, causing a lower pitch. If not speaking, the vocal cords are moved farther apart in the back, widening the passage for breathing. In males, the vocal cords are generally more relaxed than in females which thus leads to the distinctive lower pitch in a male's voice. Typically, in humans pitch frequency ranges from about 50 Hz – 500 Hz. This is approximately 50 Hz to 250 Hz for males and as high as 500 Hz for females [3].

b) Voiced and Unvoiced Speech

The decision whether a given segment of speech is classified as voiced or unvoiced is crucial to the accurate operation of the pitch detector that is implemented. Forcing air through the glottis while vibrating the vocal cords produces voiced speech. It produces a quasi-periodic signal in the time-domain and results in a spectrum of clearly defined pitch and harmonics at multiples of this frequency. Vowel sounds are typical of voiced speech, e.g. "eee". Pitch can only be measured if a particular segment of speech is classified as voiced.

Unvoiced speech is produced when no vibrations take place in the vocal cords. The resulting speech signal is non-periodic and random in the time-domain. The spectrum is broadband which makes it almost indistinguishable from noise. Some consonant sounds are typical of unvoiced speech, e.g. "s". If a segment of speech is classified as unvoiced, the pitch cannot be measured. Therefore, a reliable pitch detector must make voiced/unvoiced decisions and only during periods of voiced speech provides a measurement of the pitch period, T.

Some speech segments consist of voiced and unvoiced excitation simultaneously such as "z". These speech sounds are more difficult to classify [4].

Figure 1 shows segments of approximately 40 ms of voiced (top) and unvoiced (bottom) speech in the time-domain sampled at 8 kHz. The periodic nature of voiced speech can be seen with the distance between the largest peaks corresponding to the pitch period. The non-periodic noise-like nature of unvoiced speech is also shown in Figure 1.

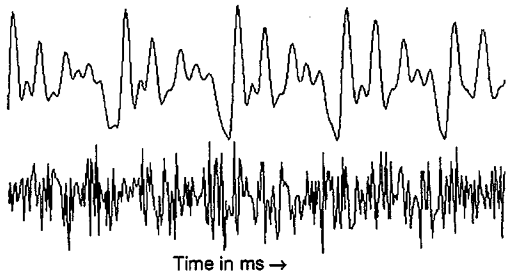

Figure 1: Voiced and Unvoiced Speech (40 ms)

II PITCH DETECTION ALGORITHM

Due to its importance in various applications, a wide variety of algorithms have been developed for pitch detection. These include the Autocorrelation method, Simplified Inverse Filtering Technique (SIFT), Data Reduction Method (DARD) and the Average Magnitude Difference Function (AMDF) [5]. Pitch detection algorithms can be divided into the following three broad categories [5]:

i) Algorithms that utilise the time-domain properties of speech signals.

ii) Algorithms that utilise the frequency-domain properties of speech signals.

iii) Algorithms that utilise a hybrid of both i) and ii).

The AMDF is a time-domain pitch detection algorithm. It is chosen because:

1) It provides a good estimate of pitch contour.
2) It has no multiply operations (unlike Autocorrelation).
3) It has relatively low computational cost.
4) The nature of its operations makes it suitable of implementation in special purpose hardware [6].

The AMDF is defined by the relation [6]:

$$D_{\tau} = \frac{1}{L}\sum_{j=1}^{L} | S_j - S_{j-\tau} | \text{ , } \tau{=}0,1,\ldots\tau_{max} \quad (1)$$

S_j are the samples of the input speech $(=S_1, S_2, \ldots S_L)$
$S_{j-\tau}$ are the samples shifted τ seconds
L is the number of delays or "lags"
τ is the delay value
τ_{max} is the maximum delay shift

In the AMDF, a difference signal is formed between the shifted speech and the original. At each delay, the absolute magnitude of the difference is taken. The difference signal is always zero when the delay, $\tau =$ 0, and exhibits deep nulls at delays corresponding to the pitch period of voiced sounds. In most AMDF pitch detectors the lag for which the magnitude of the difference function is a global minimum is chosen as the pitch period for that speech segment. The number of delays or "lags" per sample that is chosen is 64. This is a suitable number for an 8 kHz sampling rate and to obtain an accurate reflection of pitch period while keeping the computation low. In addition, for hardware implementation division by 64 in (1) is easily achieved by discarding the 6 least significant bits. Figure 2 shows a typical AMDF waveform of duration 20 ms. The global minimum corresponds to the pitch period as shown. Each AMDF point is calculated using the relation in (1). A second minimum is also apparent at twice the pitch period due to the second harmonic. This can sometimes fall lower than the actual pitch period, particularly in high noise signals. This is known as pitch period doubling.

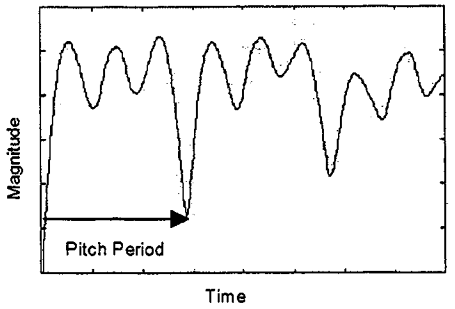

Figure 2: AMDF Waveform (20 ms)

III VOICED/UNVOICED DECISION

Three methods were implemented to determine whether a segment of speech is voiced or unvoiced. They are Short-Term Energy, Zero Crossings and the AMDF Max/Min Ratio. The Short-Term Energy and Zero Crossings functions complement each other and, therefore, can be used more accurately together to label different parts of speech [4]. By combining the three methods and using all of their advantages, a more accurate decision becomes increasingly probable.

a) Short-Term Energy

Short-Term Energy allows calculation of the amount of energy in a sound at a specific interval in time. It is purely dependent upon the energy in the signal, which depends on the method in which the sound is recorded. For example, if a person records the same phrase twice, one while whispering and one while shouting, then the Short-Term Energy values will be vastly different [4]. This is the main weakness in using energy as the sole method of determining the voiced/unvoiced decision. However, the standard property is that the energy is higher for voiced than unvoiced speech and should be zero for silent regions in clean recording of speech. Energy computations are performed on the original speech sample. The total energy of a 160-sample frame is defined by:

$$E = \sum_{n=1}^{160} n^2 \quad (2)$$

If the total energy of the frame crosses the defined energy threshold, the frame is classified as voiced. From observation, this threshold was chosen to be one quarter of the maximum energy in a frame. The reason for this is that the amplitude of most samples in a voiced frame is greater than one quarter of the normalised maximum. The reverse is the case for unvoiced frames.

b) Zero Crossings

Zero Crossings is defined as the number of times in a frame of speech that the amplitude of the sound wave changes sign. For a 20 ms 160-sample frame of clean speech, the zero crossing rate is approximately 24 or less for voiced speech, 100 for unvoiced speech and 0 for silence [4]. However, very few recordings consist of perfectly clean speech. This means that often there is some level of background noise, that interferes with the speech, causing the silent regions to actually have quite a high zero crossing rate. This is the reason why a shifted zero axis is used which is above the normal zero amplitude mean and thus eliminates the problem of high zero crossings during silent regions. The zero crossings rate in this implementation is chosen to be 18 positive zero crossings. If the number of crossings is below this number, the frame is labelled as voiced. If this number is exceeded, the frame is labelled as unvoiced.

c) AMDF Max/Min Ratio

When the AMDF for a frame has been calculated, another voicing decision is made by examining the ratio between the global maximum and global minimum in the frame [5]. The ratio for a voiced frame was found to be approximately greater than 3 for a voiced frame. If the ratio exceeds 3, the frame is classified as voiced. If it is less, the frame is unvoiced. This is because the relative amplitudes between samples in an unvoiced frame are less than that of a voiced frame.

III CMOS IMPLEMENTATION

The pitch detector was coded and tested in VHDL. The VHDL code was synthesised and tested using the Synopsys Design environment. Two different versions of the detector were developed – a serial version and a parallel version. A block diagram of the serial pitch detector is shown in Figure 3. The 8-bit speech samples are stored in a Latch-Multiplexer structure. The AMDF needs the first 64 samples of the previous frame and so the structure consists of 224 storage locations to store the samples as opposed to 160 (the frame size). The energy computation of the circuit uses add-shift multiplication. The AMDF Max/Min Ratio is computed as a final voicing decision and operates with subtract-shift division. The Min Location in the diagram is the pitch period, which corresponds to the location of the global minimum. The Moore Controller synchronises all modules and will only enable the AMDF to compute the pitch period if the energy and zero crossings modules verify that a frame is voiced. This saves power because the pitch will not be computed if the frame is unvoiced.

The Latch-Multiplexer structure, the AMDF Calculator and the Controller were implemented both serially and in parallel. The parallel version is more computationally efficient at 2724 clock cycles per frame but results in a larger silicon area. Depending on whether timing or silicon area is at a premium, a different version of the detector may be chosen. Assuming that the speech is sampled at 8 kHz, the clock speed of the parallel detector must be 21.8 MHz and 84.4 MHz for the serial version. The serial version would therefore lead to greater power consumption due to the required faster clock cycles of 10504 per frame – an unattractive property, particularly for use in battery-powered devices such as mobile phones.

IV PERFORMANCE AND RESULTS

The pitch detector was tested with 8 kHz quantised 8-bit speech samples. It was synthesised using the European Silicon Structures 0.7 µm technology. The silicon area of each module and the total silicon area (in μm^2) is shown in Figure 4. As can be seen, the total area consumption of the parallel version is approximately 30% greater than that of the serial implementation. The main reason for this is the larger Multiplexer. However, the parallel version needs only 25% of the clock frequency required for the serial one. The total silicon area for the serial version is about 7 mm^2 and just under 10 mm^2 for the parallel one – either version is small enough to fit into a small communication device. The data arrival time of both schematics is nearly equal, i.e. the propagation delay produced by the schematic is not greatly influenced by the varying functionalities. The different timing information is shown in Table 1.

	Serial Version	Parallel Version
Module	Data Arrival Time (ns)	Data Arrival Time (ns)
Latches	2.03	2.03
Multiplexer	2.02	2.04
Energy Calc.	1.97	1.97
Zero Crossings	1.92	1.92
AMDF Calc.	1.98	1.99
Min/Max Ratio	1.92	1.92
Controller	2.07	2.07
Combinedmodules	2.07	2.08

Table 1: Propagation Delays of Modules

The slower clock of the parallel implementation implies lower power consumption due to their proportional relationship. The serial version is smaller but due to necessity of a clock speed 4 times larger, this adversely affects the power efficiency.

Tests were carried out on the pitch detector using real speech samples with varying Signal-to-Noise ratios (SNRs) ranging from –10 dB to +25 dB. The performance of the detector worked well with high SNRs above 10 dB. The voiced/unvoiced classification worked perfectly and the pitch was computed accurately and updated every frame. The pitch extraction of the detector was very consistent and was capable of computing the pitch with SNRs better than or equal to 10 dB. For poor SNRs (below 10 dB) the results were not as good. With a large amount of noise, the voiced/unvoiced decision was sometimes inconsistent, particularly due to the AMDF Max/Min Ratio being too high. If this is the case, the pitch period will not be computed and thus can be overcome by using a lower ratio. This is apparent in Table 2 where a small change in global minimum can cause the ratio to change by a significant margin which can cause the frame to be labelled as unvoiced when a lot of noise is present. However, the threshold values are easy to change and the structure can be adapted for particular usages and thus operation under worse conditions than 10 dB SNR is possible.

Figure 5 (a) and (b) shows the performance of the detector with a SNR of 25 dB and an SNR of 10 dB. As can be seen, pitch period doubling occurs more often with a poorer SNR. However, for use in an adaptive noise canceller, pitch period doubling is not a problem since the delayed speech is still highly correlated with the original speech [1]. For use in a voice conversion system, this must be eliminated. Figure 5 (b) shows the number of zero crossings but does not show the energy computation since the energy threshold is very high.

Time/ms	Global MIN	Global MAX	Ratio	Label
0	0	0	0	-
20	7	70	10	Voiced
40	4	68	17	Voiced
60	4	68	17	Voiced
80	3	67	22	Voiced
100	4	68	17	Voiced
120	3	67	22	Voiced
140	3	69	23	Voiced
160	4	68	17	Voiced
180	4	69	17	Voiced
200	4	68	17	Voiced

Table 2: Voiced/Unvoiced Decision by the Ratio Calculator

V CONCLUSIONS

A pitch detector was successfully implemented in hardware using high-level VLSI design techniques and VHDL. Two different versions of the detector were realised and their relative merits were assessed. Depending on the chosen application, either version could be selected. The computational efficiency of the parallel version was better but had larger area. The pitch detector worked well with high signal-to-noise ratios. For poorer signal-to-noise ratios, the pitch was not always computed due to incorrect voicing decision of the Min/Max Ratio Calculator. However, adjusting the threshold of the Ratio Calculator can solve this problem. Further improvements may be adapted into the design by using the probabilistic approach to AMDF pitch detection [7] which improves gross error probability from 6% to 3%. In addition, power consumption tests on both implementations will be run to assess which version is more power efficient.

In summary, the detector was robust and worked very effectively with signal-to-noise ratios higher than 10 dB. Due to its efficiency, the detector can be incorporated into many small communication devices such as mobile phones, hearing aids and dictaphones.

VI REFERENCES

[1] M. R. Sambur. "Adaptive Noise Cancellation for Speech Signals". IEEE Trans. on Acoustics, Speech and Signal Processing, Vol. ASSP-26, Oct. 1978.

[2] D.G Childers, K. Wu, D. M Hicks and B. Yegnanarayana. "Voice Conversion", Speech Communication, Vol. 8, No. 2, June 1989.

[3] S. Saito and K. Nakata, *Fundamentals of Speech Signal Processing*. Academic Press, 1985.

[4] M. Greenwood and A. Kinghorn. "SUVing: Automatic Silence/Unvoiced/Voiced Classification of Speech". Department of Computer Science, University of Sheffield, 2001.

[5] L. R. Rabiner, M. J. Cheng, A. E. Rosenberg and C. A. McGonegal. "A Comparative Performance Study of Several Pitch Detection Algorithms". IEEE Transactions on Acoustics, Speech and Signal Processing, Vol. ASSP-24, No. 5, October 1976.

[6] M. J. Ross, H. L. Shaffer, A. Cohen, R. Freudberg and H. J. Manley. "Average Magnitude Difference Function Pitch Extractor". IEEE Transactions on Acoustics, Speech and Signal Processing, Vol. ASSP-22, No. 5, October 1974.

[7] Goangshiuan S. Ying, L. H. Jamieson and C. D. Michell. "A Probabilistic Approach to AMDF Pitch Detection". Proceedings of the ICSLP Fourth International Conference on Spoken Language, Vol. 2, October 1996.

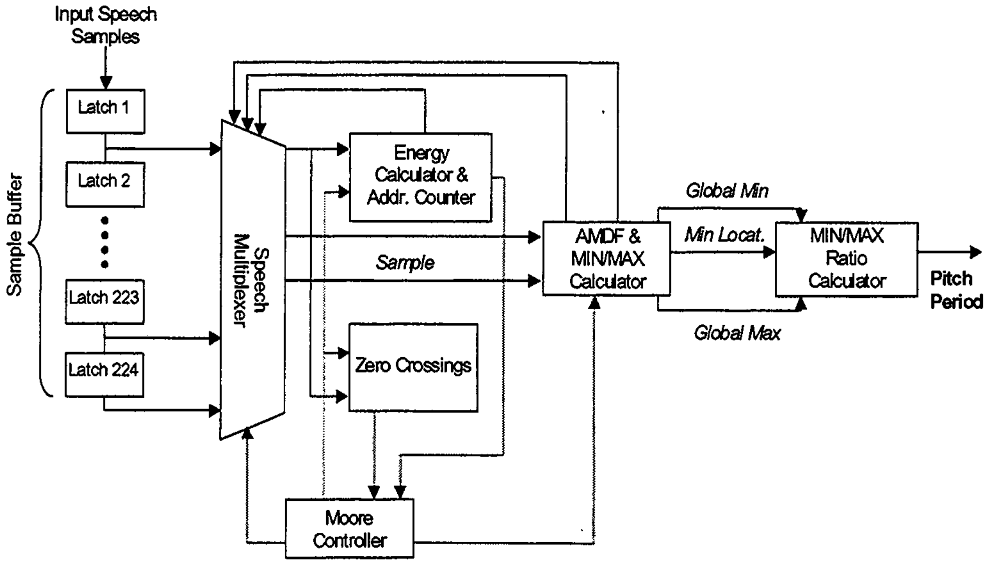

Figure 3: Block diagram of pitch detector

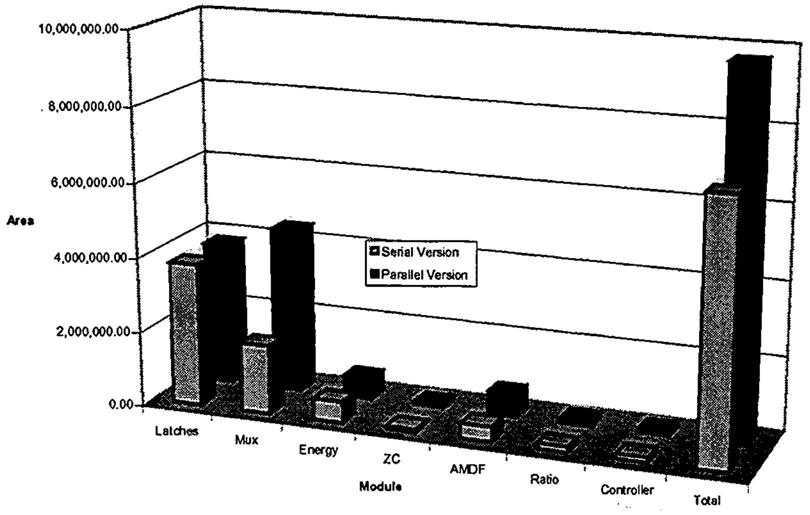

Figure 4: Silicon Area of Modules and Total Silicon Area of Both Versions

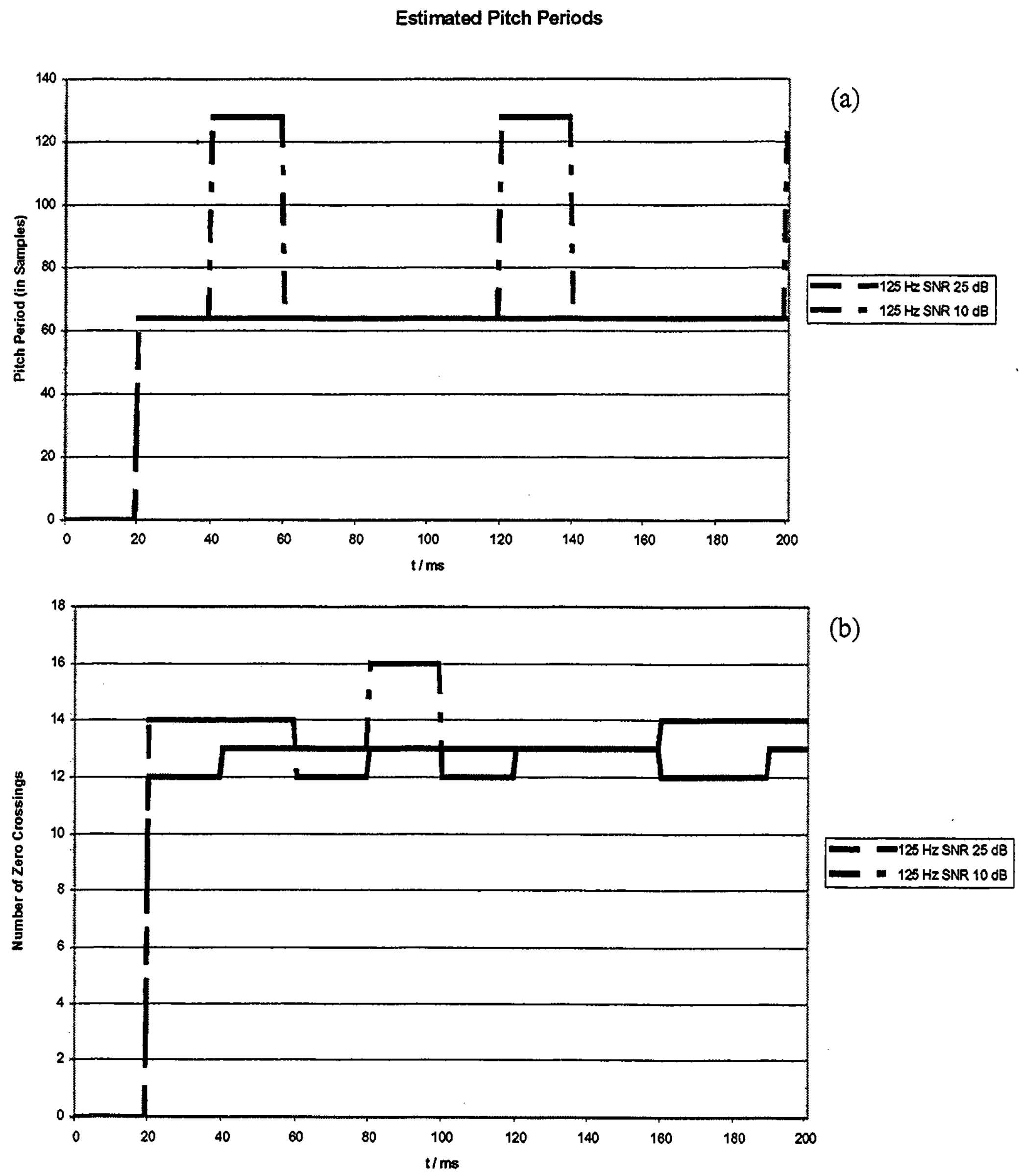

Figure 5: Output Result for two different Input Signals: a) Pitch = 125 Hz, SNR = 25 dB, b) Pitch = 125 Hz, SNR = 10 dB

Design Trade-offs for Implementation of LDPC Encoders

Gary Murphy[†], Emanuel Popovici[†] and William Marnane[§]

[†]*Department of Microelectronic Engineering*
National University of Ireland, Cork
Ireland

[§]*Department of Electrical and Electronic Engineering*
National University of Ireland, Cork
Ireland

E-mail: [†]g.c.murphy@student.ucc.ie [†]e.popovici@ucc.ie [§]l.marnane@ucc.ie

Abstract — **This paper examines a number of architectures for the practical implementation of a Low-Density Parity Check (LDPC) encoder. The encoding problem for LDPC codes can be reduced essentially to vector-matrix multiplication. In order to ensure that a LDPC code has good performance (relative to the Shannon limit) a large codeword and generator matrix are required. This however results in a number of issues with regard to a practical implementation. These issues include interconnect routing, memory size, pipelining and parallelism, all of which will be examined in this paper. A versatile LDPC encoding architecture is mapped on a FPGA (Field Programmable Gate Array) and the cost of the implementation is evaluated as function of area, speed and routability.**

Keywords — **LDPC, encoding, FPGA architectures**

I Introduction

Low-Density Parity Check (LDPC) codes were developed by Gallager [1] in the 1960's and though the code's performance was remarkable they remained largely unnoticed for the next 30 years. The main reason for this was probably the fact, that in the 1960's a practical implementation would have been unrealistically complex [2].

LDPC codes are set to challenge Turbo codes as the coding scheme of choice for the future. Two reasons in particular make them superior to Turbo codes: an LDPC decoder is of an order of magnitude less complex than that of a Turbo code with similar Bit-Error Rate (BER) performance, and an LDPC decoder is inherently parallel [2]. Also, there is no need for interleaving as the interleaver can be distributed in the code. Translated into hardware implementation, these properties make LDPC codes ideally suited to communication applications that require a fast, low-power encoder/decoder solution.

Recent advances, especially in the development of practical LDPC decoding algorithms, have resulted in a resurgence in interest in these codes. LDPC codes are being considered for use in a wide variety of applications ranging from wireless LAN, and ADSL to hard disk drives.

Before being sent over a noisy channel, a message is first encoded and then a modulation scheme is applied. At the other end, the received word, which may contain some errors due to channel noise, has to be decoded in order to remove the errors. In particular the challenges involved in the practical implementation of the decoder, have been the focus of a quite a number of papers [2][3][4]. Although the decoding is similar in principle to the decoding of Turbo codes, the encoding is more complicated. Encoding of LDPC codes provide a number of unique challenges most of which arise from the relatively large parity check matrices involved. The implementation of an LDPC encoder is of complexity $O(n^2)$ for a block of data of length n.

This paper is organised as follows: an overview of the LDPC codes including encoding and decoding is given in Section II. Various architectures for encoding LDPC codes are introduced in Section III. The main focus of the paper is prototyping and design for re-use, which can be used as template in an ASIC development flow. A versatile architecture that will allow the encoding of codes