ISSN 2229-5518

DESIGN OF TURBO DECODER FOR LOGMAP AND SOVA ALGORITHMS USING ITERATIVE DECODING

Kulbir Kaur, Ruchi Singla

Abstract- A Design of Turbo encoder and decoder have been presented using iterative decoding algorithms using MAP and SOVA. The effect of puncturing in turbo codes have been shown using variable code rates. The role of the turbo code puncturer is identical to that of its convolutional code counterpart to periodically delete selected bits to reduce coding overhead. Parallel concatenated convolutional codes are generated at the transmitter end and decoded iteratively at the receiver end considering various factors such as reliability value of AWGN Channel and standard deviation of channel due to various fading effects. The variation in probability of bit error has been shown as graph with respect to E_b/N_0 ratio. Bit error rate have been computed at every iteration loop. The turbo encoder is employed to increase the free distance of the turbo code, hence improving its error-correction performance.

Index Terms- Iterative Decoding, Turbo Codes, Redundancy, Forward Error Correction

1 INTRODUCTION

In information theory, turbo codes are a class of high-performance forward error correction (FEC) codes, which were the first practical codes to closely approach the channel capacity, a theoretical maximum for the code rate at which reliable communication is still possible given a specific noise level. Turbo codes are finding use in (deep space) satellite communications and other applications where designers seek to achieve reliable information transfer over bandwidth- or latency-constrained communication links in the presence of data-corrupting noise. Turbo codes are nowadays competing with LDPC codes, which provide similar performance.

According to Shannon, the ultimate code would be one where a message is sent infinite times, each time shuffled randomly. The receiver has infinite versions of the message albeit corrupted randomly. From these copies, the decoder would be able to decode with near error-free probability the message sent. This is the theory of an ultimate code, the one that can correct all errors for a virtually signal. Turbo code is a step in that direction. But it turns out that for an acceptable performance we do not really need to send the information infinite number of times, just two or three times provides pretty decent results for our earthly channels.

- *Kulbir Kaur is currently pursuing masters degree program in electronics and communication engineering in Punjab Technical University(PTU), India, PH-+91-8699524045. E-mail:kulbir.kaur39@gmail.com*
- *Ruchi Singla is masters degree program in electronics and communication in Punjab Technical University(PTU), India, . E-mail:cecm.ece.rps@gmail.com*

In Turbo codes, particularly the parallel structure, Recursive systematic convolutional (RSC) codes working in parallel are used to create the "random" versions of the message. The parallel structure uses two or more RSC codes, each with a different interleaver. The purpose of the interleaver is to offer each encoder an uncorrelated or a "random" version of the information, resulting in parity bits from each RSC that are independent. How "independent" these parity bits are, is essentially a function of the type and length/depth of the interleaver. The design of interleaver in itself is a science. In a typical Viterbi code, the messages are decoded in blocks of only about 200 bits or so, where as in Turbo coding the blocks are on the order of 16K bits long. The reason for this length is to effectively randomize the sequence going to the second encoder. The longer the block length, the better is its correlation with the message from the first encoder, i.e. the correlation is low. On the receiving side, there are same number of decoders as on the encoder side, each working on the same information and an independent set of parity bit. This type of structure is called Parallel Concatenated Convolutional Code or PCCC. The prevalence of turbo codes in communication systems has also nurtured the usage of decoding techniques that iteratively exchange messages based on the probability of decoded bits, also known as "soft" information. This means that the decision on the outcome of a received bit is predicated on the existence of a spectrum of values indicating the *likelihood* of a "0" or a "1" value. To compare, traditional methods of decoding such as those that employ the Viterbi algorithm make "hard" decisions—it is either a "0" or a "1" and nothing in between. Figure 1 below illustrates the role of message passing in the decoding process.

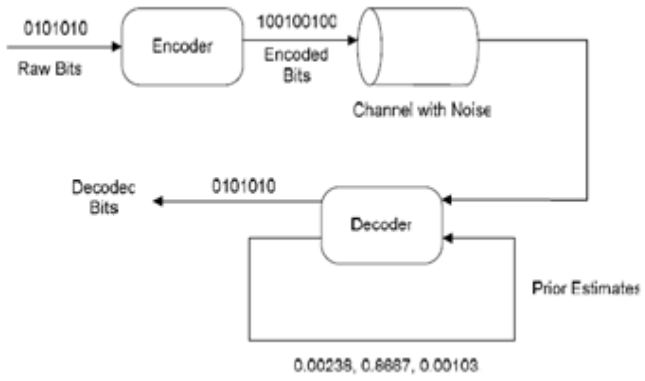


Figure 1: Message Passing from Transmitter to Receiver

The convolutional codes used in turbo codes usually have small constraint length. Where a longer constraint length is an advantage in stand-alone convolutional codes, it does not lead to better performance in TC and increases computation complexity and delay. The codes in PCCC must be RSC. The RSC property allows the use of systematic bit as a standard to which the independent parity bits from the different coders are used to assess its reliability. The decoding most often applied is an iterative form of decoding.

2 LITERATURE REVIEW

[1] shows design and implementation aspects of parallel turbo-decoders that reach the 326.4 Mb/s LTE peak data-rate using multiple soft-input soft-output decoders that operate in parallel. [2] shows the design of new turbo codes that can achieve near-Shannon-limit performance. The design criterion for random interleavers is based on maximizing the e®ective free distance of the turbo code, i.e., the minimum output weight of codewords due to weight-2 input sequences. An upper bound on the e®ective free distance of a turbo code has been derived. A review on multiple turbo codes (parallel concatenation of q convolutional codes), which increase the so-called \interleaving gain" as q and the interleaver size increase, and a suitable decoder structure derived from an approximation to the maximum a posteriori probability decision rule has been shown. A new rate 1/3, 2/3, 3/4, and 4/5 constituent codes have been developed to be used in the turbo encoder structure.[3] shows development of an application specific design methodology for low power solutions. The methodology starts from high level models which can be used for software solution and proceeds towards high performance hardware solutions. The effect on performance due to variation in parameters like frame length, number of iterations, type of encoding scheme and type of the interleaver in the presence of additive white Gaussian noise has been studied with the floating point C model. In order to obtain the effect of quantization and word length variation, a fixed point model of the application has also been developed.

3 PROBLEM FORMULATION

Our problem is to make a design to generate the turbo code and decode the code iteratively using MAP decoders and SOVA decoders.

OBJECTIVE

1. To design a turbo decoder using MAP and SOVA Algorithms
2. Functionally simulate the model for performance evaluation
3. Check the variation in results due to punctured and non-punctured codes for MAP and SOVA Algorithms by varying the code rate.

4 RESEARCH METHODOLOGY

Using MAP Detectors for turbo decoding and compute the performance of the system receiver by computing the Bit Error Rate and Using SOVA decoders for turbo decoding and compute the performance of the system receiver by computing the Bit Error Rate. Iterative Decoding scheme would be implemented for the same.

Following figure shows the block diagram for the design of turbo encoder and decoder.

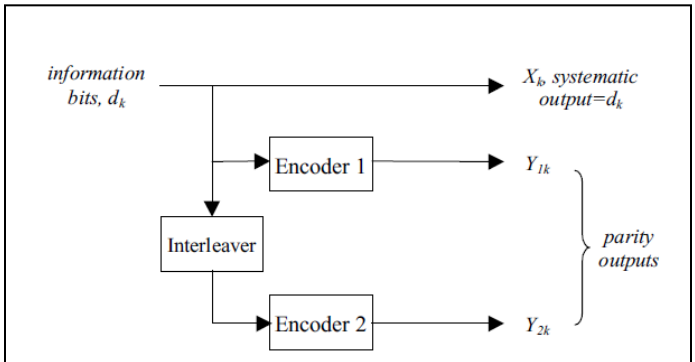


Figure 2: A Turbo Encoder.

A turbo encoder is the parallel concatenation of recursive systematic convolutional (RSC) codes, separated by an interleaver, as shown in Fig. 2. The data flow dk goes into the first elementary RSC encoder, and after interleaving, it feeds a second elementary RSC encoder [3]. The input stream is also systematically transmitted as X_k, and the redundancies produced by encoders 1 and 2 are transmitted as Y_{1k} and Y_{2K}. For turbo codes, the main reason of using RSC encoders as constituent encoders instead of the traditional non-recursive non-

systematic convolutional codes is to use their recursive nature and not the fact that they are systematic [3]. The interleaver is an important design parameter in a turbo code. It takes a particular stream at its input and produces a different sequence as output. Its main purpose at the encoder side is to increase the free distance of the turbo code, hence improving its error-correction performance.

A turbo code is far too complex to decode with a single decoder. Instead, each convolutional code in the turbo code is decoded separately with soft information being passed from one decoder to the next. The decoding scheme of a turbo code is shown in Fig. 4. The above decoder consists of two serially interconnected soft-in soft-out (SISO) decoders, which can be SOVA or MAP decoders. x_k and y_k are the channel outputs with x_k corresponding to the systematic encoder output, and y_k is a multiplexed stream of the two punctured encoder outputs[3]. Hence, a demultiplexer is necessary at the receiver. z_k is called the a priori information and is equal to zero for the first iteration.

The decoder soft output, L (dk), also called log-likelihood ratio (LLR), can be separated into three components:

$L(d_k) = L_{sys} + L_{apr} + L_{ext}$

d_k denotes the actual hard decision of the decoder at step k. L ext is called the extrinsic information. It is a function of the redundant information introduced by the encoder and has the same sign as d_k. (L_{sys} + L_{apr}) constitutes the intrinsic information. L_{sys} is the LLR for the channel output and is the received systematic input to the decoder, scaled by the channel reliability. L_{apr} is equal to the extrinsic information produced by the previous decoder [3].

The intrinsic information is subtracted from the soft decoder output (LLR1, LLR2). The resulting output is the extrinsic information, which is passed on to the next decoder. This process is repeated until a desired performance is attained after a number of iterations. The function of the Interleaver is to take each incoming block of *N* data bits and rearrange them in a pseudorandom fashion prior to encoding by the second encoder [5]. Permuting the input bits effectively randomizes the noise across the channel and minimizes burst errors. In this work, the interleaver follows a very simple permutation pattern and hence may not be the most effective for high-performance systems: inputs are written into a matrix by row, and the outputs are fed column wise. Thus, the de-interleavers used to reconstruct the original order are easily constructed since they are identical structures. The role of the turbo code puncturer is identical to that of its convolutional code terpart to periodically delete selected bits to reduce coding overhead. For the case of iterative decoding, it is preferable to delete only parity bits.

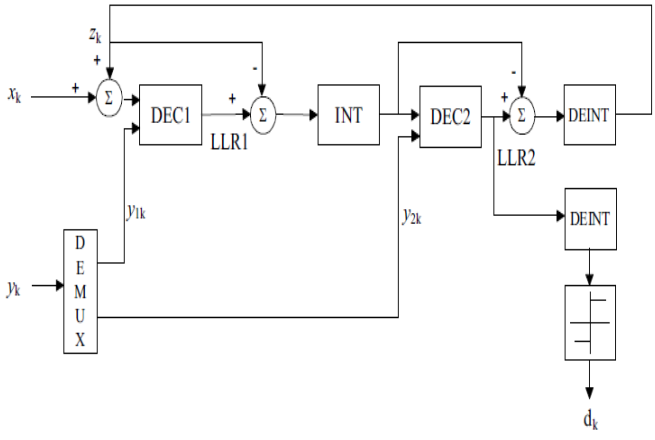


Figure 4: Turbo Decoder Block Diagram

5 MAP Algorithm

Let u = (u_1, u_2 … u_N) be the binary random variables representing information bits. In the systematic encoders, one of the outputs x_s = (x^s_1, x^s_2… x^s_N) is identical to the information sequence u. The other is the parity information sequence output x_p = (x^p_1, x^p_2… x^p_N). We assume BPSK modulation and an AWGN channel with noise spectrum density N_o. The noisy versions of the outputs is y_s = (y^s_1, y^s_2… y^s_N) and y_p = (y^p_1, y^p_2… y^p_N), and y = (y_s, y_p) is used for simplicity. In the MAP decoder, the decoder decides whether u_k = +1 or u_k = -1 depending on the sign of the following log-likelihood ratio (LLR)

$$L_R(u_k) = \log \frac{P(u_k = +1 \mid y)}{P(u_k = -1 \mid y)} .$$

---------- (1)

Let S_k denote the state of the encoder at time k. It can take values from 0 to 2M-1 where M is the number of memory elements in the encoder. LLR can be rewritten as

$$L_R(u_k) = \log \frac{\sum_{s_k} \sum_{s_{k-1}} \gamma_1(y_k, S_{k-1}, S_k).\alpha_{k-1}(S_{k-1}).\beta_k(S_k)}{\sum_{s_k} \sum_{s_{k-1}} \gamma_0(y_k, S_{k-1}, S_k).\alpha_{k-1}(S_{k-1}).\beta_k(S_k)}$$

---------- (2)

Where α is the forward recursion metric, β is the backward recursion metric and γ_i is the branch metric. They are defined as

$$\alpha_k(S_k) = \sum_{S_{k-1}} \sum_{i=0} \gamma_i(y_k, S_{k-1}, S_k).\alpha_{k-1}(S_{k-1})$$

---------- (3)

$$\beta_k(S_k) = \sum_{S_{k+1}} \sum_{i=0} \gamma_i(y_{k+1}, S_k, S_{k+1}).\beta_{k+1}(S_{k+1})$$

---------- (4)

$$((y_k^s, y_k^p), S_{k-1}, S_k) = q(u_k = i \mid S_k, S_{k-1}).p(y_k^s \mid u_k = i).p(y_k^p \mid u_k = i, S_k, S_{k-1}).P_r(S_k \mid S_{k-1})$$

---------- (5)

The parameter q (u_k = i/S_k, S_{k-1}) is either one or zero depending on whether u_k = i is possible for the transition from state S_{k-1} to S_k or not. Calculating p (y^s_k | u_k = i) and p (y^P_k | u_k = i, S_k, S_{k-1}) is trivial if the channel is AWGN. The last component P_r (S_k | S_{k-1}) usually has a fixed value for all k. However, this is not the case in the iterative decoding. The 'a priori' probability of information bits generated by the other MAP decoder must be considered in iterative decoders.

6 SOVA ALGORITHM

It is known that the performance of a SOVA (soft output Viterbi algorithm) turbo decoder can be improved, as the extrinsic information that is produced at its output is over optimistic. A new parameter associated with the branch metrics calculation in the standard Viterbi algorithm is introduced that affects the turbo code performance. Different parameter values show a simulation improvement in the AWGN channel as well as in an uncorrelated Rayleigh fading channel [10]. There are different efficient approaches proposed to improve the performance of soft-output Viterbi algorithm (SOVA)-based turbo decoders. In the first approach, an easily obtainable variable and a simple mapping function are used to compute a target scaling factor to normalize the extrinsic information output from turbo decoders. The scaling factor can be a variable scaling factor or a fixed scaling factor. In Variable scaling factor method, a scaling factor „c" of should be employed to normalize the soft output of SOVA decoders. In practice, to compute the mean and variance of the soft output from SOVA decoders, multiplication and addition operations must be performed at each symbol-processing cycle within each iterative decoding. Also, to compute the final scaling factor, a division operation must be performed before the next iteration begins. All of these imply that a practical SOVA-based turbo decoder with the normalization process embedded may work either with a larger clock cycle period or with a considerable extra latency when pipeline techniques are employed [10].

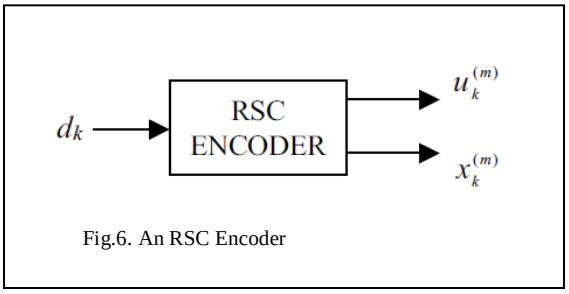


Fig.6. An RSC Encoder

Also, let $u_k^{(m)}$ be the systematic encoder output (in bipolar voltage form) for path m and $x_k^{(m)}$ be the corresponding parity output (Fig. 6). If $y_{k,1}$ and $y_{k,2}$ are the channel outputs corresponding to the systematic and parity outputs of the RSC encoder, then the metric used for the SOVA algorithm becomes

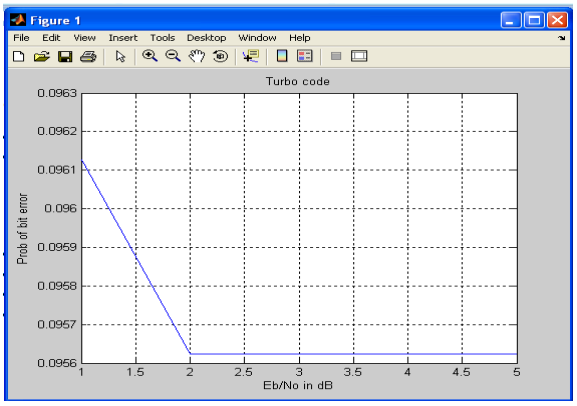
$$M_k^{(m)} = M_{k-1}^{(m)} + u_k^{(m)} L_c y_{k,1} + x_k^{(m)} L_c y_{k,2}$$

7 SIMULATION AND RESULTS

For LOG - MAP Decoder (probability of BER vs SNR) Curve [PUNCTURED CODES]

INPUT PARAMETERS:

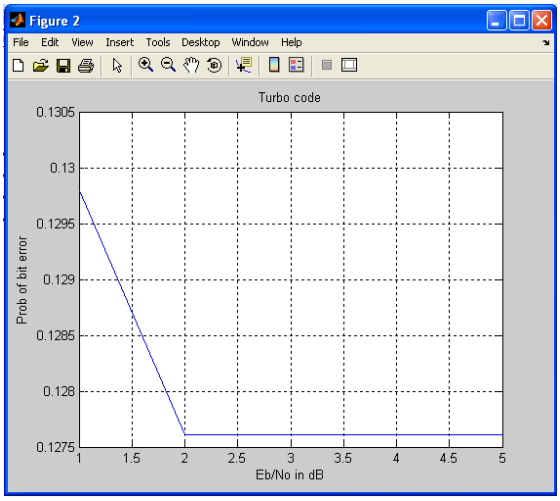
Frame size = 400
Punctured, code rate = 1/2
iteration number = 5
terminate frame errors = 15
Eb / N0 (dB) = 2.00



Probability of Bit error rate vs SNR for punctured codes curve

For LOG - MAP Decoder (probability of BER vs SNR) Curve [UNPUNCTURED CODES]

Unpunctured, code rate = 1/3



Probability of Bit error rate vs SNR for unpunctured codes using LOG-MAP

As, it is clear, that probability of error using LOG-MAP algorithm has increased with unpunctured codes and reduces to a significant value by using punctured codes due to increased code-rate.

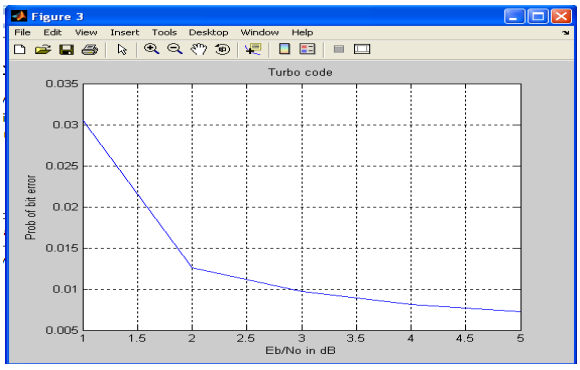
Maximum probability of bit error attained with punctured codes : 0.0961

Maximum probability of bit error attained with unpunctured codes : approx. 0.13

It is clear, that bit error rate using LOG-MAP algorithm has also increased for each iteration with unpunctured codes and reduces to a significant value by using punctured codes due to increased code-rate.

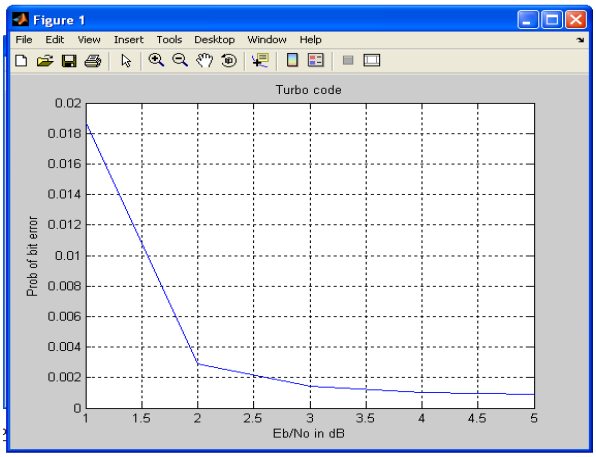
For SOVA Decoder (probability of BER vs SNR) Curve [PUNCTURED CODES]

Unpunctured, code rate = 1/3



Probability of Bit error rate vs SNR for punctured codes using SOVA DECODER

.



Probability of Bit error rate vs SNR for unpunctured codes using SOVA DECODER

It has been seen that probability of bit error has reduced significantly upto three times less in SOVA Decoder as compared to LOG-MAP Decoder. However, time for transmission and BER calculation in SOVA has increased significantly as compared to LOGMAP Decoder.

- Maximum probability of bit error in SOVA decoder (punctured codes): 0.03,
- whereas for LOG-MAP decoder (punctured codes), it has come out to be 0.0961
- It has also been seen that bit error rate and frame error rate reduces significantly with respect to SNR of the channel for each iteration at every frame transmission.

Also, the number of frames transmitted without error has been significantly increased in SOVA decoder as compared to LOG-MAP decoder for particular frame error termination. As studied, for 15 termination frames, number of frames decoded is 15 in MAP decoder, whereas it has increased to 129 frames in SOVA decoder.

8 CONCLUSION AND FUTURE WORK

The turbo code system which is going to be the defacto standard in 3G communication systems has been selected as the specific application in this thesis. The following are the key results of our investigations. From the application software model the application is characterized. The choice of the type of encoders has been reviewed. Since decoding unit has a number of interleaving and deinterleaving tasks, properly designed, less-complex interleaver is an attractive option. This follows because the choice of interleavers is associated with the performance of the turbo codes. As an experimental result, it is demonstrated that symmetric interleavers are less complex with no performance degradation and the same can be

used for deinterleaving as well. The performance improves in the first five to six iterations, subsequent iterations give diminishing returns. The number of iterations is fixed to five. A number of standards are being used worldwide for the implementation of turbo code structures. It would be advantageous to collect the information of the different functional and architectural parameter variations in these standards and come out with relevant comparison for the different implementations.

REFERENCES

[1] Design and Implementation of a Parallel Turbo-Decoder ASIC for 3GPP-LTE Christoph Studer, Student Member, IEEE, Christian Benkeser, Member, IEEE, Sandro Belfanti, and Quiting Huang, Fellow, IEEE, January 2011

[2] On the Design of Turbo Codes, D. Divsalar and F. Pollara, Communications Systems and Research Section, November 1995

[3] A Lowpower Design Methodology For Turbo Encoder And Decoder, Rajeshwari. M. Banakar, Department Of Electrical Engineering, Indian Institute Of Technology, Delhi, July 2004

[4] Sae-Young Chung, G. David Forney, Thomas J. Richardson, and Rüdiger Urbanke, "On the Design of Low-Density Parity-Check Codes within 0.0045 dB of the Shannon Limit" IEEE Communications Letters, Vol. 5, No. 2, page(s):58-60, February 2001.

[5] Sadjadpour, H.R. Sloane, N.J.A. Salehi, M. Nebe, G. "Interleaver design for turbo codes", *IEEE Journal on Selected Areas in Communications*, Volume: 19, Issue: 5, page(s): 831-837, May 2001.

[6] T.P. Fowdur, K.M.S. Soyjaudah, "Joint source channel decoding and iterative symbol combining with turbo trellis-coded modulation**",** *Signal Processing*, Volume 89, Issue 4, page(s):570-582, April 2009.

[7] Erl-Huei Lu, Yi-Nan Lin, Wei-Wen Hung ,"Improvement of turbo decoding using cross-entropy", *Computer Communications*, Volume 32, Issue 6, page(s):1034-1038,27 April 2009.

[8] Fan Zhang and Henry D. Pfister, "On the Iterative Decoding of High-Rate LDPC Codes with Applications in Compressed Sensing", arXiv: 0903.2232v2 [cs.IT], 17 June, 2009.

[9] David Haley, Vincent Gaudet, Chris Winstead, Alex Grant, Christian Schlegel,"A dual-function mixed-signal circuit for LDPC encoding/decoding", *Integration, the VLSI Journal*, Volume 42, Issue 3,page(s): 332-339 ,June 2009.

[10] T. Indermaur and M. Horowitz, "Evaluation of charge recovery circuits and adiabatic switching for low power CMOS design," *Proceedings of the 1994 Symposium on Low-Power Electronics: Digest of Technical Papers*, Oct 1994, pp. 102-103.

[11] H. Mehta, R.M. Owens, M. J. Irvin, R. Chen and D. Ghosh, "Techniques for low energy software," *International Symposium on Low Power Electronics and Design*,Aug. 1997, pp. 72-75.

[12] H. Kojima, D.Gomy, K. Nitta, and K. Sasaki, "Power Analysis of a Programmable DSP for Architecture/Program Optimization," *In IEEE Symposium on Low Power Electronics, Digest of Tech. Papers*, Oct. 1995, pp. 26-27.

[13] M. Kamble and K. Ghose, "Analytical Energy Dissipation Models for Low Power Caches," *Proceedings of the International symposium on Low Power Electronics and Design*, Aug. 1997, pp.143-148.

[14] C. Y. Tsui, J. Monterio, M. Pedram, S. Devadas and A. M. Despai, "Power estimation in Sequential logic circuits," *IEEE Transaction on Low Power Systems*, vol. 3, no. 3, Sep. 1995, pp.404-416.

[15] D. L. Liu and C. Svensson, "Power Consumption Estimation in CMOS VLSI Chips,"*IEEE Journal of Solid State Circuits*, vol. 29, no. 6, June 1994, pp.663-670.

[16] Q. Wu, C. Ding, C. Hsieh and M. Pedram, "Statistical Design of Macro-models For RT-Level Power Evaluation," *Proceedings of the Asia and South Pacific Design Automation Conference*, Jan. 1997, pp. 523-528.