ABSTRACT

A VLSI circuit complexity analysis for low-power decoder designs is presented. Two low-complexity adaptive decoder architectures incorporating the recursive systematic codes (turbo-codes) and the block-codes are considered in this paper. The system performance degradation due to the algorithm approximations for realizing the low complexity decoders is also investigated. The decoders are implemented with 0.6-μm CMOS standard cell technology where their power dissipation and size information are obtained. Furthermore, the issues concerning the throughput and latency of the decoder architectures are discussed. This study provides insights into system design trade-offs involving low-power VLSI decoders for portable wireless mobile communication systems.

I. INTRODUCTION

In wireless mobile communication systems, powerful channel coding is mandatory in order to obtain sufficient reception quality. Two types of codes, binary parallel concatenated recursive systematic convolutional codes termed turbo-codes [1] and iterative product codes [2], have amazing error correcting capabilities. Therefore, they are very attractive for application in digital mobile radio.

However, there are a number of drawbacks associated with such decoders for practical VLSI implementation. The decoding algorithms are very complex because they typically require many complex operations such as exponential and logarithmic calculus as well as an extensive channel statistic estimation process. In addition, the decoders typically require a large memory for data buffering and intermediate storage requiring a large silicon chip area. Furthermore, there is an inherent decoding latency associated with the decoding processes required for achieving the theoretical performance.

As the demands for battery powered wireless communications increases, the minimization of power consumption becomes an increasingly critical concern for the VLSI decoder design. To reduce power consumption and size in a feasible implementation, there is a great need for low-complexity decoding algorithms without sacrificing serious degradation in performance. In addition, the throughput rate and latency are the determining factors in choosing decoder architecture. For a moderate data rate (less than 1Mbps), the throughput rate is not a serious problem for VLSI implementation. However, the latency problem still remains due to the iterative nature of the algorithms.

This paper addresses two issues. First, algorithm approximations for low-complexity decoder designs applicable to low-power mobile communication is investigated. The performance degradation due to approximation including the finite word length effect is presented. Second, circuit complexity is analyzed in terms of power dissipation and size for determining the feasibility of the decoder implementation. This study provides some insights into the system design pertaining to the codes and the decoder architectures.

II. TURBO-CODES

A. RSC Turbo-Codes

Turbo-codes are the parallel concatenation of at least two recursive systematic convolutional (RSC) encoders with interleaving [1] as shown in Figure 1(a). The input to each RSC encoder is the same information bit sequence $d_k$ but in different order due to the presence of an interleaver. For an input bit sequence $\{d_k\}$, the RSC turbo-code encoder output is $\{D_k\} = \{d_k\}, \{C_{1,k}\}$, and $\{C_{2,k}\}$. The encoding is frame oriented using a finite number of $N$ bits per frame.
B. Block-Codes

Two systematic linear block-codes $C_1$ having parameters $(n_1, k_1, \delta_1)$ and $C_2$ having parameters $(n_2, k_2, \delta_2)$ where $n_i$, $k_i$, and $\delta_i$ $(i = 1, 2)$ stand for code length, number of information symbols and minimum Hamming distance respectively. The product code $P = C_1 \otimes C_2$ is obtained by placing $(k_1, k_2)$information bits in an array of $k_1$ rows and $k_2$ columns, coding the $k_1$ rows using code $C_2$ and coding $k_2$ columns using code $C_1$, as illustrated in Figure 1(b). The parameters of the resulting product code $P$ are given by $n = n_1 \cdot n_2$, $k = k_1 \cdot k_2$, and $\delta = \delta_1 \cdot \delta_2$. And the code rate $R$ is given by $R = R_1 \cdot R_2$ where $R_i$ is the code rate of code $C_i$ [2].

For the transmission of block coded binary symbols over a Gaussian channel, the samples $R = (r_1, \ldots, r_n)$ at the output of the demodulator, conditionally to a transmitted code word $E = (e_1, \ldots, e_n)$ can be modeled by $R = E + N$ where $N = (n_1, \ldots, n_n)$ are AWGN samples of variance $\sigma^2$.

The optimum sequence decoding rule for the received vector $R$ is given by

$$D = C^i \ if \ Pr\{E + C^i | R\} > Pr\{E = C^i | R\} \quad (5)$$

And the branch transition probabilities are given by

$$\gamma_k(R, m'm') = \frac{1}{\sqrt{2\pi \sigma^2}} \cdot e^{-\frac{(x_k - 1)^2}{2\sigma^2}} \cdot \frac{1}{\sqrt{2\pi \sigma^2}} \cdot e^{-\frac{(y_1 - 1)^2}{2\sigma^2}} \cdot \frac{1}{\sqrt{2\pi \sigma^2}} \cdot e^{-\frac{(y_2 - 1)^2}{2\sigma^2}} \quad (4)$$

where $\sigma^2$ is the variance of the white Gaussian noise, $\sigma_{L2}^2$ is the variance of $L_2$, and $\gamma_k(R, m'm')$ is defined as [3][4]

$$\gamma_k(R, m'm') = \frac{1}{\sqrt{2\pi \sigma^2}} \cdot e^{-\frac{(x_k - 1)^2}{2\sigma^2}} \cdot \frac{1}{\sqrt{2\pi \sigma^2}} \cdot e^{-\frac{(y_1 - 1)^2}{2\sigma^2}} \cdot \frac{1}{\sqrt{2\pi \sigma^2}} \cdot e^{-\frac{(y_2 - 1)^2}{2\sigma^2}} \quad (4)$$

and the backward recursion is given by

$$\beta_k(m) = \frac{1}{\sqrt{2\pi \sigma^2}} \cdot e^{-\frac{(x_k - 1)^2}{2\sigma^2}} \cdot \frac{1}{\sqrt{2\pi \sigma^2}} \cdot e^{-\frac{(y_1 - 1)^2}{2\sigma^2}} \cdot \frac{1}{\sqrt{2\pi \sigma^2}} \cdot e^{-\frac{(y_2 - 1)^2}{2\sigma^2}} \quad (3)$$

where $S_{k+1}$ is the set containing the index of code words such that $c_{k+1}^j = +1$ and $S_{k}^{-1}$ is the set containing the index of code words such that $c_{k}^j = -1$. By applying Bayes rule and the
assumption that the different code words are uniformly distributed,

\[ (LLR)_j = \ln \left( \frac{\sum_{i \in S_j^+} \Pr\{ E = C^i \mid R \}}{\sum_{i \in S_j^-} \Pr\{ E = C^i \mid R \}} \right) \]

\[ = \ln \left( \frac{\sum_{i \in S_j^+} \exp(-|R - C^i|^2/2\sigma^2)}{\sum_{i \in S_j^-} \exp(-|R - C^i|^2/2\sigma^2)} \right) \]

where the probability distribution function

\[ \Pr\{ R \mid E = C^i \} = \frac{1}{\sqrt{2\pi\sigma}} \exp(-|R - C^i|^2/2\sigma^2). \]  

III. LOW-COMPLEXITY DECODERS

A. RSC Turbo-Code Decoder Integration

For a practical low complexity VLSI implementation, it is highly desirable that the decoder should avoid complex arithmetic operations and variance estimation procedure. To compute the extrinsic information, the original formulas must be simplified considerably. The \( LLR \) can be approximated as follows [5]

\[ \tilde{\lambda}_k \approx \max\{\tilde{\alpha}_k^0(m) + \tilde{\beta}_k(m)\} - \max\{\tilde{\alpha}_k^1(m) + \tilde{\beta}_k(m)\} \]  

using the inequalities

\[ \max\{\zeta_1, \zeta_2, \ldots, \zeta_M\} \leq \ln \sum_{m=1}^{M} \exp\{\zeta_m\} \leq \ln M + \max\{\zeta_1, \zeta_2, \ldots, \zeta_M\}. \]

The forward recursion can be expressed as

\[ \tilde{\alpha}_k^0(m) = \tilde{\gamma}_k^0(R_k, m', m) + \max\{\tilde{\alpha}_{k-1}^0(m') + \tilde{\alpha}_{k-1}^1(m')\} \]

and the backward recursion as

\[ \tilde{\beta}_k(m) = \max\{\tilde{\gamma}_{k+1}^1(R_{k+1}, m', m') + \tilde{\beta}_{k+1}(m')\} \]

where \( m' \) is a possible source state to \( m \) in the trellis. The branch metrics are defined as

\[ \tilde{\gamma}_k^0(R_k, m', m) = x_k \cdot (1 - 2u_k) + y_{1,k} \cdot (1 - 2c_m) + L_{2,k} \cdot (1 - 2u_k)/Q \]

assuming that the channel condition remains constant during the duration of the data block (so that \( \sigma^2 \) is fixed) where \( Q = \sigma^2/Q \). Initially, the value of \( Q \) is set to a large value in order to reduce the effect of \( L_{2,k} \) and becomes smaller as the number of iterations increases. For simple implementation, the value of \( Q \) can be made to be powers of 2.

B. Iterative Block-Code Decoder Integration

The computation of \( (LLR)_j \) is too complex for block-codes with a high code rate. Instead of searching for \( m \) least reliable bit positions as in [2][3][4], two binary test vectors \( T_j^+1 \) and \( T_j^-1 \) for \( y_j = +1 \) and \( y_j = -1 \) respectively for each bit \( r_{j} \) are chosen. From these test vectors, two binary code words \( C_{j}^{\min(+)\text{ }} \) and \( C_{j}^{\min(-)}\text{ } \) are obtained by a simple syndrome decoding utilizing only a feedback shift register. \( C_{j}^{\min(+)\text{ }} \) is the code word at the minimum Hamming distance from \( Y \) with \( y_j = +1 \) and \( C_{j}^{\min(-)} \) is the code word at minimum Hamming distance from \( Y \) with \( y_j = -1 \). In this case \( LLR \) can be approximated by [2][4]

\[ (LLR)_j \approx \frac{1}{2\sigma^2}(|R - C_{j}^{\min(-)}|^2 - |R - C_{j}^{\min(+)}|^2) \approx \frac{1}{2\sigma^2}(r_j + \sum_{l=1,l\neq j}^{n} r_i c_{l}^{\min(+)\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }\text{ }�0-7803-4902-4/98/$10.00 (c) 1998 IEEE
function of the code words at minimum Euclidean distance from \( R \) and \( \{ r_l \} \) with \( l \neq j \). \( LLR_j \) has the same sign as \( d_j \) and its absolute value indicates the reliability of the decision. The idea is to generate soft decision outputs from hard decision decoding.

Figure 3: Block Diagram of the Component Decoder

Figure 3 shows the functional block diagram of the component decoder. On receiving the matrix \([R]\) corresponding to a transmitted code word \([E]\) of the product code, the first component decoder performs the soft decoding of the rows (or columns) of the matrix, estimates the normalized \( LLR [R'] \) and gives as output \([W(1)]\). Then the next component decoder performs the same operations on the columns (or rows) using

\[
[R'(1)] = [R] + \alpha(1)[W(1)]
\]

as input where the coefficients \( \alpha(k) \) are used to reduce the influence of \([W(k)]\) in the first iterations when BER is relatively high and thus when \([W(k)]\) is not absolutely reliable. The syndrome of the test vector is decoded with two parallel shift registers of length \( N \). The coefficients \( \alpha(k) \) can be set to a powers of 2 for simpler circuit implementation where the binary shift operation replaces the power consuming multiplication. The throughput is determined by the code word size \( N \) where \( N \) determines the speed of the shift registers for code word decoding. The speed of the shift register for code word decoding is \( N \) times of the symbol rate. The size of the syndrome table depends on \( n - k \).

C. Power-Down Capability

In the decoding algorithm discussed in the previous sections, the decoding procedure is repeated several times before the final decision is taken. The number of iterations required for a given performance level depends on the channel condition and is usually not known beforehand. Less iteration is highly desirable for a low power operations. A practical way to conserve the power dissipation of the decoder is to make the number of iteration be adaptive such that fewer iterations are processed in good channel condition. However, the termination condition checking processes proposed in the original algorithms is extremely complex and thus simple methods need to be devised for the low-power applications. One way is to compare the Hamming distance between the decoded bits from the two consecutive iteration stages. When decoding is reliable and performance becomes adequate, the Hamming distance between the data from two successive iterations will be very small. The value of the Hamming distance can be used as a threshold to control the iteration. Another approach is to incorporate a CRC. These approaches are much simpler than the estimating the channel statistics.

IV. EVALUATIONS

A. Decoding Performance

One RSC code with generator \((35,23)\) for \( N = 256 \) and one block-code \( \text{BCH}(31,26) \) are evaluated. Figure 4 illustrates the simulated decoding performance of the rate \( 1/3 \) RSC turbo-code and the block-code as a function of the word length (2 to 8) of the input and the metric. As shown in the figure, there is no significant improvement beyond 3-4 bit word lengths.

B. Circuit Complexity

For the low-power VLSI decoder design, the architecture needs to be selected so that voltage scaling and power-down design techniques are feasible. In order to incorporate voltage scaling technique, the architectures should be pipelined so that the operating speed requirement for a iteration is greatly reduced. Each decoding iteration corresponds to a single pipelined stage. The pipelined architecture also helps make the power down capability implementable because inactive iteration stages can be identified. The depth of pipeline and the number of iterations are limited by the available silicon real estate. In order to support the worst case channel condition, the depth of pipeline needs to be large. The pipeline architecture is well suited for implementing the discussed decoding algorithm due to the sequential nature of the algorithm.

Even though simple arithmetic operations for reducing the power dissipation replace many complex decoding operations, the size of memory required by the decoders is a bottleneck in the VLSI design. Both decoders
in general require a large amount of the memory for storing data which represents a major fraction of the area occupied by the circuit. The size of memory depends on the quantization, the number of trellis (code words), and the data block size. Even though the power dissipated by the memory is not a serious concern because only a very small portion of the memory is active, the reduction is nevertheless highly desirable.

Due to backward recursive nature of the RSC turbo-code decoding algorithm, it is required to keep quantized metric information until final extrinsic information is generated. The entire memory required per each stage of the decoder is expressed in bits as follow

\[
MEM = 3N + 4NW_L + 2NW_{in} + 4NMW_{\mu} + 4NW_{\delta} + 2NMW_c. \tag{17}
\]

where \(N\) represents the size of data block, \(M\) represents the number of states in the trellis, \(W_{in}\) represents word length of the inputs, \(W_{\mu}\) represents word length of the branch metric, \(W_{\delta}\) represents word length of the branch metric, \(W_c\) represents word length of the branch metric, and \(W_L\) is the word length of the extrinsic information.

Similarly, due to the alternating row and column decoding of the iterative block-code decoding algorithm, it is necessary to keep received data and reliability information between the stages. Thus, the entire memory required per each stage of the decoder is expresses as follow in terms of bits

\[
MEM = (2W_R + 2W_W + 1)N^2 + 2^{n-k}N. \tag{18}
\]

where \(N\) represents the code length, \(W_R\) represents the word length of the quantized inputs, and \(W_W\) represents the word length of the reliability information.

The above formulas clearly indicate that the smaller word lengths are favorable for low complexity designs as well as for small data block size. For a large data block, the requirement of memory is so excessive that external memories may be required.

The size of the decoder chip is dominated by the storage area required whereas the power dissipation of the decoder chip is dominated by the processing units. For the turbo-code decoder, (35,23) codes and \(N = 256\) are synthesized. On the other hand, (31,26) BCH codes are used for the block-code decoder. The decoder chip size and power dissipation (for 1 pipeline stage) at the supply voltage of 3.3V are evaluated with the EPOCH CAD tools using 0.6-\(\mu\)m CMOS standard cell technology where their power dissipation and size information are obtained. Furthermore, the issues concerning the throughput and latency of decoder architectures are discussed. This study provides insights into system design trade-offs involving low-power VLSI decoders for portable wireless mobile communication systems.

V. CONCLUSION

A VLSI circuit complexity analysis for low-power decoder designs is presented. Two low-complexity adaptive decoder architectures incorporating the recursive systematic codes and the block-codes are considered in this paper. The system performance degradation due to algorithm approximations for realizing the low complexity decoders is also investigated. The decoders are implemented with 0.6-\(\mu\)m CMOS standard cell technology where their power dissipation and size information are obtained. Furthermore, the issues concerning the throughput and latency of decoder architectures are discussed. This study provides insights into system design trade-offs involving low-power VLSI decoders for portable wireless mobile communication systems.

References


