Selection of Channel Coding for Low-Power Wireless Systems

Claude Desset, Andrew Fort
IMEC vzw — DESICS/WISE
Kapeldreef, 75 — B-3001 Leuven — Belgium
Phone: +32 16 28 11 63
e-mail: desset@imec.be

Abstract— **The wireless communications world is moving towards the so-called 4G picture, by integrating many different subsystems. Some of them have to provide short-range connectivity at very low power, in order to enable battery-operated sensors or devices. This low-power requirement can be achieved by selecting channel codes of high coding gain. However, the power consumption of encoding and decoding operations also has to be taken into account.**

This paper analyzes this trade-off between coding gain and digital power consumption, considering different wireless scenarios. It shows that turbo codes can be used even for relatively low-power applications. For very low-power systems, simple Hamming codes provide a good trade-off, as well as convolutional codes. The Golay code is another strong candidate, thanks to a very efficient implementation. Considering current CMOS technology, the different codes are in competition for systems sending bits with an energy between 0.1 and 10 nJ. Systems working at lower values do not gain anything in coding, while systems working above will always advantageously use turbo-codes.

I. Introduction

The field of wireless communications is continuously evolving, with more and more new devices and applications. The combination of these devices targets seamless connectivity, leading to the so-called 4G picture. In order to make this widespread apparition of wireless devices possible, most of them have to be low-cost and low-power. This will enable their integration in laptops, mobile phones, PDAs, or as remote sensors.

Bluetooth is a technology going into that direction, as an example of wireless personal area network. It serves as a cable-replacement tool between consumer electronics products, but it is not extremely power efficient, working at less than 1 Mbps between 0 and 20 dBm (for transmit power only) [1]. Future applications will include health or home automation, putting even more constraints on the design, e.g. to build long-life battery-operated sensors.

In a search for minimal power, the consideration of coding gain is of prior importance. However, the limited energy budget also constrains error-correction implementation, and only codes of limited complexity can be considered. In order to find the lowest-power solution, both transmitted analog power and digital computation power have to be taken into account. Indeed, the power consumed by the encoder and decoder can be of huge importance and ruin the coding gain.

As different wireless devices have different levels of power constraints, we can expect different channel codes to be selected. Systems which have enough power to transmit data at long distance over difficult channels can easily accommodate complex error-correcting codes in order to reduce the transmitted power. On the other hand, if the system conditions are better, the required power can also be lower, meaning that in order to reduce it thanks to coding, very low complexity codes have to be used.

Section II derives the energy that has to be transmitted per bit in various wireless scenarios. It also provides information about the energy required for basic digital operations used for channel coding and decoding, considering state-of-the-art technology.

Based on these references, we can tell how much analog power is spared by a given coding gain, and how much digital power is consumed for channel coding and decoding, meaning that we can select the best code for a given system. This is done in two steps. Section III considers different families of codes, looking at their performance and complexity. Section IV combines these results with the digital and analog power references, in order to discover which code gives the minimal power for a given wireless systems.

II. Requirements in analog and digital power

In order to look at different error-correcting codes as tools used to reduce the power consumption of wireless systems, some power references are needed. A link budget can provide us the required transmit energy per bit, while technology data sheets give us the consumption of basic digital blocks.

Both encoding and decoding power consumption are considered as relevant in order to minimize the global system power, assuming a bidirectional communication between equivalent nodes having to run both operations. In case of one-way transmission from a power-limited device to a central point with more resources, decoding power should not be considered in the budget.

A. Estimation of analog power

The link budget analysis enables the computation of the transmit energy of wireless systems. A very low-power system can use the following values, assuming a 5-meter transmission at 2.4 GHz and uncoded QPSK with 0.33 roll-off:

- path loss: 54 dB
- noise floor: -114 dBm/MHz
- bandwidth efficiency: 1.5 bit/s/Hz
- required E_b/N_0: 8 dB
- noise factor and matching loss: 8 dB
- attenuation margin (incl. fading): 15 dB
- front-end efficiency: 20%

With these parameters, the average energy required per bit can be computed, which is -24 dBm/MHz, or 4 pJ/bit. This is an extremely small value, meaning that wireless transmissions do not have to be power-hungry. The corresponding transmitted power of 40 μW for 10 Mbps is almost certainly negligible with respect to the front-end processing power.

Conditions can also be less ideal: higher frequency and range (10 meters at 5 GHz), and larger attenuation and fading margin (40 dB). As the corresponding transmitted power is higher, the front-end efficiency should increase, e.g. to 70%. This leads to a required power of 9 dBm/MHz, or 8 nJ/bit.

As a third example, the low-power 1-mW mode of Bluetooth leads to some 2 nJ/bit. Considering higher-power operation at 50 mW (up to 100 mW is possible), we get 115 nJ/bit. Compared to the first figure (4 pJ/bit in ideal conditions), this shows how large the wireless power ranges can be, even considering only short-distance communications.

Based on this fact, we decided to consider 4 different situations: a very low-power wireless system working at 5 pJ/bit, a low-power system at 100 pJ/bit, an average-power case at 2 nJ/bit, corresponding to the lower edge of Bluetooth, and a high-power case at 50 nJ/bit.

B. *Digital power consumption in available technology*

In order to study the power consumption related to channel encoding and decoding, some information about the basic components used is needed. Due to the continuous evolution of microelectronics, we have to consider what is achievable in current technologies in order to enable a quantitative analysis. We use figures for 0.18 μm CMOS, which can be considered as state of the art, even if the move towards 0.13 and below has already begun. The supply voltage is assumed to be 1.8 V.

Based on power consumption taken from data sheets for basic elements, and on figures taken from design practice (including some overhead related e.g. to interconnect), the following estimations of energies per operation can be made:

- logical gate (incl. inverter): 0.4 pJ/op
- flip-flop: 2 pJ/write
- n-bit adder: $1.5n$ pJ/op
- n-bit multiplier: $1.5n^2$ pJ/op
- RAM: 10 pJ/read or write (for 1 bit)
- m-input ROM: m pJ/read (for 1 bit)
- $GF(2^m)$ addition: $0.4m$ pJ/op
- $GF(2^m)$ mult.: $2.2m(m+1)$ pJ/op

Galois-field values are computed assuming a polynomial representation, leading to simple XORs for the addition, and a shift-register-based multiplication.

These values enable the computation of the power consumption of channel encoders and decoders. We first consider the complexity of encoding and decoding in the following section, while numerical comparisons appear in Section IV. We assume that all information bits are available either in a buffer memory — with an additional write plus read cost of 20 pJ/bit — of as a set of shift-register (at 2 pJ/bit), according to the needs of encoder and decoder in random addressing.

III. Complexity and performance of error-correcting codes

This section mentions coding gain and derives encoding and decoding complexity as average number of operations per information bit, for various classical error-correcting codes. Prior results have been presented in [2], insisting on the complexity computation for the different decoding algorithms. We consider both block (Hamming, Reed-Muller, Golay, Reed-Solomon) and convolutional (including turbo) codes. These results will be used to get numerical values of global power gain in Section IV, in order to enable code selection for different applications.

A. *Hamming codes*

A first solution is simply using Hamming codes, in their systematic form. The encoding is simply achieved by directly implementing the part of the generator matrix related to parity bits. For a $(2^m-1, 2^m-m-1)$ code, this leads to a bit more than $m/2$ logical operations (XORs) per information bit, which is very low.

Hamming codes have the advantage of a very small bandwidth expansion, and yet they can provide a significant coding gain, around 1.5 dB at a BER of 10^{-5} for codes of length 31 and above with hard decoding. Asymptotically, long Hamming codes reach 3 dB at high E_b/N_0.

Decoding Hamming codes is almost as simple as encoding. The syndromes are computed by simply encoding the received systematic bits, and comparing the obtained parity bits to the ones received from the channel. This is achieved through m additional XOR operations with respect to the encoder, m flip-flops to hold the results, and $m-1$ ORs in order to find whether there was an error. If the syndrome is non-zero, the simplest way to decode is through a m-input look-up-table, providing the position of the bit to toggle; we assume information bits are stored in a memory in order to easily implement this bit change.

This bit-toggling will not occur for each received word. Assuming a target BER of 10^{-5} with a $(63, 57)$ code (which is the best at this BER), it is easily found by using the channel BER that only 1% of the words will need some correction. However, even if correction is rare, the decoder has to implement a RAM in order to access more easily the bit to correct, meaning that more power is consumed than in the encoder for data storage; memory power dominates logic power in this case.

B. *Reed-Muller codes*

Reed-Muller codes are not extremely popular, but they have the reputation of providing good performance (only a little less than long BCH codes) at a very low cost thanks to simple majority decoders. This is especially true for first-order Reed-Muller codes. They have length, dimension, and minimum distance $(8,4,4)$ (extended Hamming), $(16,5,8)$, $(32,6,16)$, $(64,7,32)$, etc. A problem with first-order Reed-Muller codes appears to be their rather large bandwidth expansion.

Encoding such codes can be simply achieved by a m-bit counter, m AND gates, and m XOR gates, everything being clocked 2^m times to encode $(m+1)$ bits [3]. However, for short codes, it is even more efficient to encode the data by directly implementing the generator matrix. Its cost is of the order of $n/2$ XORs per information bit. Above that, flip-flop storage of data is considered. The higher values obtained for Reed-Muller codes, compared to Hamming, mainly come from the fact that the code rate is rather low (so more operations are required per information bit), and because all the bits have to be encoded, as the encoder is not systematic.

Decoding is achieved through the use of a Fast Hadamard Transform. Reed-Muller codes have the big advantage that complete decoding comes at the same price as bounded decoding. However, even with complete decoding, the coding gain remains rather low (1.8 dB for the (64,7) at 10^{-5}). Reed-Muller codes only become useful when using soft decoding. Thanks to the complete decoding algorithm used, soft decoding is a very simple extension. This leads to a significantly higher gain, i.e. 2.6 dB for the (16,5,8) code, 3.3 dB for the (32,6,16), and 3.7 dB for the (64,7,32), considering 3-bit soft information (above the sign bit) at 10^{-5}.

Using the Fast Hadamard Transform, the core of the decoder is made of m stages, with a structure based on shift registers, butterflies achieving basic Hadamard decomposition (mainly adders), and control logic in order to steer the bits [3]. Words propagating between stages start with one bit and end with m in case of hard decoding, while 3 additional bits are added at each stage for soft decoding. Taking all these elements into account, we get a total of $(m+2)2^{2m}$ shift register 1-bit storage operations, $(m^2+3m+8)2^{m-1}$ 1-bit arithmetic operations, and about $(m+3)2^{m+2}$ logical operations [2]. All this is used in order to decode $m+1$ bits. The decoding power is dominated by shift registers; it could probably be reduced by designing a more complex decoder.

C. Golay code

The binary (23,12,7) Golay code has attracted a lot of attention from coding theorists, due to its exceptional structure. It can also be of high practical interest, being as a perfect good able to provide high-performance decoding. While hard-decoded Golay provides 2.1 dB coding gain at 10^{-5}, soft decoding brings 4.0 dB. Encoding is very simple for this short code: implemented in a systematic way, only 5.5 logical XORs are needed per information bit.

Several authors have looked for efficient implementations of soft decoders for this code [4], [5]. They propose a soft ML decoder requiring only 121 real operations, mainly comparisons and additions, although some other algebraic operations are needed above these 121. Assuming 5-bit words to be used for computations, the global cost of this decoding is estimated to 605 1-bit additions, or 50 per information bit. As authors of [5] neglect algebraic operations in order to focus on more costly real operations, we take them into account as a 50% increase in decoding power.

D. Reed-Solomon codes

Reed-Solomon codes are one of the most popular family of error-correcting codes, due to their good performances for a reasonable decoding complexity. Binary BCH codes are also rather popular, but they will not be considered in this paper (except Hamming codes), as Reed-Solomon codes achieve better performances for a given complexity. Various encoder architectures are possible, and we consider an implementation based on polynomial representation and shift registers, classically used in many applications.

Systematic encoding leads to the following gains: 2.0 dB with a (15,9,7) on GF(16), 2.9 dB with a (31,19,13) on GF(32), and 3.4 dB with a (63,41,23) on GF(64). Encoding simply comes at the global cost of $4t^2$ or $(n-k)^2$ additions and the same number of multiplications, in the Galois field.

The decoder classically begins by computing the syndromes. This can be achieved by $2(t-1)(n-1)$ multiplications and $2t(n-1)$ additions. The second stage is to compute the coefficients of the error-locator polynomial. Using the Berlekamp-Massey algorithm, this is done in $2t$ steps. Step i requires shifting k times a recursive register of $\lceil i/2 \rceil$ taps, $(k+2)\lceil i/2 \rceil$ multiplications, and $k(\lceil i/2 \rceil - 1)$ additions, all these operations being run in the Galois field. The last output is compared to the i^{th} syndrome element. If they differ, the temporary error-locator polynomial is updated, using an auxiliary polynomial (for a deeper description of this algorithm, refer to [6], [3]).

The roots of the error-locator polynomial are then computed, in order to find the positions of errors. This is generally achieved by a Chien search, enabling with nt multiplications and additions the evaluation of the polynomial over the 2^m elements of the Field. Finally, error values are computed and added to the received word.

The global cost of decoding one word is $kt(t+1)m$ shift-register operations, $2tm$ logical operations, $((k+2)t(t+1)+(3t-2)n+2(t+1)(t+3))$ Galois multiplications, and $(kt(t+1)+(3t-2)(n-1)+2t(t+1))$ Galois additions.

E. Convolutional codes

Besides block codes, convolutional codes can be good candidates, thanks to the efficient Viterbi soft decoding algorithm. Considering codes of constraint length between 3 and 7, and rate 1/3 or 1/2, selected from tables of "best known codes" [3], gains between 3.3 and 4.8 dB at 10^{-5} are achieved, while short constraint lengths are expected to keep their complexity low enough for our applications.

The convolutional encoder is extremely simple. For a code of rate $1/n$ and constraint length K, it only needs a $(K-1)$-element shift register, and on the average $nK/2$ logical XORs, which leads to a very low power consumption (see Table I).

The Viterbi decoder is more demanding. A significant part is related to the storage of information sequences corresponding to surviving paths. This is achieved through a bank of 2^{K-1} shift registers, of a length generally between 5 and 10 times the constraint length; we consider a length $8K$ here. Using an implementation based on "trace-back" enables to reduce exchanges of information between shift registers, a power-hungry part. In order to reduce the logic related to trace-back, we use a block-based system, only tracing back for every Kth symbol.

Some arithmetic computations also have to be achieved, and we assume 8-bit finite precision for the memory of cumulated path metrics. For each new information bit, the decoder has to compute $n2^K$ additions and 2^{K-1} comparisons (of similar complexity), but despite these computations, the global power consumption remains mainly limited by the shift registers.

F. Turbo codes

We can obviously not end this review of possible coding techniques without considering turbo codes. However, even if their very good performances are well known, we can expect a prohibitive complexity for our strongly power-limited application.

As a matter of fact, 6 iterations lead to results close to ML decoding. Each iteration requires a forward and a backward recursion, and it is generally admitted that both of them are twice as complex as a simple Viterbi. This leads to a rough estimate of 24 times the complexity of a Viterbi decoder for the same constraint length. The turbo encoder energy consumption is computed by adding one memory write and read operation per bit to the corresponding convolutional encoder, to take the interleaver into account, above the two convolutional encoders.

Besides this first estimation, we also consider a low-power and low-latency optimized implementation of a turbo codec [7]. Selecting the best parameters in order to maximize the coding gain, we can reach a BER of 10^{-5} for an E_b/N_0 value of 2.9 dB with rate 1/2 and 2.5 dB with rate 1/3. This is about 0.5 dB above the theoretical values, which can be explained by factors such as the use of finite-precision numbers. This implementations has a very low power consumption of the order of 3 nJ/bit for .18 μm CMOS [8]. This is in fact three times less than the power consumption estimated in the previous paragraph, thanks to the in-depth optimization of the decoder.

IV. Selection of Error-Correcting Codes in Wireless Systems

This section compares the different codes considered for the four wireless scenarios presented in Section II. In all these scenarios, it compares the digital coding and decoding power to the analog power, and derives the resulting global coding gain (or loss). Table I provides the results of these comparisons.

For extremely low-power systems, no coding should be used, as it is never possible to gain enough with respect to the encoding and decoding consumption. With a somewhat higher available power, simple Hamming codes are the only affordable ones. They lead to a global coding gain of 0.5 dB for a system working around 100 pJ/bit, and soon reach 1 to 1.5 dB if a higher bit energy is allowed. They also have the advantage of presenting a very limited bandwidth expansion, which can be an additional requirement for some systems. In that range, a highly optimized Golay decoder can also lead to good performance.

Above 300 pJ/bit, more efficient coding schemes can be selected. Short-constraint-length convolutional codes are the best candidates. With a constraint length of 3, a coding gain of 2 dB can be obtained at 500 pJ/bit, and 4 dB are available at 4 nJ/bit with a longer constraint length.

Starting from 5 nJ/bit, turbo-codes become the best choice. In this study, they even provide the best coding gain at 2 nJ/bit, but under a power-optimized implementation, which is not the case for other codes except the Golay one. Depending on the decoder optimization, 5 to 6.5 dB of coding gain can be achieved at 15 nJ/bit, and above 7 dB near 100 nJ/bit. Figure 1 gives an idea of the evolution of global coding gain with the wireless bit energy.

An interesting point is the way these results are related to the technology. For the moment, Bluetooth-like systems work close to the threshold above which turbo-codes should be used, and can gain 3 or 4 dB thanks to coding. A reduction in CMOS power consumption translates into a left shift of the curves in Figure 1, meaning that higher coding gains will be achieved for lower-power systems. However, even moving three CMOS generations ahead (in some five years), one should multiply the energy for a given operation by 0.35^3, which is around 0.04. This is a factor 25, while Figure 1 spans four orders of magnitude, meaning that even if the different thresholds will move with time, the general conclusions will hardly be different.

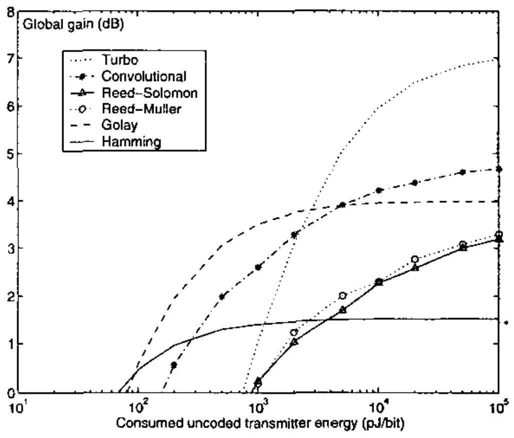

Fig. 1. Global coding gain achieved by using different codes, as a function of the transmit bit energy of the application.

V. Conclusions

This paper has investigated the reduction in power consumption of wireless systems thanks to the use of channel coding. Considering the power levels achieved in wireless personal area networks and the consumption of channel encoders and decoders, it was shown that channel coding can effectively reduce the power consumption in different

Code		Coding	Consumption (pJ/bit)		Global gain (dB) for power being:			
Type	Parameters	gain (dB)	Encoder	Decoder	Very low	Low	Average	High
Hamming	(15, 11, 3)	1.16	3	32	-	-	1.1	1.16
	(63, 57, 3)	1.52	3.5	24	-	0.46	1.46	1.52
Reed-Muller	(16, 5, 8)	2.6	4.5	736	-	-	1.23	2.54
	(32, 6, 16)	3.3	7.5	2650	-	-	-	3.08
	(64, 7, 32)	3.7	13	9920	-	-	-	2.91
Reed-Solomon	(15, 9, 7)	2.0	48	453	-	-	1.03	1.96
	(31, 19, 13)	2.9	105	1490	-	-	0.35	2.76
	(63, 41, 23)	3.4	188	4650	-	-	-	3.00
Golay	(23, 12, 7)	4.0	4	116	-	0.57	3.75	3.99
Convolutional	1/2, K=3	3.3	5	171	-	-	2.93	3.28
	1/2, K=5	4.3	10	522	-	-	3.28	4.25
	1/2, K=7	4.75	15	1790	-	-	1.96	4.60
	1/3, K=5	4.5	11	714	-	-	3.16	4.44
Turbo (rate 1/3, K=4)	classical	7.1	29	9400	-	-	-	6.35
	optimized	7.1	29	3000	-	-	2.31	6.76

TABLE I

Comparison of error-correcting codes in terms of gain and power consumption, for four different wireless scenarios — very low-power (5 pJ/bit), low-power (100 pJ/bit), average-power (2 nJ/bit), and high-power (50 nJ/bit). All comparisons are done for a BER of 10^{-5}, and a global gain is derived by combining the coding gain with the loss related to encoder and decoder consumption. Soft decoding is used for convolutional, turbo, and Reed-Muller codes.

scenarios, except for extremely low power. The opposite conclusion was found in some prior literature [9], but this seems to be related to their hypothesis of low-efficiency front-ends, assuming a fixed overhead of some 100 nJ/bit, which is too high for low-power applications. In this paper, the front-end consumption was taken into account by first computing the required wireless power and then considering an efficiency factor. In a given system for which besides power proportional to the rate of sent bits, the actual power overhead is provided, this overhead gives a threshold below which coding is not really useful anymore, meaning that the front-end should be further optimized to reduce the global power.

Considering applications with an average bit energy going from 5 pJ to 50 nJ/bit, we successively recommend no coding at all, Hamming coding, Golay coding or short constraint length convolutional coding, and in the end turbo coding. These results hold for symmetric systems, for which both encoding and decoding are achieved in the transmitter and receiver. In more specific situations, like "sensor networks" with low-power nodes sending information to a central higher-power receiver, we can use more efficient codes even for lower-power situations, as the power-hungry decoders will not be as critical any more. It was also shown that despite the downscaling of digital CMOS, the general conclusions should not be changed in the coming years, as the orders of magnitude considered are large.

The fading character of wireless channels has not been considered, except through a power margin. However, it is well-known that coding is even more appealing in fading situations, as the gain in BER or the increased order of diversity can save larger amounts of power. This means that for such situations, we can expect to use more coding than for the AWGN channel.

As a last remark, let us mention other low-power coding solutions that could be worth studying, like low-complexity sequential decoding of convolutional codes, or trellis-coded modulations for systems which are also strongly constrained in bandwidth.

References

[1] Bluetooth Special Interest Group. http://www.bluetooth.com.

[2] Claude Desset. Error control coding for wireless personal area networks. In *23rd Symposium on Information Theory in the Benelux*, pages 107–114, Louvain-la-Neuve, Belgium, May 2002.

[3] Stephen B. Wicker. *Error Control Systems for Digital Communication and Storage.* Prentice-Hall, 1995.

[4] Alexander Vardy and Yair Be'ery. More efficient soft decoding of the golay codes. *IEEE Transactions on Information Theory*, 37(3):667–672, May 1991.

[5] Alexander Vardy. Even more efficient bounded-distance decoding of the hexacode, the golay code, and the leech lattice. *IEEE Transactions on Information Theory*, 41(5):1495–1498, September 1995.

[6] E. R. Berlekamp. *Algebraic Coding Theory.* McGraw-Hill, 1968.

[7] A. Giulietti, B. Bougard, V. Derudder, S. Dupont, J.-W. Weijers, and L. Van der Perre. A 80 Mb/s low-power scalable turbo codec core. In *Custom Integrated Circuits Conference (CICC)*, Orlando, FL, USA, May 2002.

[8] Bruno Bougard, Alexandre Giulietti, Liesbet Van der Perre, and F. Catthoor. A class of power efficient VLSI architectures for high speed turbo-decoding. submitted to Globecom'02, 2002.

[9] Eugene Shih, Seong-Hwan Cho, Nathan Ickes, Rex Min, Amit Sinha, Alice Wang, and Anantha Chandrakasan. Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks. In *MOBICOM 2001, 7th Conference on Mobile Computing and Networking*, Rome, Italy, July 2001.