Due to the type 3 fonts used, please increase the magnification to view

Encodings for High-Performance Energy-Efficient Signaling

Alessandro Bogliolo
DIF - University of Ferrara
44100 Ferrara, Italy
abogliolo@ing.unife.it

ABSTRACT

Energy efficiency, performance and signal integrity are conflicting critical requirements for on-chip signaling. We propose a code-based solution that improves bit rate while reducing communication energy and preserving noise margins. Our technique is based on the observation that RC lines can be used at twice their limiting bit rate to transmit bit streams with no isolated bits. We propose new encodings (called minimum run-length guaranteed codes, MRLG) that eliminate isolated bits, thus enabling double-bit-rate signaling. We show that our encodings can be combined with any low-power code to achieve both energy reduction and performance improvement.

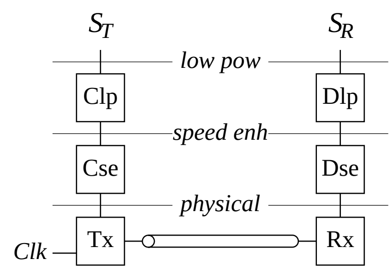

Figure 1: Schematic representation of the proposed signaling system.

1. INTRODUCTION

Energy efficiency and signal integrity are major concerns for the designers of deep sub micron (DSM) integrated circuits. The reduction of the cross section and spacing of wires causes the increase of RLC parasitics responsible of propagation delay, noise and power consumption [1], thus increasing the impact of interconnects on design metrics.

Signal propagation on interconnects is affected by several noise sources that may impair signal integrity: inter symbol interference, supply noise, timing uncertainty, crosstalk, ... Such noise sources have been deeply investigated and specific techniques have been proposed either to reduce them or to mitigate their effects on digital signals [1]. As for power consumption, a large fraction of the power budget of a digital system is spent in switching the capacitance of long internal wires. Formidable efforts have been spent in the last decade to reduce the power consumption on global interconnects [2, 3, 4].

However, signal integrity and low power needs are often conflicting, making it difficult to find unified optimal solutions (for instance, reducing supply voltages leads to a quadratic benefit in terms of power, but impairs noise margins). Moreover, both noise effects and power consumption become more critical at higher frequencies, thus actually imposing to trade-off performance for reliability and power.

In this work we address the problem of on-chip signaling by looking simultaneously at energy efficiency, performance and signal integrity. We start from a baseline communication channel consisting of a transmitter (Tx), a long on-chip interconnect and a receiver (Rx). As illustrated in Figure 1, we treat the channel as the physical layer in the ISO/OSI protocol-stack model, and we work at a higher level (say, the data link layer) by performing preliminary encodings of the bit stream to be sent across the channel. Our target is to realize a *virtual channel* working at higher bit rates and consuming less energy than the original one, while providing the same noise margins.

Encoding strategies have been successfully exploited by the low-power community to save energy by reducing the switching activity on long wires [3, 4]. We will take advantage of such techniques (hereafter referred to as *low power encodings*) to meet our energy-reduction goal.

Binary encodings granting error detection/correction capabilities [5] have been also proposed as a viable solution to reduce the error rate of the received bit stream, thus providing the opportunity of pushing the transmission rate beyond the inherent error-free limit of the channel. In particular, Pamunuwa et al. [6] demonstrated the effectiveness of triple error correcting codes to compensate the effects of crosstalk. Such codes, however, do not meet our requirements, because: i) they significantly increase communication energy and ii) they work under the assumption of random white noise (RWN), that does not model inter symbol interference and correlated noise sources.

To overcome such limitations we propose new encodings that transform arbitrary bit streams into streams having: ii) no isolated bits and ii) no more transitions than the original ones. The first feature is the key for speed enhancement.

ISLPED'01, August 6-7, 2001, Huntington Beach, California, USA.

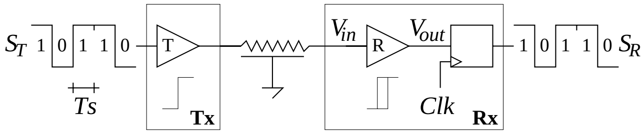

Figure 2: Reference signaling scheme.

We say that the n-th bit of a stream is *isolated* if it takes value b while bits in time slots n-1 and n+1 take value $\overline{b}$. Transmitting an isolated bit represents the worst case situation for the physical channel, since the line has to switch from $\overline{b}$ to b in a single time slot (also called *symbol time*). Satisfying noise margins on isolated bits is a difficult task that imposes the actual limitation to the bit rate of interconnects. Sequences with no isolated bits could be transmitted on the same channel at much higher bit rates without violating noise margins. The second property of our codes allows us to decouple speed enhancement from energy reduction, since the encoding for speed enhancement will not impair the benefits of low-power encodings used to reduce the average activity of the communication channel. Figure 1 shows the schematic view of the proposed signaling system that offers a virtual channel consisting of three layers: *low-power encoding* (LP), *speed-enhancement encoding* (SE) and *physical channel*.

The rest of the paper is organized as follows. In Section 2 we analyze inherent properties of physical channels, showing that interconnects can be used at twice their nominal bit rate to transmit bit streams with no isolated bits. In Section 3 we present speed-enhancement encodings generating bit streams with no isolated bits to be transmitted at double bit rate. In Section 4 we discuss the energy efficiency of the proposed codes and their use in conjunction with low-power encodings. Results are discussed in Section 5.

2. PHYSICAL LAYER

We refer to the baseline signaling scheme of Figure 2. The transmitter is a CMOS buffer (T) fed by a bit stream S_T, while the receiver consists of two components: a Schmitt trigger[1] (R) that regenerates the signal and a flip flop that samples the output of R on the rising edges of Clk to provide the received stream S_R. We denote by Vin and $Vout$ the input and output of R, by T_s the symbol time (corresponding in our scheme to the period of Clk) and by Br the corresponding bit rate ($Br = 1/T_s$).

For our simulations we modeled the wire as a distributed RC line with parameters extracted for a $5mm$ metal-2 wire of a $0.25\mu m$ process. In particular we assumed a line width of $1.5\mu m$, a line thickness of $0.5\mu m$ a lateral space of $1\mu m$ and an inter-layer oxide of $1\mu m$. We used a supply voltage of 2.5V and a driver 30 times larger than a minimum-size inverter. The rising and falling thresholds of the Schmitt trigger were 1.7V and 0.8V, respectively.

[1]We use a CMOS Schmitt trigger instead of a CMOS buffer at the receiver since it provides improved noise margins and faster transitions by means of voltage hysteresis. Hysteresis increases propagation delay, but it doesn't impair bit rate. Hence, Schmitt triggers are not suitable receivers for short local wires, where propagation delay is the main concern, while they can be used to receive digital signals sent across global interconnects, where bit rate and noise margins are the key design metrics.

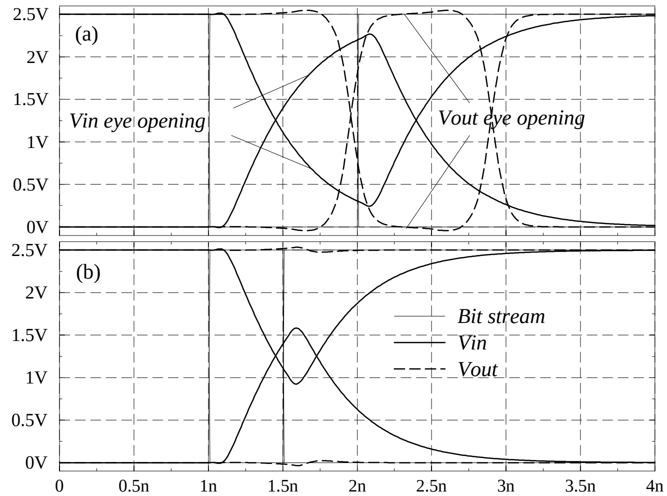

Figure 3: Response of Vin and $Vout$ to an input pulse representing the transmission of a single bit of value 1 (0) in a sequence of 0's (1's). Graph (a) was obtained with a symbol time of 1ns, graph (b) with a symbol time of 0.5ns.

Figure 3 shows the behavior of S_T, Vin and $Vout$ corresponding to the transmission of a single 1 (0) after a long sequence of 0's (1's). Graph (a) refers to a symbol time of 1ns, graph (b) to a symbol time of 0.5ns. In the first case, the eye opening of Vin and $Vout$ allows the receiver to reconstruct the correct bit stream. In the second case, the symbol time is not sufficient to switch the line capacitance, causing the isolated bit to be filtered out.

For our purposes, cross talk, supply noise and timing uncertainty are external conditions on which we have no control. Such external noise sources impose noise margins to Vin and $Vout$ in order to guarantee the correct sampling of the received bits. Without loss of generality, we assume noise margins to be exactly met by the waveforms of Figure 3(a), meaning that our physical channel has a limit bit rate of 1Gb/s. Needless to say, noise margins are violated by the waveforms of Figure 3(b).

2.1 Doubling bit rate

Figure 4 shows the behavior of the signaling system of Figure 2 when used to transmit a bit stream with *no isolated bits* at 2Gb/s. Dots on $Vout$ represent sampling points, while vertical lines in the graph represent symbol-time boundaries at the receiver. Notice that the switching time of the line exceeds the symbol time (see trace Vin in the lower graph), but all bits are sampled with the required noise margins at the receiver: the upper graph in Figure 4 shows that $Vout$ reaches nominal voltage levels corresponding to all sampling points (denoted by dots). In practice, the RC inertia of the line causes only a right-shift of S_R with respect to S_T, that does not impair the correctness of the received stream. The logical values of S_T and S_R are reported in Figure 4 above each graph.

The above discussion leads to the following general observation: *a line with limiting bit rate Br (imposed by external noise margins and ISI) can be used at $2Br$ with the same noise margins to transmit bit streams with no isolated bits.*

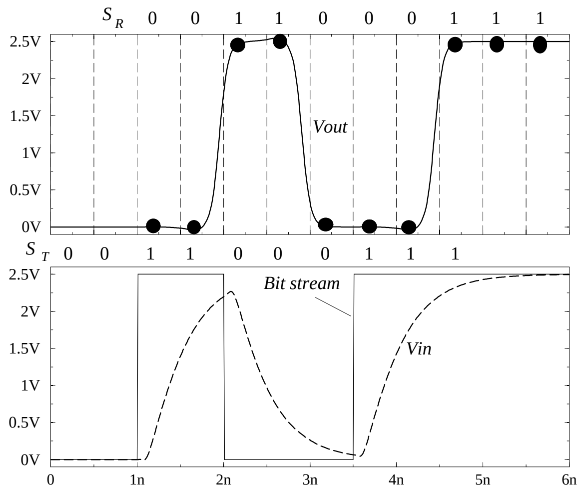

Figure 4: Signal waveforms corresponding to a 2Gb/s bit stream without isolated bits. The original bit stream and the Vin response are shown in the lower of the graph, while the upper graph shows the behavior of $Vout$. The transmitted and received bit streams are also reported on the top.

The same conclusion can be drawn by looking at the *power spectral density* (PSD) of signal S_T. The average PSD of bit streams with bit rate $2Br$ and no isolated bits is similar to that of generic bit streams with bit rate Br. Hence, bit streams without isolated bits can be sent at double bit rate across RC lines acting as a low-pass filters.

3. SPEED-ENHANCEMENT ENCODINGS

In this section we address the problem of transforming an arbitrary bit stream S_0 into an encoded stream S with no isolated bits. According to the results of Section 2, such encodings will allow us to safely transmit the encoded stream over the physical channel at twice its limiting bit rate. If we denote by N_0 the length of S_0 and by N the length of the encoded stream, the ratio N_0/N represents the encoding efficiency. The effective bit rate provided by the speed-enhanced virtual channel will be:

$$Br_{se} = 2Br\frac{N_0}{N}$$

3.1 Run-length limited (RLL) codes

Run-length limited (RLL) codes were developed in the 80's to improve the efficiency of magnetic recording devices (MRDs) [7]. MRDs required a minimum distance D_{min} between subsequent transitions, but they were able to determine the position of each transition with a resolution $D_{res} < D_{min}$. Let m be the (integer) ratio between D_{min} and D_{res}. Encodings guaranteeing a minimum distance of m bits between subsequent transitions would allow MRDs to use a minimum spacing of D_{res} (instead of D_{min}) between bits.

The term *run* denotes a sequence of bits of the same polarity. $RLL(m, M)$ codes generate encoded streams with run lengths bounded by m and M. Lower bound m was suggested by the physical properties of MRDs, while upper bound M was introduced to obtain the global effect of a phase modulation. The efficiency of $RLL(m, M)$ codes has been also discussed in the context of low-power signaling [3].

Encodings producing bit streams with no isolated bits belong to the class of $RLL(2,\infty)$ codes. The lack of upper bounds to the run length will allow us to find codes that are more efficient and easier to implement than upper-bounded $RLL(2, M)$ codes. To stress the fact that we impose only lower bounds to the run length of the encoded stream, we introduce the term $MRLG(2)$, instead of $RLL(2,\infty)$, to denote codes with *minimum run length guaranteed.*

$x^{(n)}$	$x^{(n-1)}$ 0	1
0	0	00
1	11	1

Table 1: Variable-length $MRLG(2)$ encoding.

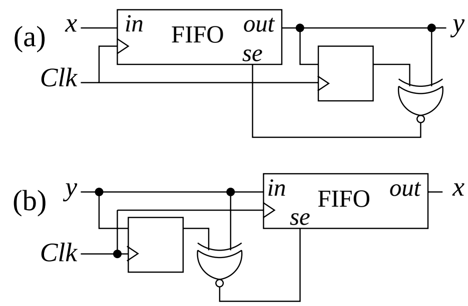

Figure 5: Gate-level netlists of (a) the encoder and (b) the decoder for the variable-length $MRLG(2)$ code of Table 1.

3.2 Variable-length (VL) MRLG codes

The simplest $MRLG(2)$ encoding consists of inserting an extra 1 at the beginning of each run of 1's and an extra 0 at the beginning of each run of 0's. Table 1 shows the codewords used to represent the n-th bit of S_0 (denoted by $x^{(n)}$) depending on the value of the previous bit (denoted by $x^{(n-1)}$). The original sequence will be reconstructed at the receiver by simply discarding the first bit of each run.

Gate-level implementations of the encoder and decoder are shown in Figure 5. Both circuits consist of a flip-flop and an exNor gate used to compare the current and previous bits of S_T and S_R, respectively. Comparison is used by the encoder to generate the control signal (se) that enables the next bit of S_0 to be taken from the queue and transmitted. If the current bit is not equal to the previous one, a transition is detected and the queue is stopped for one clock cycle to transmit twice the first bit of the new run. Otherwise, next bit is shifted out from the queue and passed to the physical channel.

Similarly, comparison is used by the decoder to skip extra bits by means of the control signal (se) that enables the current bit of S_R to be stored in a queue and become part of the reconstructed bit stream. If the current bit in not equal to the previous one it is discarded, otherwise it is stored in the queue.

In practice, the extra bits added to the original sequence have the only effect of providing to the physical channel

$\mathbf{x}^{(n)}$	$x_2^{(n-1)}$ 0	1
000	00000	10000
001	00001	10001
010	00110	11000
011	00011	10011
100	01100	11100
101	00111	11001
110	01110	11110
111	01111	11111

Table 2: Constant-length (3-to-5) $MRLG(2)$ encoding.

an extra time slot to switch the line. The total number of extra bits (i.e., extra time slots) corresponds to the number of transitions in the original stream. Hence, the efficiency of the code is data dependent and the effective bit rate varies over time. If we denote by N_{t0} the total number of transitions in S_0, the length of the encoded stream will be $N = N_0 + N_{t0}$, from which:

$$Br_{se} = 2Br\frac{N_0}{N_0 + N_{t0}}$$

In the worst case of a square waveform with $N_{t0} = N_0$, the proposed approach provides no speed enhancement: $Br_{se} = Br$. Typical interconnects, however, have $N_{t0} << N_0$, giving rise to $Br_{se} >> Br$.

Data-dependent code length actually limits the applicability of such code to serial links with loose synchronization constraints.

3.3 Constant-length (CL) MRLG codes

Constant-length encoding of a bit stream can be obtained by segmenting the stream in words of length L_0 and replacing each word with a codeword of length L. For our purposes, $MRLG(2)$ codewords must satisfy 4 requirements. First, they must be uniquely associated with each original word in order to allow the decoder to reconstruct the original sequence. Second, they must contain no isolated bits. Third, it must be possible to put them in sequence without generating isolated bits. Fourth, L should be lower than $2L_0$, or otherwise the inefficiency of the code will overcome the advantage of doubling the bit rate of the physical channel.

EXAMPLE 1. *Consider the case of $L_0 = 2$ and $L = 3$.* 010 *is not a possible codeword since it contains an isolated 1, thus violating the second requirement. On the other hand, codeword* 001 *does not contain isolated bits, since 0's are not isolated and the single 1 appears at the end of the word, making it possible (in principle) to form a longer run of 1's with the following codewords. Suppose, however, that codeword* 001 *is used to replace word* 01 *wherever it appears in the original stream. If the original stream is* ..01 01.., *the encoded stream will be* ..001 001.., *that doesn't meet the third requirement.*

It can be shown that the shortest $MRLG(2)$ constant-length encoding can be obtained for $L_0 = 3$ and $L = 5$. A valid 3-to-5 encoding is shown in Table 2. It associates with each original word one of two possible codewords, depending on the value taken by the last bit of the last word. We denote by $\mathbf{x}^{(n)}$ and $\mathbf{y}^{(n)}$ the n-th word and the corresponding codeword, respectively, and we use subscripts to refer to their single bits (using subscript 0 for the left-most bit of each word). In Table 2, row labels are the 8 configurations of word $\mathbf{x}^{(n)}$, while column labels are the 2 values of the last bit of the last word ($x_2^{(n-1)}$). It can be easily verified that the code meets requirements 1, 2 and 4. As for the third requirement, it is met by providing, for each original word, two possible codewords starting either by 0 or by 1. Codeword selection is performed by matching the first bit of the current codeword with the last bit of the previous one. This domino-like encoding guarantees that extreme bits are never isolated.

A possible implementation of the encoder and decoder for the 3-to-5 code of Table 2 is shown in Fig. 6. The code was chosen, among those satisfying the $MRLG(2)$ requirements, because of some additional features that make it simple to implement. In particular, the first and the last bits of codeword $\mathbf{y}^{(n)}$ are always equal to $x_2^{(n-1)}$ and $x_2^{(n)}$, respectively, so that they can be directly taken from the original stream without being processed by the encoder. On the other hand, the three inner bits of codewords starting by 0 and by 1 have dual encoding functions that can be realized by the same hardware according to De Morgan's law. These features are exploited by the implementation of Figure 6: the three inner bits are encoded using simple AndOrInverters, while input/output multiplexers are used to switch signal polarities according to the value of $x_2^{(n-1)}$. Decoding is even simpler, since only two AndOrInverters are required.

4. LOW-POWER ENCODING

The energy required to transmit a bit stream S on a physical channel can be expressed as

$$E_{physical}(S, C) = E_{sw}(C) \cdot N_t(S)$$

where $E_{sw}(C)$ is the energy required to switch the channel capacitance C and $N_t(S)$ is the total number of signal transitions in S. Low-power encodings affect communication energy because i) they change the switching activity of the physical channel and ii) they require extra energy to perform encoding and decoding. The energy consumption of the virtual channel can be expressed as

$$E_{virtual}(S, C) = E_{sw}(C) \cdot N_t(S_{encoded}) + E_{codec}(S)$$

where the first term represents transmission energy and the second term the encoding overhead.

We represent the effectiveness of low-power encodings in terms of energy ratio $E_{ratio}(S, C)$:

$$\begin{aligned} E_{ratio}(S, C) &= \frac{E_{virtual}(S, C)}{E_{physical}(S, C)} \\ &= \frac{E_{sw}(C) \cdot N_t(S_{encoded}) + E_{codec}(S)}{E_{sw}(C) \cdot N_t(S)} \end{aligned} \quad (1)$$

Low-power encodings reduce switching activity at the cost of energy overhead E_{codec}. If the line capacitance is too small, the energy overhead overcomes energy savings, making the encoding counterproductive. For large line capacitances, on the contrary, the energy overhead becomes negligible and

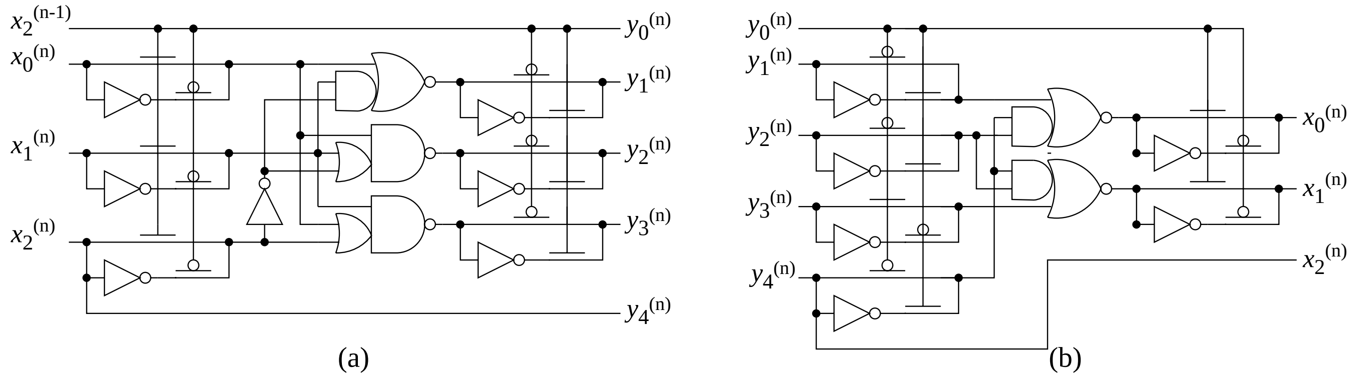

Figure 6: Gate-level netlists of (a) the encoder and (b) the decoder for the 3-to-5 $MRLG(2)$ code of Table 2.

the energy ratio can be approximated by the activity ratio

$$E_{ratio}(S, C) \simeq \frac{N_t(S_{encoded})}{N_t(S)} \quad (2)$$

As discussed in the result section, the energy cost of encoding and decoding operations can then be expressed in terms of minimum line capacitance C_{min} required to achieve energy savings.

The signaling scheme proposed in Figure 1 combines low-power (LP) and speed enhancement (SE) encodings, assuming that they do not impair each other effectiveness. In particular, LP codes should not introduce time redundancy, while SE codes should not increase switching activity.

The $MRLG(2)$ codes of Section 3 meet the second requirement, since they do not introduce extra transitions. This can be easily verified for the VL encoding of Section 3.2, while a more careful analysis is required for the CL encoding of Section 3.3. Referring to Table 2, we compare all possible 4-bit sequences in the original stream (represented by $x_2^{(n-1)}\mathbf{x}^{(n)}$) with the corresponding 6-bit sequences ($y_4^{(n-1)}\mathbf{y}^{(n)}$) of the encoded one. For instance, the original sequence ..1 001... is transformed in ..1 10001..., showing the same number of transitions. The encoding never introduces extra transitions. Rather, it saves two transitions for each occurrence in the original stream of patterns ..1 010.. and ..0 101.., that become ...1 11000... and ...0 00111... respectively.

Many time-irredundant low-power encodings have been proposed [3]. They are either based on space redundancy (extra bus lines) or on the auto-correlation of the original bit stream. A thorough overview of LP codes is beyond the scope of this work. We used only the *adaptive encoding* proposed in [2] to demonstrate the feasibility of our scheme. The n-th bit of the encoded stream ($y^{(n)}$) is obtained as a Boolean function of $x^{(n-1)}$ and $x^{(n)}$. To guarantee decodability, the original bit $x^{(n)}$ must be uniquely determined by $x^{(n-1)}$ and $y^{(n)}$. There are 4 different encoding functions that guarantee decodability. The energy savings they provide depend on signal statistics. Adaptive encoding consists of adaptively selecting one of the 4 encodings on the basis of approximate statistical information collected by observing the bit stream over a sliding window of fixed size. Run-time statistics are independently computed by transmitter and receiver on previously transmitted (received) bits. Hence, no extra information needs to be transmitted to agree on the transmission code.

As suggested in [2], we used a window size of 64 bits to compute run-time statistics.

5. RESULTS AND CONCLUSIONS

We tested the effectiveness of our encodings on a large set of random, deterministic and real-world bit streams. Random streams were generated by simulating Markov chains with randomly assigned transition probabilities; deterministic streams were square waveforms with different periods and duty cycles[2]; real-world traces where GIF, JPEG and MPEG binary files. LP and SE encodings were applied to each stream to evaluate energy and speed enhancements. Energy enhancements are represented by the energy ratio E_{ratio} defined in Equation 2, while speed enhancements are represented in terms of *transmission-time ratio*, T_{ratio}, that is the ratio between the total transmission times of the encoded and original streams:

$$T_{ratio} = \frac{T_{enhanced}}{T_{original}} = \frac{Br}{Br_{se}} \quad (3)$$

Notice that the insertion of encoders and decoders affect transmission latency, but it doesn't affect the cycle-time (i.e., the bit rate) because of pipelining. Hence, the propagation delay of encoders and decoders can be neglected when computing T_{ratio}.

Results are represented in Figure 7 as points on a E_{ratio}-T_{ratio} scatter plot. By construction, point (1,1) represents all original streams. Three sets of points are plotted. Circles represent low-power encoding (LP) without speed enhancement: many points are well below 1, demonstrating that sizeable energy savings can be achieved. Triangles represent low-power encoding combined with CL-MRLG(2): time ratio is 0.83 (corresponding to a speedup of 20%), while energy ratio is less or equal than that achieved by the LP encoding. Squares represent low-power encoding combined with VL-MRLG(2): time ratio depends on the stream (in many cases in approaches 0.5, corresponding to a speedup of 100%), while energy ratio is exactly the same achieved by LP. We remark that, when using variable-length encodings, the effectiveness of LP is also beneficial for speed enhancement. This will be also shown in Figure 8 and Table 3.

When reducing both energy and time, power could be either reduced or increased depending on the relative energy

[2]Square waveforms represent worst-case situations for adaptive LP encodings.

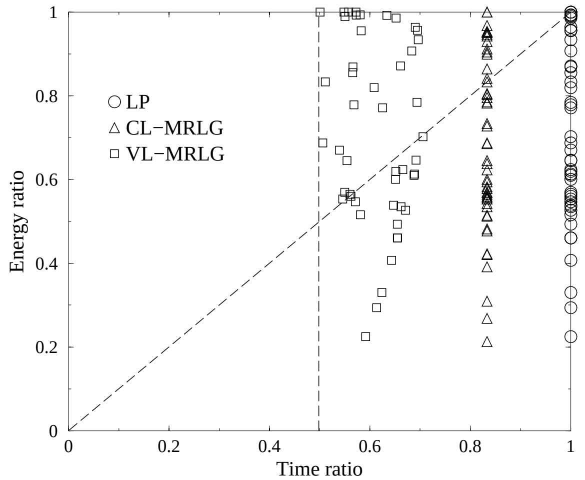

Figure 7: Scatter plot of experimental results.

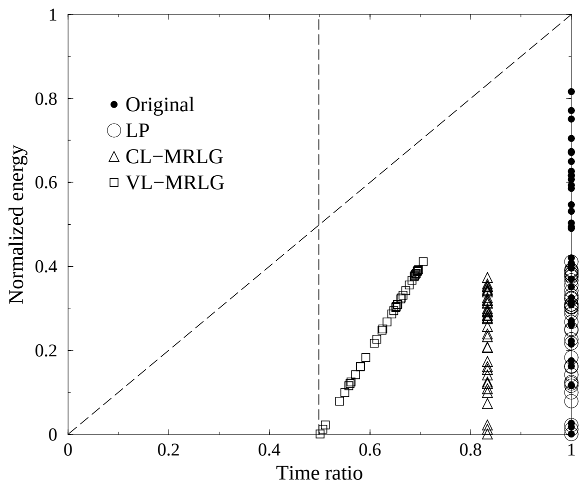

Figure 8: Scatter plot of experimental results.

and speed improvements. In Figure 7, the dashed diagonal represents equation $E_{ratio} = T_{ratio}$. Points below the diagonal represent cases where the double encoding also reduces average power, while points above the diagonal represent cases where the speedup exceeds energy reduction, thus increasing average power.

Figure 8 represents the same results using *normalized energy* instead of E_{ratio}. Normalized energy is the actual energy divided by the the maximum transmission energy, i.e., the energy spent to transmit a square waveform switching the line capacitance at every clock cycle. Dots in Figure 8 represent original streams. Notice that some of the original streams consume more than 80% of the maximum energy, while all encoded streams have total energy well below 0.5. Moreover, all points are far from the dashed diagonal representing maximum power consumption. As a final remark, the linear relation between normalized energy and T_{ratio} for VL-MRLG clearly shows that the lower the normalized energy (i.e., the switching activity) the greater the speed-up produced by variable-length encoding.

Table 3 reports the average values of E_{ratio} and T_{ratio}

Encoding	E_{ratio}	T_{ratio}	C_{min}
LP	0.73	1	36
CL-MRLG	0.95	0.83	> 200
VL-MRLG	1	0.75	∞
LP and CL-MRLG	**0.69**	0.83	48
LP and VL-MRLG	0.73	**0.61**	44

Table 3: Average results.

provided by low-power and speed-enhancement encodings and by their combinations. The minimum wire capacitance C_{min} required (on average) to compensate the energy overhead of encoders and decoders. In order to provide scalable information, C_{min} is expressed in terms of number of gate capacitances of minimum-size inverters (i.e., $C_{min} = Cwire_{min}/Cgate_{min}$). Notice that the average energy ratio achieved by combining LP with CL-MRLG codes is lower than that achieved by LP alone. In fact, CL-MRLG encoding further reduces switching activity. Similarly, VL-MRLG codes provide a greater speed-up (i.e., a lower time ratio) if combined with LP encodings that reduce the number of transitions, thus reducing the number of extra bits needed to remove isolated bits.

6. CONCLUSIONS

In this paper we have proposed a code-based approach for concurrent optimization of energy and performance of on-chip communication channels.

We have observed that RC lines can be used at twice their limiting bit rate to transmit bit streams with no isolated bits. We have presented general encodings that provide bit streams without isolated bits, suitable to be transmitted at double bit rate. Finally, we have shown that such encodings can be combined with low-power encodings to obtain both energy reduction and speed enhancement.

In our signaling scheme, as long as speed-enhancement encodings do not increase the number of signal transitions and low-power encodings do not increase the total stream length, performance and energy concerns are orthogonal. Hence, solutions can be independently developed and then combined without impairing each other.

7. REFERENCES

[1] W. J. Dally *et al.*, *Digital Systems Engineering*, Cambridge University Press, 1998.

[2] L. Benini *et al.*, "Architectures and Synthesis Algorithms for Power-Efficient Bus Interfaces," *IEEE Trans. on CAD*, Vol. 19, No. 9, pp. 969-980, 2000.

[3] M. R. Stan *et al.*, "Low-Power Encodings for Global Communication in CMOS VLSI," *IEEE Trans. on VLSI*, Vol. 5, No. 4, pp. 444-455, 1997.

[4] S. Ramprasad *et al.*, "Signal Coding for Low Power: Fundamental Limits and Practical Realizations," *IEEE Trans. on CAS-II*, Vol. 46, No. 7, pp. 923-929, 1999.

[5] S. Lin *et al.*, *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, 1983.

[6] D. Pamunuwa *et. al.*, "Combating Digital Noise in High-Speed ULSI Circuits Using Binary BCH Encoding," in *Proc. of ISCAS*, vol. IV, pp. 13-16, 2000.

[7] P. H. Siegel, "Recording codes for digital magnetic storage," *IEEE Trans. Magnet.*, pp. 1344-1349, 1985.