Serial T0: Approximate Bus Encoding for Energy-Efficient Transmission of Sensor Signals

Daniele Jahier Pagliari
Politecnico di Torino
Turin, Italy
daniele.jahier@polito.it

Enrico Macii
Politecnico di Torino
Turin, Italy
enrico.macii@polito.it

Massimo Poncino
Politecnico di Torino
Turin, Italy
massimo.poncino@polito.it

ABSTRACT

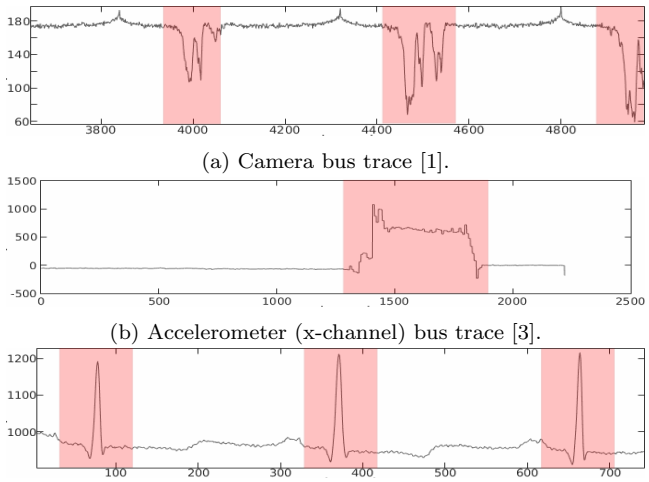
Off-chip serial buses are common in embedded systems, and due to the long physical lines, can contribute significantly to their energy consumption. However, these buses are often connected to analog sensors, whose data is inherently affected by noise and A/D errors. Thus, communication can tolerate small approximations, without a significant impact on the system outputs quality.
In this paper we propose an energy-efficient approximate serial encoding for sensors data, inspired by the T0 technique for parallel buses. Despite its small encoder/decoder overheads, this method is capable of significantly reducing dynamic power ($> 90\%$) in off-chip serial lines, with negligible average error ($<1\%$) on decoded data.

Keywords

Low Power; Serial Interfaces; Bus Encoding; Approximate Computing; Sensors;

1. INTRODUCTION

Off-chip serial communication interfaces are common in most computing systems, due to a variety of advantages over multi-bit parallel buses. In high performance systems, serial links are chosen to overcome skew and jitter issues, thus increasing the maximum operating frequency. In embedded devices with tight cost constraints, moreover, they are preferred for the reduced pin count and wire area and for easier PCB routing layout [13]. In the latter case, serial links are particularly common for the connection between the processing subsystem and sensor ICs (e.g. accelerometers, temperature sensors, cameras, etc.), relying on protocols such as I^2C, *SPI* or *CAN* [13].
However, off-chip interconnects technology does not follow the same scaling trend as semiconductor devices. Wire capacitances per unit length do not decrease as fast as for on-chip links, and so does the dynamic energy consumption. For a single PCB trace, energy consumption is estimated to be around $2pJ/bit$ [12], comparable to the tens of pJs per 32-bit instruction consumed by low-power embedded cores [7]. In this setting, the continuous exchange of data between a single sensor and a processor, at a rate of $100Mbit/s$ for 10 minutes, consumes a total of $600s \cdot 2pJ/bit \cdot 100Mbit/s = 0.12J$. Moreover, a system can easily contain 10 or more sensors operating concurrently.
In summary, serial buses are becoming relevant contributors to the total energy consumption of embedded systems, and the optimization of I/O energy costs is particularly critical for battery operated mobile or wearable devices [12]. Several literature works on this subject have addressed dynamic energy due to electrical level transitions, which is the dominant source of consumption in long serial connections [4,8–12,14]. Hence, most existing approaches try to reduce the number of level transitions by *encoding* the data that has to be sent on the bus. The vast majority of the proposed encodings are *lossless*, meaning that (in absence of channel errors) the transmitted data can be decoded exactly [4, 8, 10, 11, 14].
However, sensor data are inherently affected by environmental noise and A/D conversion errors. Therefore, as long as communication errors are negligible with respect to these other sources of inaccuracy, a *lossy* encoding could be used to achieve larger power savings, without a significant impact on the final quality of results produced by the system. When



(a) Camera bus trace [1].

(b) Accelerometer (x-channel) bus trace [3].

(c) ECG bus trace [5].

Figure 1: Real bus traces produced by three embedded system sensors. High-information areas are highlighted in red.

DAC '16, June 05-09, 2016, Austin, TX, USA

DOI: http://dx.doi.org/10.1145/2897937.2898089

this energy versus quality tradeoff is considered, the problem of finding an energy-efficient serial bus encoding falls within the general framework of Approximate Computing [6].

Approximate encodings have already been considered in some previous works [9,12]. However, none of them fully exploited a common characteristic of sensor data, which we refer to as *burstiness*, i.e. the fact that most information content is concentrated in short time windows. This phenomenon is shown in Fig. 1 for three completely different sensors: when looking at the trace of words transmitted on the serial bus, "events" such as edges of an image, movements of an accelerometer, and heart pulses in a ECG all appear as localized variations with respect to a "default" value which is approximately constant.

In this paper, we propose a novel low power approximate encoding for serial buses, that is particularly effective against the bursty nature of sensor data. We name this technique *Serial T0*, as its working principle is inspired by the T0 encoding for parallel address buses [2]. Although they share the same basic principle (the "locality" of values), our method differs significantly from classic TO, as several of its features specifically target the serial nature of the code.

We show that the savings obtained with Serial T0 for realistic sensor data are superior with respect to both lossless encodings and state-of-the-art approximate encodings. In the best case, more than 90% power reduction can be obtained with less than 1% error on the decoded data.

We also implement the encoding and decoding circuits in hardware, and demonstrate that they introduce negligible energy and latency overheads.

2. BACKGROUND AND RELATED WORK

An off-chip digital serial connection can be modeled as a purely capacitive channel [4,8,9,11,12], in which all dissipation happens in correspondence of electrical level changes. According to this model, the total power coincides with the dynamic power, and can be evaluated as:

$$P_{chan} = P_{dyn} = \alpha \cdot C_{tot} \cdot V_{DD} \cdot V_{swing} \cdot f \qquad (1)$$

where C_{tot} is the total channel capacitance, accounting for driver, pin and wire contributions, V_{DD} is the driver supply voltage, V_{swing} is the voltage swing between electrical levels, and f is the signaling frequency. Finally, $\alpha \in [0,1]$ is the switching activity factor that accounts for the probability that a level transition happens in a given clock cycle.

Most approaches to optimize energy consumption in such channels, including the one presented in this paper, focus on reducing α. In practice, this is done by reducing *intra-word* and *inter-word* transitions, i.e. logic-level changes between adjacent bits of a word and of subsequent words, respectively. To this end, an *encoding* algorithm is developed, so that the switching activity on the bus when transmitting encoded data (*codewords*) is less than the one obtained with unencoded (*raw*) data.

Serial data encodings for low power have been studied only relatively recently with respect to algorithms for parallel buses, despite the fact that these interconnects are very popular in modern embedded systems. They can broadly be categorized in two groups: *lossless* or *accurate* encodings, and *lossy* or *approximate* encodings.

The lossless category includes encodings for which the transmitted data can be reconstructed exactly at the receiver's end, in absence of channel errors [4,8,10,11,14]. Most lossless approaches exploit a priori information on the words to be transmitted, and in particular on their temporal correlation. For example, the *SerIalized Low ENergy Transmission* (SILENT) algorithm works by transmitting on the bus the bitwise difference (XOR) between subsequent words [8]. The transition count is reduced when subsequent data are strongly positively or negatively correlated (i.e. when the number of bits that differ between subsequent words is very small or very large) because the encoded words are dominated by 0s and 1s respectively.

The SILENT approach fails when data are uncorrelated, situation in which it can even lead to an *increase* in power consumption. The authors of [10] tackle this issue by proposing a hybrid encoding that combines words in different ways based on a prediction of the data correlation. The algorithm of [14], instead, works by inverting bit values in the even positions of a word, when the number of intra-word transitions is more than half the length of the word. To allow decoding, an *invert* bit is added to each codeword; a further *parity* bit, shared among multiple codewords, is used for channel errors recovery. A similar idea is described in [4], where the encoding swaps the positions of some bits within words to reduce the transition count. These swaps are signaled to the decoder via some additional *parallel* control lines. Finally, in [11] a lossless bus encoding for *display interfaces* is described. This algorithm exploits data locality in images, by transmitting pixel differences rather than pixel values on the bus. Moreover, it uses a look-up-table to map the most common differences to low intra-word transition count codewords.

Lossy encodings sacrifice total accuracy in the transmission, in order to reach larger transition count reductions. With other purposes (e.g. compression) this concept is very popular in the field of multimedia, but it has only been applied recently to serial bus I/O. One of the first works in this field is [9], again targeting display interfaces. The authors propose two techniques for two of the most popular data-link layer bus standards for LCDs (*LVDS* and *TMDS*). Both algorithms rely on saturating to all zeros or to all ones some LSBs of the transmitted pixels, also accounting for inter-pixel transitions. Recently, a new encoding called *Rake* has been introduced in [12]. Rake works by approximating each word with a lower transition count one, under a maximum value-deviation constraint. The approximate code is generated by inverting some bits of the original word in order to reduce transitions, and the algorithm has linear complexity in the bit-width of the data.

Lossy bus encodings can be considered as an instance of the recent design paradigm called Approximate Computing (AC) [6]. AC formalizes and groups all those design techniques that sacrifice the maximum computational accuracy constraint, and instead trade-off results quality with other metrics (e.g. power consumption). This idea has recently gained a lot of interest, and AC techniques for low power have been proposed at many levels of the computing stack [6]. However, very few works, among which is the already mentioned Rake encoding [12], have applied AC principles to on-chip and off-chip I/O interfaces.

3. SERIAL T0 ENCODING

In this paper, we propose a novel approximate encoding for serial connections. In particular, we focus on a precise category of off-chip buses, very common in embedded systems;

those that connect a processing element (e.g. a System-on-Chip) with external sensors ICs. Complex embedded systems can easily contains tens of sensors, most of which are interfaced with the processor via a serial bus; common protocols include I^2C, *SPI*, *MIPI* and *CAN* [13].
As explained in Sec. 1, typical sensor signals (and thus the relative bus traces) share a common characteristic: most of the information is localized in short time windows [1, 3, 5]. The sequence of data transmitted by the sensor towards the processing element can be thought of as a sequence of "idle" and "active" phases (see Fig. 1). We call this characteristic signal *burstiness*. The nature of the active phases clearly depends on the considered sensor (e.g. edges in a picture from a camera, hearth pulses in an ECG, etc.).
It is intuitive that a good approximate encoding for bursty signals should preserve as much as possible the information content in active phases, whereas larger approximations can be tolerated in idle phases. Following this intuition, we develop a simple encoding that approximates input words with low transition count codewords when the sequence of data is varying slowly, and forwards the accurate words to the bus when variations are larger.

3.1 Parallel T0 Encoding

The main inspiration for this technique comes from a classic encoding for parallel buses, called T0 [2]. It must be noted however, that the objective of T0 is completely different from the one that we try to address in this paper. Firstly, T0 is an *accurate* encoding, i.e. does not perform any kind of approximation; secondly, it is meant for *parallel* buses. Last, T0 targets *address* buses, and it exploits the specific correlation of typical address patterns. Conversely, in this work we focus exclusively on *serial* connections for *data* (the only type of information that can tolerate approximations).
In its basic form, the T0 algorithm encodes addresses on a parallel bus as follows:

$$(B^{(t)}, INC^{(t)}) = \begin{cases} (B^{(t-1)}, 1) & \text{if } b^{(t)} = b^{(t-1)} + S \\ (b^{(t)}, 0) & \text{otherwise} \end{cases} \quad (2)$$

where $b^{(t)}$ and $B^{(t)}$ are the input word and corresponding codeword at time t, $INC^{(t)}$ is the value of an additional redundant bit, and S is a constant power of 2 called *stride*, which represent the parallelism of the memory (e.g., $S = 4$ for 32-bit addresses).
The principle that is exploited is the locality of memory references, especially for instructions. In fact, instruction memory locations are mostly accessed in ascending sequences with a fixed stride, which are interrupted by control flow operations (e.g. jumps). When such sequences occur, the T0 encoder freezes all lines of the address bus to a constant value. The additional INC line is driven to logic 1, so that the receiver can autonomously compute the correct address. For an infinitely long ascending sequence with stride equal to S, T0 reduces the transition count to 0.

3.2 Approximate Serial T0 Encoding

In a sensor data trace, the alternation between idle and bursty phases is somehow similar to the one between fixed-stride sequences and jumps in instruction addresses.
Therefore, we propose to freeze the serial line to a constant value in idle phases, reducing the transition count to 0, and to transmit the actual values of the input words only during bursty phases. This can be achieved with the following scheme:

$$B^{(t)} = \begin{cases} \text{0-TC pattern} & \text{if } \|b^{(t)} - b^{(t')}\| \leq T_h \\ b^{(t)} & \text{otherwise} \end{cases} \quad (3)$$

where 0-TC stands for zero transition count, and T_h represents a tunable *maximum error threshold*. Time t' represents the last instant in which a data word was directly sent on the bus. Notice the clear similarity between Eq. 2 and Eq. 3. However, in a serial bus, repeating the last transmitted word does not in general reduce the number of transitions. Therefore, an appropriate 0-TC pattern is transmitted instead. The decoding phase of Serial T0 works as follows:

$$b^{(t)} = \begin{cases} b^{(t')} & \text{if } B^{(t)} = \text{0-TC pattern} \\ B^{(t)} & \text{otherwise} \end{cases} \quad (4)$$

The received codeword is simply copied to the output, except for the 0-TC pattern. In that case, the decoder assumes as output the value of the last valid word (non 0-TC) received. Approximations occur when the decoder output is different from the transmitted word, that is when:

$$0 < \|b^{(t)} - b^{(t')}\| \leq T_h \quad (5)$$

A larger value of T_h produces more approximations, but allows to transmit a larger percentage of words as 0-TC patterns. Hence, this parameter can be used to explore the tradeoff between energy consumption and output quality, typical of Approximate Computing.
Notice that Serial T0 produces approximations when the input signal varies slowly (idle phases), while it transmits accurately in high activity phases (bursts). Therefore, as shown in Sec. 4, this simple idea is very effective to reduce power consumption in sensor buses.

3.3 Selection of the 0-TC pattern

In a N-bit word, there are only two patterns with 0-TC: all-zero (00..0) and all-one (11..1). Thus, in order to minimize the TC, one of these two patterns has to be sent on the bus during "approximate phases". However, in most data representations, all binary patterns are already used to represent valid values. For example, in unsigned binary (00..0) represents decimal 0, whereas (11..1) is $2^N - 1$. There are only two ways to overcome this issue: adding a redundant bit to distinguish between the 0-TC pattern and a real binary word with zero transitions, or sacrificing one of those two patterns, devoting it to be used *only* as 0-TC pattern.
Adding a redundant bit to a serial code is not as convenient as for the parallel case. If the bit is on a separate physical line (i.e. in *parallel* with the data line), as in [4], the routing and area cost of the connection is effectively doubled. Alternatively, the redundant bit can be transmitted on the same physical line with the data, adding it as MSB or LSB of the codewords, as in [14]. However, this approach incurs in a maximum bandwidth overhead, since a fraction $1/(N + 1)$ of the bandwidth is "wasted" for transmitting the redundant bit. In both cases, power overheads due to transitions of the redundant bit are also present.
To avoid these overheads, in Serial T0 we do not add any redundant bits, and the 0-TC pattern is exclusively devoted to that function. In particular, the encoding "sacrifices" the (11..1) pattern. The reason for this choice (as opposed to the (00..0) pattern) is that in typical integer or fixed point

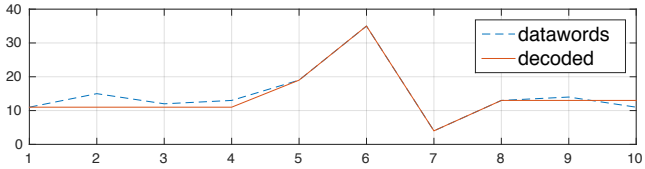


Figure 2: Example of operation of the Serial T0 algorithm for a trace of 8-bit unsigned data, under a maximum relative error constraint of 3.125%.

Dataword	TC	Codeword	TC	Decoded	Error
00001011	3	00001011	3	00001011	0
00001111	5	11111111	3	00001011	4
00001100	8	11111111	3	00001011	1
00001101	11	11111111	3	00001011	2
00010111	15	00010111	7	00010011	0
00100011	19	00100011	11	00100011	0
00000100	22	00000100	14	00000100	0
00001101	25	00001101	17	00001101	0
00001110	28	11111111	17	00001101	1
00001011	31	11111111	17	00001101	2

Table 1: Example of operation of the Serial T0 algorithm for a trace of 8-bit unsigned data (see Fig. 2).

representations used by sensors, i.e. unsigned, two's complement and module and sign (M&S), the all-zero code is very common. Conversely, the all-one pattern is unlikely to occur for unsigned data, as sensors outputs seldom reach full-scale values. Furthermore, in M&S (11..1) is a redundant pattern which corresponds to one of the two possible ways to represent 0.

In all three representations, Serial T0 deals with the occurrence of a real all-one word by replacing it with another pattern. In two's complement and M&S, (11..1) can be replaced with (00..0). This does not introduce an error in M&S, whereas the error in two's complement is $\|-1-0\| = 1$. In unsigned, (11..1) can be replaced by (11..10), incurring again in an error of $\|(2^N - 1) - (2^N - 2)\| = 1$. For two's complement and unsigned, this substitution represents an *additional approximation* with relative error equal to $1/FS$, FS being the full-scale value for the selected representation.

3.4 Example of Operation

An example of operation of the Serial T0 encoding is reported in Fig. 2 and Table 1, for a trace of unsigned data words represented on 8-bit. The threshold on the maximum allowed error is set to $T_h = 8$, i.e. 3.125% of the full-scale. In this example, the total transition count (TC) is reduced of $\frac{31-17}{31} = 45.2\%$, taking into account both intra-word and inter-word transitions (we assume a MSB-first bit ordering), with just 0.4% average error on the decoded data. Notice that, for visualization purposes, the example signal has a comparably long idle phase and burst, whereas in a real sensor trace, idle periods are normally longer than active ones (see Fig. 1). Thus, in real applications, Serial T0 can obtain greater TC reductions with similar quality constraints.

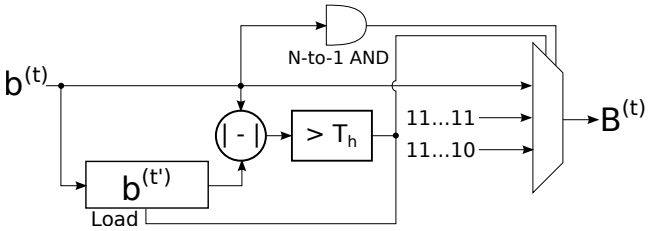


Figure 3: Serial T0 encoder block diagram.

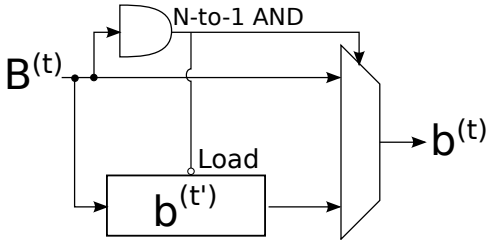


Figure 4: Serial T0 decoder block diagram.

3.5 Encoder and Decoder Implementation

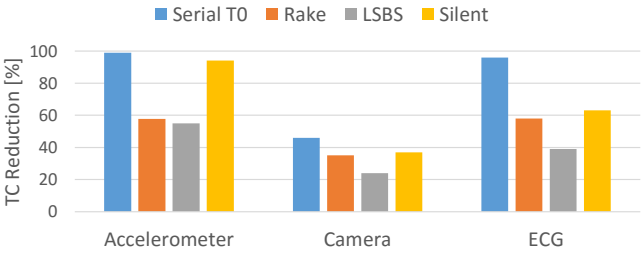
Fig. 3 shows a conceptual block diagram of the encoding circuitry for Serial T0. There are six main components. A N-bit register stores the last word transmitted without approximations $b^{(t')}$. A subtractor and a comparator compute whether the absolute value difference between the register content and the current word $b^{(t)}$ is larger than the threshold T_h. An N-input/1-output AND gate is used to detect the occurrence of a real all-one pattern. Based on the outputs of the comparator and of the AND gate, a multiplexer is used to send on the bus either the input data word, the 0-TC pattern, or the replacement pattern (11...10 in the case of unsigned data). Another N-bit register is used to store T_h (hidden inside the comparator in the figure), so that the maximum error constraint can be changed at runtime. This allows to dynamically tune the quality of the transmission, and to switch points in the energy versus quality design plane, for instance in response to a change in the state of the overall system (e.g. battery charge).

The decoder block diagram is shown in Fig. 4. In this case, the circuitry is even simpler. Once again, a N-bit register stores the last valid received word. Another N-to-1 AND gate detects the occurrence of a 0-TC pattern, and a multiplexer produces the final output selecting between the current codeword and the last valid word.

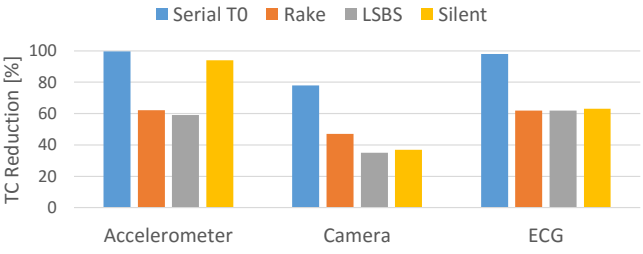
4. EXPERIMENTAL RESULTS

4.1 Comparative Analysis

To assess the goodness of Serial T0, we compared it with two state-of-the-art approximate encodings for serial buses: Rake, presented in [12], and Least Significant Bits Saturation (LSBS) described in [9]. Moreover, we also evaluated our approach against a popular *accurate* encoding. For a fair comparison, we did not consider techniques that include redundancy bits such as [4] and [14]. Instead, we selected SILENT [8] as a representative of state-of-the-art irredundant accurate encodings.



(a) Maximum relative error constraint: 2%.



(b) Maximum relative error constraint: 5%.

Figure 5: Comparison between serial encodings in terms of transition count reduction for different input sets.

Serial T0, Rake and LSBS all make approximations based on a constraint in terms of *maximum allowed error*. Therefore, they can be easily compared under the same quality settings. SILENT, instead, represents a "baseline" reference in terms of power saving, which all lossy encodings should beat in order to justify the presence of approximations.
We selected the following three sets of input stimuli, each composed of 10000 or more words for these experiments:

- To mimic the data transmitted on the bus by a camera sensor, we used a set of images from the Kodak online database [1]. The images are represented on 24-bit RGB (8-bit for each color component).
- As a representative of biomedical sensors, which typically produce bursty bus traces, we used a set of real electrocardiogram (ECG) records from the Physionet database [5]. These signals are sampled on 11-bit with unsigned binary representation.
- A third set of sensor inputs consists of realistic bus traces generated by the axis of an accelerometer IC [3] for a shock detection application. These samples are represented in 2's complement on 12-bit.

Comparative results are shown in Fig. 5. The bar charts report the transition count reduction for all input sets, under two quality constraints (maximum relative error of 2% and 5% respectively). As explained, TC reduction directly corresponds to power saving (Eq. 1). To transform error constraints into parameters for the lossy encodings (T_h for Serial T0 and analogous for Rake and LSBS) the percentages have been multiplied by the full-scale value of the data representation used in each input set, and then rounded to the nearest valid parameter (e.g. T_h must be an integer).
Serial T0 is superior to the other algorithms for all the three input sets considered. In particular, with the 5% constraint, it saves over 30% more transitions w.r.t. all other algorithms

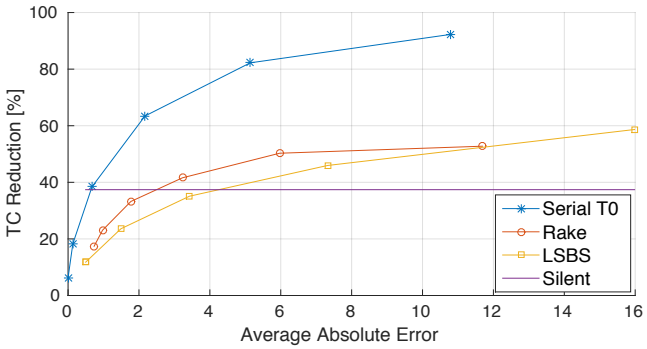
(a) Input Image. (b) Decoded Image (77% power saving on the bus).

Figure 6: Effect of Serial T0 encoding and decoding with 5% error constraint on a popular test image.

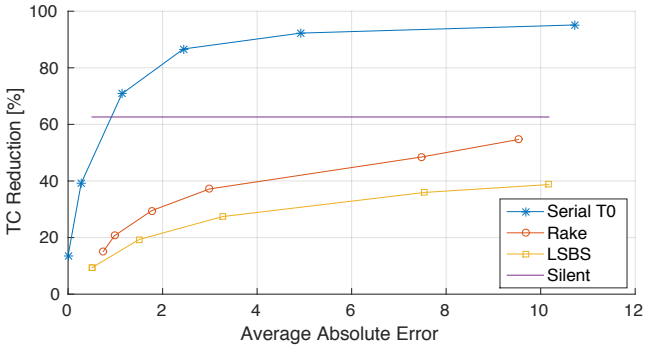
for the Camera and ECG traces. SILENT is very competitive for the accelerometer data, but this depends on the nature of the particular trace considered, a section of which is shown in Fig. 1b: with respect to ECG and camera, the data of this trace are more constant in the idle phase. Thus, both SILENT and Serial T0 reduce the TC to a value close to 0. Nevertheless, the power saving obtained by our algorithm is 4% better than SILENT (98% against 94%) under the 5% error constraint. A larger difference is expected for accelerometer data with shorter or less constant idle phases.
To further demonstrate the quality of our algorithm, Fig. 6 shows a popular benchmark image in its original form [1], and the corresponding decoded image after a simulated Serial T0 transmission. The error constraint has been set to 5%, which allows to reduce the TC by 77%, consistently with Fig. 5. Despite this significant power saving, the two images look very similar, and the only differences can be noticed in areas that do not contain edges (e.g. background and skin), as expected. The similarity is also confirmed by quantitative analysis, as the Mean Structural Similarity Index between the two images is 0.99.

4.2 Quality versus Power Tradeoff

Serial T0 and other lossy algorithms (e.g. Rake and LSBS) permit to dynamically change error constraints at runtime. This allows to obtain multiple configurations, which can be represented as points in the power saving versus quality plane [6]. Fig. 7 shows this tradeoff in the case of the Camera and ECG input sets. As measure of power saving we use once again TC reduction, shown on the vertical axis, whereas the horizontal axis reports the *average* error, computed after decoding. Since error is the inverse of quality, optimal configurations are found towards the upper left corner. For approximate encodings, the different configurations (points in the graphs) have been obtained by changing the maximum error constraint: the allowed error in *absolute* terms has been swiped in steps of ascending powers of 2, from 1 to 32 for both input sets. The horizontal purple line represents the TC reduction obtained by SILENT, which being accurate does not make any error.
These graphs highlight the effectiveness of Serial T0 for sensor data. The power saving obtained for a given output quality is significantly superior w.r.t. other approximate encodings. Interestingly, for the ECG data trace, Serial T0 becomes convenient with respect to SILENT if an average error greater than 1 (0.04% of the full-scale value 2047) is ac-



(a) Camera bus trace.



(b) ECG bus trace.

Figure 7: Power saving versus quality (error) tradeoff for two sensors bus traces.

Circuit	f_{max} $[GHz]$	Area $[\mu m^2]$	Power $[mW]$
Encoder	1.61	787.10	0.18
Decoder	3.57	170.05	0.07

Table 2: Synthesis results for 12-bit Serial T0 encoder and decoder on 45nm CMOS.

cepted. On the contrary, all other encodings remain inferior to SILENT even for a 10 times larger average error.
Finally, notice that *more than* 90% *power saving is obtained for the ECG inputs with an average error of* ≈ 5*, which corresponds to less than* 0.5% *of the full-scale.*

4.3 Encoder and Decoder Overheads

The final experiment that we performed is the synthesis of Serial T0 encoder and decoder, shown in Figs. 3 and 4. We described the two block in VHDL and synthesized them with Synopsys DC K-2015.06, targeting a 45nm CMOS standard cell library from ST Microelectronics. We then verified the correctness of the designs by performing both RTL and post-synthesis simulations in Mentor QuestaSim v10.4. For these simulations, we used some simple input sequences similar to the one in Table 1. We also annotated switching activity information for internal nodes of the post-synthesis netlists, which we then loaded in Synopsys PT J-2014.12 to perform accurate power estimation.
Synthesis results are summarized in Table 2 for a 12-bit implementation. We used this data-width as it matches with the typical accuracy of many sensor ICs. It is evident from these results that, even at very high operating frequencies, the encoder and decoder overheads are negligible.

5. CONCLUSIONS

We have presented Serial T0, a very simple yet effective approximate encoding for serial buses. We have shown that this encoding is particularly interesting for sensor data. The number of sensors in modern embedded systems is typically very large, but the data they produce can accept some approximations without a significant impact on the final quality of results. Therefore, Serial T0 configures as an effective way to reduce power in long off-chip lines that connect processing elements and sensors ICs.
A possible future improvement of this work is to consider a hybrid encoding, that combines the best features of Serial T0, superior for bursty signals, with those of LSBS or Rake, which are instead very effective for general data.

6. REFERENCES

[1] Kodak image database. http://r0k.us/graphics/kodak.
[2] L. Benini et al. Asymptotic zero-transition activity encoding for address busses in low-power microprocessor-based systems. In *Proc. GLS-VLSI, 1997.*, pp. 77–82.
[3] C.M.N. Brigante et al. Towards Miniaturization of a MEMS-Based Wearable Motion Capture System. In *IEEE Trans. on Indust. Electron.*, 58(8):3234–3241, Aug. 2011.
[4] S. Ghosh et al. Data correlation aware serial encoding for low switching power on-chip communication. In *Proc. ISVLSI, 2014*, pp. 124–129.
[5] A. Goldberger et al. Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.
[6] J. Han and M. Orshansky. Approximate computing: An emerging paradigm for energy-efficient design. In *Proc. ETS, 2013*, pp. 1–6.
[7] N. Ickes, D. Fichelstein, and A.P. Chandrakasan. A 10-pJ/instruction, 4-MIPS micropower DSP for sensor applications. In *Proc. A-SSCC, 2008*, pp. 289-292.
[8] K. Lee, S.J. Lee, and H.J. Yoo. Silent: serialized low energy transmission coding for on-chip interconnection networks. In *Proc. ICCAD, 2004*, pp. 448–451.
[9] M. Poncino and E. Macii. Low-energy RGB color approximation for digital LCD interfaces. *IEEE Trans. on Consum. Electron.*, 52(3):1004–1012, Aug. 2006.
[10] X. Ren et al. Adaptive low-power transmission coding for serial links in network-on-chip. *Procedia Engineering*, 29:1618 – 1624, 2012.
[11] S. Salerno et al. Limited intra-word transition codes: an energy-efficient bus encoding for LCD display interfaces. In *Proc. ISLPED, 2004*, pp. 206–211.
[12] P. Stanley-Marbell and M. Rinard. Value-deviation-bounded serial data encoding for energy-efficient approximate communication. Technical Report MIT-CSAIL-TR-2015-022, Massachusets Institue of Technology (MIT), 2015.
[13] F. Vahid and T. Givargis. Embedded System Design: A Unified HW/SW Introduction Wiley, 2001
[14] J. Zeng, J.-Y. Zhou, and R.-B. Lin. Transition inversion coding with parity check for off-chip serial transmission. In *Proc. ICECS, 2014*, pp. 634–637.