Probabilistic Value-Deviation-Bounded Source-Dependent Bit-Level Channel Adaptation for Approximate Communication

Bilgesu Arif Bilgin and Phillip Stanley-Marbell

Abstract—Computing systems that can tolerate effects of errors in their communicated data values can trade this tolerance for improved resource efficiency. Many important applications of computing, such as embedded sensor systems, can tolerate errors that are bounded in their distribution of deviation from correctness (distortion). We present a channel adaptation technique which modulates properties of I/O channels typical in embedded sensor systems, to provide a tradeoff between I/O power dissipation and distortion of communicated data. We provide an efficient-to-compute formulation for the distribution of integer distortion accounting for the distribution of transmitted values. Using this formulation we implement our value-deviation-bounded (VDB) channel adaptation. We experimentally quantify the achieved reduction in power dissipation on a hardware prototype integrated with the required programmable channel modulation circuitry. We augment these experimental measurements with an analysis of the distributions of distortions. We show that our probabilistic VDB channel adaptation can provide up to a 2× reduction in I/O power dissipation. When synthesized for a miniature low-power FPGA intended for use in sensor interfaces, a register transfer level implementation of the channel adaptation control logic requires only 106 flip-flops and 224 4-input LUTs for implementing per-bit channel adaptation on serialized streams of 8-bit sensor data.

Index Terms—Approximate computing, approximate communication, sensors, information theory.

1 Introduction

Most computing systems are designed to prevent errors in computation, memory, and communication. Guarding against errors however requires energy, temporal redundancy, or spatial redundancy and therefore consumes resources. But not all systems need to be free of errors: In some systems, either by explicit design or by the nature of the problems they solve, system output quality degrades gracefully in the presence of errors.

Several important applications of computing systems, ranging from wearable health-monitoring systems to neuromorphic computing architectures [1] often dissipate a significant fraction of their energy moving data. For these systems the effects of errors are best quantified in terms of their integer distance, rather than using a Hamming distance. At the same time, the computations that consume this data can often tolerate errors with a wide range of integer distances with limited system-level consequences [2], [3], [4].

Because electrical communication interfaces do not benefit as much as processors and other digital logic from semiconductor process technology scaling, the fraction of system energy dissipated by sensor data accesses and sensor data movement is currently large [5], [6], [7] and will only grow in future embedded sensor-driven and neuromorphic systems. It is therefore an important challenge to find ways to reduce the power dissipated in sensor data acquisition and transfer. One way to do this is by exploiting knowledge of the properties and uses of the transported data, to reduce communication energy expenditure in exchange for, e.g., bounded inaccuracy of the communicated values [8], [9], [10], [11], [12], [13].

Reducing I/O energy usage in exchange for bounded inaccuracy requires a physical channel property that provides such a tradeoff. One example of such a tradeoff is between energy dissipated in the pull-up resistors on a serial communication interface such as I2C and the error or erasure rate. The choice of the pull-up resistor value influences signal integrity at a given I/O speed and at the same time affects communication power dissipation.

1.1 Contributions

We exploit value representation and hardware properties of communication channels to design channel adaptation techniques that trade lower power dissipation for value-bounded distortion, for a given bound on its distribution. Given a target bounding distribution, our method varies the channel bit-error rates across the ordinal bit positions in a word to reduce energy used in transmission in such a way that the distribution of the resulting distortion is enclosed

• *B. Bilgin and P. Stanley-Marbell are with the Department of Engineering, University of Cambridge, Cambridge CB3 0FA, UK.*
For correspondence e-mail: phillip.stanley-marbell@eng.cam.ac.uk

Manuscript received December 2019; revised 11th July 11, 2020.
Author contributions: PSM formulated the problem of probabilistic value-deviation-bounded source-dependent bit-level channel adaptation, designed the experimental PCB testbed, and assisted with the I2C experiments. BAB developed the methods to calculate the integer value distortion distributions, the search algorithms to find optimal channel adaptation, devised and processed I2C experiment results, derived I2C channel error analysis, and developed the channel adaptation hardware synchronization algorithm. Both authors contributed to writing.

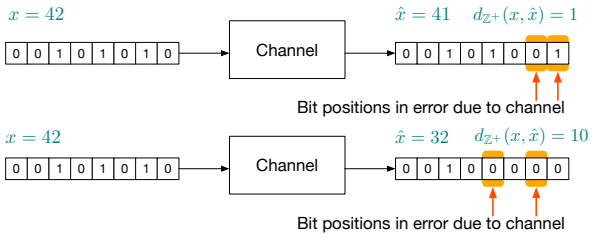


Fig. 1. Channel model and distance functions: The same number of bit errors (same Hamming distance) can lead to vastly different value deviations (integer distances).

within the target distribution. We present a proof-of-concept hardware implementation of the technique which operates by changing properties of the I2C communication channel for individual bit positions according to provided channel adaptation specifications. We calculate these specifications at system design time to generate the appropriate channel adaptation hardware. The generated channel adaptation hardware is efficient. Because the technique we present changes the physical properties of only the channel, its implementation requires no changes to the transmitter (e.g., a sensor), easing potential adoption.

A new technique for approximate communication, *probabilistic value-deviation-bounded (VDB) bit-level channel adaptation* (Section 4): In a system with integer-representing binary-valued random variable input X and output $\hat{X}$, let the random variable $M = d_{\mathbb{Z}+}(X, \hat{X})$ denote the absolute value of the difference between the integers X and $\hat{X}$. Figure 1 shows an example. We refer to the random variable M as *integer-value distortion*, or just *distortion*. It takes on instance values m and has tail distribution $\Pr(M > m)$. We denote integer distance errors tolerated by an application using an upper bound $\widehat{\mathrm{T}}_M(m)$ on the tail distribution of m seen at the output of the system (e.g., the receiver in Figure 1). Given any instance of the function $\widehat{\mathrm{T}}_M(m)$, our method derives an application-specific per-bit channel adaptation such that $\Pr(M > m) \leq \widehat{\mathrm{T}}_M(m)$ for all possible distortion values m. In contrast, rate distortion codes [14], [15] typically only bound the expected value of the channel distortion. To facilitate channel adaptation, we developed a computationally-efficient algorithm for calculating the distortion distribution of a given channel (i.e., given its bit-error probabilities) with given input distribution (Section 4.1). Using this algorithm, we demonstrate channel adaptation by implementing the offline search for finding optimal bit-error probabilities with bounded distortion distribution in the cases when the bit-error probabilities are identically-distributed over the bit positions and when they are non-identically distributed (Section 4.2). The latter enables bit-level channel adaptation, while the former only achieves word-level adaptation.

A case study for the proof of concept of the proposed method for the I2C communication channel (Section 5): We use pull-up resistance modulation as a means for I2C channel adaptation and experimentally verify power savings of up to $2\times$ ($0.8\,\mathrm{mW}$ absolute saving) on a custom hardware research platform (Section 5.3). This saving in I/O power is larger than the operation power dissipation of many state-of-the-art sensors in low-power embedded systems and is therefore significant. Combining experimental measurements (Section 5.3 and Section 5.4) with a model of I2C interface circuitry (Section 5.1), we estimate circuit parameters (not always known for commercial integrated circuits) of our hardware platform (Section 5.5). We analytically derive the dependence of the bit-error probabilities of an I2C channel with given circuit parameter values on the value of the external pull-up resistance (the channel modulator). We use this analytic model to calculate the distortion distribution as a function of pull-up resistance value and identify the range of resistances for modulation to have plausible control over bit errors for reliable channel adaptation (Section 5.6).

An analysis of the hardware cost for implementing channel adaptation (Section 6): We demonstrate the feasibility of our probabilistic VDB bit-level channel adaptation technique by synthesizing the digital logic required to control the channel adaptation, targeting a small-footprint low-power sensor interface FPGA. We report the required FPGA resource usage for sensor sample resolutions common in embedded sensor systems (e.g., 8–16 bits).

1.2 Adaptation versus Source / Channel Coding

The method we present is hard to classify as a coding scheme in the conventional nomenclature of the literature. Traditional *source coding* changes the words transmitted on a channel to adapt to properties of the inputs (e.g., for compression). Traditional *channel coding* changes transmitted words to adapt to properties of the channel (e.g., for forward error correction). In contrast, the method we present (probabilistic VDB bit-level channel adaptation) takes the peculiar approach of changing the properties of the communication channel for different ordinal positions in a transmitted word in order to reduce the energy needed to transmit the data. Probabilistic VDB bit-level channel adaptation does this with the objective of reducing power dissipation by inducing channel errors that lead to values at the channel's receiver which are at small integer distances from those sent by the channel's transmitter. We present a detailed comparison to the related literature next, in Section 2.

2 Related Research

VDB integer codes [16] were proposed as a means of trading energy efficiency for correctness in the context of program variable encodings when program variables can tolerate some distribution of integer value distortions. For serial communication interfaces such as I2C and SPI, which are common in energy-constrained embedded systems, serial VDB codes (VDBS encoding) [9], [10] provide a concrete encoding method. VDBS encoding is deterministic and has the effect of re-quantizing the number representation of values. This deterministic distortion may be undesirable for certain applications: When applied to images with encoder settings to maximally reduce I/O energy dissipation, VDBS encoding leads to regular quantization banding in images. For these and similar applications, encoding methods that trade energy-efficiency for accuracy and whose induced distortion is stochastic are therefore desirable. Existing VDBS encoding techniques must be implemented at the data transmitter. Although, being a re-quantization, they require no

decoding, their adoption requires their integration into data sources such as sensors. Because the consumers of sensor data, such as microcontrollers, provide the opportunity for programmability, a variant of VDBS encoding that could be enforced from the side of consumers of sensor data could enable widespread adoption of VDBS encoding and its benefits.

Rate distortion theory [14], [17] investigates the tradeoff of encoding efficiency (*rate*) for deviation of encoded values (*distortion*). A *distortion function* or *distortion measure* specifies this distortion as a function of the original data and its encoded form. The value deviations we consider in this work are integer distance distortions. Unlike the convention in rate distortion theory, the method we present guarantees bounds on the entire distribution of distortions, rather than only guaranteeing bounds on the expected value of the distortion.

Relation to previous probabilistic VDB methods: Prior work [8] on probabilistic VDB codes with integer value distortion adopts the same approach of trying to guarantee bounds on the entire distribution of distortions. The method we present in Section 4 builds on and improves on that work in several dimensions. First, we provide a fast method for exact calculation of the distortion distribution for the general case when bit-error probabilities may depend on the bits transmitted. This is in contrast with previous [8] work, which used an SMT solver to search for optimal bit-level channel adaptation probabilities. That method only attempts at an approximate calculation of distortion distribution by considering only up to k-bit concurrent errors for L-bit words transmitted. Second, we propose an alternative search algorithm through the space of bit-error probabilities, which utilizes the exact calculation method we developed. This method, besides being exact, outperforms the SMT search based on approximate constraints in terms of computation time. Third, we verify the feasibility of channel adaptation on I2C systems by carrying out a proof-of-concept implementation. We demonstrate power reduction using empirical measurements of power dissipation and carry out analytical investigation of induced channel errors to explore the design space. Fourth and finally, we demonstrate the adoptability of the approach by estimating the resource usage of the digital logic block required to control the analog channel adaptation circuitry.

3 Definitions

Table 1 summarizes the notation we use in the rest of this work. Let X denote the random variable with instance value $x \in \mathbb{Z}_2^L$, an L-bit binary sequence transmitted on a channel. Let the channel output be $\hat{x} \in \mathbb{Z}_2^L$ with corresponding random variable $\hat{X}$.

We consider systems where channel inputs encode unsigned integers and we define channel errors as the absolute difference of these integer values. Let x_i denote the i'th bit of x and let $u : \mathbb{Z}_2^L \to \mathbb{Z}$ denote the conversion from binary to an integer with

$$u(x) = \sum_{i=0}^{L-1} x_i 2^i.$$

TABLE 1
Terminology and notation.

Notation	Definition	Example
L	Word length (in bits)	8
x, X	Error-free binary value	`00101010`
$\hat{x}$, $\hat{X}$	Binary value after channel error	`00100000`
k	Number of perturbed bits, Hamming distance $d(x, \hat{x})$	2 (grayed-out above)
u	Conversion from binary to unsigned integer	u(`00101010`)$= 42$
v, V	Error-free integer value, $u(x)$	u(`00101010`)$= 42$
w, W	Integer value after channel error, $u(\hat{x})$	u(`00100000`)$= 32$
m, M	Integer value distortion, $d_{\mathbb{Z}^+}(x, \hat{x}) := \lvert v - w \rvert$	10
ε	Signed error vector, $x - \hat{x}$	$(0, 0, 0, 0, -1, 0, -1, 0)$
$\mathbb{E}_L$	Set of all possible ε in words of length L	
$\mathbb{E}_L^m$	Set of all possible ε in words of length L with integer value distortion m	
$\mathbb{W}_L$	Set of all possible x of length L	
$\mathbb{W}_{L,\varepsilon}$	Set of all possible $x \in \mathbb{W}_L$ that permit a given error vector ε	
$f_X(x)$	PMF of transmitted words x	
$f_V(v)$	PMF of transmitted integer values, $\Pr(V = v)$	See Figure 5
$\mathrm{F}_V(v)$	CDF of transmitted integer values, $\Pr(V \leq v)$	
$f_M(m)$	PMF of integer value distortion, $\Pr(M = m)$	See Figure 2(a)
$\mathrm{F}_M(m)$	CDF of integer value distortion, $\Pr(M \leq m)$	See Figure 2(b)
$\widehat{\mathrm{T}}_M(m)$	Constraint tail distribution	See Figure 3
$\mathrm{T}_M(m)$	Tail distribution, $\Pr(M > m)$	$\begin{cases} 1 & m < 4 \\ 1/100 & m = 4 \\ 0 & m \geq 10 \end{cases}$

We define the random variable $V := u(X)$ with instance values $v = u(x)$ and $W := u(\hat{X})$ with values $w = u(\hat{x})$. Then, we define the distance $d_{\mathbb{Z}^+}(x, \hat{x})$ between channel input and output as

$$d_{\mathbb{Z}^+}(x, \hat{x}) = \lvert u(x) - u(\hat{x}) \rvert = \lvert v - w \rvert .$$

The random variable M denoting values of channel distortion, with instance values m, takes on values determined by the distance function $d_{\mathbb{Z}^+}(x, \hat{x})$, i.e., $m = d_{\mathbb{Z}^+}(x, \hat{x})$. We will use the variable m when we wish to refer to specific instances of channel integer distortion and we will use $d_{\mathbb{Z}^+}(x, \hat{x})$ when we want to emphasize the distortion function.

For each of the random variables defined, we respectively denote its probability mass function (PMF), cumulative distribution, and tail distribution by f, F, and T with the variable as subscript, e.g., f_M, F_M, and T_M. In the case of variable M, for L-bit words transmitted over the channel, the domain of T_M is $\{0, \ldots, 2^L - 1\}$ and we have

$$\mathrm{T}_M(m) = 1 - \mathrm{F}_M(m) = \sum_{i>m} f_M(i), \quad \forall m \in \{0, \ldots, 2^L - 1\}.$$

Finally, we define the *constraint tail distribution* $\widehat{\mathrm{T}}_M$ as the distribution of integer distances $d_{\mathbb{Z}^+}(x, \hat{x})$ which the system consuming the output of the communication channel can tolerate. $\widehat{\mathrm{T}}_M$ has the same domain as T_M and it may be any non-increasing function with range $[0, 1]$.

4 Probabilistic VDB Source-Dependent Bit-Level Channel Adaptation

We consider a communication channel that transmits integer data represented by L-bit binary words x. Suppose we have a tunable method at our disposal to induce errors in this communication channel that lets one vary the error probabilities of each bit in a controllable manner. We assume that the method of inducing errors on the channel comes with a benefit (e.g., a reduction in power dissipation) and that we have a mapping, $\mathrm{B} : \mathbb{R}^{2L} \to \mathbb{R}$, that quantifies the benefit for a given set of bit-level error probabilities.

Given the knowledge of bit-error probabilities and the distribution of input words x to the channel, one can calculate f_M, the PMF for the distribution of induced distortion. This is equivalent to finding the tail distribution T_M of the induced distortion.

Given the foregoing conditions, probabilistic VDB source-dependent bit-level channel adaptation consists of finding the correct set of parameters for the mapping B, i.e., the bit-error probabilities, such that: ❶ $\mathrm{T}_M(m) \leq \widehat{\mathrm{T}}_M(m)$ for each $m \in \{0, \ldots, 2^L-1\}$ and ❷ the benefit B is maximized. We first present our method for calculating f_M and then present our search method for finding bit-error probabilities that satisfy conditions ❶ and ❷.

4.1 Calculating f_M

Recall, from Table 1, that f_M is the PMF of integer value distortions that occur during transmission of L-bit words. One can calculate it exactly provided that one knows the bit-error probabilities suffered during a transmission and the PMF f_X of input words x. Given these, to calculate $f_M(m)$ for each $m \in \{0, \ldots, 2^L-1\}$, we first characterize all possible word-level error events resulting in a distortion value of m. Then, for each possible input x, we determine the probability of each event resulting in m-distortion given x is the input. The sum of all these event probabilities is equal to $f_M(m)$.

Characterization of all error events with distortion m: We can identify all possible word-level error events in L-bit words with the set $\mathbb{E}_L$ of all error vectors ε of length L with entries from the set of error polarities $\{-1, 0, 1\}$, i.e.,

$$\mathbb{E}_L := \{\varepsilon \,:\, \varepsilon(i) \in \{-1, 0, 1\}, \quad 0 \leq i \leq L-1\},$$

where $\varepsilon(i)$ denotes the i'th entry of the error vector $\varepsilon := x - \hat{x}$. If a word-level error event involves a bit-level error of `0` $\to$`1`, `1` $\to$`0`, or no error at the i'th bit, then the corresponding error vector ε has $\varepsilon(i) = 1$, $\varepsilon(i) = -1$, or $\varepsilon(i) = 0$, respectively. Let the integer value distortion corresponding to a given error vector ε be

$$\mathrm{m}(\varepsilon) = \left| \sum_{i=0}^{L-1} \varepsilon(i) 2^L \right|, \quad \forall \varepsilon \in \mathbb{E}_L.$$

Given input and output words of the channel $x, \hat{x} \in \mathbb{Z}_2^L$ and channel error $\varepsilon = x - \hat{x}$, we have $\mathrm{m}(\varepsilon) = d_{\mathbb{Z}^+}(x, \hat{x})$.

Let $\mathbb{E}_L^m$ be the set of all possible error vectors causing m-distortion, that is

$$\mathbb{E}_L^m := \{\varepsilon \in \mathbb{E}_L \,:\, \mathrm{m}(\varepsilon) = m\}.$$

The set $\mathbb{E}_L^m$ characterizes all possible word-level error events with distortion m. Computing f_M requires computing the sets $\mathbb{E}_L^m$ for each $m \in \{0, \ldots, 2^L-1\}$.

The computation of the sets $\mathbb{E}_L^m$ makes use of the fact that a word-level error event is a disjoint union of its constituent bit-level error events (even if distinct bit-level events may depend on each other). An error at the i'th bit introduces a signed distortion of 2^i, -2^i or 0 depending on its polarity. The distortion resulting from the word-level error event is the absolute value of the sum of the signed distortions resulting from all constituent bit-level errors. In analogy to the fact that the distribution of the sum of two random variables is the convolution of their distributions, one can construct the sets $\mathbb{E}_L^m$ all at once with an $(L-1)$-step generalized convolution of set-valued functions.

Let α and β be two set-valued functions on $\mathbb{Z}$ such that for each $n \in \mathbb{Z}$ the sets $\alpha(n)$ and $\beta(n)$ contain elements from the same group, with group operation "+". The set-valued convolution $f \diamond g$ between two set-valued functions is again a set-valued function defined by

$$\alpha \diamond \beta(n) := \sum_{k+l=n}^{\oplus} \alpha(k) \star \beta(l), \tag{1}$$

where $\sum^{\oplus}$ and $\star$ are the summation and multiplication of sets, respectively. Given two sets

$$\mathbb{A} = \bigcup_{i=1,\ldots,n} \{a_i\} \quad \text{and} \quad \mathbb{B} = \bigcup_{j=1,\ldots,m} \{b_j\}$$

we define the sets $\mathbb{A} \oplus \mathbb{B}$ and $\mathbb{A} \star \mathbb{B}$ as

$$\mathbb{A} \oplus \mathbb{B} := \mathbb{A} \dot{\cup} \mathbb{B},$$

and

$$\mathbb{A} \star \mathbb{B} := \dot{\bigcup_{\substack{i=1,\ldots,n \\ j=1,\ldots,m}}} \{a_i + b_j\},$$

respectively, where $\dot{\cup}$ stand for disjoint union. From the observation that

$$|\mathbb{A} \oplus \mathbb{B}| = |\mathbb{A}| + |\mathbb{B}| \text{ and } |\mathbb{A} \star \mathbb{B}| = |\mathbb{A}| \times |\mathbb{B}|, \tag{2}$$

the set-valued convolution defined by (1) reduces to the standard convolution if we only consider the sizes of the sets. For a given set-valued function f, if we define $|f| : \mathbb{Z} \to \mathbb{N}$ by

$$|f|(m) := |f(m)|, \tag{3}$$

then

$$|f \diamond g| = |f| * |g|. \tag{4}$$

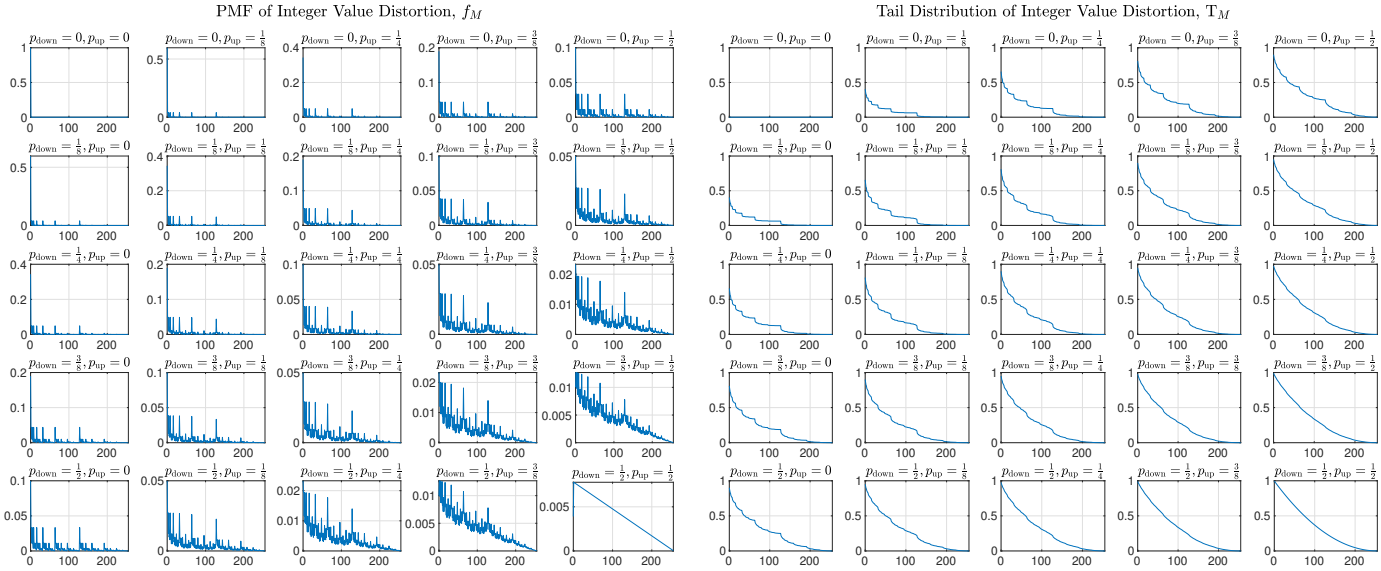
Coming back to the calculation of the sets $\mathbb{E}_L^m$, let the unit positive error vectors $\hat{\varepsilon}_i$ for each $i \in \{0, \ldots, L-1\}$ be

$$\hat{\varepsilon}_i(j) := \begin{cases} 1 & \text{, when } j = i, \\ 0 & \text{, otherwise,} \end{cases} \quad 0 \leq j \leq L-1,$$

and consider the vector-set-valued functions χ_i defined on $\{-2^L-1, \ldots, 0, \ldots, 2^L-1\}$ by

$$\chi_i(n) = \begin{cases} \{-\hat{\varepsilon}_i\}, & \text{when } n = -2^i, \\ \{\vec{0}_L\}, & \text{when } n = 0, \\ \{\hat{\varepsilon}_i\}, & \text{when } n = 2^i, \\ \emptyset, & \text{otherwise,} \end{cases} \quad 0 \leq i \leq L-1,$$

where $\emptyset$ is the empty set and $\vec{0}_L$ is the zero error vector. For a given $i \in \{0, \ldots, L-1\}$, χ_i encodes the possible error events at the i'th bit only according to their polarities by storing the corresponding error vector in the set labeled by the signed-integer distortion the error causes. Then, the



(a) PMF of integer value distortion, f_M. (b) Tail distribution of integer value distortion, T_M.

Fig. 2. (a) f_M and (b) T_M induced by bit-position-independent errors with different probabilities p_{down} and p_{up} for the case of 8-bit words and when the input is uniformly distributed. As expected, T_M rises as bit-error probabilities increase.

set-valued function, defined by the $(L-1)$-step set-valued convolution

$$\mathbb{C}_L := \chi_1 \diamond \cdots \diamond \chi_L, \tag{5}$$

encodes all possible word-level error events by storing the corresponding error vector in the set labeled by its signed-integer distortion. Consequently, we have

$$\mathbb{E}_L^m = \mathbb{C}_L(m) \cup \mathbb{C}_L(-m).$$

Determining event probabilities and finding f_M: Once we have calculated $\mathbb{E}_L^m$ we can calculate f_M provided that we know the bit-level error probabilities of the channel and the occurrence probabilities of the input words. For a given input word x, let $p^x_{i,\mathrm{down}}$ and $p^x_{i,\mathrm{up}}$ denote the probabilities of $\mathtt{1} \to \mathtt{0}$ and $\mathtt{0} \to \mathtt{1}$ erroneous bit transitions for the i'th bit x_i given it has the correct initial value, i.e., given $x_i = 1$ for negative polarity error and given $x_i = 0$ for positive polarity error. A given error vector ε can occur only if the input word x is in the set

$$\mathbb{W}_{L,\varepsilon} := \left\{ x \in \mathbb{W}_L : \begin{cases} x_i = 1, & \text{if } \varepsilon(i) = -1, \\ x_i = 0, & \text{if } \varepsilon(i) = 1, \end{cases} 0 \le i \le L-1 \right\}.$$

Hence, the occurrence probability P_ε of an error vector ε is

$$P_\varepsilon = \sum_{x \in \mathbb{W}_{L,\varepsilon}} \Pr(x - \hat{x} = \varepsilon | X = x) f_X(x), \tag{6}$$

where the probability that ε occurs given $X = x$ is

$$\Pr(x - \hat{x} = \varepsilon | X = x) = \prod_{i=1}^{L} p_i^x, \tag{7}$$

with

$$p_i^x = \begin{cases} p^x_{i,\mathrm{down}}, & \text{if } \varepsilon(i) = -1, \\ p^x_{i,\mathrm{up}}, & \text{if } \varepsilon(i) = 1, \\ 1 - p^x_{i,\mathrm{down}}, & \text{if } \varepsilon(i) = 0 \text{ and } x_i = 1, \\ 1 - p^x_{i,\mathrm{up}}, & \text{if } \varepsilon(i) = 0 \text{ and } x_i = 0, \end{cases} \tag{8}$$

for $0 \le i \le L-1$.

As $\mathbb{E}_L^m$ contains all word-level error events ε that cause integer value distortion of m, summing the error occurrence probabilities P_ε over this set gives us the probability that the integer value distortion is m. That is, we have

$$f_M(m) = \sum_{\varepsilon \in \mathbb{E}_L^m} P_\varepsilon.$$

Using the PMF f_M we calculate the tail distribution T_M as

$$\mathrm{T}_M(m) = 1 - \sum_{i=0}^{m} f_M(i).$$

In the special case of maximal entropy input distribution we have

$$f_X(x) = 2^{-L}, \quad \forall x \in \mathbb{W}_L.$$

Assuming the individual bit-error probabilities are independent of the transmitted word x (which is not the case for the I2C channel for instance, see Section 5), in view of the relations (7) and (8) and the structure of the set $\mathbb{W}_{L,w}$, the occurrence probability of ε given by (6) reduces to the product

$$\begin{aligned} P_\varepsilon &= 2^{-L} \sum_{x \in \mathbb{W}_{L,\varepsilon}} \Pr(x - \hat{x} = \varepsilon | X = x), \\ &= 2^{-L} \prod_{\varepsilon(i)=-1} p_{i,\mathrm{down}} \prod_{\varepsilon(i)=1} p_{i,\mathrm{up}} \prod_{\varepsilon(i)=0} (2 - p_{i,\mathrm{down}} - p_{i,\mathrm{up}}). \end{aligned}$$

Figure 2 shows the results of the calculations carried out in the case of 8-bit words with maximal entropy input when the error probabilities are independent of the bit position, i.e., $p_{i,\mathrm{down/up}} = p_{\mathrm{down/up}}$ for all $1 \le i \le 8$. The plots sweep p_{down} and p_{up} in the range of $[0, 1/2]$ with resolution $\frac{1}{8}$. Figure 2 (a) and (b) respectively depict the dependencies of the PMF and the tail distribution on p_{down} and p_{up}. All plots illustrate the expected channel behavior of elevated integer value distortion with increasing bit-error probabilities.

4.2 Searching for Optimal Bit-Error Probabilities

In practice, an implemented channel adaptation technique will only be able to modulate the channel to finitely many

distinct states. This will yield a finite number of possible collections of bit-error probabilities for all possible words $\{\mathbf{p}^x\}_{x\in\mathbb{W}_L}$, where we define

$\mathbf{p}^x := (\mathbf{p}^x_{\text{down}}, \mathbf{p}^x_{\text{up}}) := (p^x_{1,\text{down}}, \dots p^x_{L,\text{down}}, p^x_{1,\text{up}}, \dots p^x_{L,\text{up}},)$.

For each such state, using the corresponding collection of bit-error probabilities, we can calculate the induced tail distribution $\mathrm{T}_M(m)$ as Section 4.1 describes. Given a benefit functional $\mathrm{B} : \mathbb{R}^{2L} \to \mathbb{R}^+$ we would then select after an exhaustive search the state that yields the maximal average benefit

$$\widetilde{\mathrm{B}} := \sum_{x\in\mathbb{W}_L} f_X(x)\mathrm{B}(\mathbf{p}^x), \tag{9}$$

while also satisfying the constraint $\mathrm{T}_M(m) \le \widehat{\mathrm{T}}_M(m)$ for all $m \in \{0, \dots, 2^L-1\}$.

We demonstrate the results for the case when the word length is $L = 8$ and assume that the input words to the channel are uniformly distributed. We also assume that the bit-error probabilities are independent of the transmitted word, i.e., $p^x_{i,\text{down/up}} = p_{i,\text{down/up}}$ for all $x \in \mathbb{W}_L$. In this case, the average benefit defined by (9) reduces to $\widetilde{\mathrm{B}} = \mathrm{B}(\mathbf{p})$.

We consider two sub-scenarios, first when the bit errors are independent of the bit positions, i.e., $p_{i,\text{down/up}} = p_{\text{down/up}}$ and second when they depend on the bit positions. We define the benefit functional as

$$\mathrm{B}(\mathbf{p}) := \|\mathbf{p}\|^2 = \sum_{i=1,\dots,L} (p^2_{i,\text{down}} + p^2_{i,\text{up}}). \tag{10}$$

B is an increasing function of its argument. Hence, this choice of B obeys the principle that the cost (e.g, error suffered) directly correlates with benefit. **Searching with bit errors independent of the bit positions:** In the first scenario we carry out an exhaustive search through the space of possible bit-error probabilities $(p_{\text{down}}, p_{\text{up}}) \in [0, \frac{1}{2}]^2$ with a probability resolution of $p_{\text{res}} = 2^{-7}$. **Searching with bit errors dependent on the bit positions:** In this scenario, the space to search through ($\mathbf{p} \in [0, \frac{1}{2}]^{2L}$) is too large to carry out an exhaustive search with a fine probability resolution (e.g., $p_{\text{res}} = 2^{-7}$). Thus, we employ a search algorithm with adaptive probability resolution. Specifically, in the first step, we do an exhaustive search with the initial probability resolution of $p^{(1)}_{\text{res}} = 2^{-2}$ to find an optimal probability vector $\mathbf{p}^{(1)}_{\text{opt}}$, that is, one that maximizes the average benefit $\widetilde{\mathrm{B}}$. The probability vectors we search through have the form

$$\mathbf{p}^{(1)} = \mathbf{0}_{2L} + \left((p^{(1)}_{\text{res}})^{\delta_1}, \dots, (p^{(1)}_{\text{res}})^{\delta_{2L}}\right),$$

where $\mathbf{0}_{2L} \in \mathbb{R}^{2L}$ is the 0-vector and $\delta_i = 0$ or $\delta_i = 1$ for $1 \le i \le 2L$, so that the search involves 2^{2L} probability vectors. In the subsequent steps, we continually halve the probability resolution until we reach to the probability resolution of 2^{-7} (i.e., $p^{(n)}_{\text{res}} = 2^{-(n+1)}$, $1 \le n \le 6$) and in the n'th step, to find $\mathbf{p}^{(n)}_{\text{opt}}$ that maximizes $\widetilde{\mathrm{B}}$, we search through the probability vectors of the form

$$\mathbf{p}^{(n)} = \mathbf{p}^{(n-1)}_{\text{opt}} + \left((p^{(n-1)}_{\text{res}})^{\delta_1}, \dots, (p^{(n-1)}_{\text{res}})^{\delta_{2L}}\right), \; 2 \le n \le 6.$$

Evaluation of search methods: To generate an example constraint tail distribution $\widehat{\mathrm{T}}_M$, we generate a random probability vector $\mathbf{p}_{\text{rand}} \in \mathbb{R}^L$ and consider the scenario when only the $\mathbf{0}_L \in \mathbb{R}^L$ is transmitted over the communication channel. In this scenario the only possible bit-errors are `0` $\to$ `1` errors and we assume that $\mathbf{p}_{\text{rand}}$ describes the probabilities of these errors, i.e., we take $\mathbf{p}_{\text{down}} := \mathbf{0}_L$ and $\mathbf{p}_{\text{up}} := \mathbf{p}_{\text{rand}}$. We set the example constraint tail distribution as the distribution of the integer value distortion suffered in this scenario. Then, for $0 \le m \le 2^L-1$, the PMF of the suffered integer value distortion is

$$f_M(m) = \prod_{i=1}^{L} \mathbf{p}_{\text{rand}}(i)^{u^{-1}(m)_{i-1}} \left(1 - \mathbf{p}_{\text{rand}}(i)\right)^{\left(1-u^{-1}(m)_{i-1}\right)},$$

where u is defined as in Section 3 (see Table 1) and $u^{-1}(m)_i$ corresponds to the value of the i'th bit of the binary representation of m. The example constraint tail distribution is given by

$$\widehat{\mathrm{T}}_M(m) = 1 - \sum_{0\le i\le m} f_M(i).$$

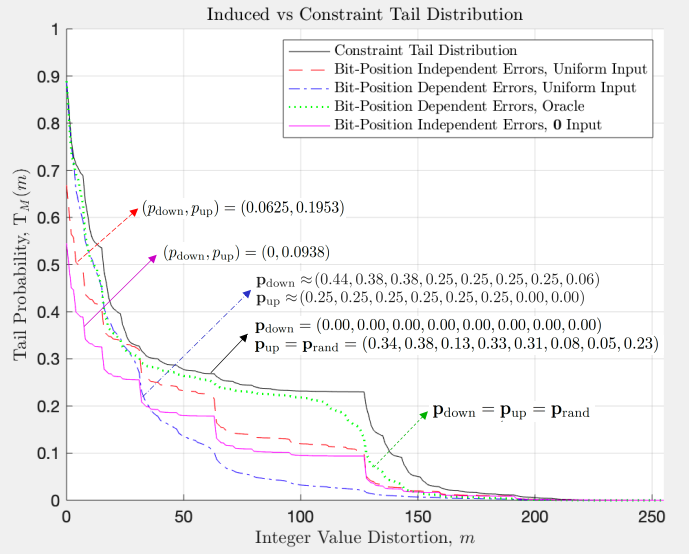


Fig. 3. Tail distributions of induced integer value distortion corresponding to the bit-error probabilities with maximum average benefit $\widetilde{\mathrm{B}}$ found by the search algorithms in the cases when bit-error manipulation is same for all bit positions with uniform input (red dashed line), distinct at the bit-level with uniform input (blue dash-dot line), in the case of the oracle with uniform input (green dotted line) and when bit-error manipulation is same for all bit positions with zero input. All induced tail distributions are bounded by the constraint tail distribution $\widehat{\mathrm{T}}_M(m)$ (black solid line). In the case of the uniform input, bit-level error manipulation achieves more than $3\times$ higher benefit than the bit-independent error manipulation and higher than even what the oracle can achieve.

Figure 3 shows for the case when the input is uniformly distributed the constraint tail distribution together with the induced tail distributions of integer value distortion corresponding to the benefit-maximizing bit-error probabilities encountered in our search algorithms. Figure 3 also plots the tail distribution of induced integer value distortion when the bit-error probabilities satisfy $\mathbf{p}_{\text{down}} = \mathbf{p}_{\text{up}} = \mathbf{p}_{\text{rand}}$. We refer to this as the *oracle tail distribution*, as it requires the knowledge of the random vector $\mathbf{p}_{\text{rand}}$, which the search algorithms do not possess. The oracle tail distribution is bounded by the constraint tail distribution, because in the case of error induction with same bit-error probabilities (for both $\mathbf{p}_{\text{down}}$ and $\mathbf{p}_{\text{up}}$), when input is uniformly distributed as in the case of the constraint tail distribution scenario (where only $\mathbf{0}_L$ is transmitted and only `0` $\to$ `1` bit-errors can occur), the induced integer value errors by distinct bit-errors may have different signs and do not compound as in the oracle

scenario where distinct bit-errors always have the same sign. Finally, Figure 3 also plots the tail distribution of integer value distortion of the exhaustive search for bit position independent errors for the case when the input is always $\mathbf{0}_L$.

In the case of uniform input distribution, the results reveal that the separate control of error rates in individual bit positions allows one to induce integer value distortion on the channel with a larger corresponding average benefit $\widetilde{\mathrm{B}}$ than achievable by bit-position-independent error induction, as well as than that the oracle tail distribution achieves. Even though the improvement due to this capability is undetectable when induced tail distributions are visually compared, in terms of average benefit maximization the adaptive bit-dependent search algorithm achieves the largest average benefit with $\widetilde{\mathrm{B}} \approx 1.102$ in arbitrary units compared to $\widetilde{\mathrm{B}} \approx 0.336$ of the exhaustive bit-position independent search (more than $3\times$ improvement) and $\widetilde{\mathrm{B}} \approx 1.087$ that the oracle tail distribution achieves. On the other hand, when only $\mathbf{0}_L$ is transmitted over the channel the bit-position-independent error induction can only achieve an average benefit of $\widetilde{\mathrm{B}} \approx 0.07$ compared to $\widetilde{\mathrm{B}} \approx 0.543$ achievable with bit position dependent error manipulation, which is the average benefit corresponding to the probabilities of the constraint tail distribution. In general, independent of the shape of the input distribution and whether the bit errors depend on the words transmitted, it should be expected that bit-level error manipulation will achieve better performance in terms of achievable average benefit, because the set of bit-error probabilities admitted in the case of bit-position-independent error induction is a subset of the set of bit-error probabilities admitted in the case of bit-position-dependent error induction.

Finally, in comparison to the approximate method of prior work [8] for calculating the distribution of integer value distortion for given bit-error probabilities, the exact calculation method provided in this paper is more computationally efficient. This fact reflects itself in the computation time the respective search algorithms take for finding the benefit-maximizing bit-error probabilities inducing a tail distribution of integer value distortion bounded by a given constraint tail distribution. For the special case of bit-position-independent bit-error probabilities, while the method of prior work takes ~ 21 minutes, the method we describe in this section takes only ~ 50 seconds on the same workstation.

5 Case Study: Channel Adaptation on the I2C Serial Bus

The Inter-Integrated Circuit serial bus (I2C) is a serial communication bus widely used in embedded sensor interfaces and consists of two signals: a serial clock (SCL) and a serial data line (SDA). Figure 4(a) shows how a processor (the master) can communicate with multiple sensors (the slaves) over I2C. Data transfers in I2C occur in multiples of a byte. Drivers on the I2C bus are open-drain and the I2C standard requires systems implementing it to provide pull-up resistors on both SDA and SCL. To implement the channel adaptation method of Section 4, we implement these pull-up resistors with two digitally-controlled potentiometers (DCPs). Because of the pull-up resistors, when the I2C bus is idle, both SDA and SCL voltages are close to the I/O supply voltage. To pull SDA or SCL low, either the master or the slave sinks the current through its drive transistor to ground.

5.1 I2C Electrical Parameters

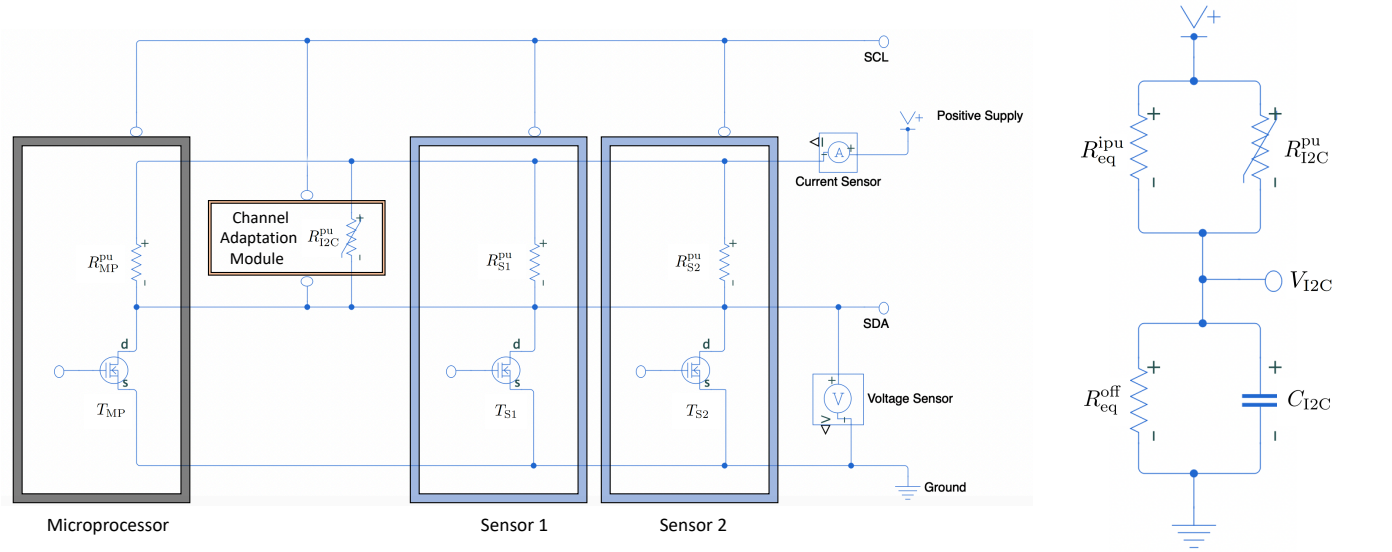
Typically, designers choose a significantly larger value for the pull-up resistance $R_{\mathrm{I2C}}^{\mathrm{pu}}$ compared to the on-resistance R_T^{on} of the transistor in order to achieve logic-`0` voltage close to zero. Furthermore, for the bus to be able to charge/discharge to these voltage levels within an I2C clock cycle t_{I2C} one requires the charging/discharging time constants of the I2C data bus τ_{up} and τ_{down} to be comparably small. As transistors with both very small on- and very large off-resistances are available, choosing $R_{\mathrm{I2C}}^{\mathrm{pu}}$ small enough to be able to charge the bus in time is the main constraint determining its value from the perspective of reliable transmission. On the other hand, the main source of power dissipation in I2C data transmission is due to current flow through $R_{\mathrm{I2C}}^{\mathrm{pu}}$ during a logic `0` transmission. Consequently, during transmission, using values of $R_{\mathrm{I2C}}^{\mathrm{pu}}$ higher than those used for reliable operation will reduce reliability, but it will also reduce power dissipation.

Reduction in transmission reliability corresponds to an increase in bit-error probabilities, whereas reduction in power dissipation corresponds to an increase in power savings. Accordingly, the benefit functional B defined in Section 4, which takes the bit-error probabilities as input and whose value translates to power savings within the I2C channel adaptation framework, should be an increasing function of its arguments. The definition (10) in Section 4.2 obeys this principle.

5.2 Implementing Bit-Level Channel Adaptation for I2C

We implement source-dependent bit-level channel adaptation for the I2C channel, establishing a tradeoff between transmission reliability and power dissipation by manipulating the value of the pull-up resistance $R_{\mathrm{I2C}}^{\mathrm{pu}}$ during each bit transmission of an I2C transaction. Here, we focus on the analog electrical behavior of the bus for different per-bit values of the pull-up resistance achieved by the combination of digital modulator logic and the variable resistor. We present the digital logic for controlling the DCP in Section 6.

The channel adaptation module in Figure 4(a) takes as input the I2C SCL and SDA lines and modulates the value of the pull-up resistor on a per-bit-clock basis. For each sensor S on the I2C channel that transmits data in chunks of L_S bits, the channel adaptation module contains offline-calculated pull-up resistance selections corresponding to resistances $\{R_{\mathrm{I2C}}^{\mathrm{pu},i}\}_{i=1}^{L_S}$, which will lead to associated bit-error probabilities that are permissible for that bit position (calculated in Section 4). The channel adaptation module synchronizes with the I2C transaction and switches the pull-up resistance to $R_{\mathrm{I2C}}^{\mathrm{pu},i}$ at the beginning of transmission of the i'th bit of each data chunk of size L_S, $1 \leq i \leq L_S$. The switching of the pull-up resistances on a per-bit-clock basis requires the channel adaptation module to switch the DCP at a frequency much higher than the clock frequency of the I2C transaction, which may go up to $5\,\mathrm{MHz}$ (referred to as



(a) Schematic of an I2C bus with one microprocessor as master and two sensors as slaves. The schematic includes the current and voltage meters we use in our experiments of Section 5.4.

(b) The equivalent circuit around the I2C data bus during reception event.

Fig. 4. An example I2C system with exposed internal circuitries of agents around the I2C data bus SDA and the experimental setup we used.

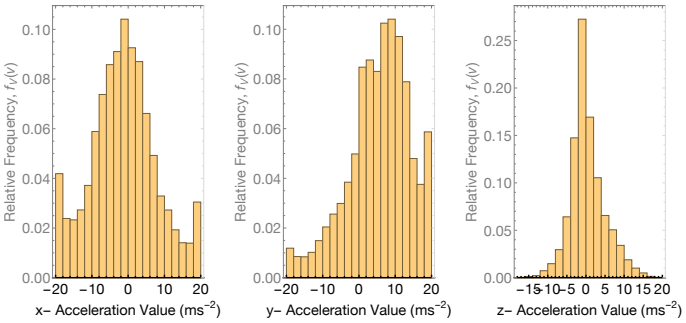


Fig. 5. Empirical PMFs (i.e., $f_V(v)$, see Table 1) for 29 978 samples (25 minutes of sensor data sampled at 20 Hz) from the three axes of an accelerometer worn by a person engaged in a variety of activities [18].

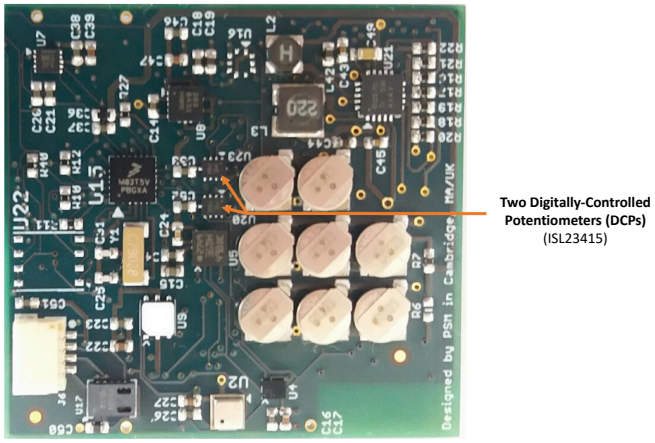


Fig. 6. The Warp embedded hardware-research platform contains two DCPs configurable to serve as pull-up resistors to its I2C channel.

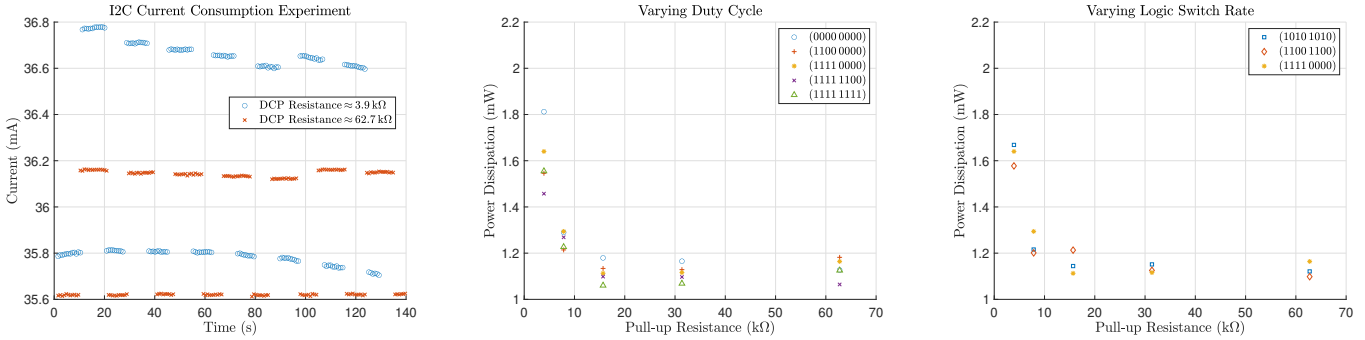
the ultra-fast mode). On the other hand, the CMOS switches on a typical application-specific integrated circuit with a 90 nm process can switch their output in 120 ps and can do so every 10 ns, which permits bit-level channel adaptation even for ultra-fast mode I2C transactions with the pull-up resistance switching occupying less than 0.1% of an I2C clock cycle.

The user input of our method is the expected sensor data distribution f_V and an upper bound tail distribution $\widehat{\mathrm{T}}_M$ on the permitted integer value distortion. Figure 5 shows an example of real-world distributions of the error-free integer values $u(x)$ (see Section 3). Using these inputs in an offline step performed at system design time, we derive a specification of L_S pull-up resistance selections. The resulting communication on the I2C bit-level-adapted channel which switches pull-up resistance according to this specification yields a tail distribution of integer value distortion T_M globally bounded by $\widehat{\mathrm{T}}_M$. The resulting approximate channel consumes less power than the unadapted channel.

To be able to find optimal pull-up resistance values for I2C channel adaptation given the input distribution and user-defined bound on distortion distribution and to estimate the resulting power savings, we need to characterize the dependence of power dissipation and induced distortion distribution on pull-up resistance value. We measure the power dissipation dependence experimentally. This also lets us estimate the parameters of the equivalent circuit around the I2C data bus during reception of logic 1 provided in Figure 4 (b). For the dependence of integer value distortion incurred during transmission we present an analytic model in Section 5.6.

5.3 Setup for Measuring the I2C Power Dissipation Dependence on Pull-up Resistance

To determine the dependence of I2C bus power dissipation on the value of the pull-up resistor, we ran experiments on the I2C communication interface in Warp [19], an open-



(a) Raw current data for all the bytes transmitted for the experiment at two extreme values of the pull-up resistance.

(b) Average I2C power consumption as a function of pull-up resistance for transmitted bytes with varying duty cycles.

(c) Average I2C power consumption as a function of pull-up resistance for transmitted bytes with varying logic switch frequency.

Fig. 7. Power measurement experiment results show clear inverse dependence of I2C power consumption on the I2C pull-up resistance value.

source hardware platform for research into tradeoffs between accuracy and power dissipation (see Figure 6).

We use two ISL23415 DCPs [20] as configurable pull-up resistors on the SDA and SCL lines of the I2C bus. The ISL23415 is an integrated circuit that occupies an area of less than $2\,\mathrm{mm}^2$. It dissipates less than $0.5\,\mu\mathrm{W}$ of power over its CMOS switches when switching between different resistance settings. Because we cannot control the ISL23415 DCPs to switch resistance values between I2C clock cycles, we use them primarily to demonstrate the feasibility and potential power savings for the case where we use the same pull-up resistance value for all clock cycles. A custom integrated circuit combining the digital interface from Section 6 with the internals of a device such as the ISL23415 would be primarily an engineering task and would not provide much additional research insight.

We measured the dependence of I2C bus power dissipation on the pull-up resistance at $f_{\mathrm{I2C}} = 200\,\mathrm{kHz}$. We selected $3.92\,\mathrm{k\Omega}$, $7.84\,\mathrm{k\Omega}$, $15.69\,\mathrm{k\Omega}$, $31.37\,\mathrm{k\Omega}$, and $62.75\,\mathrm{k\Omega}$ as pull-up resistance values for investigation. The lowest value is in the range of typical external I2C pull-up resistance values.

We used an MMA8451Q digital accelerometer [21] as the target sensor in our measurements. To control the bytes transmitted by the sensor over the I2C bus, we write a prescribed value to a sensor configuration register and request the sensor to send that value back $60\,000$ times. We used two sets of bytes to characterize the power dissipation on the I2C bus. The first set (**`0000 0000`**, **`1100 0000`**, **`1111 0000`**, **`1111 1100`**, **`1111 1111`**) has bytes with varying duty cycles of logic `0` and `1` and have fixed logic switch frequency. The second set of bytes (**`1010 1010`**, **`1100 1100`**, **`1111 0000`**) all have equal duty cycles for both logic values but have varying number of transitions between logic states.

We use a Keithley SMU2450 6.5-digit laboratory source-measure unit with a current measurement resolution of $10\,\mathrm{nA}$ to power the Warp and determine the power consumed by I2C communication by measuring the current drawn by the power supply. Figure 7(a) shows raw current measurement data at the lowest and highest DCP settings. Each data point corresponds to a current measurement. The first five high current plateaus correspond (in order) to transmissions of the five bytes in the first set given above and the last two correspond (in order) to the first two bytes in the second set.

From the raw current data we estimate average current consumption during repeated I2C transmissions of a given byte by taking the difference between the averages of data points on the plateau and the averages of points around it (four on each side). We then find the average power consumption by multiplying the average current consumed by the positive supply voltage, which we set to $2.4\,\mathrm{V}$ in the experiment.

5.4 Measurements Results

Figure 7(b) and (c) show the dependence of I2C power consumption on the I2C pull-up resistance value for transmitted bytes with varying duty cycles (and fixed logic switch frequency) and with varying logic switch frequency (and fixed duty cycle), respectively. The results show a clear inverse dependence of the I2C power dissipation on the pull-up resistance value with an absolute value of power savings of up to $0.8\,\mathrm{mW}$. The power dissipation, which can be as high as $2\,\mathrm{mW}$ for the lowest used resistance value of $3.92\,\mathrm{k\Omega}$ (typically used in practice), reduces below $1.2\,\mathrm{mW}$ and settles to a plateau above $1\,\mathrm{mW}$, implying up to $2\times$ power reduction. The reason for this plateau is the existence of the internal pull-up pathways embedded into sensors and/or processors as Figure 4 shows. Furthermore, the results confirm the expected behavior of increasing power consumption with increasing duty cycle of logic `0` and reveal the increasing power consumption with increasing logic switch frequency.

Because our experimental procedure involved a repetitive register read request of a fixed byte from a sensor register, the actual reading process was only a portion of the total I2C transaction. As a result, the observed differences in current measurements only reflect changes in power dissipation due to the transmission of a prefixed byte during that portion of the whole process. Consequently, observed power consumption dependencies in Figure 7(b) and (c) are less than what would be in a real-word usage scenario, where a sensor transmits streams of uninterrupted data.

5.5 Estimation of I2C High-Logic-Reception Channel Model Parameters

Both the analysis of I2C circuitry (shown in Figure 4(a)) and the experimental results indicate that one can decrease the

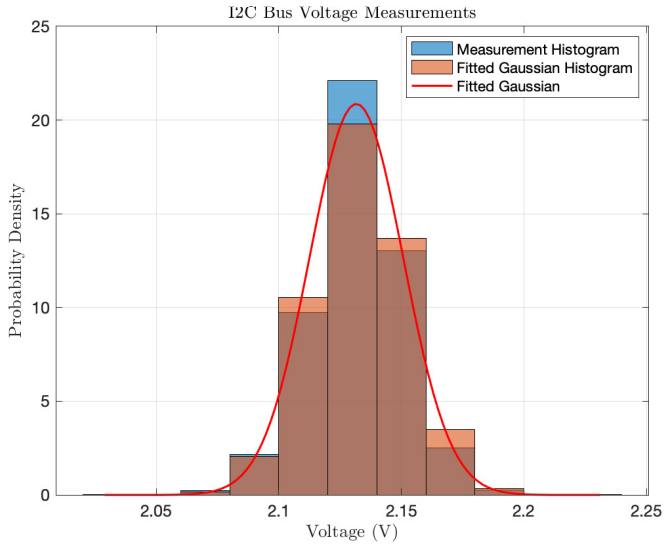


Fig. 8. Density histogram (the area under the histogram integrates to unity) of voltage measurements of the I2C bus with the minimum DCP pull-up resistance configuration (44 235 samples), the fitted Gaussian distribution with same mean and standard deviation as the data and the density histogram of samples drawn from the fitted Gaussian with same resolution as the measurement data. Two-sample Cramér-von Mises test rejects the null hypothesis that the measurement data is distributed as the samples from the fitted Gaussian at the 1% level.

power consumption of I2C communication by increasing the pull-up resistance of the I2C data bus. Similarly, it follows that an increase in pull-up resistance will only have detrimental effects on the transmission fidelity of high logic values. Specifically, it results in a decrease of the steady-state logic `1` I2C voltage level (bringing it closer to threshold) and in an increase of `0`-to-`1` transition time (risking reaching to steady-state level in an I2C clock cycle). In stark contrast, an increase in pull-up resistance causes a decrease of the steady-state logic `0` I2C voltage level (taking it further away from threshold) and a decrease of `1`-to-`0` transition time. Based on this observation, which we also verified with measurements provided below, the potential of our proposed method lies in the analysis of logic `1` transmission.

We evaluated the two extreme configurations of the pull-up DCP by measuring the steady-state logic `0/1` voltage levels $V_{\mathrm{I2C}}^{0/1,\min}$ and $V_{\mathrm{I2C}}^{0/1,\max}$, as well as the rise/fall times (10%–90%) of the I2C voltage, for the minimum ($R_{\mathrm{I2C}}^{\mathrm{pu},min} \approx 3.9\,\mathrm{k\Omega}$) and maximum ($R_{\mathrm{I2C}}^{\mathrm{pu},max} \approx 62.7\,\mathrm{k\Omega}$) DCP pull-up resistance configurations, respectively.

Figure 8 plots the distribution of the steady state voltage measurements of the I2C bus with the minimum DCP pull-up resistance configuration at supply voltage $V_{\mathrm{supply}} = 2.5\,\mathrm{V}$, together with the fitted Gaussian distribution with the same mean and standard deviation as the measurement data and the density histogram of samples drawn from the fitted Gaussian. Even though the plot visually confirms that the density histogram of the measured steady-state voltage of the I2C bus is close to the density histogram of samples drawn from the fitted Gaussian, the two-sample Cramér-von Mises test rejects the null hypothesis that the measurement data is distributed as the samples drawn from the fitted Gaussian at the 1% level. Measurements yielded the following means and standard deviations for the low and high logic voltage levels of the I2C bus with the minimum and maximum DCP pull-up resistance configurations:

$$V_{\mathrm{I2C}}^{0,\min} = 58.9 \pm 19.4\,\mathrm{mV}, \quad V_{\mathrm{I2C}}^{1,\min} = 2131.6 \pm 19.1\,\mathrm{mV},$$
$$V_{\mathrm{I2C}}^{0,\max} = 25.2 \pm 10.0\,\mathrm{mV}, \quad V_{\mathrm{I2C}}^{1,\max} = 2072.3 \pm 19.6\,\mathrm{mV},$$

and rise/fall times

$$\tau_{\mathrm{rise}}^{\min} \approx 454\,\mathrm{ns}, \qquad \tau_{\mathrm{fall}}^{\min} \approx 18\,\mathrm{ns},$$
$$\tau_{\mathrm{rise}}^{\max} \approx 1092\,\mathrm{ns}, \qquad \tau_{\mathrm{fall}}^{\max} \approx 17\,\mathrm{ns}.$$

These results verify the previous hypothesis that increasing pull-up resistance increases the rise time and decreases the fall time of the I2C bus voltage.

We denote the equivalent resistance of the pull-up branch in Figure 4(b) when the DCP is in minimum/maximum pull-up resistance configuration by

$$R_{\mathrm{eq}}^{\mathrm{pu},\min/\max} = R_{\mathrm{eq}}^{\mathrm{ipu}} \parallel R_{\mathrm{I2C}}^{\mathrm{pu},\min/\max}.$$

$R_{\mathrm{eq}}^{\mathrm{ipu}}$ is the equivalent resistance for the internal pull-up paths (connected to the supply in parallel) of all devices on the I2C bus. The resistance $R_{\mathrm{eq}}^{\mathrm{off}}$ in the pull-down path of the equivalent circuit in Figure 4(b) is the equivalent resistance for the off-resistances of the internal transistors in the pull-down path (connected to the ground in parallel) of all devices on I2C bus.

The steady-state voltage level measurements provided above let us estimate all the resistances in Figure 4(b). In particular,

$$\frac{R_{\mathrm{eq}}^{\mathrm{pu},\min/\max}}{R_{\mathrm{eq}}^{\mathrm{off}}} = \frac{V_{\mathrm{supply}} - V_{\mathrm{I2C}}^{1,\min/\max}}{V_{\mathrm{I2C}}^{1,\min/\max}}. \tag{11}$$

There are two equalities in (11). Taking the ratio of these equalities and plugging in the steady state voltage values gives

$$\frac{R_{\mathrm{eq}}^{\mathrm{ipu}} \parallel R_{\mathrm{I2C}}^{\mathrm{pu},\max}}{R_{\mathrm{eq}}^{\mathrm{ipu}} \parallel R_{\mathrm{I2C}}^{\mathrm{pu},\min}} = \frac{R_{\mathrm{eq}}^{\mathrm{pu},\max}}{R_{\mathrm{eq}}^{\mathrm{pu},\min}} \approx 1.19.$$

With the knowledge of the DCP pull-up resistance values $R_{\mathrm{I2C}}^{\mathrm{pu},\min/\max}$ this ratio yields an equivalent internal pull-up resistance of $R_{\mathrm{eq}}^{\mathrm{ipu}} \approx 0.82\,\mathrm{k\Omega}$, which in turn yields an equivalent transistor off-resistance of $R_{\mathrm{eq}}^{\mathrm{off}} \approx 3.94\,\mathrm{k\Omega}$.

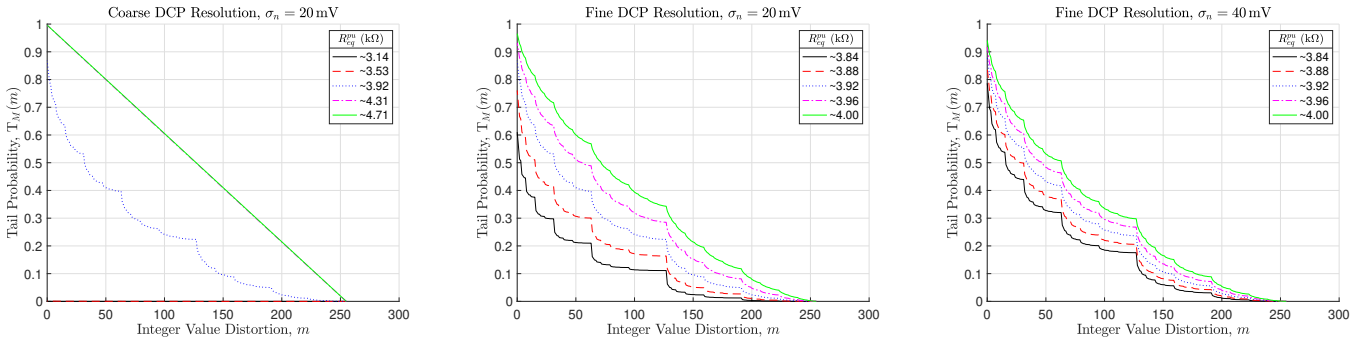
After estimating the resistance values we can estimate C_{I2C} using the rise/fall time measurements and the analytic solution of the voltage on the I2C data bus,

$$V_{\mathrm{I2C}}(t) = V_{\mathrm{I2C}}(0) e^{-\frac{t}{\tau_{\parallel}^{\min/\max}}} + V_{\mathrm{I2C}}^{1,\min/\max}\left(1 - e^{-\frac{t}{\tau_{\parallel}^{\min/\max}}}\right), \tag{12}$$

where $\tau_{\parallel}^{\min/\max} = R_{\parallel}^{\min/\max} C_{\mathrm{I2C}}$ and $R_{\parallel}^{\min/\max} = R_{\mathrm{eq}}^{\mathrm{off}} \parallel R_{\mathrm{eq}}^{\mathrm{pu},\min/\max}$. Using this equation we estimate the I2C capacitance of our system as $C_{\mathrm{I2C}}^{\min} \approx 522\,\mathrm{pF}$.

5.6 Analytic Derivation of Dependence of I2C Channel Integer Value Distortion on Pull-up Resistance

During bit reception process over I2C from a sensor, the sensor sets the voltage of the I2C bus at the beginning of an I2C clock cycle (when the I2C clock SCL is low), either pulling the SDA voltage down to logic `0` or letting the supply pull the SDA voltage up. We assume that an erroneous bit read from a sensor on I2C occurs when the



(a) Induced I2C errors are sensitive to $R_{\text{eq}}^{\text{pu}}$. Consecutive settings of the DCP in Warp (8–12) are too coarse to have control over error dynamics. The straight line is the worst possible operation, where all 1's are wrongly perceived as 0.

(b) Increasing the resolution of the DCP ($10\times$) improves control over induced I2C error characteristics. The resistance values vary around the same value ($3.92\,\text{k}\Omega$) as in (a).

(c) Induced I2C errors are sensitive to the noise levels on the I2C channel. Thus, control over I2C errors induced requires monitoring of and compensation for noise levels.

Fig. 9. Analysis of tail distributions of integer value distortion suffered in I2C channels for uniform input distribution. I2C error dynamics are highly sensitive to the equivalent pull-up resistance value $R_{\text{eq}}^{\text{pu}}$ and the standard deviation σ_n of the noise on I2C channel.

I2C voltage sampled by the microprocessor at the end of the clock cycle (during the high-to-low transition of SCL) is on the wrong side of the threshold voltage V_{th}. We take it to be half of the supply voltage, i.e., $V_{\text{th}} = \frac{V_{\text{supply}}}{2}$. The possible causes for such an event can be long transition times between logic levels compared to the I2C clock period t_{I2C} and logic levels which are on the wrong side of V_{th}, possibly due to noise.

Let $R_{\text{eq}}^{\text{pu},i}$ denote the equivalent resistance on the total pull-up path in the equivalent circuit in Figure 4(b) when the DCP is at the i'th setting. The ISL23415 DCP we use in the experiments of Section 5.4 has 256 programmable levels, so $0 \leq i \leq 255$. Let $V_{\text{I2C}}^{0/1,i}$ be the corresponding steady-state logic `0/1` voltage levels on the I2C bus and let τ_i be the corresponding rise time constant. Let $\{i_1, \ldots, i_8\}$ denote the DCP setting values for the pull-up resistances we switch through during transmission of each integer-encoding byte, where i_1 corresponds to the first bit of the byte sent over I2C. This bit is the least or most significant bit of the byte if the sensor transmits data in little- or big-endian fashion, respectively. Also, let $0 \leq i_{\text{nom}} \leq 255$ be the nominal index for the DCP setting with corresponding equivalent pull-up resistance $R_{\text{eq}}^{\text{pu},i_{\text{nom}}}$ yielding reliable operation without any read errors. We assume that the DCP module switches to this resistance for operation-critical bus transactions, e.g., during transmission of START/STOP or ACK/NACK, and bytes which are not integer-encoded or which cannot tolerate non-zero integer value distortion.

During data reception on I2C from a sensor, the sensor sequentially transmits sensor data in chunks of bytes. During the clock cycle, just before each such byte transmission, SDA is always low. Either the sensor sets it low just before the transmission of the initial byte as an acknowledgement (ACK) to the read request from the microprocessor or the master sets it low as an ACK in between two consecutive bytes the sensor transmits. Since logic `0` transmission on I2C is always reliable, at the beginning of transmission of each integer-encoding byte the SDA voltage will be steady at $V_{\text{I2C}}^{0,i_{\text{nom}}}$.

For $1 \leq j \leq 8$ and a given byte x sent over the I2C bus let $x(j)$ denote the value of j^{th} transmitted bit (in time). In the I2C protocol the most significant bit is transmitted first, so that with the notation introduced in Section 3 we have $x(1) = x_7$. We find the probabilities of erroneous bit reception during transmission of x given the DCP configuration profile $\{i_j\}_{j=1}^{8}$ via the following iterative calculations.

Assuming that the transmission of x occurs during the time interval $[0, 8 * t_{\text{I2C}}]$ and that the initial I2C voltage at time $t = 0$ is $V_{\text{I2C}}(0) = V_{\text{I2C}}^{0,i_{\text{nom}}}$, iteratively on $1 \leq j \leq 8$ we solve an equation similar to the Equation (12) over the time interval $[(j-1)t_{\text{I2C}}, jt_{\text{I2C}}]$ to find $V_{\text{I2C}}(jt_{\text{I2C}})$. If $x(j) = 0$, then transitions are rapid and I2C voltage reaches steady-state level before the end of clock cycle, so that at the end of the j'th cycle we set the I2C voltage to be $V_{\text{I2C}}(jt_{\text{I2C}}) = V_{\text{I2C}}^{0,i_j}$. If $x(j) = 1$ however, then transitions are slow and we calculate the voltage levels at the end of each clock cycle as

$$\begin{aligned} V_j &:= V_{\text{I2C}}(jt_{\text{I2C}}) \\ &= V_{\text{I2C}}((j-1)t_{\text{I2C}})e^{-\frac{t_{\text{I2C}}}{\tau_{i_j}}} + V_{\text{I2C}}^{1,i_j}\left(1 - e^{-\frac{t_{\text{I2C}}}{\tau_{i_j}}}\right). \end{aligned}$$

We take the I2C voltage levels at the end of each I2C clock cycle to be Gaussian distributed (see Figure 8) with mean V_j and standard deviation σ_n, $1 \leq j \leq 8$. Then, given $x(j) = 1$, the probability that the receiver erroneously perceives it as logic `0` is

$$p_j^x = \frac{1}{\sigma_n\sqrt{2\pi}} \int\limits_{V \leq V_{\text{th}}} e^{-\frac{1}{2}\left(\frac{V - V_j}{\sigma_n}\right)^2} dV.$$

After appropriate relabeling of indices (from $1 \leq j \leq 8$ to $0 \leq i \leq 7$) the transmitted-byte-dependent error probabilities of Equation (8) in Section 4.1 read

$$p_{i,\text{down}}^x = p_j^x, \quad x_i = x(j);$$

and

$$p_{i,\text{up}}^x = 0, \quad \forall x \in \mathbb{W}_8, \quad 0 \leq i \leq 7.$$

Finally, using these probabilities we calculate the distribution of the integer value distortion of the channel with the given DCP configuration profile as described in Section 4.1.

Figure 9 shows the dependence of the tail distribution of integer value distortion T_M on the equivalent I2C pull-up resistance values $R_{\text{eq}}^{\text{pu}}$ in the case of uniform input

distribution. Each plot corresponds to a DCP configuration profile with a fixed resistance across a whole byte. The fixed resistances correspond to the DCP settings used in power measurement experiments of Section 5.3. We choose the channel parameters same as the values estimated for our system in Section 5.5 with the exception of the bus capacitance, which we set equal to $C_{\mathrm{I2C}} = 100\,\mathrm{pF}$, more representative of an embedded system with fewer sensors connected to the I2C bus than in the Warp experimental system described in Section 5.3. We assume devices on I2C have no internal pull-up paths, so that the external DCP pull-up dominates the pull-up resistance.

Figure 9(a) shows the distribution of induced integer value distortion for the DCP (ISL23415TFUZ) on the Warp system used in the power dissipation experiments described in Section 5.3. The resolution ($0.39\,\mathrm{k}\Omega$) of the ISL23415TFUZ ($100\,\mathrm{k}\Omega$) was too coarse to have the desired accuracy of I2C error manipulation. Resistance values corresponding to consecutive DCP settings (8–12) yielded big jumps in the error performance of the I2C channel. However, Warp also supports the DCP ISL23415WFUZ ($10\,\mathrm{k}\Omega$), which is pin compatible with the ISL23415TFUZ and has $10\times$ finer resolution. Figures 9(b) and (c) demonstrate the improved control on the I2C error one can achieve when using the ISL23415WFUZ as the choice of the DCP. There is no fundamental reason for not using a DCP with an even smaller granularity in a custom silicon implementation.

Figures 9(b) and (c) also demonstrate the sensitivity of the behavior of I2C error to the I2C noise levels. Figure 9(b) shows results of calculations for I2C voltage noise standard deviation $\sigma_n = 20\,\mathrm{mV}$, whereas Figure 9(c) shows results for $\sigma_n = 40\,\mathrm{mV}$. As expected, increased noise deteriorates the control over the I2C error and shifts the tail distribution of integer value distortion of the channel upwards. Thus, control of the induced I2C channel errors requires one to monitor the noise levels on I2C bus and compensate for changes in noise levels. Because primarily `1` $\rightarrow$ `0` errors can occur as a direct result of increasing pull-up resistor value, I2C channel errors result in received values that are numerically smaller than the actual value transmitted and the integer value distortion (which by definition is nonnegative) shown in Figure 9 correspond to negative errors only.

6 Hardware Overheads of Channel Adaptation

In Section 5 we implemented channel adaptation to I2C via a DCP module. In general, channel adaptation applies to any binary channel that transmits integer data and will require a channel adaptation module similar to the DCP module for each such channel. In this section we estimate the hardware cost of the channel adaptation module that is independent of the channel communication protocol for serial communication channels such as I2C and SPI, which are widely used in embedded systems.

Utilization of bit-level channel adaptation exhibits a number of technical complications from the processor side, such as the operational load introduced to the processor, its accompanying power consumption and the difficulty in implementation from the timing perspective. However, these are straightforward to mitigate by providing a channel

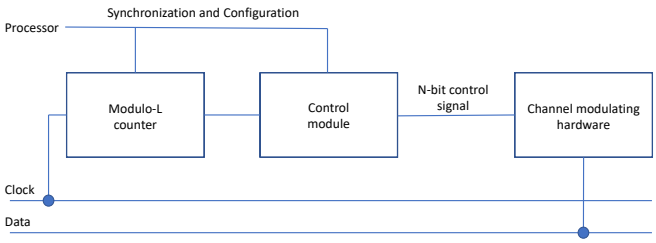


Fig. 10. The channel adaptation module consists of a channel modulating hardware (e.g., a DCP) facilitating adaptation of the data bus, an L-bit counter and a control module that provides the N-bit control signal to the channel modulating hardware The processor feeds the configuration parameters to the control module only once and synchronizes the modulo-L counter with the serial communication transactions.

adaptation module that acts as the glue logic between the processor and the channel modulating hardware, e.g., the DCP module in the case of I2C communication discussed in Section 5.

Figure 10 illustrates a block diagram for the architecture of such a channel adaptation module. For a single sensor transmitting L-bit words on a serial channel, $L + 1$ N-bit registers (1 register for each bit position and 1 for default value) suffice to store a bit-level channel adaptation configuration, which consists of selection values to choose from the 2^N possible states the channel modulating hardware can assume. The approach of feeding these values constantly through the CPU is energy inefficient and does not meet the the timing requirements of bit-level adaptation. To control the channel modulating hardware we propose a minimal control module implemented in a miniature low-power FPGA, which can be easily integrated, for instance, into existing DCP designs. The module receives from the processor the channel adaptation lookup table only once and executes adaptation when instructed. This enables fast and low-power switching of the channel modulating hardware and facilitates bit-level adaptation of high-frequency serial communication channels.

To quantify the hardware footprint of the protocol-agnostic part of the channel adaptation module, we implement its control logic in an FPGA using the Verilog hardware description language. Algorithm 1 encapsulates our implementation. We design the hardware module as a two-state finite state machine that modulates the communication channel at each negative edge of the transmission clock during data transaction. During the transmission of the i'th bit of a word, the module modulates the channel according to the selection specified by register R_i.

We have synthesized the Verilog implementation on an iCE40 FPGA for word lengths $L = 8\,/12\,/16$. The synthesis revealed that, the protocol-agnostic part of the synchronization logic for the channel adaptation module utilizes $61\,/61\,/61$ carry logic, $34\,/34\,/34$ negative edge clock D flip-flops, $72\,/104\,/136$ negative edge clock D flip-flops with clock enable and $224\,/248\,/291$ 4-input LUTs.

7 Conclusions

We present a new communication channel adaptation technique that trades integer distance in induced errors for

Algorithm 1: Pseudocode for synchronizing the channel adaptation module to a serial channel with transmitted words of size L.

Inputs: SCL (clock signal), S_{start} (start signal) and L (word length).
Outputs: S_{select} (an 8-bit selection signal to manipulate channel modulation hardware.
Hardware Components:
– A channel modulating hardware with 8-bit selection option.
– $L+1$ 8-bit registers (R_0 to R_L) to hold the $L+1$ channel modulation selections. R_0 holds the default selection value and R_1 to R_L hold the selections to switch through during sensor data transmission.
State Transitions:
– Start at the initial state s_0
– **In** s_0: `// The idle state`
  i=0;
  `/* Beginning of a word transmission */`
  if $S_{\text{start}} == 1$ *and* SCL *at negative edge* **then**
    i=i+1
    Switch to s_1
– **In** s_1: `// Counting state`
  if $i < L$ *and* SCL *at negative edge* **then**
    i=i+1
    else if $i = L$ *and* SCL *at negative edge* **then**
      i=0
      Switch to s_0
Always: $S_{\text{select}} \leftarrow R_i$

lower power usage. The technique achieves this by modulating the channel to vary the bit-error rate across the ordinal bit positions in a word to reduce power dissipation. To predict channel parameters, it utilizes an efficient method for calculating the exact distribution of integer value distortion suffered during bit-level adaptation. We carry out a case study for implementation of the technique on I2C channel. We empirically verify power reduction of up to $2\times$ ($0.8\,\text{mW}$) by means of modulating the pull-up resistance of the I2C bus of the Warp hardware platform. We explore the design space and feasibility of the proposed scheme utilizing the developed theoretical tools for calculating the integer value distortion distribution and analytical derivations of I2C channel errors. Results indicate the need for noise compensation for reliable channel adaptation. Finally, by means of synthesizing in FPGA the protocol-agnostic part of bit-level synchronization to serial communication channels, we give an estimate for the footprint of a channel modulation hardware for sensor sample resolutions common in embedded sensor systems (e.g., 8–16 bits).

ACKNOWLEDGMENTS

This research is supported by an Alan Turing Institute award TU/B/000096 under EPSRC grant EP/N510129/1, and by Royal Society grant RG170136. We thank Richard Hopper for assisting with power dissipation measurements in Section 5, Srisht Fateh Singh for help with the FPGA synthesis in Section 6, and James Meech, Nathaniel Tye, Chatura Samarakoon, and Vasileios Tsoutsouras for providing feedback on the manuscript.

REFERENCES

[1] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.
[2] M. Shafique, R. Hafiz, S. Rehman, W. El-Harouni, and J. Henkel, "Cross-layer approximate computing: From logic to architectures," in *Proceedings of the 53rd Annual Design Automation Conference*, 2016, pp. 1–6.
[3] P. Stanley-Marbell, A. Alaghi, M. Carbin, E. Darulova, L. Dolecek, A. Gerstlauer, G. Gillani, D. Jevdjic, T. Moreau, M. Cacciotti *et al.*, "Exploiting errors for efficiency: A survey from circuits to algorithms," *arXiv preprint arXiv:1809.05859*, 2018.
[4] P. Stanley-Marbell, V. Estellers, and M. Rinard, "Crayon: Saving power through shape and color approximation on next-generation displays," in *Proceedings of the Eleventh European Conference on Computer Systems*, ser. EuroSys '16. New York, NY, USA: Association for Computing Machinery, 2016.
[5] H. Hatamkhani, F. Lambrecht, V. Stojanovic, and C.-K. K. Yang, "Power-centric design of high-speed i/os," in *Proceedings of the 43rd annual Design Automation Conference*. ACM, 2006, pp. 867–872.
[6] F. O'Mahony, G. Balamurugan, J. E. Jaussi, J. Kennedy, M. Mansuri, S. Shekhar, and B. Casper, "The future of electrical i/o for microprocessors," in *2009 International Symposium on VLSI Design, Automation and Test*. IEEE, 2009, pp. 31–34.
[7] K. T. Malladi, B. C. Lee, F. A. Nothaft, C. Kozyrakis, K. Periyathambi, and M. Horowitz, "Towards energy-proportional datacenter memory with mobile dram," in *Proceedings of the 39th Annual International Symposium on Computer Architecture*, ser. ISCA '12, 2012, pp. 37–48.
[8] P. Stanley-Marbell and P. Hurley, "Probabilistic value-deviation-bounded integer codes for approximate communication," *arXiv preprint arXiv:1804.02317*, 2018.
[9] P. Stanley-Marbell and M. Rinard, "Reducing serial i/o power in error-tolerant applications by efficient lossy encoding," in *Proceedings of the 53rd Annual Design Automation Conference*, ser. DAC '16. New York, NY, USA: ACM, 2016, pp. 62:1–62:6.
[10] ——, "Efficiency limits for value-deviation-bounded approximate communication," *IEEE Embedded Systems Letters*, vol. 7, no. 4, pp. 109–112, 2015.
[11] Y. Kim, S. Behroozi, V. Raghunathan, and A. Raghunathan, "Axserbus: A quality-configurable approximate serial bus for energy-efficient sensing," in *2017 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE, 2017, pp. 1–6.
[12] P. Stanley-Marbell, P. A. Francese, and M. Rinard, "Encoder logic for reducing serial i/o power in sensors and sensor hubs," in *Hot Chips 28 Symposium (HCS)*. IEEE, 2016, pp. 1–2.
[13] C.-H. Huang, Y. Li, and L. Dolecek, "Acoco: Adaptive coding for approximate computing on faulty memories," *IEEE Transactions on Communications*, vol. 63, no. 12, pp. 4615–4628, 2015.
[14] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE National Convention Record*, vol. 7, no. 4, pp. 142–163, 1959.
[15] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.
[16] P. Stanley-Marbell, "Encoding efficiency of digital number representations under deviation constraints," in *IEEE Information Theory Workshop, (ITW'09)*, Oct 2009, pp. 203–207.
[17] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Urbana, Illinois: University of Illinois Press, 1963.
[18] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SigKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.
[19] P. Stanley-Marbell and M. Rinard, "Warp: A hardware platform for efficient multi-modal sensing with adaptive approximation," *IEEE Micro*, vol. 40, no. 1, pp. 57–66, Jan 2020.
[20] *ISL23415 - Single, Low Voltage Digitally Controlled Potentiometer*, Renesas Electronics, 9 2015, rev. 2.00.
[21] *MMA8451Q, 3-axis, 14-bit/8-bit digital accelerometer*, NXP Semiconductors, 2 2017, rev. 10.3.

Bilgesu Arif Bilgin is currently a postdoctoral research associate in the Physical Computation Laboratory of the Engineering Department at the University of Cambridge. Prior to that he has worked as a postdoctoral researcher at the Department of Engineering at the Koc University, Turkey (2015–2017). He has received his B.S. degree in Electrical and Electronics engineering from Middle East Technical University in 2008, M.Sc. and Ph.D. degrees in Mathematics from Koc University, in 2011 and 2015, respectively. His research interests include unconventional computing, neural information theory, nano-fabrication, partial differential equations, modelling and molecular communications.

Phillip Stanley-Marbell is currently an Assistant Professor with the Department of Engineering at the University of Cambridge, where he leads the Physical Computation Laboratory. His research focuses on exploiting an understanding of properties of the physical world and the flexibility of sensing systems, to make more efficient computing systems that interface with nature. Prior to joining the University of Cambridge, he was a Researcher with the Massachusetts Institute of Technology, from 2014 to 2017. He has held positions in industry at Bell-Labs Research (1995, 1996), Lucent Technologies and Philips (1999), and NEC Research Labs (2005). He was a permanent Research Staff Member with IBM Research—Zürich from 2008 to 2012, and then an Engineer with Apple until 2014. He received the PhD degree from CMU in 2007 and is a Senior Member of the IEEE. Contact him at phillip.stanley-marbell@eng.cam.ac.uk.