Bitline PUF: Building Native Challenge-Response PUF Capability into Any SRAM

Daniel E. Holcomb and Kevin Fu

University of Michigan, Ann Arbor MI 48109, USA
`danholcomb@umich.edu` `kevinfu@umich.edu`

Abstract. Physical Unclonable Functions (PUFs) are specialized circuits with applications including key generation and challenge-response authentication. PUF properties such as low cost and resistance to invasive attacks make PUFs well-suited to embedded devices. Yet, given how infrequently the specialized capabilities of a PUF may be needed, the silicon area dedicated to it is largely idle. This inefficient resource usage is at odds with the cost minimization objective of embedded devices. Motivated by this inefficiency, we propose the Bitline PUF – a novel PUF that uses modified wordline drivers together with SRAM circuitry to enable challenge-response authentication. The number of challenges that can be applied to the Bitline PUF grows exponentially with the number of SRAM rows, and these challenges can be applied at any time without power cycling. This paper presents in detail the workings of the Bitline PUF, and shows that it achieves high throughput, low latency, and uniqueness across instances. Circuit simulations indicate that the Bitline PUF responses have a nominal bit-error-rate (BER) of 0.023 at 1.2 V supply and 27°C, and that BER does not exceed 0.076 when supply voltage is varied from 1.1 V to 1.3 V, or when temperature is varied from 0°C to 80°C. Because the Bitline PUF leverages existing SRAM circuitry, its area overhead is only a single flip-flop and two logic gates per row of SRAM. The combination of high performance and low cost makes the Bitline PUF a promising candidate for commercial adoption and future research.

Keywords: VLSI, SRAM, PUFs, Strong PUFs

1 Introduction

An emerging alternative to classical cryptography in embedded systems is the use of physical unclonable functions (PUFs). PUFs use random manufacturing variations constructively, either to generate cryptographic keys, or to implement physical hash functions for challenge-response authentication [32]. The secret key style of PUF is sometimes called a weak PUF, and PUFs capable of challenge-response hashing are sometimes called strong PUFs [7]. We adopt the weak versus strong naming convention for this paper, and further clarify that strong PUF here denotes a circuit that natively provides physical challenge-response hashing, to distinguish it from a weak PUF that is used to key a classical hash function to provide the logical equivalent of a strong PUF.

In this paper we present a novel strong PUF termed the Bitline PUF. The Bitline PUF leverages the storage cells and support circuitry of SRAM to save area cost, and achieves high throughput by using individual SRAM columns as parallel PUFs instances. The main contributions of this paper are as follows:

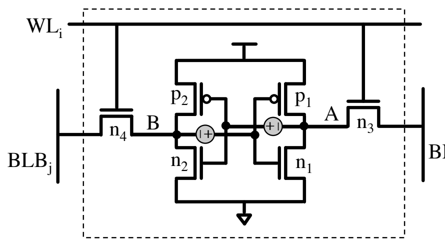
– We present the first strong PUF that creates responses from contention between cells in pre-existing circuitry.
– We show that adding a small amount of circuitry to SRAM creates a new strong PUF based on bitline contention.
– We present in detail the operation of the Bitline PUF and analyze its throughput, latency.
– We evaluate using circuit simulation the uniqueness, reliability, power consumption, and susceptibility to modeling attacks of the Bitline PUF.

2 Static Random-Access Memory

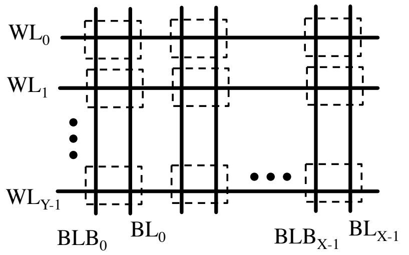
Static Random-Access Memory (SRAM) is a ubiquitous building block of integrated circuits that is found in caches, register files, and buffers. Single VLSI circuits commonly contain millions of bits of SRAM storage. Each bit of SRAM is typically implemented by a single 6-transistor cell (Fig. 1a). An SRAM cell has two stable states, and in each stable state node A or B is pulled high through transistor p_1 or p_2 while the other is pulled low through n_1 or n_2. The cell is read and written using complementary bitlines (BL) and (BLB) through two access transistors n_3 and n_4. The two access transistors of a cell are controlled by a single wordline.

The SRAM cells in a memory are arranged in a matrix of rows and columns (Fig. 1b). SRAM cells in the same column share common bitlines and hence only one cell per column is accessed at any time. SRAM cells in the same row share a wordline but have independent bitlines and are therefore read and written in parallel as data words. Each SRAM column uses support circuitry to read and write its cells. A cell is written by setting one bitline high and the other low and then asserting the wordline to transfer the bitline values to the cell.

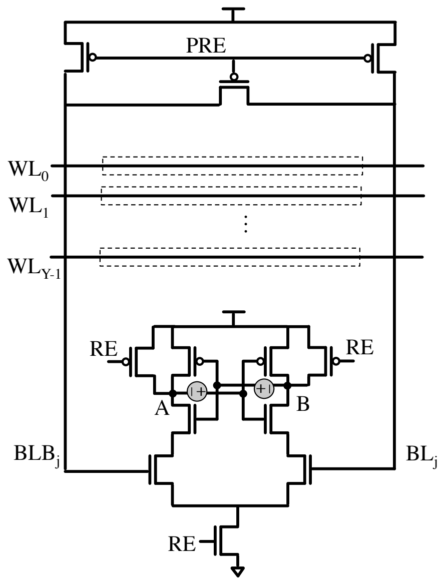
An evaluation of the Bitline PUF is similar to an SRAM read operation, and hence a detailed explanation of the SRAM read operation is given here as background. The support circuitry for a read operation comprises precharge logic at the top of each column and a sense amplifier at the bottom (Fig. 1c). Fig. 2a shows the timing of the control signals (PRE, WL, and RE) for a read operation and shows overlaid bitline waveforms from reading cells with different process variations. During an SRAM read operation, both bitlines are first charged and equalized by the precharge circuit at the top of the column. Next, the precharge signal (PRE) goes high to end the precharge phase and the wordline (WL) for a



(a) 6-Transistor SRAM cell



(b) Rows and columns in SRAM



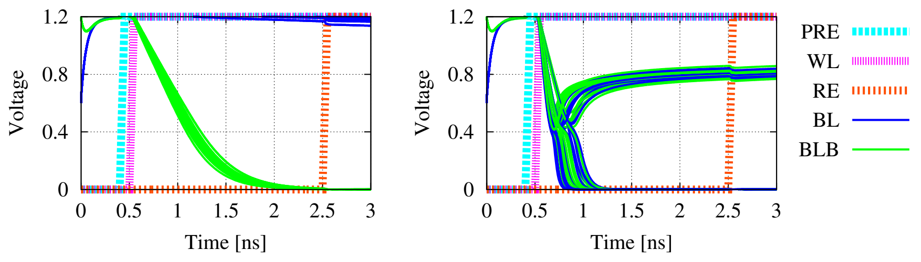
(c) Precharge circuit and sense amplifier of an SRAM column

Fig. 1: SRAM cells are arranged in a matrix of rows and columns. SRAM rows share wordlines, and columns share bitlines. Each column uses a precharge circuit and a sense amplifier to perform read operations. Note that the circuitry used for writing values to cells is not depicted.

single row is asserted. The wordline connects a cell to the precharged bitlines and depending on the state of the cell, transistor n_1 or n_2 will begin to discharge one of the bitlines through the corresponding access transistor. The discharge rate of the bitline varies depending on the random variation of the transistor that is discharging it [8]. A fixed time after the wordline is asserted, a read-enable signal (RE) is asserted to activate the sense amplifier. The sense amplifier detects the difference in voltage across the two bitlines and generates from it a digital 0 or 1 value. The digital value in the sense amplifier is the final result of the SRAM read operation, and can be sent out of the SRAM.

3 System Description of Proposed Bitline PUF

The proposed Bitline PUF is a novel PUF formulation that borrows much of its circuitry from SRAM. The operation of the Bitline PUF can be viewed as an attempt to read multiple cells in a column at the same time, creating contention that is resolved according to process variation. A challenge is applied to the PUF by pre-loading chosen values into the cells, and choosing the wordlines to concurrently activate. The PUF response is simply the value that the SRAM



(a) Reading "1" from a cell (b) Bitline PUF evaluation

Fig. 2: SRAM read operation and bitline PUF evaluation use the same control signal timing. The precharge signal (PRE) is asserted to stop charging the bitlines, and the wordline (WL) is asserted to begin the discharging of the bitline. The read enable signal (RE) is asserted 2 ns later to activate the sense amplifier that detects the voltage difference between the two bitlines. The thin lines (BL and BLB) are overlaid plots of the bitline voltages from 30 different trials; only one bitline discharges in the read operation, but in the PUF evaluation both bitlines initially discharge and then stabilize with one high and one low.

produces from a read operation when the challenge condition is applied. The Bitline PUF requires additional circuitry to enable the concurrent activation of multiple wordlines because the capability of activating multiple wordlines has no use in SRAM's traditional tasks of reading, writing, and storing data.

Let the challenge applied to a Bitline PUF be $C : \{c_0, c_1, \ldots, c_{Y-1}\}$, where Y is the number of rows in the SRAM. Each element c_i of the challenge corresponds to SRAM row i as follows, and we say that any row is *active* in a challenge if its corresponding challenge element (c_i) is either 0 or 1.

- if $c_i = 0$, then row i is loaded with 0s and WL_i is on during evaluation.
- if $c_i = 1$, then row i is loaded with 1s and WL_i is on during evaluation.
- if $c_i = 2$, then row i is loaded with 0s and WL_i is off during evaluation
- if $c_i = 3$, then row i is loaded with 1s and WL_i is off during evaluation

A single SRAM column constitutes a Bitline PUF with a 1-bit response, and Bitline PUFs are therefore inherently parallel because a challenge is applied concurrently to many SRAM columns. Let a 1-bit PUF at column i be denoted P_i, and its response to challenge C be denoted $P_i(C)$. Let an X-column Bitline PUF be denoted $P_{0:X-1}$ and its response be $P_{0:X-1}(C) = \{P_0(C), P_1(C), \ldots, P_{X-1}(C)\}$. Note that for simplicity the same challenge is applied to all columns of the SRAM PUF[1]. Therefore, a Bitline PUF with Y rows and X columns has 4^Y possible challenges and 2^X possible responses.

[1] Different challenges can be applied to different columns provided that the challenges agree on which rows are active. This can be particularly useful in the case of inactive rows that retain pre-existing data through a challenge.

3.1 Challenge-Response Operation

The sequence of events necessary to operate the Bitline PUF is shown in Fig. 3. The first two phases set up the desired challenge by loading values into SRAM cells and enabling the appropriate wordlines. The final phase evaluates the PUF response by reading the value produced when the challenge is applied. The Bitline PUF evaluation is destructive with respect to active rows only. It is therefore possible to use only some rows of SRAM as part of a Bitline PUF evaluation while others rows are being used as storage. The three phases of operation are described in the following paragraphs.

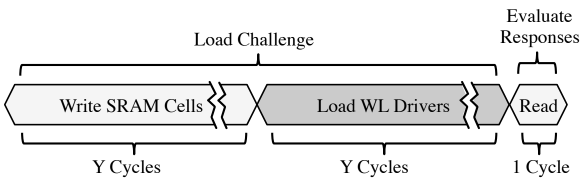


Fig. 3: Sequence of operations for evaluating the PUF response to a challenge

Write Values into SRAM Cells: The values loaded into the SRAM cells of active rows will determine which transistors will ultimately be used to discharge the bitlines during the evaluation of the PUF response. To load a specific challenge, the cells of each row i are written with the value specified by c_i. The SRAM cells, as in other write operations, are written one row at a time, so the time to write all Y rows is Y cycles.

Load Wordline Drivers using Accumulators: The proposed SRAM PUF requires augmentation to the wordline control circuitry so that multiple wordlines can be concurrently enabled during PUF evaluation. In a typical SRAM, an externally supplied $\log_2(Y)$-bit address is decoded to select exactly one of the Y rows for reading; the selected row then uses a clocked driver to set its wordline high at the appropriate time during the clock cycle. The proposed PUF requires multiple wordlines to be concurrently enabled, and this can be accomplished by having at the input of each wordline driver a flip-flop (Fig. 4) that accumulates wordline activation signals. At the start of the second phase of Fig. 3, the accumulator of every wordline is reset. In each of the subsequent Y cycles, a $\log_2(Y)$-bit select signal sets high the flip-flop of one active wordline. Once all flip-flops are appropriately loaded, an evaluation signal passes the loaded values to the wordline drivers, so that multiple wordlines are asserted in the same cycle during the PUF evaluation. One wordline accumulator per SRAM row is the only additional circuitry required to create bitline PUFs from an SRAM.

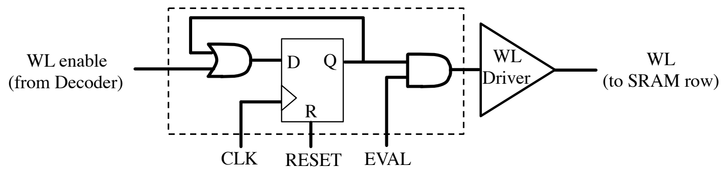


Fig. 4: The wordline accumulator circuit enables sequential loading and concurrent activation of wordlines.

Evaluate Responses: Evaluating the PUF response is identical to an SRAM read operation, except that multiple wordlines are asserted. For each column, the cells at any active row will discharge one of the two complementary bitlines, and considering that different cells in the column will discharge each bitline, this causes both bitlines of a column to be discharged during the evaluation. The discharging of bitlines for a variety of challenges are shown in Fig. 2b. While both bitlines initially discharge, there is no stable state in which both bitlines are fully discharged due to the cross-coupled inverters in the active SRAM cells. Contention thus ensues until a stable state is reached with one bitline charged and the other fully discharged. Note that the charged bitline in the stable state does not charge all the way to V_{dd}, but only charges to $V_{dd} - V_{th}$ because it is being pulled high by SRAM cells through an NMOS access transistor that causes a voltage drop of V_{th}. If the bitlines reach a stable state in the time between the assertion of the wordline and the assertion of the read enable (RE) signal, then the sense amplifier unambiguously detects the large differential voltage of $\pm(V_{dd} - V_{th})$ across the bitlines, and generates a digital output as in a normal SRAM read operation. This output is the response to the applied challenge.

3.2 Performance

The three phases of Bitline PUF operation (Fig. 3) define its latency and throughput. All cells are written in Y cycles, all wordline accumulators are loaded in Y additional cycles, and all X columns are evaluated in parallel during a single cycle. Therefore, the latency to obtain an X-bit response is $2Y + 1$ cycles and the response throughput is $\frac{X}{2Y+1}$. For a 256-column by 256-row SRAM with a 5ns cycle time, this corresponds to a latency of 2.6 μs and a response throughput of 99.8 Mbps.

4 Methodology

The results in this paper are obtained from circuit simulation using the Ngspice simulator (Rev 25). On account of the long runtimes of large SPICE simulations, the columns of the simulated bitline PUFs have only 16 rows, whereas a real SRAM would typically have hundreds of rows.

4.1 Transistor Models and Sizing

Transistor and interconnect models are from the freely-available Predictive Technology Model (PTM). More specifically, the transistor models are BSIM4 PTM models for a 90 nm process [29]. Transistor sizes are shown in Tab. 1; the six transistors in the SRAM cell are sized to match the design of Nii *et al.* [23], and the transistors in the sense amplifier and precharge circuits are upsized.

		Sizing		Process Variation			
		W [nm]	L [nm]	vth0 [mV]		lint [nm]	
				μ	σ	μ	σ
SRAM cell	n1,n2	200	90	397	13.4	7.5	3
	n3,n4	140	90	397	16.0	7.5	3
	p1,p2	140	90	-339	16.0	7.5	3
Sense Amp & Precharge	NMOS	1000	90	397	6.0	7.5	3
	PMOS	1000	90	-339	6.0	7.5	3

Table 1: Transistor sizes and process variation. The transistor sizes used within SRAM cells are adopted from Nii *et al.* [23], and threshold voltage variation depends on transistor size (Eq. 1).

4.2 Bitline Model

To better represent a real design, the 16 SRAM rows simulated are modeled as being distributed over a typical-length bitline. In this way, the 16 rows can be considered as existing among many others within a realistic-sized SRAM. Keeping with the work of Nii *et al.* [23], we assume for bitline modeling an SRAM with 520 rows and a cell height of 0.72 μm, for a total length of 374.4 μm per bitline. According to the PTM interconnect calculator [30], a 374.4 μm local interconnect in 90 nm technology has a total resistance of 183.04 Ω and capacitance of 69.67 fC. The resistance and capacitance is distributed such that the bitlines between each of pair of adjacent rows is implemented by a wire model with an 11.44 Ω resistance between two capacitors of 2.17 fC each.

4.3 Process Variation

To model process variations from fabrication, random parameter variation is applied to every transistor of each PUF instance. The transistor parameters determining threshold voltage and length are replaced by normally distributed $\mathcal{N}(\mu, \sigma^2)$ random variables. Table 1 shows the mean and standard deviation for each such parameter.

Random dopant fluctuation is represented in transistor parameter `vth0`. The mean value for threshold voltage is the default value in the transistor model, and the standard deviation depends on transistor geometry according to Eq. 1 [27];

larger devices have less threshold variation than the small devices in the SRAM cells. We use a value of 1.8 $mV\mu m$ for A_{VT} [31].

$$\sigma_{VT} = \frac{A_{VT}}{\sqrt{WL}} \tag{1}$$

Variations in effective transistor length are represented by changes to parameter `lint`[2]. The nominal value of `lint` is 7.5 nm and its standard deviation is set to 3 nm based on the observation that effective transistor length has a 3σ value that is 10% of overall transistor length [1]

4.4 Modeling Noise

Thermal noise is modeled in SPICE by transient random voltage sources. As represented by small grey circles in Fig. 1a and Fig. 1c, noise sources are added between the cross-coupled state nodes of SRAM cells [34] and sense amplifiers. The magnitude of thermal noise at each node depends on the node capacitance (Eq. 2). The standard deviation of noise for each SRAM cell node is set to 4.5mV, and for each sense amplifier node is 1.7 mV[3].

$$\sigma_{NOISE} = \sqrt{\frac{k_B T}{C}} \tag{2}$$

5 Evaluation

The simulation methodology explained in the previous section is used for experimental evaluation of the Bitline PUF. Uniqueness of responses, and reliability with respect to temperature and supply voltage variation are evaluated. Finally, power consumption and susceptibility to modeling attacks are considered. These experimental results indicate that the Bitline PUF is promising as a reliable and unique strong PUF.[4]

5.1 Unbiased Challenges to Elicit Unique Responses

The mixture of c_i values in each challenge can bias PUFs toward producing 0-responses or 1-responses, but ideal challenges should produce either response with equal probability across a population. From a circuit perspective, ideal challenges should discharge both bitlines with equal strength to increase the sensitivity of response to process variations. For a symmetric SRAM cell, where only variation differentiates n_1 and p_1 from n_2 and p_2, the two complementary

[2] `lint`, standing for internal length, represents the difference between nominal and effective transistor length

[3] Ngspice source `vxx a an dc 0 trrandom (2 100p 0 1.7m 0)`

[4] All software used in experiments is freely available, and the source code for all experiments in this paper is provided online at `https://spqr.eecs.umich.edu/papers/Holcomb-bitline-CHES2014.zip`

bitlines discharge with equal strength when the same number of NMOS transistors (i.e, n_1 or n_2 of each active cell) are discharging each one. The challenges that cause this situation are those having an equal number of $c_i = 0$ and $c_i = 1$ values, along with some unspecified mixture of inactive rows with $c_i = 2$ or $c_i = 3$; challenges satisfying this condition are therefore denoted as "unbiased".

The heat map of Fig. 5a confirms that unbiased challenges are the ones most likely to elicit different responses from different PUF instances. For each of the 64 squares in the plot, 1000 randomly generated challenges with the specified number of 0s and 1s are created. Each of the challenges is applied to two randomly selected PUF instances to check whether the responses differ. For the unbiased challenges, along the diagonal of Fig. 5a, the responses of the two PUFs differ in roughly half of all trials. For challenges that are slightly biased (i.e. close to the diagonal), the PUFs sometimes produce differing responses. For challenges that are highly biased (e.g. at the upper left and bottom right corners of Fig 5a), all PUF instances produce the same response.

The number of unbiased challenges having exactly k challenge values with $c_i = 0$ and k with $c_i = 1$ is given by $n'_k(Y)$ (Eq. 3). The number of total unbiased challenges with any number of $c_i = 0$ and $c_i = 1$ values is given by $n(Y)$ (Eq. 4). The number of unbiased challenges is exponential in the number of rows Y (i.e. the challenge size). Therefore, an adversary cannot hope to mimic a PUF by simply recording all challenge-response pairs, and must instead resort to predicting responses using a parametric model [17,33] (see Sec. 5.4).

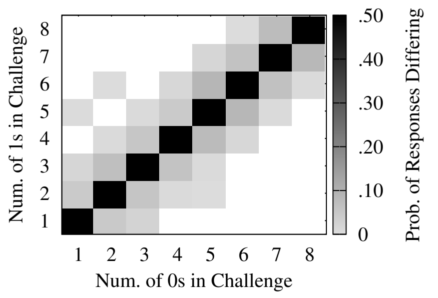
$$n'_k(Y) = \binom{Y}{k} * \binom{Y-k}{k} \tag{3}$$

$$n(Y) = \sum_{k=1\ldots\frac{Y}{2}} n'_k(Y) \tag{4}$$

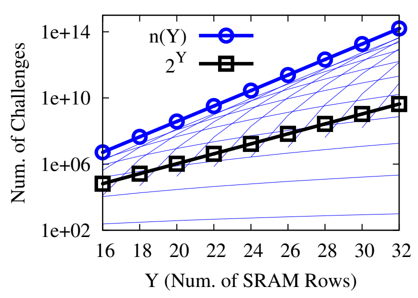
5.2 Within-Class and Between-Class Hamming Distances

A single PUF should always respond to the same challenge similarly, and two PUF instances should never respond to the same unbiased challenges similarly. For a challenge C, a comparison of two responses from the same PUF is denoted "within-class", and a comparison of responses from two different PUFs is denoted "between-class." Hamming distance (Eq. 5) is used to quantify the similarity of responses in each between-class or within-class comparison. Within-class distances are a measure of unreliability, and between-class distances are a measure of uniqueness.

Within-class and between-class Hamming distances are evaluated experimentally on 32-column bitline PUFs. For each of 200 random unbiased challenges, 5 PUF instances are generated and the challenge is applied 6 times to each. Within-class distances are obtained by comparing the responses of the same PUF to the same challenge, and between-class distances are obtained by comparing the response of different PUFs to the same challenge. The separability of within-class



(a) Challenges with an equal number of 0 and 1 values are termed unbiased



(b) The number of unbiased challenges is exponential in the number of SRAM rows

Fig. 5: Challenges with equal numbers of 0 and 1 values are most likely to produce different responses across PUF instances. We refer to these challenges as unbiased. The number of unbiased challenges grows exponentially in the number of SRAM rows; this is depicted at right where Y is the number of rows and $n(Y)$ (Eq. 4) is the number of unbiased challenges. The thin lines at right depict $n'_k(Y)$ (Eq. 3), the number of unbiased challenges with exactly k 0s and k 1s, for all values of k.

and between-class Hamming distances (Fig. 6) implies that responses are unique across Bitline PUF instances. The average within-class Hamming distance is 0.75 for a 32-bit response, and the average between-class distance is 16.01.

$$\mathrm{HD}(P_{0:X-1}, P'_{0:X-1}, C) = \sum_{i=0...X-1} P_i(C) \oplus P'_i(C) \tag{5}$$

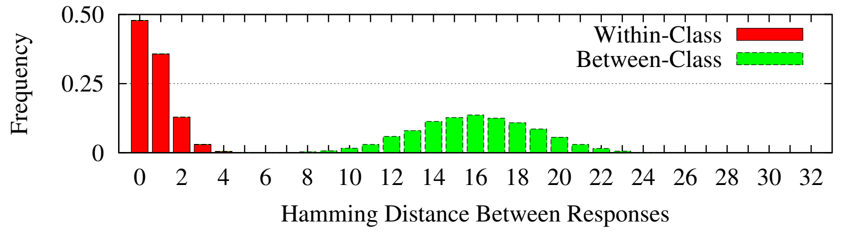


Fig. 6: Within-class and between-class Hamming distances for 32-bit PUF responses. The separation between the distributions shows uniqueness of instances.

5.3 Sensitivity to Supply Voltage and Temperature Variations

A PUF response should not be highly sensitive to changes in supply voltage or temperature, as this would restrict its useful application to tightly controlled environments. PUF responses at the nominal operating conditions of 1.2 V supply and 27°C are compared against a variety of temperatures from 0°C to 80°C and supply voltages from 1.1 V to 1.3 V (Fig. 7). For each comparison 10,000 random PUF instances are created. For each instance, a randomly chosen unbiased challenge is applied to the PUF at both conditions; the BER is the fraction of these 10,000 trials in which the two responses differ. While changing supply voltage or temperature does increase the BER of responses, at all tested conditions the BER remains less than 0.076.

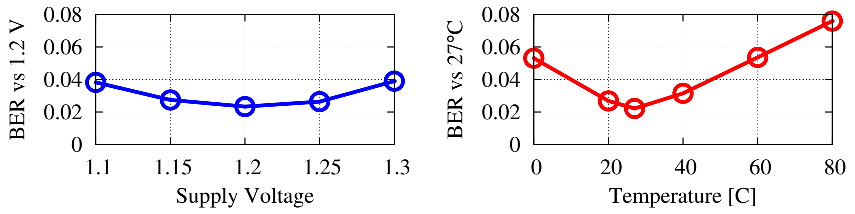


Fig. 7: Bit error rate of responses when one is collected at nominal conditions of 1.2 V and 27°C, and the second at a different supply voltage or temperature.

5.4 Modeling Attacks

The Bitline PUF is susceptible to modeling attacks if the challenge-response pairs (CRPs) can be observed, and therefore care must be taken to avoid or obfuscate the CRPs of the Bitline PUF. Otherwise, an adversary can use a parametric model to predict the PUF response to any challenge [17,33], without needing a dictionary of all possible challenge response pairs.

We demonstrate a modeling attack on bitline PUFs using support vector machine (SVM) classification. The task of the SVM classifier is, after training on some number of observed CRPs, to correctly predict responses to new challenges. To use SVM classification, each CRP is converted to a pair $(x, y)\,|x \in \{0,1\}^{4Y}, y \in \{-1,+1\}$ where Y is the number of rows in the PUF and the number of values in the challenge. In the pair (x, y), x represents the challenge and is determined according to Eq. 6, while y represents the response of the PUF to the challenge. Note that for SVM classification, negative responses are entered as the value -1 instead of 0.

$$x_{4i:4i+3} = \begin{cases} 1,0,0,0 & \text{if } c_i == 0 \\ 0,1,0,0 & \text{if } c_i == 1 \\ 0,0,1,0 & \text{if } c_i == 2 \\ 0,0,0,1 & \text{if } c_i == 3 \end{cases} \tag{6}$$

Fig. 8 shows the prediction accuracy of SVM classification using the tool SVM^{light} [13], applied to three different bitline PUF instances. For each PUF instance, 1000 CRPs are collected and cross-validation is used to examine how the prediction accuracy varies with the size of the training set. After 500 CRPs are observed, responses can be predicted with approximately 90% accuracy. While for clarity only three PUFs are plotted in Fig. 8, these three results are typical of observed prediction accuracy trends for bitline PUFs.

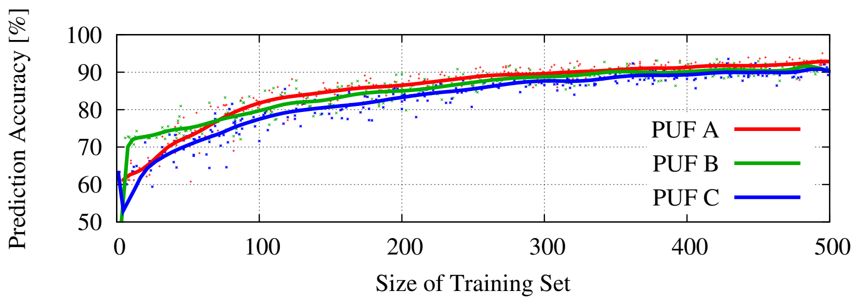


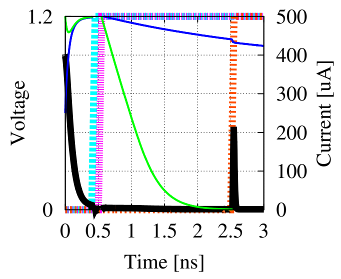
Fig. 8: Modeling attacks are possible if the challenge-response pairs of bitline PUFs are not protected. The data points in the plot are cross-validation results for prediction accuracy using support vector machine classification, and the curves are fit lines for the results.

Parametric models exist for many PUFs including the arbiter PUF [17,33]. Yet, the practical usefulness of PUFs with parametric models is not diminished because modeling resistance can be assured through obfuscating or preventing access to the PUF responses [36]. The Bitline PUF is uniquely suited to protection via access control because it uses ordinary SRAM, and as such can employ SRAM access protection mechanisms including ARM TrustZone [2].

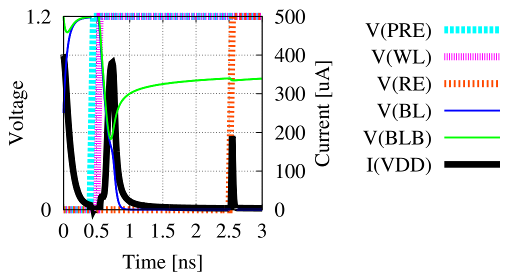
5.5 Power Consumption

The power consumption of a bitline PUF evaluation is higher than that of a standard SRAM read operation. More specifically, a bitline PUF draws significant current during metastability when the bitline potentials are approximately equal. During metastability, all of the cells that are active in the challenge are drawing current, either through p_1 and n_2, or else through p_2 and n_1.

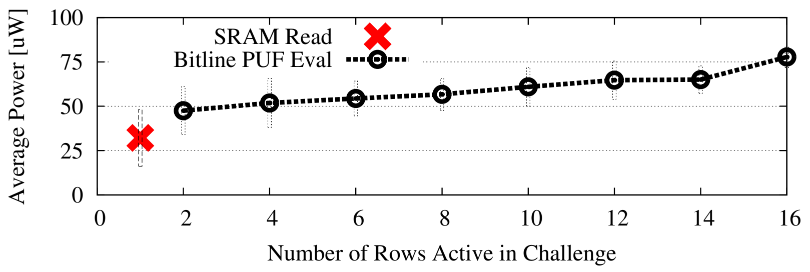
Fig. 9a shows a normal SRAM read operation and its current draw; the most significant instantaneous currents are consumed when the bitlines are precharged, and when the sense amplifier turns on. During a bitline PUF evaluation (Fig. 9b), an additional third current spike is observed during metastability. The power consumed by a bitline PUF evaluation depends on the size of this current spike. When more cells are active, there is a potential for larger instantaneous current and therefore higher power. Fig. 9c shows that average power increases with the number of rows that are active in a challenge.



(a) Read operation



(b) Bitline PUF evaluation



(c) Average power versus number of active rows. Error bars mark the standard deviation over 100 trials.

Fig. 9: While a bitline PUF is metastable, there exists a current path from supply to ground through active SRAM cells. The average power of a bitline PUF evaluation therefore exceeds that of an SRAM read operation, and increases with the number of active cells in a challenge.

6 Related Work

Strong PUFs are marked by the ability to map challenges to responses according to a function determined by random physical variations. The first such PUF was based on optical scattering [26], but the practicality of strong PUFs increased with the invention of silicon PUFs that can be integrated in VLSI circuits. The first and best-known silicon PUFs are delay-based PUFs [6] including the arbiter PUF [16] and variants thereof [21,18].

The Bitline PUF shares many similarities with two particular strong PUFs – the bistable ring PUF [5] and a low power current-based PUF [20]. The similarity to the bistable ring PUF is the use of controllable electrical contention that resolves to one of two states according to variation. The similarity to the current-based PUF is the use of a sense amplifier to detect a differential signal from a controllable set of variation-sensitive elements; the significant difference is that the Bitline PUF uses pre-existing variation-sensitive elements (SRAM cells) and sense amplifiers.

Weak PUFs do not perform challenge-response hashing, but instead function as physically obfuscated keys. Weak PUFs can either use special purpose variation-sensitive circuits or clever ways of detecting variations in existing circuits. Examples of custom-circuit weak PUFs include designs based on variations in drain currents [19], stabilization of cross-coupled devices [35], stabilization of cross-coupled devices in the presence of delay variations [22], and the skewed tendencies of sense amplifiers [3]. Examples of weak PUFs utilizing variations in existing circuitry include ones based on clock skew [14,37] and random flash memory latencies [28].

Several prior works have proposed PUFs based on ordinary or slightly modified SRAM. A common mechanism used by SRAM PUFs is the uniqueness of power-up state [11,7]. The reliability of SRAM power-up state PUFs can be enhanced by detecting and using only cells with large mismatch [10], or by electrically biasing cells to reinforce inherent tendencies [4]. Aside from power-up state, a PUF can be created from ordinary SRAM using unique minimum data retention voltage signatures [12] or failure signatures from attempted writes at low voltages [38]. PUF mechanisms in modified SRAM arrays include unique signatures based on error locations under varied wordline duty cycles [15], and the resolution of SRAM cells under a non-standard metastable write [24]. The significant difference between the Bitline PUF and prior SRAM-based PUFs is that the Bitline PUF generates responses based on mismatch across the cells within a column, instead of just mismatch within a single SRAM cell.

7 Future work

As this work is the first to propose the Bitline PUF, there are many interesting directions that warrant future research. The reliability of Bitline PUF responses with respect to circuit aging should be considered, as well as its susceptibility to cloning attacks [25,9]. For SRAM with asymmetric cells or timing mismatch in the wordline drivers, unbiased challenges may not be those with an equal number of 0 and 1 values, and future work can consider the problem of finding challenges to maximize the uniqueness of responses in these cases. Finally, we will look to fabricate an SRAM with the wordline accumulator circuits that are required for bitline PUF operation, and use data from this implementation to further evaluate the Bitline PUF.

8 Conclusion

This work presents a new PUF design termed the Bitline PUF. The Bitline PUF is a low cost solution that shares most of its circuitry with SRAM, and is created by adding two logic gates and a flip-flop to the wordline driver of each SRAM row to enable challenge-response hashing. The Bitline PUF, applied to a SRAM of typical size, has a response latency of 2.6 μs and response throughput of 99.8 Mbps. Circuit simulation indicates that responses produced by the Bitline PUF in 90 nm technology have a nominal bit error rate of 0.023, and that the bit error rate does not exceed 0.076 for any supply voltage between 1.1 V and 1.3 V, or temperature between 0°C and 80°C.

Acknowledgment: This work was supported in part by C-FAR, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA, and by NSF CNS-1331652. Any opinions, findings, conclusions, and recommendations expressed in these materials are those of the authors and do not necessarily reflect the views of the sponsors.

References

1. Anis, M., and Aburahma, M. H. Leakage current variability in nanometer technologies. In *System-on-Chip for Real-Time Applications, 2005. Proceedings. Fifth International Workshop on* (2005), pp. 60–63.
2. ARM Limited. ARM Security Technology: Building a secure system using trustzone technology. `http://infocenter.arm.com/help/topic/com.arm.doc.prd29-genc-009492c/PRD29-GENC-009492C_trustzone_security_whitepaper.pdf`. Last Viewed June 13, 2014.
3. Bhargava, M., Cakir, C., and Mai, K. Attack Resistant Sense Amplifier Based PUFs (SA-PUF) with Deterministic and Controllable Reliability of PUF Responses. *Hardware-Oriented Security and Trust (HOST), 2010 IEEE International Symposium on* (2010).
4. Bhargava, M., Cakir, C., and Mai, K. Reliability enhancement of bi-stable PUFs in 65nm bulk CMOS. *Hardware-Oriented Security and Trust (HOST), 2012 IEEE International Symposium on* (2012), 25–30.
5. Chen, Q., Csaba, G., Lugli, P., Schlichtmann, U., and Rührmair, U. The Bistable Ring PUF: A new architecture for strong Physical Unclonable Functions. *Hardware-Oriented Security and Trust (HOST), 2011 IEEE International Symposium on* (2011), 134–141.

6. Gassend, B., Clarke, D., and Van Dijk, M. Silicon physical random functions. In *Proceedings of the IEEE Computer and Communications Society* (2002).
7. Guajardo, J., Kumar, S., Schrijen, G., and Tuyls, P. FPGA intrinsic PUFs and their use for IP protection. *Cryptographic Hardware and Embedded Systems* (2007).
8. Heald, R., and Wang, P. Variability in sub-100nm SRAM designs. In *Computer Aided Design, 2004. ICCAD-2004. IEEE/ACM International Conference on* (2004), pp. 347–352.
9. Helfmeier, C., Boit, C., Nedospasov, D., and Seifert, J. P. Cloning Physically Unclonable Functions. *Hardware-Oriented Security and Trust (HOST), 2013 IEEE International Symposium on* (2013), 1–6.
10. Hofer, M., and Boehm, C. An Alternative to Error Correction for SRAM-like PUFs. *Cryptographic Hardware and Embedded Systems* (2010), 335–350.
11. Holcomb, D. E., Burleson, W. P., and Fu, K. Power-up SRAM state as an identifying fingerprint and source of true random numbers. *IEEE Transactions on Computers* (2009).
12. Holcomb, D. E., Rahmati, A., Salajegheh, M., Burleson, W. P., and Fu, K. DRV-Fingerprinting: using data retention voltage of SRAM cells for chip identification. In *RFIDSec'12: Proceedings of the 8th international conference on Radio Frequency Identification: security and privacy issues* (July 2012), Springer-Verlag.
13. Joachims, T. Making large-Scale SVM Learning Practical. In *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, Cambridge, MA, 1999, pp. 169–184.
14. Kohno, T., Broido, A., and Claffy, K. Remote physical device fingerprinting. *Security and Privacy, 2005 IEEE Symposium on* (2005), 211–225.
15. Krishna, A. R., Narasimhan, S., Wang, X., and Bhunia, S. MECCA: A Robust Low-Overhead PUF Using Embedded Memory Array. In *Cryptographic Hardware and Embedded Systems*. 2011, pp. 407–420.
16. Lee, J. W., Lim, D., Gassend, B., Suh, G. E., Van Dijk, M., and Devadas, S. A technique to build a secret key in integrated circuits for identification and authentication applications. *VLSI Circuits, 2004. Digest of Technical Papers. 2004 Symposium on* (2004), 176–179.
17. Lim, D. *Extracting Secret Keys from Integrated Circuit.* PhD thesis, Massachusetts Institute of Technology, May 2004.
18. Lin, L., Holcomb, D. E., Krishnappa, D. K., Shabadi, P., and Burleson, W. P. Low-power sub-threshold design of secure physical unclonable functions. In *ISLPED '10: Proceedings of the 16th ACM/IEEE international symposium on Low power electronics and design* (Aug. 2010).
19. Lofstrom, K., and Daasch, W. IC identification circuit using device mismatch. *International Solid State Circuits Conference* (2000).
20. Majzoobi, M., Ghiaasi, G., Koushanfar, F., and Nassif, S. R. Ultra-low power current-based PUF. *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on* (2011), 2071–2074.
21. Majzoobi, M., Koushanfar, F., and Potkonjak, M. Lightweight secure PUFs. In *Computer-Aided Design, 2008. ICCAD 2008. IEEE/ACM International Conference on* (2008), pp. 670–673.
22. Mathew, S. K., Satpathy, S. K., Anders, M. A., Kaul, H., Hsu, S. K., Agarwal, A., Chen, G. K., Parker, R. J., Krishnamurthy, R. K., and De, V. A 0.19pJ/b PVT-variation-tolerant hybrid physically unclonable function circuit for 100% stable secure key generation in 22nm CMOS. In *Solid-State Circuits*

Conference Digest of Technical Papers (ISSCC), 2014 IEEE International (2014), pp. 278–279.

23. Nii, K., Tsukamoto, Y., Yoshizawa, T., Imaoka, S., Yamagami, Y., Suzuki, T., Shibayama, A., Makino, H., and Iwade, S. A 90-nm low-power 32-kB embedded SRAM with gate leakage suppression circuit for mobile applications. *Solid-State Circuits, IEEE Journal of 39*, 4 (2004), 684–693.
24. Okumura, S., Yoshimoto, S., Kawaguchi, H., and Yoshimoto, M. A 128-bit chip identification generating scheme exploiting SRAM bitcells with failure rate of 4.45 1019. In *Proceedings of the 37th European Solid-State Circuits Conference* (2011), pp. 527–530.
25. Oren, Y., Sadeghi, A.-R., and Wachsmann, C. On the effectiveness of the remanence decay side-channel to clone memory-based PUFs. In *CHES'13: Proceedings of the 15th international conference on Cryptographic Hardware and Embedded Systems* (Aug. 2013), Springer-Verlag.
26. Pappu, R., Recht, B., and Taylor, J. Physical One-Way Functions. *Science* (2002).
27. Pelgrom, M. J. M., Duinmaijer, A. C. J., and Welbers, A. P. G. Matching properties of MOS transistors. *Solid-State Circuits, IEEE Journal of 24*, 5 (1989), 1433–1439.
28. Prabhu, P., Akel, A., Grupp, L., Yu, W., Suh, G., Kan, E., and Swanson, S. Extracting Device Fingerprints from Flash Memory by Exploiting Physical Variations. *Proceedings of the 4th International Conference on Trust and Trustworthy Computing* (2011).
29. Predictive Technology Model. 90nm NMOS and PMOS BSIM4 Models. `http://ptm.asu.edu/modelcard/2006/90nm_bulk.pm`. Last Viewed June 13, 2014.
30. Predictive Technology Model. Interconnect. `http://ptm.asu.edu/interconnect.html`. Last Viewed June 13, 2014.
31. Qazi, M., Tikekar, M., Dolecek, L., Shah, D., and Chandrakasan, A. Loop flattening & spherical sampling: highly efficient model reduction techniques for SRAM yield analysis. In *DATE '10: Proceedings of the Conference on Design, Automation and Test in Europe* (Mar. 2010).
32. Rührmair, U., and Holcomb, D. E. PUFs at a Glance. In *DATE '14: Proceedings of the Conference on Design, Automation and Test in Europe* (Mar. 2014).
33. Rührmair, U., Sehnke, F., Sölter, J., Dror, G., Devadas, S., and Schmidhuber, J. Modeling attacks on physical unclonable functions. In *CCS '10: Proceedings of the 17th ACM conference on Computer and communications security* (2010).
34. Seevinck, E., List, F. J., and Lohstroh, J. Static-noise margin analysis of MOS SRAM cells. *Solid-State Circuits, IEEE Journal of 22*, 5 (1987), 748–754.
35. Su, Y., Holleman, J., and Otis, B. A 1.6 pj/bit 96% stable chip-ID generating circuit using process variations. *International Solid State Circuits Conference* (2007).
36. Suh, G. E., and Devadas, S. Physical unclonable functions for device authentication and secret key generation. In *DAC '07: Proceedings of the 44th annual Design Automation Conference* (2007).
37. Yao, Y., Kim, M., Li, J., Markov, I. L., and Koushanfar, F. ClockPUF: Physical Unclonable Functions based on clock networks. In *DATE '13: Proceedings of the Conference on Design, Automation and Test in Europe* (2013), pp. 422–427.
38. Zheng, Y., Hashemian, M. S., and Bhunia, S. RESP: a robust physical unclonable function retrofitted into embedded SRAM array. In *DAC '13: Proceedings of the 50th Annual Design Automation Conference* (2013).