MULTI-TIERED APPROACH TO IMPROVING THE RELIABILITY OF MULTI-LEVEL CELL PRAM

Chengen Yang, Yunus Emre, Yu Cao and Chaitali Chakrabarti

School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85287
{Chengen.yang,yemre,yu.cao,chaitali}@asu.edu

ABSTRACT

Phase change RAM (PRAM) is a promising memory technology because of its fast read access time, high storage density and very low standby power. Multi-level Cell (MLC) PRAM which has been introduced to further improve the storage density, comes at a price of lower reliability. This paper focuses on a cost-effective solution for improving the reliability of MLC-PRAM. As the first step, we study in detail the causes of hard and soft errors and develop error models to capture these effects. Next we propose a multi-tiered approach that spans architecture, circuit and system levels to increase the reliability. At the architecture level, we use a combination of Gray code encoding and 2-bit interleaving to partition the errors so that a lower strength error correction coding (ECC) can be used for half of the bits that are in the odd block. We use subblock flipping and threshold resistance tuning to reduce the number of errors in the even block. For even higher reliability, we use a simple BCH based ECC on top of these techniques. We show that the propose multi-tiered approach enables us to use a low cost ECC with 2-error correction capability (t=2) instead of one with t=8 to achieve a block failure rate of $10^{-8}$.

Index Terms— Phase change memory, multi-level cell, error correction codes, multi-tiered approach.

1. INTRODUCTION

Phase-change random access memory (PRAM) is a non-volatile memory technology that has many attractive features, including fast read access time, high density, superior scalability, and very low standby leakage[1]. Unlike conventional SRAM and DRAM technologies that use electrical charge to store information, in PRAM, the state information, set or reset, corresponds to the resistance of chalcogenide material in the memory cell (GST). This material can switch between the crystalline phase corresponding to the set or ‘1’ state and the amorphous phase corresponding to the reset or ‘0’ state.

Recently, multiple level cell (MLC) PRAM has been introduced to improve the memory density even further [1]. Unfortunately, MLC type memories have more challenging reliability issues than SLC PRAM because of reduced difference between consecutive resistance levels. In an MLC PRAM, the resistance of an intermediate state drifts to that of a state with higher resistance causing soft errors [3]; these errors increase with data retention time (DRT). Again the resistance of the amorphous state decreases with the number of programming cycles (NPC) and causes hard errors [3].

Error control coding (ECC) has been used for reliable memory operation for decades. Unfortunately direct use of ECC for PRAMs results in large overhead both in terms of area and decoding latency and is not desirable. To reduce the ECC cost during decoding, many architecture-level techniques have been proposed. Techniques to reduce hard errors in SLC PRAM have been presented in [4-7]. Wear leveling techniques and a hybrid memory architecture that reduce the number of write cycles in PRAM have been proposed in [4]. The schemes in [5] and [6] are based on verify-after-write and can identify the locations of hard errors and use schemes based on error partitioning to correct them during read operation. Another scheme [7] uses fine-grained remapping with BCH code and embedded pointers, and can correct up to 6 errors. For correcting soft errors in MLC PRAM, [8] uses a time tag to record the retention time information for each memory block or page and this information is used to determine the threshold resistance that minimizes the soft error bit error rate (BER). However, tuning of threshold resistance for reducing only soft errors has an adverse effect on its hard error rate.

In this paper, we propose a multi-tiered approach to increase the reliability of MLC-PRAM cells with low cost. First, we apply Gray coding and 2-bit interleaving to distribute the odd and even bits of the 2-bit cells and process them separately. Since the error rate in the odd block is lower, it is sufficient to employ a simpler ECC such as Hamming on this block. For the even block, we propose to first use sub-block flipping to reduce the number of hard errors. Next, we use threshold resistance tuning to further reduce the soft and hard error rates. We show that there is an optimal threshold resistance for a given data retention time and number of programming cycles and this is different for different levels of storage. Application of all these techniques reduces the error rate significantly and so it is now sufficient to use an ECC with lower error correction capability, resulting in lower hardware overhead. Alternately, the proposed technique can also be used to improve the lifetime of the device with only a small increase in the overhead.

The rest of the paper is organized as follows. Section 2 describes the operation of SLC and MLC PRAM. The causes of soft errors and hard errors are given in Section 3. Section 4 describes the 2-bit interleaving and subblock flipping techniques followed by Section 5 which details the effect of threshold resistance on the total error rate. Section 6 describes the BCH-based ECC techniques and compares the performance and hardware overhead of the candidate schemes. Section 7 concludes the paper.

This work was supported in part by a grant from NSF, CSR 0910699.
2. BACKGROUND

2.1. Single Level Cell (SLC) PRAM

A SLC PRAM consist of two states, SET state corresponding to the low resistance crystalline phase or state ‘1’, and RESET state corresponding to the high resistance amorphous phase or state ‘0’. As shown in Figure 1(a), for RESET operation, a large current is passed through the top and bottom electrode that heats the programmable region over its melting point. This is followed by a rapid quench which turns this region into an amorphous state. For SET, a lower current pulse is applied for a longer period of time so that the programmable region is at a temperature that is slightly higher than the crystallization transition temperature. For READ, the read voltage is set to be sufficiently high to provide a current that can be sensed by a sense amplifier but low enough to avoid write disturbance [1].

Figure 1. (a) PCM cells are programmed and read by applying electrical pulses with different characteristics. (b) Hspice simulation model for representing the programming process.

To simulate the programming process of a PRAM cell, an Hspice model has been developed as shown in Figure 1(b). Here $R_1$ and $R_2$ are the thermal resistances of amorphous and crystalline phases, $R_t$ is the thermal resistance that equals to $R_A$ or $R_C$, and $C_t$ is the thermal capacitance of the PRAM cell. The voltage of the thermal capacitor, $V_{th}$, indicates the phase of ST material. The supply voltage $V_{dd}$ is set to 1V, so $V_x$ is between 1 and 0.

The electrical resistance of PRAM cell, $R$ can be modeled as a voltage control resistor (VCR); it is control by $V_x$ during transition from one state to another. $R$ can be represented as

$$R = R_c + (R_A - R_c)V_x$$

where $R_A$ and $R_c$ are the electrical resistance of amorphous and crystalline phase, respectively.

For SLC PRAM, the capacitance switch is controlled by the programming current. The capacitance is connected to $R_t$ when the input current is $I_{set}$, $V_x$ then equals to $V_{th}=1$ and $R$ equals to $R_A$. The capacitance is connected to $R_c$ when the input current is $I_{set}$. In this case, $V_x$ is drawn back to zero, and $R$ equals to $R_c$ for the crystalline phase.

2.2. Multiple level cell (MLC) PRAM

For MLC PRAM, the simulation model of SLC PRAM in Figure 1(b) can still be utilized. Note that while for SLC PRAM, the value of $V_x$ can only be set to ‘1’ or ‘0’ corresponding to amorphous or crystalline phase, for MLC PRAM, $V_x$ is set to an intermediate value between 0 and 1. The resistance $R$ in equation(1) can be then used to represent a mixture of amorphous and crystalline phase resistances, corresponding to the resistance of the intermediate states of MLC PRAM.

A 2-bit MLC PRAM consist of 4 states, where ‘00’ is full amorphous state, ‘11’ is full crystalline state, ‘01’ and ‘10’ are two intermediate states. The corresponding finite state machine (FSM) for modeling the WRITE strategy of a 2-bit MLC is shown in Figure 2 [9]. To go to ‘11’ state, a ramp down SET pulse is applied. To go to ‘00’ state from a ‘01’ or ‘10’ state, it first transitions to ‘11’ state to avoid over programming, and then to ‘00’ state. To write ‘01’ or ‘10’, it first transitions to ‘00’ state and then to the final state using several sequential short pulses. In this paper, the static parameters used in the simulation of 2bit MLC PRAM are listed in Table 1.

![Figure 2. Finite state machine of MLC PRAM.](image)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>$R_{00}$</th>
<th>$R_{10}$</th>
<th>$R_{11}$</th>
<th>$R_{11}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>2300</td>
<td>46</td>
<td>15</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 1. Simulation parameters of 2bit MLC PRAM(KΩ)

3. ERROR MODELS

The reliability of a PRAM cell can be represented by its data retention capability and cycling endurance [3]. Data retention depends on the stability of the resistance in the crystalline and amorphous phases. While the crystalline phase is fairly stable with time and temperature, the resistance of amorphous phase increases due to structure relaxation (SR) [10]. The impact of resistance drift due to SR can be annulled in the next programming cycle, thus this component only causes soft errors.

Hard errors occur when the data value stored in one cell cannot be changed in the next programming cycle. There are two types of hard errors in PRAM: stuck-RESET failure and stuck-SET failure [3]. Stuck-SET or stuck-RESET means the value of stored data in PCM cell is stuck in SET or RESET state no matter what value has been written into the cell. These errors increase as the NPC increases. For SLC PRAM, most of the failures are stuck-SET failure [3].

The distributions of the intermediate states (‘01’ and ‘10’) are shaped by the multiple-step programming strategy. There are three threshold resistances $R_{01(11,10)}$, $R_{01(10,01)}$, and $R_{01(01,00)}$ to identify the boundaries between the four states. These resistance can be changed by tuning the reference current of the differential current amplifier during read sensing as has been demonstrated in MLC Flash memory architectures in [11]. Due to the change in the material characteristics such as SR or re-crystallization, the resistance distribution of logical states shifts from the initial position. Memory cells fail when the distribution crosses the threshold resistance level; the error rate is proportional to the extent of overlap. In this paper, we assume that the initial resistance distribution is Gaussian. The mean values of the
resistances have been listed in Table 1 and the deviation is 0.17 as used in [8].

![Figure 3. Soft error mechanism of MLC PRAM.](image)

A simple model has been built to model resistance drift effect of PRAM. Since $R_A$ represent the amorphous active region exclusively, let $R_e$ represent the impact of all the other resistances. Then, MLC PRAM time dependent resistance is given by

$$R(t) = R_A + R_e \cdot e^{-\nu t}$$  \hspace{1cm} (3)$$

where $R_A$ and $R_e$ are varying and $\nu$ is the resistance drift coefficient, which is constant for all the intermediate states. Measured data from [12] almost match the simulated data. In this paper, $\nu$ is set to 0.11, a typical value which has been used in [8] and [10], and the standard deviation to mean ratio is 40% as defined in [8]. Based on the initial resistance in Table 1, $R_A$ and $R_e$ in this paper are listed in Table 2.

Figure 3 describes the two mechanisms that result in soft errors. The error rate due to state ‘10’ crossing $R_{th}(01,00)$ and state ‘01’ crossing $R_{th}(00,01)$ depends on the distributions of the resistances of states ‘10’ and ‘01’ and the values of $R_{th}(01,00)$ and $R_{th}(00,01)$. Increasing $R_{th}(00,01)$ results in larger reduction in the soft error rate, as will be shown later.

<table>
<thead>
<tr>
<th>Table 2. Parameters of resistance drift model</th>
</tr>
</thead>
<tbody>
<tr>
<td>State 00</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>$R_A$ ($\Omega$)</td>
</tr>
<tr>
<td>$R_e$ ($\Omega$)</td>
</tr>
</tbody>
</table>

Stuck-SET failure is due to repeated cycling that leads to Sb enrichment at the bottom electrode [3]. Sb rich materials have a lower crystallization temperature leading to data loss and crystallization of the region above the bottom electrode at much lower temperatures than the original material composition. As a result, the bottom electrode cannot heat the GST material sufficiently, and the resistance is lower than the desired level of reset state. The resistance drop can be analyzed as an exponential function of the NPC $N$ and is given by $\Delta R = aN^b$. The resistance drop model of ‘00’ state have been compared with measured data from [13]. It shows that this model is fairly accurate; here a equals 151601 and b equals 0.16036. Since the resistance of intermediate states of MLC PRAM are guaranteed by read and verify steps in the write operation, the hard error mechanism of MLC PRAM is the same as that of SLC PRAM. Figure 4 shows how the resistance of ‘00’ state drops over time. When the resistance distribution of state ‘00’ crosses $R_{th}(01,00)$, hard error occurs.

![Figure 4. Hard error mechanism of MLC PRAM.](image)

4. ARCHITECTURAL TECHNIQUES LTS

4.1 2-Bit interleaving

In section 3, major causes of hard and soft errors of 2bit MLC PRAM have been described. We see that the resistance drift of ‘10’ state to ‘01’ state and resistance drift of ‘01’ state to ‘00’ state causes soft errors and resistance drift of ‘00’ state to ‘01’ state causes hard errors. We propose a scheme based on combination of Gray code based encoding and 2-bit interleaving that helps partition these errors so that a lower strength ECC can be used for at least half of the bits.

By using Gray code based encoding for a 2bit MLC, the mapping of ‘00’ and ‘01’ remains the same, but ‘10’ is now mapped to ‘11’ and ‘11’ is mapped to ‘10’. Thus errors due to resistance drift of states ‘10’ to ‘01’ translate to resistance drifts of states ‘11’ to ‘01’. Now with 2-bit interleaving, all errors are now localized in the most significant bit or the ‘odd’ bit. Similarly the errors due to resistance drift of ‘00’ to ‘01’ that causes hard errors and resistance drift of ‘01’ to ‘00’ that causes soft errors are localized in the least significant bit or ‘even’ bit.

![Figure 5. Encoding flow of 2-bit interleaving technique.](image)

This feature is exploited in the proposed method, where during encoding, the n bits are split across two blocks each of size n/2. The odd block contains all the odd bits while the even block contains all the even bits as shown in Figure 5. The data in the odd block contain fewer errors and are encoded by a simple ECC scheme, such as Hamming code. The data in the even block contain larger number of errors and so we propose to use stronger ECC such as BCH(274,256) with capability to correct 2 errors ($t=2$). Now, the errors in the even block are low in the beginning and increase with NPC while the errors in the odd block are always low. So for the even block, when the Hamming code can no longer guarantee a specific error rate after decoding, we switch to a stronger ECC as shown in Fig. 5. Implementation results in Section
Subblock flipping technique flips part of the information block or subblock, after verify-after-write process in the write operation. The flipped subblock is marked by a ‘flag bit’ as ‘flipped’, and encoded again before write back to memory. Subblocks marked as “flipped” are flipped back before decoding in the read operation. Subblock flipping eliminates visible hard errors which are stuck at the opposite value of what has been written in and therefore can be detected by verify-after-write process. The invisible hard errors are stuck at the same value of what has been written in and cannot be found by verify-after-write. In [5], all the ‘visible’ errors are found during verify-after-write. An iterative algorithm partitions the information data into multiple subblocks such that in the end there is only a single error in each of them.

In the proposed subblock flipping technique, we partition the information bits into fixed number of subblocks before verify-after-write. After verify-after-write, the subblock is flipped when the first hard error is found in this subblock. If there is a single hard error in one subblock, subblock flipping can eliminate the visible hard error. In a multiple hard error case, invisible hard errors can be flipped to visible errors, and some visible errors eliminated by subblock flipping. For instance, if two hard errors have equal probability of being visible or invisible, then after subblock flipping, the average number of visible errors that are still left is 0.5 and thus the hard error rate is reduced by 50%.

We consider 3 subblock partitioning schemes: 8 subblocks each of size 32 bits, 4 subblocks each of size 64 bits and 2 subblocks each of size 128 bits. The hard error rates after subblock flipping for the three schemes are shown in Figure 6. All the three candidate subblock schemes provide two decades hard error rate reduction after subblock flipping for a one decade reduction in the raw error rate, the 32bits*8 scheme has 0.5db lower BER compared to the other schemes.

The overhead of subblock flipping comes from the extra hardware that is required to perform this technique. These include XOR gates that are used to flip data, extra storage of the flag bits, and the write latency and energy of the 2nd write. Overhead of the 2nd write can be estimated based on its probability of occurrence. The extra latency due to the 2nd write is the block failure rate (BFR) of the 256 bit block and is given by

\[ L_{2nd} = 1 - (1 - BER_{raw})^{256} \]  
(3)

The increase in energy due to the 2nd write is the BFR of the subblock multiplied by the number of subblocks for every set of 256 bits. It is given by

\[ E_{2nd} = \left( 1 - (1 - BER_{raw}) \right)^2 p, \]  
(4)

where \( p \) is the number of subblocks. According to equation(3) and (4), \( E_{2nd} \) and \( L_{2nd} \) are about 2.5% when the raw hard error rate is \( 10^{-3} \). The \( E_{2nd} \) of [5] equals to \( BER_{raw}*N/p \), which is larger than that of proposed scheme.

5. CIRCUIT-LEVEL TECHNIQUE

In section 3, we have shown that the soft error rate increases with DRT and that the hard error rate increases with the NPC. In this section, we show how the error rate can be controlled by tuning the threshold resistance \( R_{th(01,00)} \) for a specific DRT. Recall that threshold resistance can be tuned by changing the current reference of the sense amplifier as in [11]. DRT is different for different type of memory. Here we consider a range of DRT values from 10 sec to 10^4 sec.

5.1. Soft error rate

The soft error rate of 2bit MLC PRAM is a function of the resistance drift of ‘01’ to ‘00’ state, \( Es(‘01’->‘00’) \), and the resistance drift of ‘10’ to ‘01’ state, \( Es(‘10’->‘01’) \). While \( Es(‘01’->‘00’) \) depends on the value of \( R_{th(01,00)} \), \( Es(‘10’->‘01’) \) depends on the value of \( R_{th(10,01)} \). Note that after Gray code encoding, the ‘10’ state and ‘11’ state are switched.

\[ \text{Figure 7. } Es(‘10’->‘01’ \text{ and } Es(‘01’->‘00’) \text{ increase with data retention time.} \]

Figure 7 describes how the soft error rate increases with data retention time.

Figure 7 shows how the so ft error rate increases with DRT for different values of \( R_{th(01,10)} \). Here \( R_{th(01,10)} \) is set as the middle value between resistances of ‘01’ and ‘10’ states, which is 30.5K \( \Omega \) in this case. Tuning this resistance is difficult because of the close spacing between the distributions of the ‘01’ and ‘10’ states. This scenario, however, \( R_{th(01,10)} \) has a much higher impact on the total soft error rates; as \( R_{th(01,10)} \) increases, the soft error rate reduces. Threshold resistance tuning is an effective way of reducing the soft error rate. A technique to record the DRT for every memory block and then using this to tune the threshold resistances between all the intermediate states has been proposed in [8].

5.2. Hard error rate

The hard error rate of 2bit MLC PRAM is due to the resistance drop of ‘00’ state to the ‘01’ state as shown in Figure 8. It is a function of \( R_{th(01,00)} \), and the resistance distribution of state 00. Due to multiple pulse write strategy for intermediate states, there is no resistance drop from ‘01’ state to ‘10’ state, and thus \( R_{th(10,01)} \) has no impact on the hard error rate.
5.3 Total error rate

Consider scenarios where the NPC is $10^{5.5}$ and the DRT are $10^4 \sim 10^5$ seconds. Since both the hard error and soft error rates are a function of $R_{th(01,00)}$, we combine the two error rates for DRT of $10^4$ seconds in Figure 9 and present them as a function of $R_{th(01,00)}$. We see that while the hard error rate increases monotonically, the soft error rate curve decreases at first and then becomes constant. Soft error rate keeps decreasing till a critical $R_{th(01,00)}$ is reached, which is 480KΩ in this case. It then maintains a constant value which is determined by the error rate $E_s$ ($10^{-01}$). From the plot we see that if subblock flipping is not done, the lowest total error corresponding to $10^4$ DRT is $2\times10^{-5}$ at $R_{th(01,00)}$ of 300KΩ. This error rate drops to $6.2\times10^{-8}$ with subblock flipping at $R_{th(01,00)}$ of 410KΩ. Figure 11 also shows the error reduction proportion of subblock flipping and $R_{th(01,00)}$ tuning compared to a randomly fixed $R_{th(01,00)}$ case for an NPC of $10^{5.5}$.

For a specific DRT (given by soft error rate curve), the optimal $R_{th(01,00)}$ reduces as the NPC increases. Figure 10 provides the lowest error rate values as a function of optimal $R_{th(01,00)}$ for four DRTs. As the DRT increases, the error rate increases, as expected. Nevertheless threshold resistance tuning provides an easy way of achieving the lowest possible total error rate considering both soft and hard errors. In the next section we show how ECC can be used in conjunction with threshold resistance tuning and subblock flipping to achieve even lower error rates.

6. ECC SCHEMES

6.1 ECC performance

In this section we consider block failure rate (BFR) as the performance metric since it represents the memory performance more accurately compared to bit error rate.

$$BFR = P(error > t) = \sum_{i=t+1}^{N} \binom{N}{i} BER^i (1 - BER)^{N-i}$$ (5)

where t is the correction strength of the ECC, BER represents the bit error rate, which is the input of ECC. In this paper, the target BFR corresponding to the expected lifetime is set to $10^{-8}$. Figure 11 shows the error reduction proportion of subblock flipping and $R_{th(01,00)}$ tuning compared to a randomly fixed $R_{th(01,00)}$ case for an NPC of $10^{5.5}$.

Next we analyze the lifetime of two schemes that achieve the same target BFR. Scheme 1 uses $R_{th(01,00)}$ tuning and a BCH code of large t and Scheme 2 uses $R_{th(01,00)}$ tuning, subblock flipping and a BCH code of small t. From Figure 11, we see that for a certain DRT value, the BER is a function of NPC and $R_{th(01,00)}$. Thus, for the same target BFR, the input BER of Scheme 1 and Scheme 2 correspond to the same NPC values, described by point A and point B in Figure 11. If Scheme 2 uses BCH of $t=2$, its corresponding input BER is $1.55\times10^{-3}$ and if Scheme 1 uses $t=8$, its input BER is $1.13\times10^{-3}$. Next, we will show that Scheme 1 and Scheme 2 have the same life time and that points A and B are very close.

Figure 12 describes the procedure of finding lifetime in terms of NPC. The minimum BER in Figure 9 is the sum of its soft error rate and hard error rate and so the soft error rate and hard error rate in Figure 9 are both half of the minimum BER. Since soft error rate is determined by $R_{th(01,00)}$, we use the right half of Figure 12 to determine $R_{th(01,00)} = 230KΩ$ corresponding to soft error rate of Scheme 1 which is $5.65\times10^{-4}$. The left half of Figure 12 plots hard error rate as a function of NPC for different values of $R_{th(01,00)}$. From the hard error rate plot of $R_{th(01,00)} = 230KΩ$, and hard error rate $5.65\times10^{-4}$, we see that the life time of this scheme is $10^{6.5}$ NPC, which corresponds to point A in Figure 11. Similarly for Scheme 2, the soft error rate, which is half of the minimum BER,
is $7.75 \times 10^6$ corresponding to $R_{\text{p0}(01,00)} = 310 \text{K}\Omega$. From the left half of Figure 12, we see that the life time of Scheme 2 is also $10^6$ NPC, which corresponds to point B in Figure 11. Thus, it is reasonable to compare the implementation overhead of Scheme 1 and Scheme 2 since they both have the same lifetime.

Since the soft error rate of different storage levels vary due to different DRTs, the obtained lifetime values also vary. The lifetime values for DRT of $10^3$, $10^5$ and $10^6$ are $10^6$ cycles, $10^5$ cycles, and $10^4$ cycles, respectively. Thus for the same ECC scheme, lower DRT results in longer lifetime in terms of NPC. This is because lower DRT has lower soft error rate and thus the ECC can correct more hard errors that occur because of increase in NPC.

### 6.2 Hardware overhead

In this section we compare the hardware overhead of Schemes 1 and 2. Scheme 2 uses Hamming (266, 256) and BCH(274,256) while Scheme 1 uses BCH(592,512). The two schemes have been synthesized in 45nm technology using Nangate cell library [14] and Synopsys Design Compiler [15]. BCH decoders use pipelined simplified inverse-free Berlekamp-Massey (SiBM) algorithm. The 2t-folded SiBM architecture [16] is used to minimize the circuit overhead of Key-equation solver at the expense of increase in latency. A parallel factor of 8 is used for the syndrome calculation and Chien search blocks. From the synthesis comparison in Table 3, we can see that while the area and additional storage rate of Schemes 1 and 2 are comparable, the latency and the energy of Scheme 2 is about 43% and 12% of Scheme 1. Scheme 2 has lower DRT results in longer lifetime in terms of NPC. This is because lower DRT has lower soft error rate and thus the ECC can correct more hard errors that occur because of increase in NPC.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Energy (pJ)</th>
<th>Latency (ns)</th>
<th>Area ($\mu$m$^2$)</th>
<th>Additional Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scheme 1 BCH(t=8)</td>
<td>116</td>
<td>89.5</td>
<td>5772</td>
<td>13.5%</td>
</tr>
<tr>
<td>Scheme 1 BCH(t=8)</td>
<td>1.2</td>
<td>39</td>
<td>2358</td>
<td>6.5%</td>
</tr>
<tr>
<td>Hamming</td>
<td>4.1</td>
<td>1.8</td>
<td>3111</td>
<td>3.7%</td>
</tr>
<tr>
<td>Total</td>
<td>14.3</td>
<td>39</td>
<td>5470</td>
<td>12.3%</td>
</tr>
</tbody>
</table>

The latency of Scheme 2 is significantly lower than that of Scheme 1. This is mostly due to use of bit interleaving which causes the ECC to operate on 256 bits instead of 512 bits. Since the block size is now 256 bits, it is sufficient to use BCH(274,256) which works in Galois Field($2^t$) instead of Galois Field($2^{10}$) used for operating on 512 bits. Accordingly, the critical path is reduced from 0.72ns to 0.65ns. The reduction of energy is partly due to latency reduction but mostly due to use of BCH with lower t. As shown in Figure 6, subblock flipping enables us to use t=2 instead of t=8 to achieve the same BFR. In BCH decoder, the hardware overhead of parallel syndrome calculation and Chien search is proportional to t. The number of registers in the 2t-fold SiBM Key-equation solver is 4t. Thus, subblock flipping at architecture level contributes to significant reduction in the energy consumption.

### 7. CONCLUSION

In this paper we describe a multi-tiered approach that spans architecture-level, circuit-level and system-level techniques to improve the reliability of 2bit MLC-PRAM. At the architecture level, we apply Gray code based encoding and 2-bit interleaving to partition the information bits into odd and even blocks that have different error characteristics. We show that the odd block has fewer errors and require a simple ECC scheme to guarantee reliability. For the even block, we propose use of another architecture-level technique, namely, subblock flipping and a circuit-level technique based on threshold resistance tuning to reduce the errors. Application of these techniques reduces the error rate significantly and it is now sufficient to use a lower cost ECC scheme. We propose use of BCH (274,256) at the system level to achieve block failure rate of $10^{-8}$ for original block size of 512 bits. This code has a t=2 error correction capability compared to the t=8 code that would be required otherwise to achieve the same block failure rate. Thus the proposed multi-tiered approach is a very cost-effective solution for increasing the reliability of MLC-PRAM.

### REFERENCES


