Accurate Statistical Soft Error Rate (SSER) Analysis Using A Quasi-Monte Carlo Framework With Quality Cell Models

Yu-Hsin Kuo, Huan-Kai Peng, and Charles H.-P. Wen
Department of Electrical Engineering
National Chiao Tung University, Hsinchu, Taiwan
E-mail: {rainstar.cm97g@nctu.edu.tw, pumbaapeng@gmail.com, opwen@g2.nctu.edu.tw}

Abstract— For CMOS designs in sub 90nm technologies, statistical methods are necessary to accurately estimate circuit SER considering process variations. However, due to the lack of quality statistical models, current statistical SER (SSER) frameworks have not yet achieved satisfactory accuracy. In this work, we present accurate table-based cell models, based on which a Monte Carlo SSER analysis framework is built. We further propose a heuristic to customize the use of quasirandom sequences, which successfully speeds up the convergence of simulation error and hence shortens the runtime. Experimental results show that this framework is capable of more precisely estimating circuit SSERs with reasonable speed.

I. INTRODUCTION

Formerly only concerned in memory, soft errors have emerged to be one of the major failure mechanisms for logic circuits in sub-90nm technologies. Such errors result from radiation-induced transient faults latched by state-holding elements, which depends on three masking effects [1]: logical, electrical and timing maskings. As shown in Figure 1, logical masking occurs when the input value of one cell blocks the propagation of the transient fault under one input pattern. One transient fault attenuated by electrical masking may further disappear due to cell’s electrical properties. Timing masking occurs when the survival transient faults arrives one state-holding element outside its window of clock transition.

Subject to the three mechanisms, numerous researches are presented to evaluate soft error for logic circuits. The work in [3] propagates transient faults through one gate according to the logic function and meanwhile uses analytical models to electrically evaluate the change of transient faults. A refined model is presented in [4] to incorporate non-linear transistor current, which is further applied to all gates with different charges deposited. A static analysis is also proposed in [10] for timing masking by computing backwards the propagation of the error-latching windows efficiently.

Consequently, soft error rate (SER) has become a key metric for circuit reliability and been extensively investigated. SERA[5] computes SER by means of a wave-
and statistical-learning-based frameworks for statistical SER (SSER) analysis, their SSER results are not accurate enough, where the main challenge comes from the difficulty of constructing quality cell models for transient-fault distributions.

In this work, we first build accurate table-based models for transient-fault distributions, according to which a Monte Carlo SSER analysis framework is built. Further, we propose a heuristic to customize the use of quasirandom sequences, which successfully speed up the convergence of simulation error and hence shorten the runtime. From the experimental results, the framework is capable of yielding more accurate SSER results compared to previous works with reasonable speed.

The rest of this paper is organized as follows. In Section II, we present the SSER analysis framework. In Section III, the generation of our table-based cell models is detailed. Then, we propose a heuristic of using quasirandom sequences to speed up the framework in Section IV. Section V describes the experimental results, including the accuracy of our models, the Monte Carlo convergence and statistical-learning-based frameworks for statistical fault distributions.

II. SSER Analysis Framework

In this section, we describe the SSER analysis framework that considers process-variation impacts for cell-based designs. The proposed framework is illustrated in Figure 3 and mainly consists of four stages: (1) cell modeling, (2) electrical probability computation, (3) signal probability computation and (4) SER estimation. A stage-by-stage explanation of each component will start reversely from SER estimation to cell modeling.

A. SER estimation

We will first introduce the estimation of the overall SER in our framework. The overall SER for the circuit under test (CUT) can be computed by summing up the SER’s of each individual node in the circuit. That is,

\[
\text{SER}_{\text{CUT}} = \sum_{i=0}^{N_{\text{node}}} \text{SER}_i
\]  

(1)

where \(N_{\text{node}}\) is the total number of possible nodes to be struck by radiation particles in the CUT.

Each \(\text{SER}_i\) can be further formulated by integrating over the range \(q = 0\) to \(Q_{\text{MAX}}\) the products of particle-hit rate and the probability that a soft error can survive. Therefore,

\[
\text{SER}_i = \int_{q=0}^{Q_{\text{MAX}}} (R(q) \times P_{\text{soft-err}}(i, q))dq
\]  

(2)

B. Signal probability computation

\(P_{\text{soft-err}}(i, q)\) represents the probability that a transient fault originated from the particle of charge \(q\) at node \(i\) can result in one soft error at any flip-flop. \(R(q)\) represents the effective frequency for a particle hit of charge \(q\) in unit time according to \([1][5]\). That is,

\[
R(q) = F \times K \times A \times \frac{1}{Q_s} \times \exp(-\frac{q}{Q_s})
\]  

(3)

where \(F\), \(K\), \(A\) and \(Q_s\) denote the constants for neutron flux (> 10 MeV), the technology-independent fitting parameter, the susceptible area in \(cm^2\) and the charge collection slope, respectively.

Here \(P_{\text{soft-err}}(i, q)\) represents the probability that a transient fault originated from the particle of charge \(q\) at node \(i\) can result in one soft error at any flip-flop. \(R(q)\) represents the effective frequency for a particle hit of charge \(q\) in unit time according to \([1][5]\). That is,

\[
R(q) = F \times K \times A \times \frac{1}{Q_s} \times \exp(-\frac{q}{Q_s})
\]  

(3)

where \(F\), \(K\), \(A\) and \(Q_s\) denote the constants for neutron flux (> 10 MeV), the technology-independent fitting parameter, the susceptible area in \(cm^2\) and the charge collection slope, respectively.

Here \(P_{\text{soft-err}}(i, q)\) represents the probability that a transient fault originated from the particle of charge \(q\) at node \(i\) can result in one soft error at any flip-flop. \(R(q)\) represents the effective frequency for a particle hit of charge \(q\) in unit time according to \([1][5]\). That is,

\[
R(q) = F \times K \times A \times \frac{1}{Q_s} \times \exp(-\frac{q}{Q_s})
\]  

(3)

where \(F\), \(K\), \(A\) and \(Q_s\) denote the constants for neutron flux (> 10 MeV), the technology-independent fitting parameter, the susceptible area in \(cm^2\) and the charge collection slope, respectively.

B. Signal probability computation

\(P_{\text{soft-err}}(i, q)\) depends on all three masking effects and can be further decomposed into

\[
P_{\text{soft-err}}(i, q) = \sum_{j=0}^{N_{\text{ff}}} P_{\text{logic}}(i, j) \times P_{\text{elec}}(i, j, q)
\]  

(4)

where \(N_{\text{ff}}\) denotes the total number of flip-flops in the circuit under test. \(P_{\text{logic}}(i, j)\) denotes the overall signal probability of propagating the transient faults through all cells along the path from node \(i\) to flip-flop \(j\). It can be computed by multiplying the signal probabilities of all cells as follows.

\[
P_{\text{logic}}(i, j) = \prod_{k \in i \rightarrow j} P_{\text{sig}}(k)
\]  

(5)

where \(k\) denotes one node on the path \(i \rightarrow j\) and \(P_{\text{sig}}(k)\), accordingly, denotes the probability that all input signals of node \(k\) jointly determine such that the transient fault is not logically masked on this path.
The handling of reconvergent fanout nodes (RFONs) is an issue of computing signal probability whereas omitting it may cause considerable error [15]. In this work, a linear-time algorithm, dynamic weighted averaging algorithm (DWAA), is employed to consider the RFON effect and fix the signal probability. The main idea behind DWAA is to consider the dependency of signals between the fanout cone and the reconvergent node by forcing the reconvergent signals to the value corresponding to their respective fanins. More details of DWAA can be referred to [15].

C. Electrical probability computation

Electrical probability $P_{\text{elec}}(i, j, q)$ considers the electrical and timing masking effects and can be defined as

$$P_{\text{elec}}(i, j, q) = P_{\text{err-latch}}(pw_j, w_j) = P_{\text{err-latch}}(\lambda_{\text{elec-mask}}(i, j, q), w_j) \quad (6)$$

where $P_{\text{err-latch}}$ is defined as follows.

Definition (P_{\text{err-latch}}, error-latching probability)

Assume that the pulse width of one arrival transient fault and the latching window ($t_{\text{setup}} + t_{\text{hold}}$) of one flip-flop are random variables and denoted as $pw$ and $w$, respectively. Let $x = pw - w$ be a new random variable where $\mu_x$ and $\sigma_x$ are its mean and variance.

$$P_{\text{err-latch}}(pw, w) = \frac{1}{t_{\text{clk}}} \int_{0}^{\mu_x + 3\sigma_x} x \times P(x > 0) \times dx \quad (7)$$

While $P_{\text{err-latch}}$ accounts for the timing masking effect, $\lambda_{\text{elec-mask}}$ accounts for the electrical masking effect with the following definition.

Definition ($\lambda_{\text{elec-mask}}$, electrical masking function)

Given the node $i$ where the particle strikes to cause a transient fault and flip-flop $j$ is the destination that the transient fault finally ends at, assume that the transient fault propagates along one path $i \sim j$ through $v_0, v_1, \ldots, v_m, v_{m+1}$ where $v_0$ and $v_{m+1}$ denote node $i$ and flip-flop $j$, respectively.

$$\lambda_{\text{elec-mask}}(i, j, q) = \delta_{\text{prop}}(\cdots \delta_{\text{prop}}(\delta_{\text{prop}}(pw_0, 1), 2), \cdots, m) \quad (8)$$

where $pw_0 = \delta_{\text{strike}}(q, i)$.

In Equation (8), $\delta_{\text{strike}}$ and $\delta_{\text{prop}}$, respectively, represent the first-strike function and the propagation distribution function of transient faults.

D. Cell modeling

Since $\delta_{\text{strike}}$ and $\delta_{\text{prop}}$ are both non-linear functions of distributions, they are non-deterministic in nature and can only be only approximated by efficient and accurate models $M_{\text{strike}}$ and $M_{\text{prop}}$. As detailed in the next section, they are also the most critical components for an accurate SSER analysis framework due to the difficulty from integrating process-variation impacts.

III. Table-based Statistical Models

$M_{\text{strike}}$ and $M_{\text{prop}}$ are respectively the generation and propagation models of $pw$ that is a random variable. According to [11], $pw$ follows the normal distribution, which can be written as:

$$pw \sim N(\mu_{pw}, \sigma_{pw}) \quad (9)$$

Therefore, we decompose $M_{\text{strike}}$ and $M_{\text{prop}}$ into four models: $M_{\text{strike}}^\mu$, $M_{\text{strike}}^\sigma$, $M_{\text{prop}}^\mu$, and $M_{\text{prop}}^\sigma$ where each can be defined as:

$$M : \vec{x} \rightarrow y \quad (10)$$

where $\vec{x}$ denotes a vector of input variables and $y$ is called the model’s label or target value. For $M_{\text{strike}}^\mu$ and $M_{\text{strike}}^\sigma$, we use input variables including charge strength, driving gate, input pattern, and output loading. For $M_{\text{prop}}^\mu$ and $M_{\text{prop}}^\sigma$, we use input variables including input pattern, pin index, driving gate, input pulse-width distribution ($\mu_{pw,i}^{-1}$ and $\sigma_{pw,i}^{-1}$), propagation depth, and output loading.

To build these models, a traditional approach is to construct tables according to manually-selected corner cases. However, such approach has two difficulties: first, these models have a lot of input variables so that their combinations enumerating all corner cases are prohibitively expensive. Second, input variables such as input pulse-width distribution are dependent variables in nature, which cannot be specified directly according to pre-selected combinations. Therefore, we use a different approach, as shown in Figure 4, consisting of 3 steps: random sample generation, table fill-up, and table lookup.

A. Random sample generation

We use a unified Monte Carlo SPICE simulation framework to build the two kinds of models ($M_{\text{strike}}$ and $M_{\text{prop}}$) of distinct mapping spaces, as illustrated by Step 1 of Figure 4. The framework first generates a random path loaded with additional random cells. A charge is then injected as a current source at the beginning of the path according to the following equation [4]:

$$I(q,t) = \frac{q}{\tau_{\alpha} - \tau_{\beta}} \times (e^{-\frac{t}{\tau_{\alpha}}} - e^{-\frac{t}{\tau_{\beta}}}) \quad (11)$$

In each Monte Carlo instance, the pulse-width distributions are recorded along the path, which are later collected separately for different models. Note that this framework can be applied to all sources of process variations, as long as each of their impacts can be reflected using SPICE simulation. Also, to build accurate models, it is essential to acquire sufficiently large amount of samples in this step; in our case, for example, 500K.
B. Table fill-up

In Step 2 of Figure 4, we classify all samples according to their corresponding input variables to fill up the tables. For discrete variables such as charge strength, driving gate, input pattern, pin index, propagation depth, and output loading (in terms of equivalent-INVs), this can be done directly, which is like having multiple slices of tables, as illustrated in Figure 4.

For continuous variables such as the width and height of input pulse, however, we must discretize them to form a number of table cells. It can be done through determining the upper/lower bounds and the number of partitions. For the two bounds, we use the MIN and MAX values of samples sharing the same discrete input variable combination. For the number of partitions, there is a trade-off between table resolution and size: with sufficient samples, a larger number of partitions leads to finer table resolution and accuracy, in expense of a larger table size.

To achieve the balance the table size and resolution, an estimate of the table error is:

\[
\text{MEAN}_{C_i \in \text{all cells}} \left( \frac{\text{MAX}(C_i) - \text{MIN}(C_i)}{\text{MEAN}(C_i)} \right) \leq \hat{\epsilon}
\]

(12)

\(C_i\) represents the samples within a specific cell; \(\hat{\epsilon}\) represents the error rate threshold. MAX, MIN, and MEAN respectively represent the maximum, minimum, and mean of the sample labels of \(C_i\). We iteratively increase the number of partitions and calculate the mean error estimate until it falls below the target threshold. In our case, we found good accuracy can be reached with the number of partitions no more than 25 for all tables.

C. Table lookup

After all samples are allocated into table cells, there are two types of cells: non-empty cells with a number of samples and empty cells with none. For non-empty cells, we calculate its lookup value according to the samples within. While there are many ways to do it, we found the mean a good and efficient representative.

For the lookup values of empty cells, a traditional approach would be extrapolating them from non-empty ones. However, under sufficiently large amount of random samples, it is very likely that the empty cells originate from unrealistic situations. For example, as in Step 3 of Figure 4, the empty cells are distributed only in the top-right and lower-left corners, representing the extremely flat and the extremely sharp transient faults, respectively. Although neither of the two kinds of transient faults exists in reality, accesses to these cells happen during the SSER analysis occasionally as a result of error propagation. In such cases, we use the lookup value of the nearest non-empty cell instead to offset the expected error.

IV. USING QUASIRANDOM SEQUENCES

Pseudorandom number generation plays a key role to the success of the Monte Carlo method. However, using \texttt{rand()} function for sampling points often suffers from the clustering problem [16] in high dimensional spaces. Figure 5(a) illustrates this problem on an example of generating a \((X,Y)\)-distribution by the Monte Carlo method using the \texttt{rand()} function. The sampling points are observed not evenly scattered among the \((X,Y)\) plate, which means that these sampling points from pseudorandom generation may not be representative enough for the entire space.

The clustering problem motivates research of finding a deterministic sequence such that well-chosen points are distributed in the high-dimensional spaces uniformly. Such sequences are named quasirandom sequences. Figure 5(b) shows the same number of sampling points using quasirandom sequences on the \((X,Y)\) plate. Sobol algorithm [16] is used to generate the corresponding se-
Monte Carlo methods with quasirandom sequences are termed Quasi-Monte Carlo (QMC) methods. Given a sampling number \( N \) and a dimension \( d \), Monte Carlo methods converge with \( O(1/\sqrt{N}) \) simulation errors whereas QMC methods converge with \( O(1/N) \) for optimal cases. Previous research works have demonstrated better results for QMC than MC methods for the problems with \( \leq 360 \) dimensions in finance and physics.

Since each gate in the circuit becomes a free dimension (regardless of spatial correlations), the total dimension in the corresponding SSER system can be very high. However, for a large \( d \) and moderate \( N \), quasirandom sequences perform no better than the pseudorandom sequences [16]. Besides, high dimensional quasirandom sequences tend to suffer from the clustering problem again. In the worst cases, QMC’s convergence rate, \( O((\ln N)^d/N) \), are even worse than MC’s \( O(1/\sqrt{N}) \) as \( d \) goes larger. Therefore, we are motivated to apply dimension reduction to ensure the effectiveness of the proposed QMC framework for SSER analysis.

Effective dimensions of circuits can be observed through experiments. Figure 6 shows the convergence rates for four sample circuits where the vertical lines indicate the logic depths (a.k.a. levels) of each circuit. All convergence rates drops quickly as the dimension numbers increase. Such phenomenon implies their underlying SSER systems can be properly described using much lower dimensions.

Fig. 5. Distributions from the Monte Carlo methods with random number generation and quasirandom sequences

Fig. 6. Convergence rate, dimension number, and logic depth of benchmark circuits

V. Experimental Results

A series of table-based models are built and evaluated in accuracy. These models are then integrated into our SSER analysis framework to evaluate their SER estimation capability.

A. Model accuracy

We build the table-based models according to Figure 4 for three cells under 45nm technology. Assuming 5% process variation (\( \sigma_{\text{proc}} = 5\% \)), the models are built using 500K training samples. The total size of cell models in our experiments is 9.5MB. Then, we examine these models’ accuracy using another 10K test samples.

The average errors of the models are summarized in Table I according to model types. Accordingly, two messages can be observed: (1) For \( M_{\text{strike}}^\mu \), \( M_{\text{strike}}^\sigma \), and \( M_{\text{prop}}^\mu \) the models are highly accurate with average errors no more than 0.4%. For the \( M_{\text{prop}}^\sigma \) models, the average error is still within 2.1%. (2) In [11], the \( M_{\text{strike}}^\mu \), \( M_{\text{prop}}^\mu \), and \( M_{\text{prop}}^\sigma \) models have average errors up to 3.9%. For its \( M_{\text{strike}}^\sigma \) models, the average error further reaches 12.9%. In summary, our models exhibit much better quality.

B. SSER measurement

The proposed framework is implemented in C/C++ and exercised on a Linux machine with a Pentium Core Duo (2.4GHz) processor and 4GB RAM. The 45nm Predictive Technology Model (PTM) [17] is used for cell modeling. For all circuits, each node under every input pattern combination is injected with four levels of electrical charges: \( Q_0 = 34fC \), \( Q_1 = 66fC \), \( Q_2 = 99fC \) and \( Q_3 = 132fC \), where 32fC is observed to be the weakest
charge capable of generating a transient fault with positive pulse width under the settings in our experiments.

Both circuit SER and SSER are measured and compared. For SER, we use static SPICE simulation; for SSER, we use Monte Carlo SPICE simulation as well as the proposed framework with (QMC) and without (MC) quasirandom sequences. Considering the extremely long runtime of Monte Carlo SPICE simulation (w/ 100 runs), we can only afford to perform tests on small circuits (i4, i6, i18 and c17), with the largest containing 7 gates, 12 strike nodes and 5 inputs. The runtime of the Monte Carlo SPICE simulation ranges from 8 hours to slightly more than one day. The runtime of our framework requires less than 1 second with an average of $10^6$ speedup.

![Fig. 7. SSER comparison from static and Monte Carlo SPICE simulations, the proposed MC and QMC frameworks](image)

Figure 7 compares the results from SPICE simulation and our frameworks. The three facts are observed: (1) Considering 5% process variations, the SER obtained by Monte Carlo SPICE simulation are 35% $\sim$ 52% above the SER obtained by static SPICE analysis (indicated by the black bars). Since the process variation worsens the stability of circuits beyond the deep submicron era, statistical effect should be considered to avoid increasingly underestimated circuit SER. (2) The proposed MC and QMC frameworks yield very similar SSERs with each others where the mismatches are within 0.4%. This means that we can use the faster QMC without serious accuracy degradation. (3) Compared to the results of Monte Carlo SPICE simulation, the proposed QMC framework has error rates of 2.5%, 0.8%, 2.9%, and 2.8%, respectively. Compared to [14] and [11] where the error rates are around 10%, our framework is quite accurate, which can be well attributed to our models.

To more closely investigate the SER difference between static and statistical analysis, we breakdown the results in Figure 7 by charge strength levels, and present the results in Figure 8. Comparing the results between static and statistical SPICE simulations across all test circuits, it is observed that the results of the two SPICE simulations and the proposed framework are very similar for $Q_1 \sim Q_3$ parts (within 1% difference). However, the static SPICE simulation dramatically underestimate the SERs for the $Q_0$ part (indicated by the white bars), whereas the proposed framework gives much closer results with $Q_0$-part errors of 11.4%, 2.3%, 5.6%, and 4.1%, respectively. It also discloses that the analysis of weak charges is the most challenging task of SSER analysis.

C. SSER estimation on benchmark circuits

Using the proposed MC/QMC frameworks, we conduct SSER analysis on a variety of circuits including the ones in Figure 7, the ISCAS’85 benchmark circuits, and a series of multipliers. Table II first lists the name, the total number of nodes, and the total number of outputs for each circuits. The following four columns report the SSER values and the runtime required by the MC and QMC frameworks, respectively. The last two columns compute the SER difference and speedup, respectively, by comparing results from the MC and QMC frameworks.

From Table II, SSER is clearly related to the number of nodes and primary outputs of a circuit, which correspond to the possibility of the circuit struck by radiation particles and the possibility of the transient faults observed at primary outputs, respectively. The runtime, however, depend on not only the number of strike nodes, but also the number of convolutions between nodes. SER difference is computed by $|SSER_{MC} - SSER_{QMC}|/SSER_{MC}$ and the average of 0.88% difference implies that the QMC and MC frameworks are of the same quality. For all benchmark circuits, the overall speedup brought by QMC is 2.95X in average and the QMC runtime is comparable to that of [11].

VI. Conclusion

Traditional SER analysis techniques intend to mimic the static SPICE simulation. However, in presence of process variation, all static techniques tend to unavoidably underestimate true SERs and thus the research of statistical SER analysis is emerging. In this paper, we propose a method for building quality statistical cell models, based on which a Monte Carlo SSER framework is built. A heuristic is particularly proposed to apply quasirandom
sequences to the framework for faster convergence and shorter runtime. According to the experimental results, the SSER errors are within 3% compared to Monte Carlo SPICE simulations, which are more accurate than those from previous works. Furthermore, the use of quasirandom sequences demonstrates an average of 2.95X runtime improvement over the baseline MC framework while preserving the same SSER quality.

SSER analysis is a thriving research topic in sub-90nm technologies. Future works include further reduction of dimensions to improvement runtime, applications of various variance reduction techniques to the QMC framework, and the inclusion of spatial correlation over cells.

REFERENCES


### Table II

<table>
<thead>
<tr>
<th>circuit</th>
<th>N_nodes</th>
<th>N_vars</th>
<th>MC SSER (FIT)</th>
<th>T_MC (sec)</th>
<th>QMC SSER (FIT)</th>
<th>T_QMC (sec)</th>
<th>MC/QMC Comparison</th>
<th>speedup (X)</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>4</td>
<td>1</td>
<td>24.22E-05</td>
<td>&lt; 1</td>
<td>24.31E-05</td>
<td>&lt; 1</td>
<td>0.37</td>
<td>-</td>
</tr>
<tr>
<td>18</td>
<td>6</td>
<td>2</td>
<td>37.60E-05</td>
<td>&lt; 1</td>
<td>37.63E-05</td>
<td>&lt; 1</td>
<td>0.03</td>
<td>-</td>
</tr>
<tr>
<td>118</td>
<td>12</td>
<td>3</td>
<td>64.26E-05</td>
<td>&lt; 1</td>
<td>64.21E-05</td>
<td>&lt; 1</td>
<td>0.02</td>
<td>-</td>
</tr>
<tr>
<td>c17</td>
<td>12</td>
<td>3</td>
<td>64.06E-05</td>
<td>&lt; 1</td>
<td>64.27E-05</td>
<td>&lt; 1</td>
<td>0.17</td>
<td>-</td>
</tr>
<tr>
<td>c1423</td>
<td>233</td>
<td>7</td>
<td>1047.12E-05</td>
<td>114.37</td>
<td>1045.23E-05</td>
<td>30.43</td>
<td>0.18</td>
<td>3.76</td>
</tr>
<tr>
<td>c1499</td>
<td>638</td>
<td>32</td>
<td>1150.61E-05</td>
<td>870.61</td>
<td>1161.77E-05</td>
<td>269.71</td>
<td>0.97</td>
<td>3.23</td>
</tr>
<tr>
<td>c880</td>
<td>443</td>
<td>26</td>
<td>1519.24E-05</td>
<td>173.25</td>
<td>1516.46E-05</td>
<td>36.90</td>
<td>0.18</td>
<td>4.69</td>
</tr>
<tr>
<td>c1355</td>
<td>629</td>
<td>32</td>
<td>1188.16E-05</td>
<td>891.80</td>
<td>1169.25E-05</td>
<td>273.20</td>
<td>1.59</td>
<td>3.26</td>
</tr>
<tr>
<td>c1908</td>
<td>425</td>
<td>25</td>
<td>1124.75E-05</td>
<td>365.07</td>
<td>1145.27E-05</td>
<td>109.25</td>
<td>2.09</td>
<td>3.34</td>
</tr>
<tr>
<td>c2670</td>
<td>841</td>
<td>157</td>
<td>3479.23E-05</td>
<td>401.02</td>
<td>3463.73E-05</td>
<td>120.23</td>
<td>0.45</td>
<td>3.34</td>
</tr>
<tr>
<td>c3540</td>
<td>901</td>
<td>22</td>
<td>2411.57E-05</td>
<td>1070.61</td>
<td>2395.72E-05</td>
<td>309.53</td>
<td>0.66</td>
<td>3.46</td>
</tr>
<tr>
<td>c5315</td>
<td>1806</td>
<td>123</td>
<td>41587.66E-05</td>
<td>818.22</td>
<td>41592.29E-05</td>
<td>403.17</td>
<td>0.01</td>
<td>2.03</td>
</tr>
<tr>
<td>c6288</td>
<td>2788</td>
<td>32</td>
<td>3860.03E-05</td>
<td>15703.05</td>
<td>3769.48E-05</td>
<td>4710.04</td>
<td>2.35</td>
<td>3.33</td>
</tr>
<tr>
<td>c7525</td>
<td>2114</td>
<td>126</td>
<td>14311.19E-05</td>
<td>1406.70</td>
<td>14359.10E-05</td>
<td>658.37</td>
<td>0.89</td>
<td>2.14</td>
</tr>
<tr>
<td>mul_4</td>
<td>158</td>
<td>8</td>
<td>883.35E-05</td>
<td>98.82</td>
<td>890.33E-05</td>
<td>34.85</td>
<td>0.79</td>
<td>2.84</td>
</tr>
<tr>
<td>mul_8</td>
<td>728</td>
<td>16</td>
<td>2127.35E-05</td>
<td>710.21</td>
<td>2094.05E-05</td>
<td>271.03</td>
<td>1.57</td>
<td>2.62</td>
</tr>
<tr>
<td>mul_16</td>
<td>3156</td>
<td>32</td>
<td>4775.07E-05</td>
<td>9565.03</td>
<td>4843.29E-05</td>
<td>5010.40</td>
<td>1.47</td>
<td>1.91</td>
</tr>
<tr>
<td>mul_24</td>
<td>7234</td>
<td>48</td>
<td>7636.46E-05</td>
<td>39628.50</td>
<td>7478.02E-05</td>
<td>29630.01</td>
<td>2.07</td>
<td>1.35</td>
</tr>
</tbody>
</table>

Average | 0.88 | 2.39 |