Dependable Systems: Final Project

Lorenzo Di Tucci - lorenzo.ditucci@mail.polimi.it

A.Y 2014/2015

1 Introduction

Nowadays, one of the biggest challenges in modern multicore systems is reliability. Elevated operating temperatures and high power densities leads to an acceleration of device wear-out.
If the wear-out failures are not taken into consideration during task allocation and scheduling processes, some processors might age much faster than the others and become the reliability bottleneck for the system, significantly reducing the system's service life.[2]
It's clear that both power and thermal management techniques should be exploited in order to have a low power and reliable system on chip. Lot's of techniques have been proposed: Dynamic Voltage Frequency Scaling (DVS) for example , act by changing the voltage frequency, lowering the power consumption and reducing the chip average temperature resulting in improving the reliability of the system by reducing hard failures. Some power management policies decrease the average temperature by turning off some idle core improving reliability and diminishing the probability of phenomena like Electromigration (EM) and Time-Dependet-Dieletric-Breakdown (TDDB).
However, all this techinques do not concentrate on deleting the thermal stresses as Thermal Cycle (TC) because they do not consider the deteriorating impacts of TC on the overall system reliability.[3]

In this project, we focus on creating a model to estimate the reliabiliy of a system considering the wear-out due to Thermal Cycling.

2 Thermal Cycling

As Thermal Cycling (TC) we refer to the wear caused by thermal stress, resulting from mismatched coefficient of thermal expansion for adjacent matherial layers; run-time temperature variation results in inelastic deformation, eventually leading to failure.[4]
A thermal Cycle occurs when the temperature starts from some initial value, reaches an extreme point, and returns to the starting values. The MTTF (Mean Time To Failure) due to Thermal Cycling, strongly depends on the peak temperature and cycle amplitude. A cycle can be denoted bu a non-adjacent peak/valley pair.
An important problem to be addressed here is the method to count how many cycles there are in a set of temperatures. A proper estimation is complex, this because, double counting the number of cycles (counting them in a naive way) could carry to an underestimation of the MTTF. In contrast, if we count only the neighbouring peak/valleys we would overestimate the MTTF by 2.7x [4]. In our model, we decided to use the Rainflow Counting Algorithm in order to count the number of cycles in a profile of temperatures. We've chosen this method as it's said to be the "most widely used cycle counting method in material science and is commonly accepted as the best method for estimating the damage of a

material due to random loading fluctuation.[4]

3 Rainflow Counting Algorithm

The rainflow-counting algorithm is used in the analysis of fatigue data in order to reduce a spectrum of varying stress into a set of simple stress reversals. It is very important, as it allows the application of Miner's Rule, key formula to calculate the MTTF. The algorithm was developed by Tatsuo Endo and M. Matsuishi in 1968, however, in this model we are implementing the newer version of Downing and Socie, explained as Algorithm II in [5].

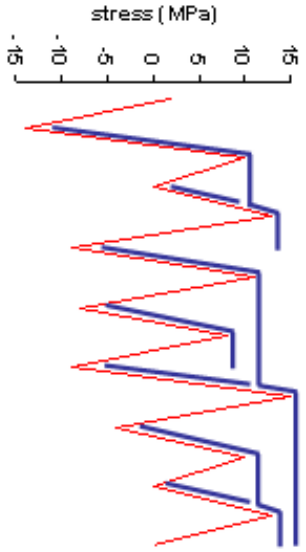

Figure 1: Example of Rainflow Counting

The Naive version of the Algorithm, works in 8 steps:[6]

1. Reduce the time history to a sequence of peaks and Valleys
2. Imagine that the time history is a template for a rigid sheet (pagoda roof)
3. Turn the sheet clockwise 90 degrees (earliest time to the top).
4. Each tensile peak is imagined as a source of water that "drips" down the pagoda.

5. Count the number of half-cycles by looking for terminations in the flow occurring when either:
   - It reaches the end of the time history;
   - It merges with a flow that started at an earlier tensile peak; or
   - It flows when an opposite tensile peak has greater magnitude.
6. Repeat step 5 for compressive valleys.
7. Assign a magnitude to each half-cycle equal to the stress difference between its start and termination.
8. Pair up half-cycles of identical magnitude (but opposite sense) to count the number of complete cycles. Typically, there are some residual half-cycles.

4 Implementation

The model presented here, calculates the Realiability and the MTTF due to thermal cycles in two scenarios: a static and a dynamic one. The static scenario, works in three simple steps. It reads the data from a file, applies the Rainflow Counting algorithm on the data and, once the cycles have been identified, calculates the MTTF of the system.

For what concern the dynamic scenario, the three steps are implemented as three different modules that works in parallel. The first module has the task of periodically checking the input file, and to identify if the new values read are peak, valleys or none of the previous.
The second module iteratively applies the Rainflow Counting Algorithm to the data read by the first module, but without doing the last step of the algorithm. In particular, this version of the Rainflow Counting Algorithm, once it has read all the data and goes out of it, does not re-start from the beggining but simply stop.
For this version of the algorithm, it didn't really make sense to calculate the MTTF as stated in [7] as we are reading data dynamically. Calculating the MTTF in this case would means that we are inferring about the future (we do not have the complete temperature profile), and that is not possible. For this reason, the third module, instead of calculating the MTTF we simply calculates the Realiability due to TC.

As said before, in the second scenario we are applying a version of the Rainflow Counting algorithm that has been slightly modified. In fact, once the algorithm reads all the data one time, it does not restart from the beginning. It could seems that in this ways we loose a considerably amount of data. Actually the amount of data we loose is around 1% as it's observable from the table below.

Peak/Valleys Numb.	Static Cycles	Dynamic Cycles
10 000	5000	4995
100 000	50 000	49 994
1 000 000	500 000	499 996

As an example, we provide a chart representing a complete profile of temperature, and the result after applying the Rainflow Counting algorithm both in the dynamic and static case.

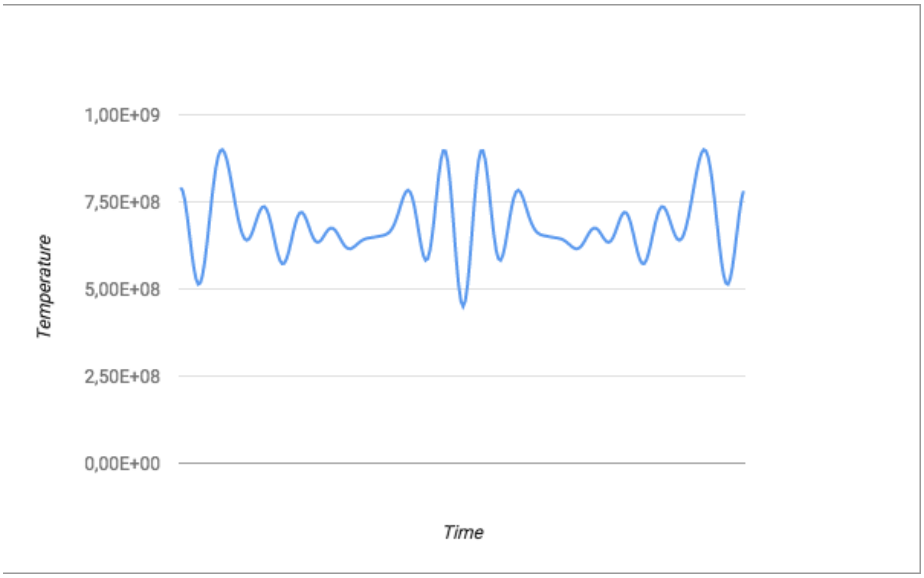

Figure 2: Rainflow Example: raw data

In Figure 2, is observable the raw input data fed to the algorithm. The result of this step of the algorithm is the number of cycles identified. From an input of 256 values, the static algorithm identified 13 cycles, while the dynamic one identified 10 cycles. In particular the cycles identified are:

Number	Range	Time
1	9.64844	31
2	4.10089	63
3	10.4526	56
4	20.1244	104
5	32.7361	20
6	4.10089	189
7	10.4526	180
8	20.1244	146
9	9.64844	219
10	32.4338	138
11	0.989052	256
12	0	249
13	0.302368	56

The first 10 results have been identified by both the algorithm while the last three, are identified by the last algorithm only, during its last pass.

4.1 Formulas Used

The formulas used to calculate the parameters for the outputs (MMTF or Reliability) are the same between the two implementations. Obviously, as the outputs are different, we used two different formulas to calculate the Reliability and the Mean Time to Failure.
All the formula I will present here are taken from [7].
In this paper [7] is stated that Thermal Cycling related MTTF can be calculated in three steps:

1. Calculating the Thermal Cycles from a thermal profile using Rainflow Counting Algorithm [5]
2. Calculating, from each Thermal Cycle, the number of cycles to failure using Coffin-Mansion's Rule

$$N_{TC}(i) = A_{TC}(\delta T_i - T_{Th})^{-b} * e^{\frac{E_a}{KT_{max}(i)}} \tag{1}$$

   Where

   - $N_{TC}(i)$: number of cycles to failure due to the i_{th} thermal cycle
   - A_{TC}: Empirically Determined constant
   - δT_i: Amplitude of the i_{th} thermal cycle
   - T_{Th}: Amplitude at which elastic deformation begins
   - b: Coffin-manson exponent constant
   - E_a: Activation Energy
   - $T_{max}(i)$: Maximum temperature in the i_{th} thermal cycle

3. Calculating the MTTF using Miner's Rule

$$MTTF = \frac{N_{TC} \sum_{i=1}^{m} t_i}{m} \tag{2}$$

   Where
   t_i is the temperature of the i_{th} thermal cycle, m the number of thermal cycles calculated in step 1 and N_{TC} the effective number of cycles calculated using

$$N_{TC} = \frac{m}{\sum_{i=1}^{m} \frac{1}{N_{TC}(i)}} \tag{3}$$

In the second implementation, Step 3 is replaced by the calculation of the reliability as stated in formula 5 of this paper [1]. Here the Reliability at time

t is calculated as follow:

$$R(t) = e^{-\sum_{j=1}^{i} \frac{\tau_j}{\alpha_j(T)}} \tag{4}$$

where τ_i is the duration of each period of time with temperature T_j up to time t.
In our scenario the formula has been rearranged as follow:

$$R(t) = e^{-\sum_{i=1}^{m} \frac{\delta t_i}{N_T C(i)}} \tag{5}$$

with m number of cycles identified, δt_i duration of the $i_t h$ cycle and $N_T C(i)$ number of cycles to failure due to the ith thermal cycle.

4.2 Tuning Parameters

As it's observable from the formulas, there is the needing of tuning some parameters. In particular, the parameters that needs to be tuned are:

- A_{TC}
- T_{Th}
- b
- E_a
- k

By default, in our software implementations, the parameters have been set as in section 9.2 of [8]. Here the Coffin-Manson exponent has been set to 6, the activation energy to 0.5 and the threshold temperature to 0. In this paper is also stated that the coefficient of proportionality is not significant (A_{TC}) if you are interested in the relative improvement.
Other parameters can also be taken from [9].

References

[1] Cristiana Bolchini, Matteo Carminati, Marco Gribaudo, Antonio Miele - Dipartimento di Elettronica, Informazione e Bioingegneria -Politecnico di Milano, Italy *A Lightweight and Open-source Framework for the Lifetime Estimation of Multicore Systems*

[2] Lin Huang, Feng Yuan and Qiang Xu - CUhk Reliable computing laboratory, The Chinese University of Hong Kong, Shatinm N.T., Hong Kong *Lifetime Reliability-Aware Task Allocation and Scheduling for MPSoC Platforms.*

[3] Mehdi Kamal, Arman Iranfar, Ali Afzali-Kusha, Massoud Pedram - University of Tehran, Iran *A Thermal Stress-Aware Algorithm for Power and Temperature Management of MPSoCs - 2015*

[4] Yun Xiang, Thidapat Chantem, Robert P. Dick, X. Sharon Hu, Li Shang - University of Michigan, University of Notre Dame, University of Colorado *System-Level Reliability Modeling for MPSoCs*

[5] S.D. Downing, D.F.Socie *Simple Rainflow Counting Algorithm, 1982*

[6] https://en.wikipedia.org

[7] Anup Das, Rishad A. Shafik, Geoff V. Merrett, Bashir M. Al-Hashimi, Akash Kumar, Bharadwaj Veeravalli - University of Singapore and University of Southampton *Reinforcement Learning-Based Inter- and Intra-Application Thermal Optimization for Lifetime Improvement of Multicore Systems*

[8] Ivan Ukhov, Min Bao, Petru Eles, Zebo Peng - Linkoping University- Sweden *Steady-State Dynamic Temperature Analysis and Reliability Optimization for Embedded Multiprocessor Systems*

[9] *Failure Mechanisms and Models for Semiconductor Devices - JEP122F - November 2010*