On Modeling the Lifetime Reliability of Homogeneous Manycore Systems

Lin Huang and Qiang Xu
CUhk REliable computing laboratory (CURE)
Department of Computer Science & Engineering
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
Email: {lhuang,qxu}@cse.cuhk.edu.hk

ABSTRACT

Advancements in technology enable integration of a large number of cores on a single silicon die. At the same time, aggressive technology scaling has an ever-increasing adverse impact on the lifetime reliability of such large integrated circuits. In this work, we model the lifetime reliability of homogeneous manycore systems using a load-sharing nonrepairable k-out-of-n:G system with general failure distributions for embedded cores. In manycore systems, an embedded core can be in operational, cold standby, or warm standby state depending on system redundancy schemes and their workloads. We then use the proposed model to analyze the impact of different redundant schemes and configurations on the lifetime reliability of manycore systems.

1. INTRODUCTION

While the relentless scaling of CMOS technology has brought with it enhanced functionality and improved performance in every new generation, the associated ever-increasing on-chip power and temperature densities make the lifetime reliability of high-performance integrated circuits (ICs) one of the major concerns for the industry [3, 19].

The failure mechanisms that contribute to IC's permanent failures (e.g., time dependent dielectric breakdown (TDDB) and electromigration) have been extensively studied at the circuit level in the past, and they are shown to be strongly related to the temperature and voltage applied to the circuit. Recently these failure mechanisms have been revisited at the processor microarchitecture level due to their increasing impact with technology scaling [18, 19].

The above models mainly target unicore processor chips. State-of-the-art computing systems (e.g., multi-digital signal processor (DSP) system [11], general-purpose processors), however, have started to employ multiple cores on a single silicon die to improve performance through parallel execution instead of frequency increase, which have the benefits of power-efficiency and short time-to-market [7]. A 128-core GPU [15] and a 64-core general-purpose multiprocessor [22] have already been released to the market. Various research teams have projected that thousand-core processor chips will become commercially available in the foreseeable future [1, 4]. For such large-scale manycore systems fabricated with latest technology, how to model its lifetime reliability is an interesting and relevant problem.

Assuming the failure distribution of an embedded core is known a priori, we analyze the lifetime reliability of manycore systems in this work. We make the following observations during the modeling process:

- embedded cores will age in operation. That is, we expect an increasing failure rate (IFR) when a core gets older.
- manycore systems are *k-out-of-n:G systems*[1], in which n is the total number of processor cores fabricated on-chip and k is the number of cores for the system to function correctly. Generally speaking, the value of n is larger than the value of k to provide fault tolerance [24].
- manycore systems are *load-sharing systems*, i.e., each embedded core is designed to carry only part of the load assigned by the operating system (OS). In fact, a core's failure rate and the associated lifetime depends significantly on its workload that determines the temperature and voltage applied to the circuit.
- manycore systems are *nonrepairable systems*. That is, unlike traditional board-level multiprocessor systems that can be easily repaired by replacing defective processor chip, embedded cores are integrated on silicon die in manycore systems and it is extremely difficult to repair or replace a faulty core, if not impossible.

Based on the above observations, we model the lifetime reliability of manycore systems using a load-sharing nonrepairable k-out-of-n:G system with general lifetime distributions for embedded cores. To the best of our knowledge, this is the first comprehensive reliability model for such complex systems.

Manycore systems can be configured in two ways to achieve reliability: (i). *gracefully degrading systems* that use all failure-free cores to execute tasks. When a core failure is detected, these systems attempt to reconfigure to a system with one fewer module; (ii). *standby redundant systems* that execute tasks on active cores. Upon detection of the failure of an active core, these systems attempt to replace the faulty unit with a spare unit. Depending on the above configurations and current workload, cores can be in normal functional mode, warm standby, or cold standby state, which have direct implications on the ageing effect of the manycore system. In this paper, we use the proposed model to analyze the impact of different configurations and redundant schemes on manycore systems' lifetime reliability. This will facilitate designers to make architecture decisions to achieve their design objectives.

The remainder of this paper is organized as follows. In Section 2, we present preliminaries and motivation for this work. The proposed lifetime reliability model for manycore systems is then discussed in detail in Section 3. Experimental results for different manycore system configurations are presented in Section 4. Finally, Section 5 concludes this paper.

[1]An n-component system that works (or is "good") if at least k of the n components work is called a k-out-of-n:G system.

2. PRELIMINARIES

In this paper, we consider homogeneous manycore systems that have n identical embedded cores fabricated on-chip. In order to function correctly, at least k ($k \leq n$) cores need to be good. These cores will share the workload designated by operating system. Apparently, this is a k-out-of-n:G load-sharing system. Before discussing the technical details of the proposed lifetime reliability model, we present some preliminaries in this section.

2.1 IC Lifetime Reliability

Integrated circuit errors can be broadly classified into two categories: soft errors and hard errors. As soft errors caused by radiation effects do not fundamentally damage the circuit, they are not viewed as lifetime reliability threats. In this paper we mainly consider those hard errors that are permanent once they manifest, such as TDDB in the gate oxides, electromigration (EM) and stress migration (SM) in the interconnects, and thermal cycling (TC).

The above failure mechanisms have an increasingly adverse effect with technology scaling, and therefore have re-attracted research interests recently. Srinivasan *et al.* [19] described a so-called RAMP model that is able to dynamically track lifetime reliability of a processor according to changes in application behavior. Their model, however, is inherently inaccurate because it assumes a uniform device density over the chip and an identical vulnerability of devices to failure mechanisms. To address this problem, Shin *et al.* [18] introduced a structure-aware model that takes the vulnerability of basic structures of the microarchitecture (e.g., register files, latches and logic) to different types of failure mechanisms into account. Coskun *et al.* proposed a cycle-accurate lifetime reliability simulation methodology as well as a statistical one in [5] and used them to optimize the processor power management policy. In [20], Srinivasan *et al.* studied the vulnerability of FPGAs to TDDB and EM effects.

2.2 Modeling Processor Core Behavior

We assume embedded cores execute tasks independently (an application however may consist of a series of tasks [14]) and one core can perform at most one task at a time. In addition, the tasks assigned to a certain core is assumed to be stored in a first-in-first-out (FIFO) buffer with infinite capacity when the core is busy. Once the core becomes available, it starts to process the next task in the FIFO promptly. As shown in Fig. 1, a core can be in active mode or spare mode in the manycore system (depending on redundancy configurations). For spare processor cores, their power supply can be reduced significantly or turned off completely, we therefore treat them as cold standby components with zero failure rate. For active cores, depending on the current workload, they can be in two states: *processing* or *wait*, which denote the state that the cores are performing tasks or waiting for task allocation, respectively. Generally speaking, cores operate at higher temperature in processing state and hence will wear out more quickly than in wait state. We therefore regard cores in wait state as warm standby components in this work, and we use $R_p(t)$ and $R_w(t)$ to denote the reliability functions of cores in processing state and wait state, respectively, where they have the same shape but different scale parameter. For example, $R_p(t) = e^{-(\frac{t}{\theta_p})^\beta}$ and $R_w(t) = e^{-(\frac{t}{\theta_w})^\beta}$, wherein θ_p and θ_w are scale parameters.

According to the above discussion, if manycore systems are configured as a graceful degrading system, embedded cores cannot be in spare mode and hence they are in either processing or wait state. The number of cores in either state at a particular moment is dependent on the current workload and hence is uncertain. If, however, manycore systems are configured as a standby redundant system, an embedded core can serve as: *cold* standby, *warm* standby or *processing* core. As k cores are active, we know exactly how many cores are cold standbys but again not sure about the number of cores in processing or wait state at a specific time.

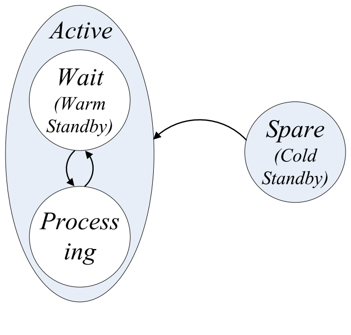


Figure 1: The Embedded Core Behavior.

2.3 Related Work on Modeling k-out-of-n:G Systems

While there has been a large amount of research work on modeling the lifetime reliability of multi-component systems, most of them focused on parallel systems that are designed to carry full load, as shown in [10, 23].

In the literatures on load-sharing k-out-of-n:G systems, for the sake of simplicity, many studies (e.g., [16, 12]) assume an exponential lifetime distribution for every component, which implies a constant failure rate during a component's entire life cycle. With this assumption, the system can be represented by a discrete-state, continuous-time homogeneous Markov chain and analyzed using mature techniques [16]. The above assumption, however, implies that there is no difference between a brand-new unit and a 10-year old one in terms of failure rates, which is obviously not true. In fact, the popularity of this assumption is mainly due to its mathematical tractability rather than accuracy.

In real-life systems, we expect components to experience increasing failure rate over its life cycle, i.e., exponential lifetime distribution does not apply. For systems with general lifetime distributions for the internal components, Markov model cannot be used to analyze their lifetime reliability because whether a component is good or not depends on its past usage and hence the memoryless property required for Markov modeling does not hold. This makes the mathematical analysis for systems with general lifetime distributions much more complicated. [9] studied a 1-out-of-2:G system with time-varying failure rates in a general polynomial expression format. Later, [13] presented an analytical model for components with various general lifetime distributions.

An idle component in computing systems can serve as a cold, hot, or warm standby unit, which has a zero failure rate, the same failure rate as active components or a failure rate between cold and hot, respectively. In the above models (e.g., [9, 13]), every component in the system is assumed to conform to a single failure distribution and hence can only be applied to analyze systems with hot standby components. In manycore systems, as discussed earlier, an active embedded core in wait state should be treated as a warm standby component. Consequently, the above models are not applicable. [17] provided a closed-form expression for the k-out-of-n:G systems with warm standby components. However, they assume the failure rates of both active and standby units are constant. Recently, [25] presented an analytical model for k-out-of-$(M+N)$:G repairable warm standby systems that consists of two different types of components.

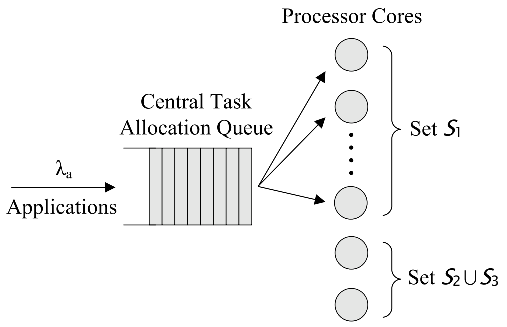


Figure 2: Queueing Model for Task Allocation in Manycore System.

Most prior work (e.g., [16, 9, 13]) focuses on analyzing the lifetime reliability of gracefully degrading systems, in which all embedded cores are active. For standby redundant systems, an embedded core can be a "spare" unit, and such systems involve both warm standby (wait state) and cold standby (spare state) units as well as processing cores. [8] presented a mixture model for a 1-out-of-3 standby system that contains a dedicated warm standby unit and a dedicated cold standby unit. [21] analyzed a system in which a module can alternate between cold and warm standby state. However, the system investigated in this work contains only two components. Similar to many prior work on this topic, both [8] and [21] are difficult to extend to the general k-out-of-n:G systems.

2.4 Notations

The most widely-used concepts in reliability engineering and their representations are listed as follows and will be used throughout this paper without further explanations.

$R(t)$	reliability function[2]
$F(t)$	cumulative distribution function, $1-R(t)$
$f(t)$	probability density function, $\frac{\mathrm{d}}{\mathrm{d}t}F(t)$
$h(t)$	failure rate function (hazard function), $\frac{f(t)}{R(t)}$
$MTTF$	mean time to failure, $\int_0^\infty R(t)\mathrm{d}t$

In addition, in the rest of the paper, we use superscript *sys* to distinguish the functions for the entire manycore system from that for a single core. We also use *de* and *st* to represent the system in degrading configuration and standby configuration, respectively. For manycore systems in standby configuration, subscripts i and j are used to indicate the number of active and spare cores.

3. PROPOSED LIFETIME RELIABILITY MODEL FOR MANYCORE SYSTEMS

3.1 Queueing Model for Task Allocation

Consider a manycore system composed of a set $\mathcal{S} = \{1,2,\ldots,n\}$ identical embedded cores. Among these cores, the set of active cores is $\mathcal{S}_1$, the set of spare cores is $\mathcal{S}_2$, the set of faulty cores is $\mathcal{S}_3$, and $\mathcal{S}_1 \cup \mathcal{S}_2 \cup \mathcal{S}_3 = \mathcal{S}$. To capture the key features, we model a general-purpose parallel processing system with a central queue as a bulk arrival $M^X/M/|\mathcal{S}_1|$ queuing model, as shown in Fig. 2. More specifically, the application arrival to the manycore system is assumed to be a Poisson process with rate λ_a. An application may consist of a series of tasks that can be processed independently of each other [14]. We denote the number of tasks in an application as X. Let γ_i be the probability that an application consists of i tasks (i.e, $\Pr\{X = i\}$). Apparently, $i = 1,2,\cdots$; and we have $\sum_{i=1}^{\infty}\gamma_i = 1$. By using z-transform, the probability generating function of X is $G_X(z) = \sum_{i=1}^{\infty}\gamma_i z^i$. Each task is executed by an individual active core and the service time is exponentially distributed with mean $\frac{1}{\mu}$. Consequently, the entire manycore system is modeled as an $M^X/M/|\mathcal{S}_1|$ queuing system. The probability that a certain active core is occupied by tasks, i.e., traffic intensity (also called utilization), is $\rho = \frac{E[X]\cdot\lambda_a}{|\mathcal{S}_1|\cdot\mu}$, where $E[X]$ is the s-expected value of X and can be computed as $\frac{\mathrm{d}}{\mathrm{d}z}G_X(z)\big|_{z=1} = G'_X(1)$. Let $\lambda \equiv E[X]\cdot\lambda_a$, $\rho = \frac{\lambda}{|\mathcal{S}_1|\cdot\mu}$. Our approach could be easily extended to represent other kinds of manycore systems and/or task allocation mechanisms (e.g., modeling the entire manycore system as a set of $M/M/1$ queue).

3.2 Lifetime Reliability of A Single Core

To obtain the lifetime reliability of the manycore system, we need to first calculate the lifetime reliability of an individual core. In this section, we examine the manycore system with two different redundant schemes: gracefully degrading and standby redundant.

3.2.1 Gracefully Degrading System

In gracefully degrading manycore systems, all good cores in the system are active and they alternate between *wait* state (as warm standby) and *processing* state in their lifetimes with different ageing effects. To accommodate this issue, we define a core's *accumulated time* in a certain mode at time t as how long it has spent in such a state up to time t. For example, suppose a core has its first task executed from time zero to time T_1, and its second task arrives at time T_2 ($T_2 \geq T_1$), and suppose at time t this task has not finished yet. In this case, this core's accumulated time in processing state is $(T_1 - 0) + (t - T_2)$ and its accumulated time in wait state is $(T_2 - T_1)$. We thus have the following theorem.

Theorem 1 *Suppose a manycore system with gracefully degrading scheme has experienced ℓ core failures, in the order of occurrence time at $t_1, t_2, \cdots, t_\ell$, respectively, for any core that has survived until time t ($t > t_\ell$)*

(a) its accumulated time in the processing state up to time t

$$\psi_p^{de}(t,t_1,t_2,\cdots,t_\ell) \approx \frac{\lambda}{(n-\ell)\mu}t - \sum_{j=1}^{\ell}\frac{\lambda}{\mu(n-j+1)(n-j)}t_j \quad (1)$$

(b) its accumulated time as warm standby up to time t

$$\psi_w^{de}(t,t_1,t_2,\cdots,t_\ell) \approx t - \frac{\lambda}{(n-\ell)\mu}t + \sum_{j=1}^{\ell}\frac{\lambda}{\mu(n-j+1)(n-j)}t_j \quad (2)$$

Proof Before the first core failure, there are n active cores in the system. In such an $M^X/M/n$ queuing system, the utilization of each core $\rho_n = \frac{\lambda}{n\mu}$. Since usually the time scale of lifetime reliability (in years) is much larger than that of task processing, the accumulated time in the processing state from time zero to t_1 can be approximated as $\rho_n \cdot t_1 = \frac{\lambda}{n\mu}t_1$. After each failure, the system is reconfigured to a system with one fewer module. Thus from t_j to t_{j+1}, there are $(n-j)$ active cores in the system $(0 < j \leq \ell - 1)$. The system can therefore be modeled as an $M^X/M/(n-j)$ queuing system. Similarly, the utilization of any surviving core is $\rho_{n-j} = \frac{\lambda}{(n-j)\mu}$. Its accumulated time in processing state in this period is therefore $\frac{\lambda}{(n-j)\mu}(t_{j+1} - t_j)$.

[2]Reliability function $R(t)$, also known as survival function, gives the probability that a component does not fail up to time t.

From t_ℓ to t, it is $\frac{\lambda}{(n-\ell)\mu}(t-t_\ell)$. The summation of these $(\ell+1)$ terms is Equation (1).

Since a surviving core can be in either processing or wait state from time zero to t, $\psi_w^{de}(t,t_1,t_2,\cdots,t_\ell) = t - \psi_p^{de}(t,t_1,t_2,\cdots,t_\ell)$. Hence, Equation (2) holds. □

With different aging effects in processing state and wait state for a particular core, we need to combine the accumulated time in these two modes in a unified manner to calculate the lifetime reliability of embedded cores. Recall that we assume the reliability functions in wait state and processing state have the same shape but different scale parameters (see Section 2). The *scale parameter* is a value by which t is divided and we use θ_w and θ_p to denote it in wait state and processing state, respectively. Given a general reliability function defined as $R(t,\theta)$ (abbreviated as $\mathcal{R}(t)$), the reliability functions of processing state and wait state are $R(t,\theta_p)$ (denoted as $R_p(t)$) and $R(t,\theta_w)$ (denoted as $R_w(t)$) respectively. After unification, we have relationships: $R_p(t) = \mathcal{R}(\frac{\theta}{\theta_p}\cdot t)$ and $R_w(t) = \mathcal{R}(\frac{\theta}{\theta_w}\cdot t)$. The following theorem provides the relationship between a single core's reliability and its accumulated times in different states, and enables their integration into one analytical model.

Theorem 2 *Given a gracefully degrading manycore system that has experienced ℓ core failures which occur at $t_1,t_2,\ldots,t_\ell$, respectively, the probability that a certain core survives at time t ($t>t_\ell$) provided that it has survived until time t_ℓ is given by*

$$R^{de}(t|t_1,t_2,\cdots,t_\ell) = \mathcal{R}(\psi^{de}(t,t_1,t_2,\cdots,t_\ell)) \tag{3}$$

where

$$\psi^{de}(t,t_1,t_2,\cdots,t_\ell) = \frac{\theta}{\theta_p}\psi_p^{de}(t,t_1,t_2,\cdots,t_\ell) + \frac{\theta}{\theta_w}\psi_w^{de}(t,t_1,t_2,\cdots,t_\ell) \tag{4}$$

Proof The proof of this theorem is presented in Appendix. □

We are also interested in the probability density function (p.d.f.) of core failures. By the definition of reliability and the corresponding p.d.f., given a manycore system has experienced ℓ core failures, at $t_1,t_2,\cdots,t_\ell$, respectively, the probability that a certain core fails in an infinitesimal interval $\mathrm{d}t_{\ell+1}$ at time $t_{\ell+1}$ is given by

$$f^{de}(t_{\ell+1}|t_1,t_2,\cdots,t_\ell)\mathrm{d}t_{\ell+1} = \frac{\mathrm{d}}{\mathrm{d}t}(1-R^{de}(t|t_1,t_2,\cdots,t_\ell))\Big|_{t=t_{\ell+1}}\mathrm{d}t_{\ell+1} \tag{5}$$

3.2.2 Standby Redundant System

In a standby redundant manycore system, spare cores are put aside in the beginning and are activated only when failures occur. In other words, different from that in gracefully degrading system, the active cores at time t in standby redundant system may be initially configured as spare ones. Once a spare core converts into active mode, it starts its aging process and will never return to spare mode (see Fig. 2). To capture this feature, we define a core's *birth time t^b* as the time point when it is configured as an active one. Before birth time t^b, a processor core is in the cold standby mode and has zero failure rate. After that, it alternates between processing state and wait state.

Theorem 3 *In a standby redundant manycore system, for any core with birth time t^b that has survived until time t ($t>t_b$)*

(a) its accumulated time in the processing state up to time t

$$\psi_p^{st}(t,t^b) \approx \frac{\lambda}{k\mu}(t-t^b) \tag{6}$$

(b) its accumulated time as warm standby up to time t

$$\psi_w^{st}(t,t^b) \approx (1-\frac{\lambda}{k\mu})\cdot(t-t^b) \tag{7}$$

Proof A functioning manycore system with standby redundant scheme always keeps k active cores and leaves remaining good cores as spares. Therefore, the utilization of an active core remains $\rho_k = \frac{\lambda}{k\mu}$. Similar to the analysis of the system with gracefully degrading scheme, the accumulated time of a core in processing state from t^b to t can be approximated as $\rho_k\cdot(t-t^b) = \frac{\lambda}{k\mu}(t-t^b)$. As can be observed, a processor core's accumulated time in either state only depends on its birth time t^b while is independent of the past ℓ core failures. As for the accumulated time in wait state, since the core is in either processing or wait state from t^b to t, $\psi_w^{st}(t,t^b) = (t-t^b) - \psi_p^{st}(t,t^b)$. Thus, we get Equation (7). □

Theorem 4 *In a manycore system with standby redundant scheme, the probability that a certain core with birth time t^b survives at time t ($t>t_b$) is given by*

$$R^{st}(t,t^b) = \mathcal{R}(\psi^{st}(t,t^b)) \tag{8}$$

where

$$\psi^{st}(t,t^b) = \frac{\theta}{\theta_p}\psi_p^{st}(t,t^b) + \frac{\theta}{\theta_w}\psi_w^{st}(t,t^b) \tag{9}$$

Proof Because the failure rate of cold standbys is considered to be negligible, $\forall\tau<t^b: R^{st}(\tau) = \mathcal{R}(0)$. After becoming an active one, a core alternates between wait state and processing state. Except for this note, the proof for Theorem 4 is as same as the proof for Theorem 2. □

Similar to the analysis in section 3.2.1, the probability that a certain core with birth time t^b fails at time $t_{\ell+1}$ is given by

$$f^{st}(t_{\ell+1},t^b)\mathrm{d}t_{\ell+1} = \frac{\mathrm{d}}{\mathrm{d}t}(1-R^{st}(t,t^b))\Big|_{t=t_{\ell+1}}\mathrm{d}t_{\ell+1} \tag{10}$$

3.3 Lifetime Reliability of the Entire Manycore System

After obtaining the lifetime reliability of a single core, we move to study the lifetime reliability of the entire manycore system in this section. Again, we first introduce the reliability of gracefully degrading systems and then investigate that of standby redundant systems.

3.3.1 Gracefully Degrading System

As the manycore system is functioning when it contains no less than k good cores, it may contain $k,k+1,\cdots,n$ good cores and all are in active mode. Let $P_{n-\ell}^{sys,de}(t)$ be the probability that the manycore system has $(n-\ell)$ active cores at time t. The system reliability $P^{sys,de}(t)$ can therefore be expressed as

$$P^{sys,de}(t) = \sum_{\ell=0}^{n-k} P_{n-\ell}^{sys,de}(t) \tag{11}$$

Hence, the mean time to failure of the system can be written as

$$MTTF^{sys,de} = \int_0^\infty P^{sys,de}(t)\mathrm{d}t = \int_0^\infty \sum_{\ell=0}^{n-k} P_{n-\ell}^{sys,de}(t)\mathrm{d}t = \sum_{\ell=0}^{n-k}\int_0^\infty P_{n-\ell}^{sys,de}(t)\mathrm{d}t \tag{12}$$

Clearly, it is necessary to determine $P_{n-\ell}^{sys,de}(t)$ to compute $MTTF^{sys,de}$. Since all components are in active mode before the first core failure, $P_n^{sys,de}(t)$ is simply the probability that all n cores survive at time t, i.e.,

$$P_n^{sys,de}(t) = \left(R^{de}(t)\right)^n \tag{13}$$

The event that the system experiences exactly one failure before t is a union of a set of continuous elementary events in which a failure occurs in an infinitesimal interval $\mathrm{d}t_1$ at time t_1; the probability that a certain core fails during $\mathrm{d}t_1$ is $f^{de}(t_1)\mathrm{d}t_1$. After this failure, the system is reconfigured as a system with $(n-1)$ cores in a very short reconfiguration time. Because of this failure, the load on each surviving cores increases. The load strongly influence the aging effect of the remaining cores. Therefore, the probability that all other $(n-1)$ cores survive up to time t is given by

$$P_{n-1}^{sys,de}(t|t_1) = \left(R^{de}(t|t_1)\right)^{n-1}, \qquad t \geq t_1 \tag{14}$$

This elementary event consists of two independent events. Therefore, the probability for this elementary event is $P_{n-1}^{sys,de}(t|t_1) \cdot f^{de}(t_1)\mathrm{d}t_1$. By the theorem of total probability, the probability of the union event $P_{n-1}^{sys,de}(t)$ is obtained by integration over t_1. Besides, since there are n good cores before the first failure, we obtain

$$P_{n-1}^{sys,de}(t) = n \cdot \int_0^t P_{n-1}^{sys,de}(t|t_1) f^{de}(t_1)\mathrm{d}t_1 \tag{15}$$

Using the same argument, by extending the event of only one failure to include ℓ failures at time $t_1, t_2, \cdots, t_\ell$, the general term [3] can be written as Equation (16).

where

$$P_{n-\ell}^{sys,de}(t|t_1,t_2,\cdots,t_\ell) = \left(R^{de}(t|t_1,t_2,\cdots,t_\ell)\right)^{n-\ell} \tag{17}$$

3.3.2 Standby Redundant System

In a standby redundant manycore system, it is functioning if it contains at least k good cores, i.e., $|\mathcal{S}_1| + |\mathcal{S}_2| \geq k$. Among these good cores, k of them are configured as active ones. The number of spare cores in the system thus can be $0, 1, \cdots, (n-k)$. Therefore, similar to the analysis of gracefully degrading system, we have

$$MTTF^{sys,st} = \int_0^\infty \sum_{\ell=0}^{n-k} P_{k,n-k-\ell}^{sys,st}(t)\mathrm{d}t = \sum_{\ell=0}^{n-k} \int_0^\infty P_{k,n-k-\ell}^{sys,st}(t)\mathrm{d}t \tag{18}$$

where, $P_{k,n-k-\ell}^{sys,st}(t)$ is the probability that a manycore system has exactly k active cores and $(n-k-\ell)$ spare cores at time t.

The probability that no failure occurs in such a system up to time t equals to the probability that all active cores with birth time zero survives at t, that is,

$$P_{k,n-k}^{sys,st}(t) = \left(R^{st}(t)\right)^k \tag{19}$$

To compute the probability that exactly one core failure occurs up to t, we firstly analyze the event that a certain core with birth time zero fails in a small interval $\mathrm{d}t_1$ at t_1; the probability for this is $f^{st}(t_1,t_0)\mathrm{d}t_1$ (for ease of discussion, let $t_0 \equiv 0$). In addition, the probability that the remaining $(k-1)$ active cores with birth time zero and the active core with birth time t_1 survive at t can be expressed as

$$P_{k,n-k-1}^{sys,st}(t|t_1) = \left(R^{st}(t,t_0)\right)^{k-1} \cdot R^{st}(t,t_1) \tag{20}$$

Again, the unconditional probability is obtained by multiplying the probabilities of two independent events and integration over t_1. Also, because there are k cores with birth time zero in the system before the first failure, we have

$$P_{k,n-k-1}^{sys,st}(t) = k \cdot \int_0^t \mathrm{d}t_1 \left(R^{st}(t,t_0)\right)^{k-1} R^{st}(t,t_1) f^{st}(t_1,t_0) \tag{21}$$

The reliability of a core depends on its birth time, independent of the occurrence time of past failures of manycore system. According to their birth times, active cores in a manycore system may belong to more than one type. Although the first failure of the manycore system must occur on a core with birth time zero, the following ones may occur on other types of cores. For example, after the first failure, there are two types of cores in the system: $(k-1)$ cores with birth time zero and one core with birth time t_1. The second failure may happen on either one of them. Consequently, to compute the reliability of a manycore system having two failures, we should consider two cases: the birth time of two failure cores is t_0, t_0; or t_0, t_1. Each one corresponds to a different failure rate. Therefore, when $\ell \geq 2$ the expression of $P_{k,n-k-\ell}^{sys,st}(t)$ is more complex. Different from the analysis of gracefully degrading system, the i^{th} failure of the manycore system with standby redundant scheme should be described by two parameters: occurrence time t_i and index of birth time x_i, representing that the i^{th} failure occurs on a core with birth time t_{x_i} at time t_i. Then, the key features of past ℓ failures can be captured by two $1 \times \ell$ vectors: $\mathbf{t}_{1\times\ell} = (t_1, t_2, \cdots, t_\ell)$ and $\mathbf{x}_{1\times\ell} = (x_1, x_2, \cdots, x_\ell)$.

To preserve the order of failures, vector $\mathbf{t}_{1\times\ell}$ satisfies: $t_1 < t_2 < \cdots < t_\ell$. In addition, since the i^{th} failure must occur on a core that is initially configured as an active one or activated because of the past $(i-1)$ failures, the birth time $t_{x_i} \in$ set $\{t_0, t_1, \cdots, t_{i-1}\}$. In other words, vector $\mathbf{x}_{1\times\ell}$ satisfies $\forall 1 \leq i \leq \ell : x_i = 0, 1, \cdots, i-1$. Also, since there are at most one core with birth time t_i $(1 \leq i \leq \ell)$ in the manycore system having ℓ core failures, the birth time index vector $\mathbf{x}_{1\times\ell}$ also satisfies $\forall x_i, x_j \neq 0 : x_i \neq x_j$. Let $\pi_{i,j}$ be the number of i's in the first j elements of vector $\mathbf{x}_{1\times\ell}$. It is also important to note that the number of cores with birth time zero in a manycore system is no more than k. Thus, vector $\mathbf{x}_{1\times\ell}$ should also meet the constraint $\pi_{0,\ell} \leq k$.

Initially, there are k active cores with birth time zero in the manycore system. After ℓ failures, $\pi_{0,\ell}$ of them fails and the remaining $(k - \pi_{0,\ell})$ are still functioning. After the i^{th} failure, a core is activated and its birth time is t_i. At time t, it may be either functioning or failed, which is described by $\mathbf{x}_{1\times\ell}$. That is, the number of active cores with birth time t_i $(0 < i \leq \ell)$ is $(1 - \pi_{i,\ell})$. The conditional probability that, given the past ℓ failures described by $\mathbf{t}_{1\times\ell}$ and $\mathbf{x}_{1\times\ell}$, no more failures occur up to time t $(t > t_\ell)$ is therefore expressed as

$$P_{k,n-k-\ell}^{sys,st}(t|\mathbf{t}_{1\times\ell};\mathbf{x}_{1\times\ell}) = \left(R^{st}(t,t_0)\right)^{k-\pi_{0,\ell}} \cdot \prod_{i=1}^{\ell} \left(R^{st}(t,t_i)\right)^{1-\pi_{i,\ell}} \tag{22}$$

Next, consider the event that the r^{th} failure occurs in a small interval $\mathrm{d}t_r$ at t_r on a certain core with birth time t_{x_r} given past $(r-1)$ failures' description; the probability for this is $f^{st}(t_r, t_{x_r})\mathrm{d}t_r$.

Now, we are able to compute the probability that the past ℓ failures of the manycore system. It can be described by vector $\mathbf{x}_{1\times\ell}$ as Equation (23), where term $N(x_r|\mathbf{x}_{1\times\ell}^{(r-1)})$ denotes the number of active cores with birth time t_{x_r} before the r^{th} failure in the system, and $\mathbf{x}_{1\times\ell}^{(r-1)} = (x_1, x_2, \cdots, x_{r-1})$ indicates the birth time index of first $(r-1)$ failures. When it comes to the computation of $N(x_r|\mathbf{x}_{1\times\ell}^{(r-1)})$, two cases are considered: the r^{th} failure occurs on a core initially configured as active one (i.e., $x_r = 0$) or spare one (i.e., $x_r \neq 0$). For the first case, there are $(k - \pi_{0,r-1})$ cores in the system belonging to this type before the r^{th} failure; for the second case, only one core has birth time t_{x_r}. Therefore, we have

$$N(x_r|\mathbf{x}_{1\times\ell}^{(r-1)}) = \begin{cases} k - \pi_{0,r-1}, & x_r = 0 \\ 1, & \text{otherwise} \end{cases} \tag{24}$$

To cover all possible failure cases, let $\mathcal{X}_{1\times\ell}$ be the set of all possible $\mathbf{x}_{1\times\ell}$. $P_{k,n-k-\ell}^{sys,st}(t)$ can be expressed as

$$P_{k,n-k-\ell}^{sys,st}(t) = \sum_{\mathbf{x}_{1\times\ell} \in \mathcal{X}_{1\times\ell}} P_{k,n-k-\ell}^{sys,st}(t, \mathbf{x}_{1\times\ell}) \tag{25}$$

[3]Multiple integrals can be effectively and efficiently calculated by using Monte Carlo simulation [6].

$$P^{sys,de}_{n-\ell}(t) = n\cdot(n-1)\cdots(n-\ell+1)\cdot \int_0^t \mathrm{d}t_\ell \int_0^{t_\ell} \mathrm{d}t_{\ell-1} \int_0^{t_{\ell-1}} \mathrm{d}t_{\ell-2} \cdots \int_0^{t_3} \mathrm{d}t_2 \int_0^{t_2} \mathrm{d}t_1 P^{sys,de}_{n-\ell}(t|t_1,t_2,\cdots,t_\ell) f^{de}(t_1) f^{de}(t_2|t_1)\cdots f^{de}(t_\ell|t_1,t_2,\cdots,t_{\ell-1}) \tag{16}$$

$$P^{sys,st}_{k,n-k-\ell}(t,\mathbf{x}_{1\times\ell}) = \prod_{r=1}^{\ell} N(x_r|\mathbf{x}^{(r-1)}_{1\times\ell}) \cdot \int_0^t \mathrm{d}t_\ell \int_0^{t_\ell} \mathrm{d}t_{\ell-1} \int_0^{t_{\ell-1}} \mathrm{d}t_{\ell-2} \cdots \int_0^{t_3} \mathrm{d}t_2 \int_0^{t_2} \mathrm{d}t_1 P^{sys,st}_{k,n-k-\ell}(t|\mathbf{t}_{1\times\ell};\mathbf{x}_{1\times\ell}) f^{st}(t_1,t_{x_1}) f^{st}(t_2,t_{x_2})\cdots f^{st}(t_\ell,t_{x_\ell}) \tag{23}$$

4. NUMERICAL RESULTS

4.1 Experimental Setup

In this section, we present results for the lifetime reliability of manycore systems with different redundancy schemes and various workloads. Two widely-used non-exponential lifetime distributions are assumed in the experiments: Weibull and Linear Failure Rate, whose reliability functions can be written as $\mathcal{R}(t) = e^{-(\frac{t}{\theta})^\beta}$ and $\mathcal{R}(t) = e^{-a\cdot(\frac{t}{\theta})-b\cdot(\frac{t}{\theta})^2}$, respectively. θ is the scale parameter, and they are different in processing state (θ_p) and wait state (θ_w). Typically they are in unit of years or hours. Clearly, $\theta_p < \theta_w$.

The property of the Weibull distribution, whose failure rate function $h(t) = \frac{\beta}{\theta}\cdot(\frac{t}{\theta})^{\beta-1}$, highly depends on its shape parameter β. When $\beta = 1$, a Weibull distribution reduces to an exponential one, i.e., the failure rate is constant. For $\beta > 1$, it has increasing failure rate. For $0 < \beta < 1$, the failure rate is decreasing with respect to time. We set $\beta = 4$ in our experiment. Linear failure rate distribution has hazard function $h(t) = \frac{a}{\theta} + \frac{2b}{\theta^2}\cdot t$, where $a, b \geq 0$. When $b = 0$, it reduces to an exponential distribution; when $a = 0$, it becomes a Rayleigh distribution. Different from Weibull distribution, linear failure rate distribution may have non-zero failure rate at $t = 0$. We set $a = 0.03$, $b = 0.15$ in our experiments.

Also, we set the number of embedded cores in the manycore system to be $(32+m)$. Therefore, if it is configured as a gracefully degrading system, there are initially $(32+m)$ active cores; while if standby redundancy configuration is used, it consists of 32 active cores and m spare cores at time zero.

4.2 Experimental Results and Discussion

First of all, we discuss an issue that attracts the most attention: how much benefit can be expected from adding core-level redundancy into a manycore system? As shown in Table 1, if we assume an exponential lifetime distribution, the sojourn time only depends on the number of active cores in the system, independent of aging effect. We therefore observe great lifetime enhancement (around m times extension), as shown in the last column of Table 1, where we set $\theta_w = \theta_p = 7$. Apparently, this result does not conforms to our common sense. In practice, IC products experience increasing failure rates. Therefore, if we use Weibull or linear failure rate distribution to approximate such wearout effect, we are able to achieve more reasonable results.

Fig. 3 shows the lifetime enhancement achieved by core-level redundancy with Weibull and Linear failure distributions. The larger m is, the longer lifetime of the manycore system at a larger area overhead. With the increase of m, the lifetime improvement gradually slows down. For example, see the curve for the $\theta_p = 3$, $\theta_w = 25$ case in Fig. 3(b). The addition of first redundant core results in 60.8% lifetime extension; those of the second, third, and fourth one lead to 45.2%, 42.2%, 23.3% extension, respectively. Consequently, designers need to set m with an appropriate value to tradeoff area overhead with lifetime extension, rather than set m as large as possible under the area overhead constraints.

When m is fixed, from Fig. 3, we can observe that the lifetime enhancement also depends on the scale parameters (i.e., θ_p and θ_w) in the reliability functions. There are two extreme cases for the relationship between these two parameters. When $\theta_p = \theta_w$, there is no difference between wait state and processing state in terms of reliability function. Essentially, this case is the so-called hot standby scheme. Another extreme case is $\theta_w \to \infty$, which means an embedded core in wait state is essentially a cold standby component and cannot fail. In other cases (e.g., $\theta_p = 3$ and $\theta_w = 10$), embedded cores in wait state serve as the warm standbys. Given the same θ_p, since a core's accumulated time in either processing or wait state is independent of scale parameters, it is reasonable to expect that the lifetime increases as θ_w increases.

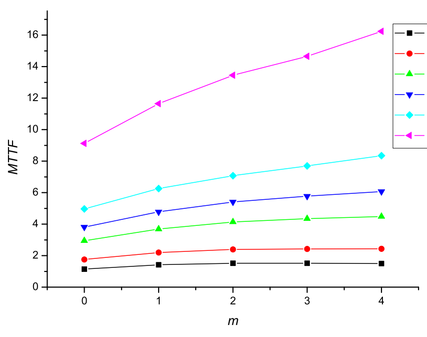
A closer observation for both redundant schemes is shown in Table 2 and Table 3. As an example, we set $\theta_p = 3$, $\theta_w = 10$, and $\frac{\lambda}{\mu} = 10$. Comparing Due to the increasing failure rate, the manycore system contains no faulty cores in most of its lifetime, especially for systems experiencing more severe wearout effects. For example, as shown in Line 7 of Table 2, the sojourn time of a gracefully degrading system with 4 redundant cores in 0-failure state is 2.2452 years, while the expected value of its whole lifetime is 3.2864 years. In such case, one core's failure may imply the entire system is old and we cannot expect much residual useful lifetime.

Next, we show the impact of workload, namely $\frac{\lambda}{\mu}$, on the lifetime reliability, as depicted in Fig. 4. We set $\theta_p = 3$; results are presented for four different cases. When the workload increases, the system's lifetime significantly decreases (e.g., see the curves for $\theta_w = 10$, Weibull distribution case). But the scale of the decrease of lifetime is smaller than that of the increase of workload. We attribute this phenomenon to the wearout effects of warm standbys. Nevertheless, the workload has significant influence on the lifetime reliability of manycore systems and should be paid much attention by designers.

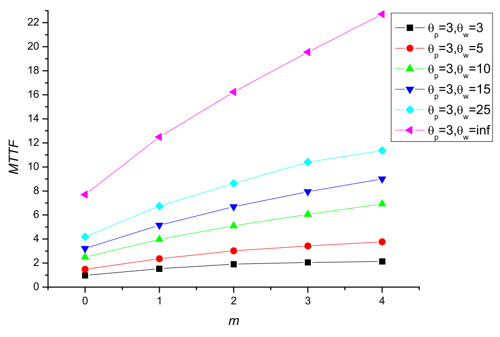
Table 4 compares the manycore system's lifetime reliability in gracefully degrading scheme and the one in standby redundant scheme. We set $\theta_p = 3$ and $\frac{\lambda}{\mu} = 20$. It can be observed that, the difference between these two schemes highly depends on the core failure function in wait state. Most prior work based on the hot standby assumption (e.g., [2]) claim that the standby redundant system has longer lifetime but worse performance when compared with the gracefully degrading system. When we assume the embedded cores in wait state have the same failure functions as that in processing state (i.e., hot standby), our model leads to the same conclusion. However, if we consider the realistic warm standby situation, the difference is smaller than that based on the hot standby assumption (see Columns 5-8). Moreover, when the aging effect in wait state is much slower than that in processing state, the lifetime of the standby redundant system may be even shorter than that of the gracefully degrading system (e.g., the eighth column). An extreme case is when it is assumed to be cold standby, the gracefully degrading system is better (the last column). In this sense, a reasonable assumption is important because it is able to prevent designers from misleading conclusions. In addition, this difference is also dependent on m. It is very small when m is small, and it becomes slightly larger with the increase of m. Both Weibull and linear failure rate distributions follows the above observations.

m	Redundancy Scheme	Sojourn Time (years)					$MTTF^{sys,de}$
		0-Failure State	1-Failure State	2-Failure State	3-Failure State	4-Failure State	
0	—	0.2188	—	—	—	—	0.2188
1	Degrading	0.2121	0.2188	—	—	—	0.4309
	Standby	0.2188	0.2188	—	—	—	0.4376
2	Degrading	0.2059	0.2121	0.2188	—	—	0.6368
	Standby	0.2188	0.2188	0.2188	—	—	0.6564
3	Degrading	0.2000	0.2059	0.2121	0.2188	—	0.8368
	Standby	0.2188	0.2188	0.2188	0.2188	—	0.8752
4	Degrading	0.1944	0.2000	0.2059	0.2121	0.2188	1.0312
	Standby	0.2188	0.2188	0.2188	0.2188	0.2188	1.0940

Table 1: Lifetime Reliability of Manycore System with Constant Failure Rate.



(a) Weibull Distribution



(b) Linear Failure Rate Distribution

Figure 3: Lifetime Enhancement of Manycore System.

Distribution	m	Sojourn Time (years)					$MTTF^{sys,de}$
		0-Failure State	1-Failure State	2-Failure State	3-Failure State	4-Failure State	
Weibull	0	2.2039	—	—	—	—	2.2039
	1	2.2153	0.5573	—	—	—	2.7726
	2	2.2260	0.5600	0.3055	—	—	3.0915
	3	2.2359	0.5626	0.3142	0.1040	—	3.2167
	4	2.2452	0.5649	0.2988	0.0955	0.0820	3.2864
Linear Failure Rate	0	1.8572	—	—	—	—	1.8572
	1	1.8463	1.1367	—	—	—	2.9830
	2	1.8354	1.1325	0.8926	—	—	3.8605
	3	1.8243	1.1282	0.8798	0.6941	—	4.5264
	4	1.8133	1.1237	0.8762	0.7055	0.6269	5.1456

Table 2: Lifetime Reliability for Non-Exponential Lifetime Distribution (Gracefully Degrading System).

Distribution	m	Sojourn Time (years)					$MTTF^{sys,de}$
		0-Failure State	1-Failure State	2-Failure State	3-Failure State	4-Failure State	
Weibull	0	2.2039	—	—	—	—	2.2039
	1	2.2039	0.5617	—	—	—	2.7656
	2	2.2039	0.5617	0.3156	—	—	3.0812
	3	2.2039	0.5617	0.3156	0.0966	—	3.1778
	4	2.2039	0.5617	0.3156	0.0966	0.0578	3.2356
Linear Failure Rate	0	1.8572	—	—	—	—	1.8572
	1	1.8572	1.1419	—	—	—	2.9991
	2	1.8572	1.1419	0.9110	—	—	3.9101
	3	1.8572	1.1419	0.9110	0.6879	—	4.5980
	4	1.8572	1.1419	0.9110	0.6879	0.6148	5.2128

Table 3: Lifetime Reliability for Non-Exponential Lifetime Distribution (Standby Redundant System).

Distribution	m	Redundancy Scheme	$MTTF^{sys}$					
			Hot Standby	Warm Standby				Cold Standby
				$\theta_w = 5$	$\theta_w = 10$	$\theta_w = 15$	$\theta_w = 25$	
Weibull	2	Degrading	1.5039	1.8232	2.1497	2.2930	2.4265	2.6258
		Standby	1.5314	1.8227	2.1133	2.2488	2.3484	2.5309
	4	Degrading	1.5046	1.8521	2.2305	2.4432	2.5771	2.8376
		Standby	1.5577	1.8545	2.1715	2.3103	2.4266	2.6261
Linear Failure Rate	2	Degrading	1.9115	2.3197	2.7070	2.8697	3.0105	3.2424
		Standby	1.9608	2.3314	2.7330	2.8851	3.0091	3.2146
	4	Degrading	2.1348	2.7122	3.3642	3.6529	3.9385	4.3590
		Standby	2.3008	2.7899	3.4307	3.6015	3.8588	4.1881

Table 4: Comparison Between Gracefully Degrading System and Standby Redundant System.

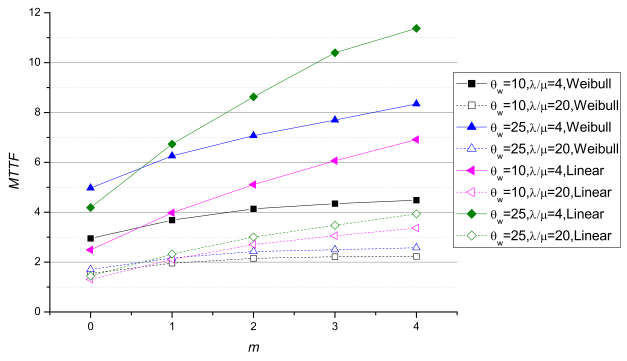


Figure 4: The Impact of Workload.

5. CONCLUSION AND FUTURE WORK

State-of-the-art technology enables the integration of a great amount of embedded cores in a single computing system. The lifetime reliability of such large circuit is a major concern because IC failure mechanisms have an increasingly adverse impact with technology scaling. In this paper, we propose a comprehensive analytical model to estimate the lifetime reliability of manycore systems, which is able to facilitate designers to make a better decision at the architecture level.

The model presented in this paper could be extended to take some other aspects into account. For example, in this model we assume the wearout of active cores can be in two discrete states with different reliability functions. In practice, as the wearout effects highly depend on temperature and voltage, cores in the same active state can experience different reliability functions. The proposed model can be extended to deal with continuous states by expressing the scale parameter as a function of temperature and voltage. Also, the assumption that the failure rate in spare state is negligible can be abandoned.

Appendix A. Proof of Theorem 2

Any core can start with either wait state or processing state at time zero. Suppose a core is in wait state from 0 to T_1, and then converts into the processing state and stays for $(T_2 - T_1)$, and so on. Assuming it is surviving at t, we obtain a a subdivision of the time $[0,t]$: $0 = T_0 < T_1 < T_2 < \cdots < T_d = t$. We know the initial reliability of this core is given by $R^{de}(0) = \mathcal{R}(0)$. By the definition of reliability function in the wait state, the reliability of this core in the first interval $[T_0, T_1)$ is

$$R^{de}(\tau) = \mathcal{R}(\frac{\theta}{\theta_w} \cdot (\tau - T_0)), \qquad T_0 \le \tau < T_1 \qquad \text{(A-1)}$$

At the end of this interval

$$R^{de}(T_1^-) = \mathcal{R}(\frac{\theta}{\theta_w} \cdot (T_1 - T_0)) \qquad \text{(A-2)}$$

Since this core follows the reliability function in the processing state in the second interval $[T_1, T_2)$, we have

$$R^{de}(\tau) = \mathcal{R}(\frac{\theta}{\theta_p} \cdot (c + \tau - T_1)), \qquad T_1 \le \tau < T_2 \qquad \text{(A-3)}$$

where, c reflects the aging effect of the first interval.

At the beginning of this interval

$$R^{de}(T_1^+) = \mathcal{R}(\frac{\theta}{\theta_p} \cdot c) \qquad \text{(A-4)}$$

By its continuity, the reliability function must satisfy $R^{de}(T_1^-) = R^{de}(T_1^+)$. Combining Equation (A-2) and (A-4) yields $c = \frac{\theta_p}{\theta_w} \cdot (T_1 - T_0)$. Therefore

$$R^{de}(\tau) = \mathcal{R}(\frac{\theta}{\theta_p} \cdot (\tau - T_1) + \frac{\theta}{\theta_w} \cdot (T_1 - T_0)), \qquad T_1 \le \tau < T_2 \qquad \text{(A-5)}$$

After finishing this procedure, we obtain the reliability of a single core at time t

$$R^{de}(t) = \mathcal{R}(\frac{\theta}{\theta_p} \cdot \sum_{\substack{i=1 \\ i \text{ is even}}}^{d} (T_i - T_{i-1}) + \frac{\theta}{\theta_w} \cdot \sum_{\substack{i=1 \\ i \text{ is odd}}}^{d} (T_i - T_{i-1})) \qquad \text{(A-6)}$$

where, $\sum_{\substack{i=1 \\ i \text{ is even}}}^{d} (T_i - T_{i-1})$ and $\sum_{\substack{i=1 \\ i \text{ is odd}}}^{d} (T_i - T_{i-1})$ are the accumulated time in the processing and wait state respectively. Apparently, the conclusion does not depend on the starting state of this core. Therefore, Theorem 2 holds.

6. REFERENCES

[1] A. Agarwal and M. Levy. The KILL Rule for Multicore. In *Proceedings ACM/IEEE Design Automation Conference (DAC)*, pages 750–753, 2007.

[2] M. D. Beaudry. Performance-related reliability measures for computing systems. *IEEE Transactions on Computers*, 27(6):540–547, June 1978.

[3] S. Borkar. Designing Reliable Systems from Unreliable Components: The Challenges of Transistor Variability and Degradation. 25(6):10–16, Nov.-Dec. 2005.

[4] S. Borkar. Thousand Core Chips - A Technology Perspective. In *Proceedings ACM/IEEE Design Automation Conference (DAC)*, pages 746–749, 2007.

[5] A. Coskun, T. Rosing, K. Mihic, G. D. Micheli, and Y. L. Lebici. Analysis and optimization of mpsoc reliability. *Journal of Low Power Electronics*, 15(2):159–172, February 2006.

[6] A. Dubi. In *Monte Carlo Applications in System Engineering*. John Wiley & Sons, 1999.

[7] D. Geer. Chip Makers Turn to Multicore Processors. *IEEE Computer*, 38(5):11–13, May 2005.

[8] L. R. Goel, R. Gupta, and R. K. Agnihotri. Analysis of a three-unit redundant system with two types of repair and inspection. *Microelectron Reliability*, 29(5):769–773, 1989.

[9] T. F. Hassett, D. L. Dietrich, and F. Szidarovszky. Time-varying failure rates in the availability and reliability analysis of repairable systems. *IEEE Transactions on Reliability*, 44:155–160, March 1995.

[10] W. Kuo and M. J. Zuo. In *Optimal Reliability Modeling: Principles and Applications*. John Wiley & Sons, 2003.

[11] Y.-H. Lee and C. Chen. A two-level scheduling method: An effective parallelizing technique for uniform nested loops on a dsp multiprocessor. *Journal of Systems and Software*, 75:155–170, 2005.

[12] H.-H. Lin, K.-H. Chen, and R.-T. Wang. A multivariate exponential shared-load model. *IEEE Transactions on Reliability*, 42(1):165–171, March 1993.

[13] H. Liu. Reliability of a load-sharing k-out-of-n:g system: Non-iid components with arbitrary distributions. *IEEE Transactions on Reliability*, 47(3):279–184, September 1998.

[14] R. Nelson, D. Towsley, and A. N. Tantawi. Performance analysis of parallel processing systems. *IEEE Transactions on Sofeware Engineering*, 14(4):532–540, April 1988.

[15] Nvidia. Geforce 8800 graphics processors. http://www.nvidia.com/page/geforce 8800.html.

[16] J. Shao and L. R. Lamberson. Modeling a shared-load k-out-of-n: G system. *IEEE Transactions on Reliability*, 40(2):205–209, June 1991.

[17] J. She and M. G. Pecht. Reliability of a k-out-of-n warm-standby system. *IEEE Transactions on Reliability*, 41(1):72–75, March 1992.

[18] J. Shin, V. Zyuban, Z. Hu, J. Rivers, and P. Bose. A Framework for Architecture-Level Lifetime Reliability Modeling. In *Proceedings IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 534–53, 2007.

[19] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers. The Case for Lifetime Reliability-Aware Microprocessors. In *Proceedings IEEE/ACM International Symposium on Computer Architecture (ISCA)*, pages 276–287, 2004.

[20] S. Srinivasan, P. Mangalagiri, Y. Xie, N. Viiayhrishnan, and K. Sarpatwari. FLAW: FPGA Lifetime Awareness. In *Proceedings ACM/IEEE Design Automation Conference (DAC)*, pages 630–635, 2006.

[21] R. Subramanian and V. Anantharaman. Reliability analysis of a complex standby redundant system. *Reliability Engineering and System Safety*, 48:57–70, 1995.

[22] Tilera. Tile64 processor family. http://www.tilera.com/products/processors.php.

[23] M. Xie, Y.-S. Dai, and K.-L. Poh. In *Computing Systems Reliability: Models and Analysis*. Kluwer Academic Publishers, 2004.

[24] L. Zhang, Y. Han, Q. Xu, and X. Li. Defect Tolerance in Homogeneous Manycore Processors Using Core-Level Redundancy with Unified Topology. In *Proceedings Design, Automation, and Test in Europe (DATE)*, 2008.

[25] T. Zhang, M. Xie, and M. Horigome. Availability and reliability of k-out-of-$(m+n)$:g warm standby systems. *Reliability Engineering and System Safety*, 91:381–387, 2006.