Homework#3 for ECE 254 / CPS 225
Due in class on Monday, October 27

All homework must be done individually or in groups of two
(and I’d really prefer groups of two!)

1 Backward Error Recovery
1.1) [10] Unlike in FER systems, BER systems have a window of opportunity in which a fault can occur during the recovery of a previous fault. For a system that performs BER by periodically checkpointing its state to disk, how can it deal with this situation?

1.2) [10] If database software can perform checkpoint/recovery that is perfectly reliable, is there any need for hardware BER on a system that runs this database? Think about the end-to-end argument and its pros and cons.

2 Fault Tolerant Microprocessors
2.1) [10] Discuss the errors that cannot be detected by Argus.

3 Fault Tolerant Memory
3.1) [10] Explain how Chipkill Memory (a.ka., RAID-M) works.

3.2) [15] A recent research paper (from DSN 2003) proposed using invalid lines in the L1 cache to hold replicas of valid blocks. For example, a store to a given cache location would be mirrored by one or more stores to other locations in the cache (determined by an address hashing algorithm). The cache controller first tries the primary copy of the block, but, if an error is detected (EDC is on each cache block), the cache controller searches for an error-free replica. What are the advantages and disadvantages of this physical redundancy scheme, in terms of fault tolerance, fault-free performance, hardware cost and complexity, and power consumption? Describe an alternative approach for protecting the L1 cache and compare it to the above scheme using the same metrics.

4 Fault Tolerant Disks
4.1) [10] Look up one commercially available hard disk and describe its fault tolerance.

4.2) [10] You have a RAID-5 system with 9 disks. Each disk has a MTTF of 1 year. MTTR is one hour. Assuming that two simultaneous disk failures (i.e., one fails and then another fails before the first failure is repaired) lead to a system failure, what is the mean time between system failures?

5 Fault Tolerant Networks
5.1) [10] Can a torus network tolerate a dead switch? If so, how? If not, why not?

5.2) [10] Communication takes an average of H hops. We can either protect each link (at a cost of 10% slower links), or we can use end-to-end retry if the message is corrupted along one or more hops of its traversal (assume we can detect this with no performance overhead). Assume that link errors occur E% of the time (per hop). For what values of E and H
is link-level protection better than end-to-end retry? (If you find yourself doing very difficult math, please see me - this probably indicates that you’re mis-interpreting the question.)

5.3) [10] Look up a commercial network product or standard (Myrinet, Infiniband, ATM), and describe its fault tolerance.

6 Fault Tolerant Software (and using software to tolerate hardware faults)

6.1) [10] Why does software “age”? Describe how we might be able to write software that doesn’t age and give at least one concrete example.

6.2) [10] How does Orbitz.com provide reliable services? I expect you to have to do a little digging to find this answer, and I expect an answer that’s around half a page in length.

6.3) [15] How does Google’s redundancy compare to that of the IBM mainframes? Explain how their differences are related to their error models and the goals of their systems (including what kind of software they are expected to run).

6.4) [15] Summarize the “SWIFT” paper (Reis et al.) in one page. Include what you believe to be the paper’s contributions, strengths, and weaknesses, and justify your answers.