The University of Western Australia
Electrical & Electronics Engineering
A/Prof Thomas Bräunl

Fault Tolerant Computer Systems
FTCS 422

Tutorial 1 | Due: week 2

Preparation for Lab No. 1.

Topics:
- Fault tolerance in embedded systems
- EyeBot embedded controller
- Serial communication
- C programming
- C++ programming
- fltk graphical user interface

The University of Western Australia
Electrical & Electronics Engineering
A/Prof Thomas Bräunl

Fault Tolerant Computer Systems

FTCS 422

Tutorial 2 Due: week 3

In a chemical plant, a controller is used to maintain the temperature of a process.

The controller comprises:

- input: temperature sensor + A/D converter
- output: D/A converter plus valve motor
- CPU component

This simple version of the controller is non-redundant.

Construct a fault tolerant version of this controller by employing **active or passive hardware redundancy**. Use block diagrams for your sketch.

Please note that there are a number of different valid approaches.
Compare and discuss the merits of different approaches.

The University of Western Australia
Electrical & Electronics Engineering
A/Prof Thomas Bräunl

Fault Tolerant Computer Systems

FTCS 422

Tutorial 3 Due: week 4

Preparation for Lab No. 2.

Topics:

- Control loops
- Electrical motor control
- Mid-Point Selection
- Triple modular redundancy
- Validity checking (data tange)
- Debugging
- C programming
- C++ programming

The University of Western Australia
Electrical & Electronics Engineering
A/Prof Thomas Bräunl

Fault Tolerant Computer Systems

FTCS 422

Tutorial 4 — Due: week 5

---

1. Parity: *bit-per-byte, alternating odd-even for odd/even addresses*
(a) is this a separable code?
(b) how much extra memory do you need for 1MByte data ?
(c) what is the advantage of alternating odd/even ?
(d) what is the Hamming distance of this code ?

2. Hamming Code
the general scheme for *single-bit-correction*, *double-bit-detection* Hamming Code is as follows:

- place check bits at positions 1, 2, 4, 8, ...
  and label them c_1, c_2, c_4, c_8, ...
- place data bits at remaining positions
  and label them according to their position d_3, d_5, d_6, d_7, d_9, ...
- connect the check bits according to the scheme:

  $$\mathbf{c_{2^i} = XOR\ \{ d_k \mid bin(k)\ [i] = 1\}}$$ (bit positions are counted from 0)

  or verbally: to create each check bit, XOR all the data bits, whose position (in binary representation) has a "1" at the check bit's position.
- Example: c1 c2 d3 c4 d5 d6 d7
  then c1 = XOR(d3,d5,d7), c2 = XOR(d3, d6, d7), c3 = XOR(d5, d6, d7)

Construct this Hamming Code for 32bit words.

3. Residue Code R7 (residue of division by 7)
(a) is this a separable code?
(b) how much extra memory do you need for 1MByte data ?
(c) what is the Hamming distance of this code ?
(d) encode/compare 55 + 23
(e) encode/compare 55 – 23

The University of Western Australia
Electrical & Electronics Engineering
A/Prof Thomas Bräunl

Fault Tolerant Computer Systems

FTCS 422

Tutorial 5 Due: week 6

1. Information redundancy

(a) Parity: bit-per-byte, alternating odd-even for odd/even addresses
Construct a circuit that generates and tests this type of parity codes

(b) Hamming Code (1-error correction, 2-error detection)
Design a circuit for automatic error detection and error correction.

(c) What Hamming distance is required for 2-error correction, 3-error detection?
How many check bits are required for 4 data bits?

(d) Design a memory system with generating and checking of Berger code
with 8 information bits

2. Time Redundancy

(a) Design a program, which uses 3TR to detect and correct transient faults in a certain calculation named "f(x: integer)".

(b) Design a program, which uses shift-encoded 2TR to detect permanent faults in combinatorial logic function "g(x,y,z: byte)".

3. Software Redundancy

Discuss the various advantages and problems involced in using software redundancy.

The University of Western Australia
Electrical & Electronics Engineering
A/Prof Thomas Bräunl

Fault Tolerant Computer Systems

FTCS 422

Tutorial 6 — Due: week 7

---

1. A company manufactured components **A** and tested samples of them subsequently. From 100 tested components:

3 failed right at the start of the test
2 more failed after 1 hour
2 more failed after 2 hours

- sketch the reliability function over time
- given the testing results, do you consider 2 hours to be long enough
- how long should the "burn-in" be in general

2. Assume components **A** and **B** are now in the "useful life" period with constant failure rates $z(t) = \lambda$ and $\lambda_a=0.01$ $\lambda_b = 0.001$ failures per hour.

Average repair times are $r_a = 5$ min. $r_b = 30$ min.

- determine the MTTF for components **A** and **B**
- determine the MTBF for components **A** and **B**

3. What is the reliability for

- series system of 5 • **A**
- series system of 4 • **B**
- series system of: **A** — **B** — **B** — **A**
- parallel system: **A** || **B** || **B**
- 2-of-3 **A**
- 3-of-5 **B**

The University of Western Australia
Electrical & Electronics Engineering
A/Prof Thomas Bräunl

Fault Tolerant Computer Systems

FTCS 422

Tutorial 7 | Due: week 8

---

1. Markov Model
- Construct a Markov model for a 5MR
- construct the full model
- collapse states into categories

2. Reliability Analysis
- determine the transision probabilities
  (assuming constant failure rate)
- construct differential equations

3. – optional –
- solve differential equations
- determine MTTF

The University of Western Australia
Electrical & Electronics Engineering
A/Prof Thomas Bräunl

Fault Tolerant Computer Systems

FTCS 422

Tutorial 8 | Due: week 9

Preparation for Lab No. 5.

Topics:
- PVM Parallel Virtual Machines
- Unix systems
- Time-out programming
- Watchdogs