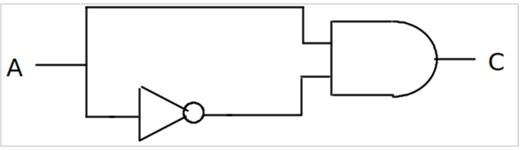
1) The following circuit contains two gates. Independent of time, it would functionally produce a constant output of C = 0. However, as this problem demonstrates, timing is a critical design parameter which must be factored into account by computer architects. The issue being examined in this problem is that of a Race or Glitch Condition. The tell-tale sign that a circuit may exhibit a timing glitch is a situation where one input path to a gate is longer that of the other. The existence of this condition is obvious and easy to spot in the circuit below; however, for complex circuits with multiple computation paths and large fan-in/fan-out between gates, the potential for hidden glitches can be quite high. It is in this area that computer-aided design tools, such as VHDL, can offer valuable assistance to the engineer in terms of predicting actual performance of the circuit as a function of time.



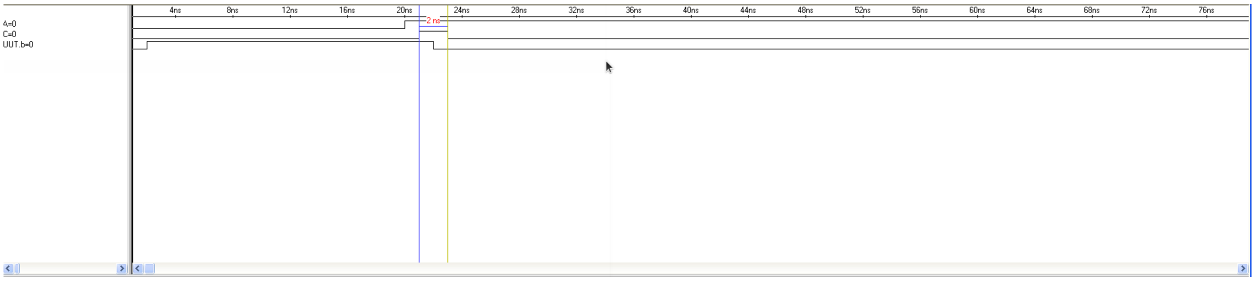
Simulate the circuit using a structural architecture to determine its output response for various combinations of delay values for each gate. For example, try all four combinations of NOT delay = (2ns or 4ns) and AND delay = (1ns or 3ns). Run a sufficient number of simulations while permuting the delay values on each of the gates so that you can answer (and provide documentation) for the questions below.

In particular, assume the actual purpose of the circuit is to deliberately produce a pulse (glitch) at output C whenever input A has a positive (0 to 1) transition. [40 pts]

Case1:

NOT gate delay 2ns, AND gate delay 1ns,

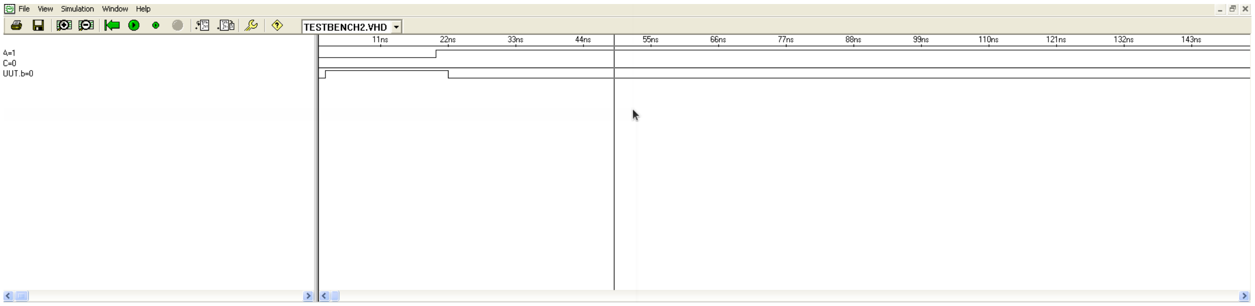
The waveform shows that 1ns (AND gate delay) after A changes, C has a pulse, and the pulse during is 2ns (NOT gate delay).



Case2:

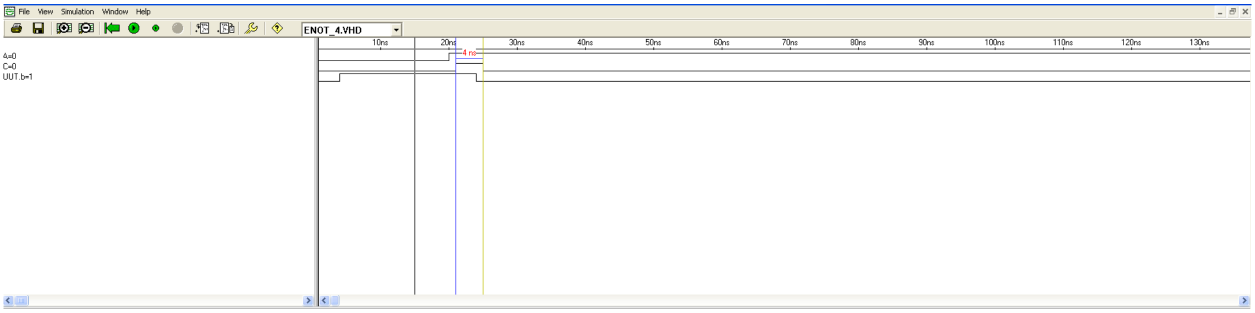
NOT gate delay 2ns, AND gate delay 3ns,

The waveform shows C has no pulse, because NOT gate delay < AND gate delay.



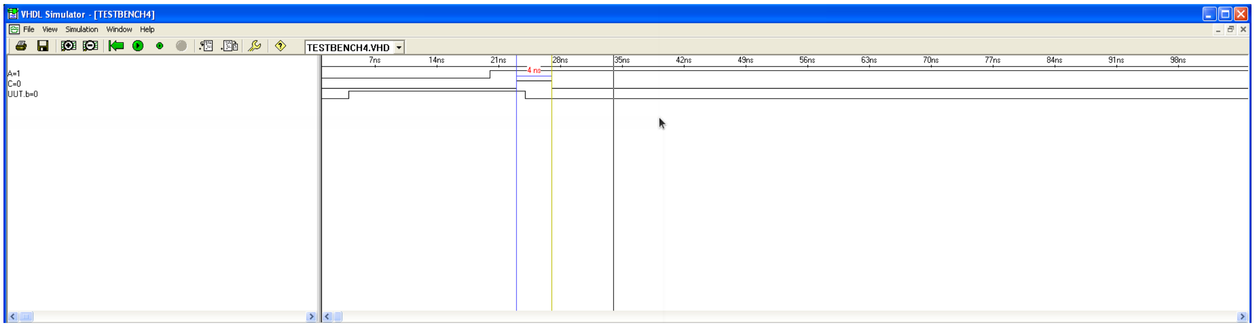
Case3:

NOT gate delay 4ns, AND gate delay 1ns,
The waveform shows that 1ns (AND gate delay) after A changes, C has a pulse, and the pulse during is 4ns (NOT gate delay).



Case4:

NOT gate delay 4ns, AND gate delay 3ns,
The waveform shows that 3ns (AND gate delay) after A changes, C has a pulse, and the pulse during is 4ns (NOT gate delay).



a) If both gates have non-zero inertial delay values, what relationship must exist between the delay time of the INVERTER and the delay time of the AND gate in order for the circuit to produce a pulse ?

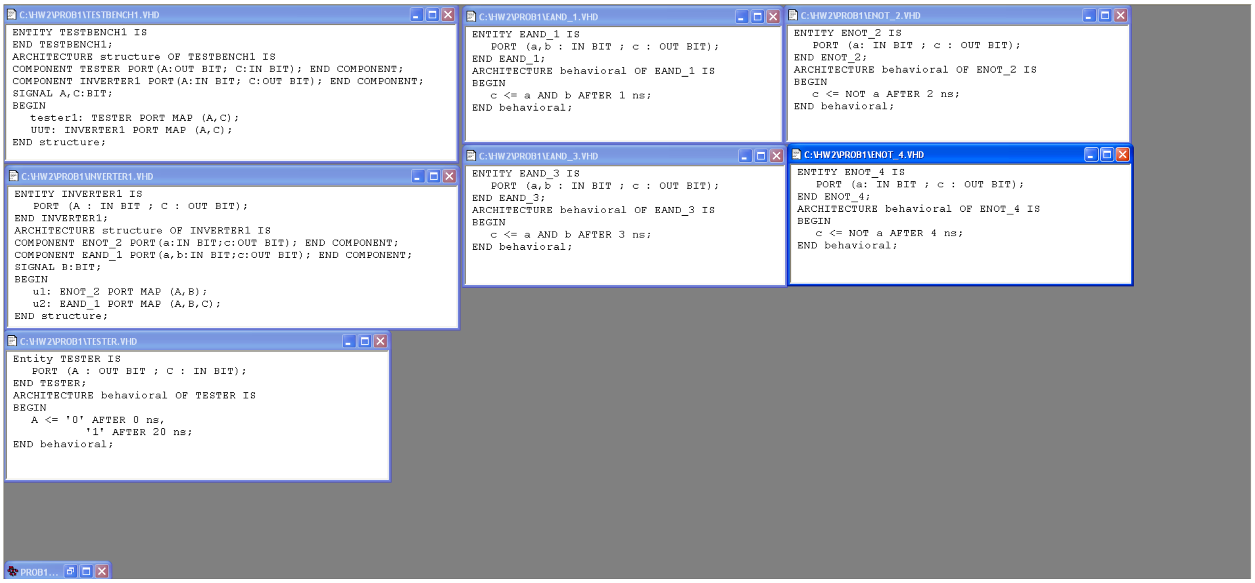
INVERTER delay time > AND gate delay time

b) Assuming the constraint in (a) above is satisfied and that a positive transition of A occurs at time T, at what time will the beginning of the pulse occur at output C ?

T + AND gate delay time

c) Assuming the constraint in (a) above is satisfied, determine an equation which will predict the width of the pulse at output C.

width of the pulse at output C = INVERTER delay time



```
ENTITY TESTBENCH1 IS
END TESTBENCH1;
ARCHITECTURE structure OF TESTBENCH1 IS
COMPONENT TESTER PORT(A:OUT BIT; C:IN BIT); END COMPONENT;
COMPONENT INVERTER1 PORT(A:IN BIT; C:OUT BIT); END COMPONENT;
SIGNAL A,C:BIT;
BEGIN
   tester1: TESTER PORT MAP (A,C);
   UUT: INVERTER1 PORT MAP (A,C);
END structure;
```

```
ENTITY INVERTER1 IS
   PORT (A : IN BIT ; C : OUT BIT);
END INVERTER1;
ARCHITECTURE structure OF INVERTER1 IS
COMPONENT ENOT_2 PORT(a:IN BIT;c:OUT BIT); END COMPONENT;
COMPONENT EAND_1 PORT(a,b:IN BIT;c:OUT BIT); END COMPONENT;
SIGNAL B:BIT;
BEGIN
   u1: ENOT_2 PORT MAP (A,B);
   u2: EAND_1 PORT MAP (A,B,C);
END structure;
```

```
Entity TESTER IS
   PORT (A : OUT BIT ; C : IN BIT);
END TESTER;
ARCHITECTURE behavioral OF TESTER IS
BEGIN
   A <= '0' AFTER 0 ns,
           '1' AFTER 20 ns;
END behavioral;
```

```
ENTITY EAND_1 IS
   PORT (a,b : IN BIT ; c : OUT BIT);
END EAND_1;
ARCHITECTURE behavioral OF EAND_1 IS
BEGIN
   c <= a AND b AFTER 1 ns;
END behavioral;
```

```
ENTITY EAND_3 IS
   PORT (a,b : IN BIT ; c : OUT BIT);
END EAND_3;
ARCHITECTURE behavioral OF EAND_3 IS
BEGIN
   c <= a AND b AFTER 3 ns;
END behavioral;
```

```
ENTITY ENOT_2 IS
   PORT (a: IN BIT ; c : OUT BIT);
END ENOT_2;
ARCHITECTURE behavioral OF ENOT_2 IS
BEGIN
   c <= NOT a AFTER 2 ns;
END behavioral;
```

```
ENTITY ENOT_4 IS
   PORT (a: IN BIT ; c : OUT BIT);
END ENOT_4;
ARCHITECTURE behavioral OF ENOT_4 IS
BEGIN
   c <= NOT a AFTER 4 ns;
END behavioral;
```

2) Because of space and power constraints, satellites often only use dual processors instead of triple processors for increased reliability. Describe various ways in which detection and/or correction can be accomplished under this architecture. Under these situations, is real-time correction of faults possible ? Specify how you define "real-time" in your answer. [15 pts]

real-time correction is possible with a dual-processor.

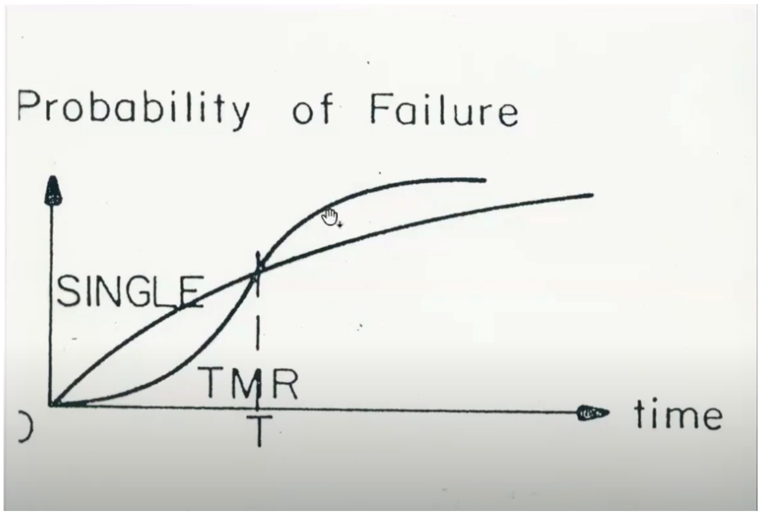
real-time correction definition: When faults are detected, a system will not restart the execution or retrieve new data from a previous task.

Generally, dual-processors will have two processors processing the same task at the same time. When any processor detects an error, the system will immediately switch to the other processor.

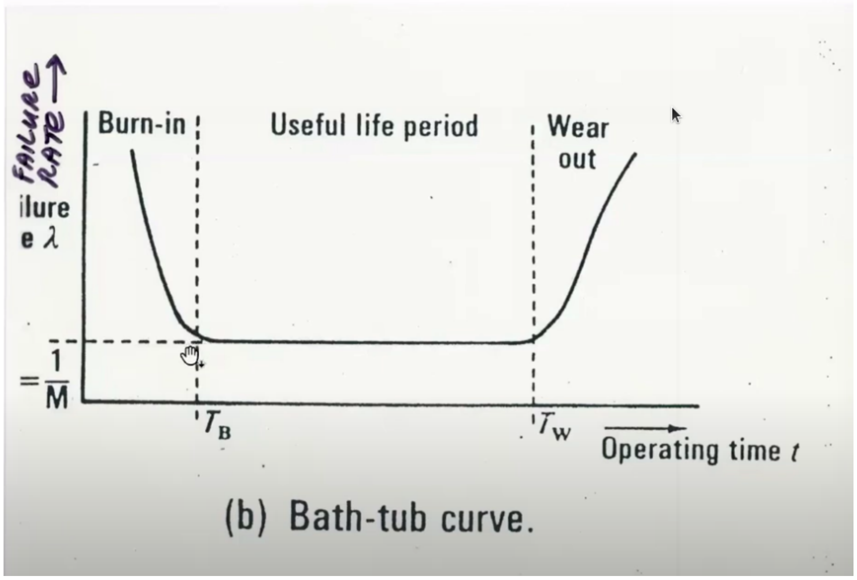
Satellites are to maximize Energy Efficient, so the two processors are divided into a primary processor and a secondary processor. A checkpoint is inserted at the completion of the primary processor. If there is no fault, the secondary processor terminates its execution. Otherwise, should the fault occur, the second processor continues and completes the application before its deadline.[1]

3) Explain the flaw in this statement: "A system with built-in TMR redundancy is always more reliable than a simplex system with no redundancy". Sketch any graphs that may be appropriate. Provide an intuitive explanation for why the graph is shaped the way it is. [15 pts]

[1] S. Hua, P. R. Pari and G. Qu, "Dual-Processor Design of Energy Efficient Fault-Tolerant System," IEEE 17th International Conference on Application-specific Systems, Architectures and Processors (ASAP'06), 2006, pp. 239-244, doi: 10.1109/ASAP.2006.27.



A system with built-in TMR redundancy is "not" always more reliable than a simplex system with no redundancy



(b) Bath-tub curve.

It is possible to analyze the Bathtub curve to explain why the Failure of TMR is higher than the simplex after T point.

Bathtub curve shows a high probability of failure during burn-in phase, That is because although all components are new, there will be manufacturer defects, algorithm defects or any defects in the components. TMR can tolerate these defects better than a simplex system. This is why in the beginning, the TMR system showed better reliability.

However, when the components enter the wear phase, the failure is increasing, and it is inevitable. TMR can tolerate defects, but it can not stop the component getting old, that depends on the material of the component rather than the algorithm. Because TMR consists of multiple simplex systems, the old parts will accumulate faster than simplex systems. That's why the TMR system dropped reliability after a certain point.

4) Newer generation cars often have adaptive cruise control (ACC). Like ordinary cruise control, it allows the driver to set a constant speed for the car without having to continually press on the gas pedal; however, it is more intelligent. A radar/laser/camera unit located behind the grille determines the speed and distance of the vehicle in front. If necessary, ACC will automatically slow down the car to maintain a safe following distance should the car in front slow. When the distance to the vehicle in front is computed to be safe again, ACC accelerates the car to resume its last speed setting. Pretend you are a designer of ACC. Identify potential failure modes that your ACC must be capable of dealing with. List as many as you can. For example, your car is running on ACC when a loose plastic grocery bag or newspaper page on the freeway is kicked up by the car in front of you and clings to your car's grille. [15 pts]

The road ahead was repaired, but the warning sign was blown down by the wind.

In heavy rain or snow, the camera or sensor will be blurred.

A "large paper box" is in the middle of the road, and it is completely safe to run over it, but the car identified it to be a dangerous obstacle.

A "small metal box" is in the middle of the road, running over from above will cause the car to tip over, but the car identified it to be a safe obstacle.

If the zombie apocalypse happens, the car will not run over the zombie, because they are identified as pedestrians crossing the road.

5) Imagine in the future, two self-driving cars get into an accident. Who's at fault ? [15 pts]

If it is only a minor accident, The collision screen can be reviewed through the car camera. The closer to the "Fail soft" principle, the less responsibility. The Fail soft principle of vehicle collision, I personally think it is braking. A stopped car will not cause any accidents, So full responsibility for the slowest braking vehicle.

If two cars collide and burn to ashes like Tesla. The principle of Bath-tub curve can be used. Check the vehicle usage history, consider the Bath-tub curve, the failure rate of new cars and old cars is relatively high. Compare the car with the worst reliability. The poor one will take full responsibility.