Master Degree in Embedded Computing Systems
A.Y. 2016- 2017

Project Report

Dependable and Secure Systems

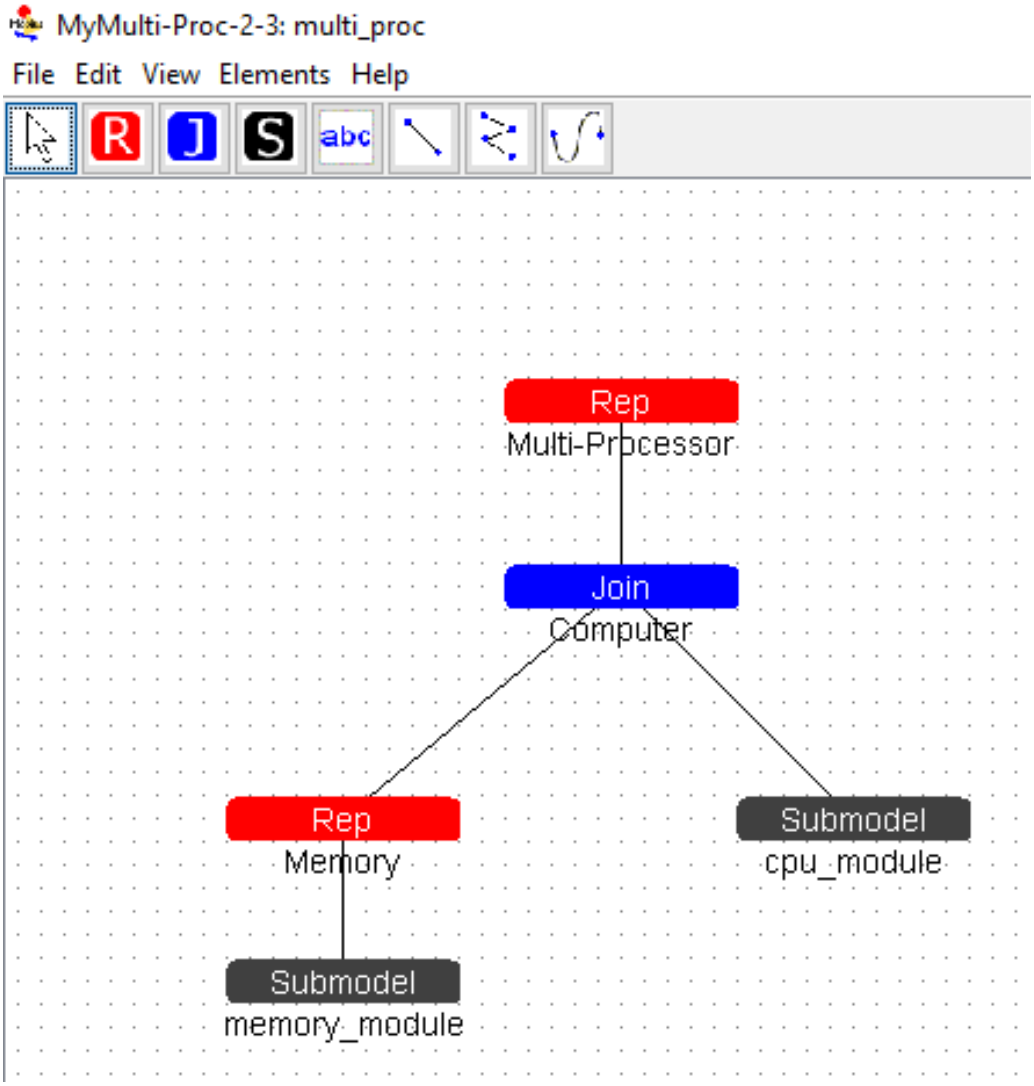
Student:

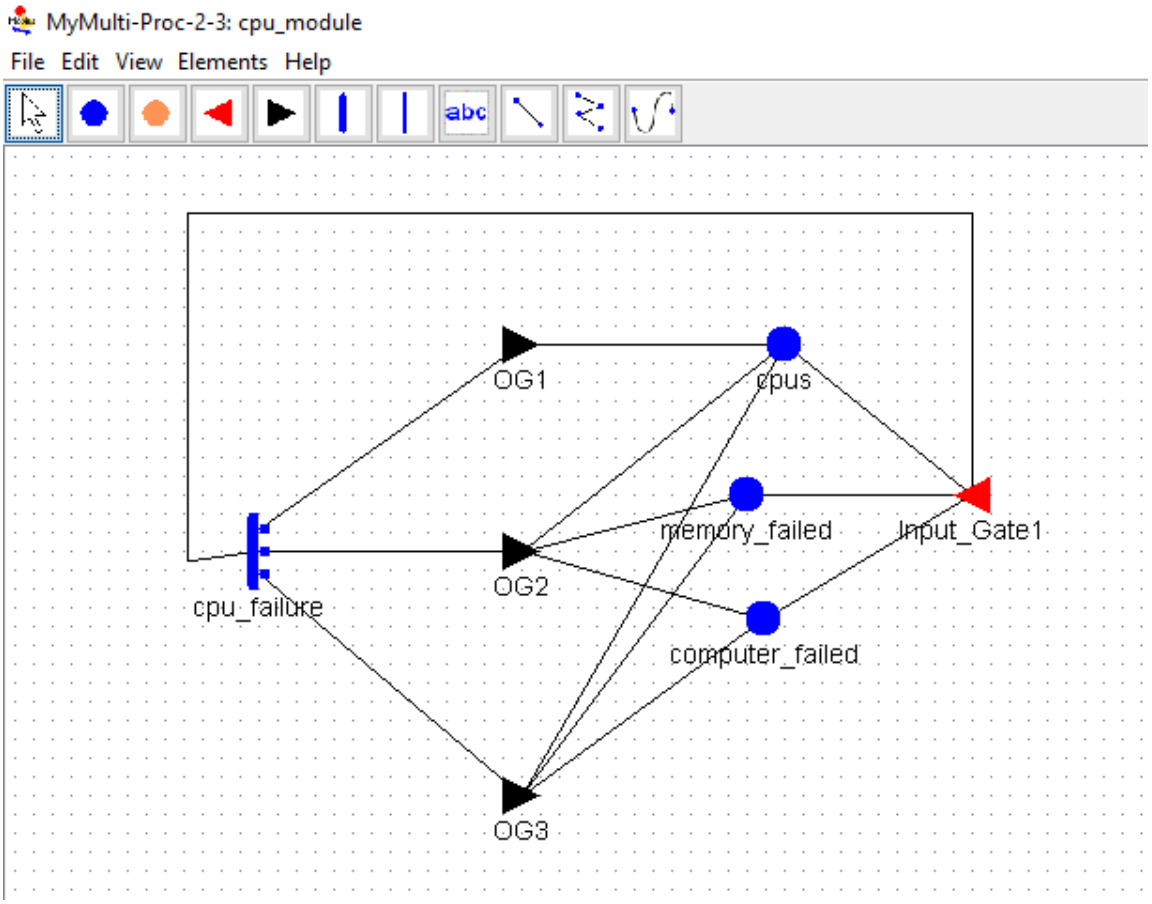
Giovanni Falzone

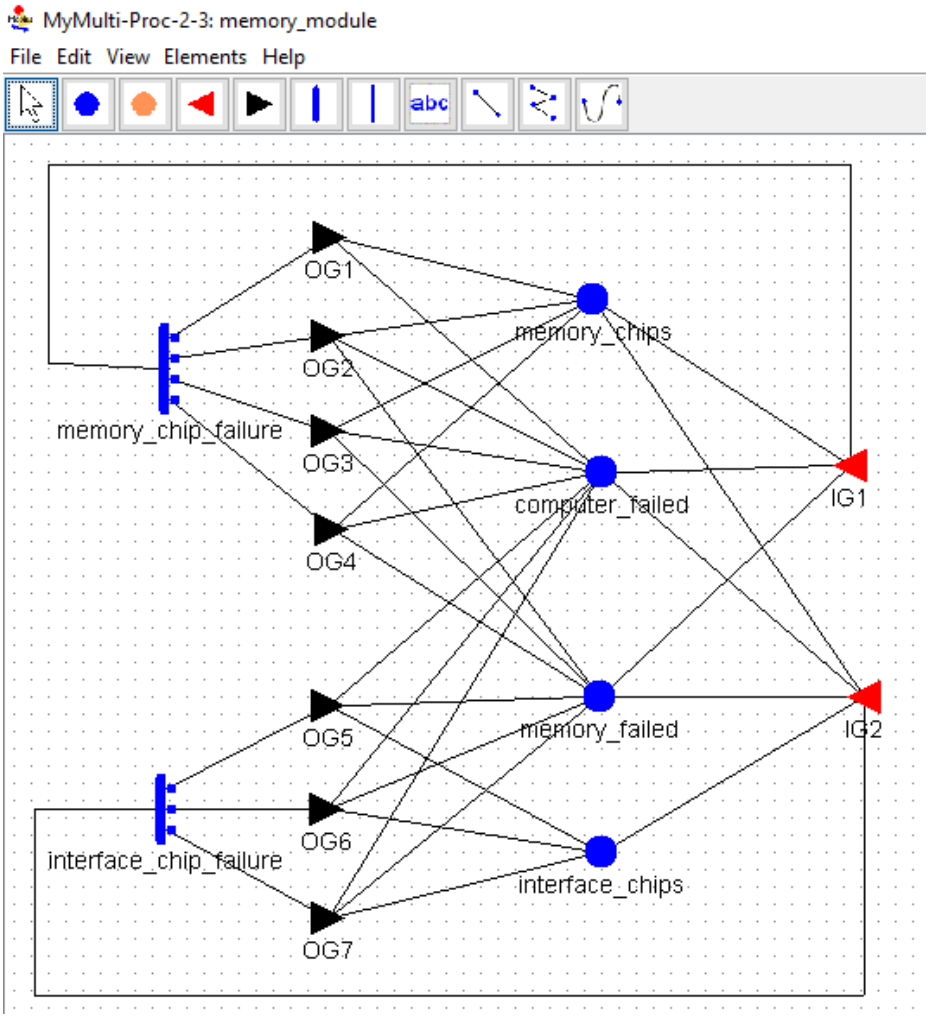
“The candidate will model the very same system introduced in the Multi-proc tutorial, with the simplifying assumption that the error handler modules and the I/O port modules are fully reliable and considering the system operational if at least two computers are working.

The coverage values, the cpu module and the memory module are the same of the tutorial.
Vary the values of the chip failure rate from 0.0008766 to 0.0035064 with a multiplicative factor of 2.0.”

To study the reliability of this system, I’ve just cut-off the components that are considered fully reliable, and study the system with only the other Components.




MyMulti-Proc-2-3: cpu_module
File Edit View Elements Help
abc
OG1
cpus
memory_failed
Input_Gate1
cpu_failure
OG2
computer_failed
OG3


MyMulti-Proc-2-3: memory_module
File Edit View Elements Help
abc
OG1
memory_chips
OG2
memory_chip_failure
OG3
computer_failed
IG1
OG4
OG5
memory_failed
IG2
OG6
interface_chip_failure
interface_chips
OG7

CPU SubModel

We have 3 cases about the CPU failure Activity, but the system must be operational so:

- At least 2 CPU's are operational;
- At least 2 memory modules are operational;
- At least 2 computers are operational.

The CPU failure rate is the **chip failure rate times the number of chips times the number of operational CPU's** (The CPU has six non-redundant chips).
The possible outcomes are:

1. **The CPU can be replaced**: decrease the number of operational CPU's.
2. **The CPU cannot be replaced**, but the computer can: increase the number of failed computers, set to zero the number of operational components.
3. **No replacement is possible**: set the number of failed computers to N-1, set to zero the number of operational components.

Case	Probability with Spares	Probability w/o Spares
1	CPU_Cov	0
2	(1-CPU_Cov)*Comp_Cov	Comp_Cov
3	(1-CPU_Cov)*(1-Comp_Cov)	1-Comp_Cov

Memory SubModel

A memory module has two sets of chips, one with spare chips and one without. So, it has two failure activities.
The **interface_chip_failure** activity is enabled if the following conditions hold:

- At least 2 chips are operational;
- At least 2 memory modules are operational;
- At least 2 computers are operational.

The failure rate is the **chip failure rate times the number of interface chips**.

The possible outcomes are:

1. **Module replacement**: increase the number of failed Memory Modules; if 2 Memory Modules are failed, increase the number of failed computers.
2. **Computer replacement**: if one Memory Module has already failed and at least **two** computers are operational, set the number of failed computers to (N-1), otherwise increase the number of failed computers.
3. **No replacement**: set the number of failed computers to (N-1).

Case	Probability with Spare	Probability w/o Spare
1	0	MemCvg
2	0	(1-MemCvg)CompCvg
3	0	(1-MemCvg)(1-CompCvg)

The memory_chip_failure activity is enabled if the following conditions hold:

- At least 39 chips are operational
- At least 2 memory modules are operational
- At least 2 computers is operational

The failure rate is the chip failure rate times the number of RAM chips.The possible outcomes are:

1. Chip replacement, if spares are available, decrease the number of RAM chips.
2. Module Replacement, increase the number of failed Memory Modules; if one has already failed, increase the number of failed computers.
3. Computer Replacement, if one Memory Module has already failed and at least two computers are operational, set the number of failed computers to N-1, otherwise increase the number of failed computers.
4. No Replacement, set the number of failed computers to N-1.

Case	Prabability with spare	Probability w/o Spare
1	Ram_Cvg	0
2	(1-Ram_Cvg)MemCvg	MemCvg
3	(1-RamCvg)*(1-MemCvg)CompCvg	(1-MemCvg)CompCvg
4	(1-RamCvg)(1-MemCvg)(1-CompCvg)	(1-MemCvg)(1-CompCvg)

Comparison of Reliability after 20 years between System, Simplex System (single computer) and Original System.

	System (2+1)	System (2+0)	Simplex System (1+0)
#States	60928	1616	66

	Original System (1+2)	Original System (1+1)	Original System (1+0)
#States	463268	10114	116

Failure Rate	System (2+1)	System (2+0)	Simplex System (1+0)
0.0008766	9.855998e-001	9.007262e-001	9.490659e-001
0.0017532	8.438526e-001	5.910448e-001	7.687944e-001
0.0035064	1.501138e-001	6.291530e-002	2.508292e-001

Failure Rate	Original System (1+2)	Original System (1+1)	Original System (1+0)
0.0008766	9.163859e-001	8.623177e-001	6.605027e-001
0.0017532	8.456845e-001	7.589375e-001	5.350427e-001
0.0035064	3.990325e-001	3.042473e-001	1.745647e-001

In the following figures, we can see the comparison between the different configuration of the System together with the Simplex System.

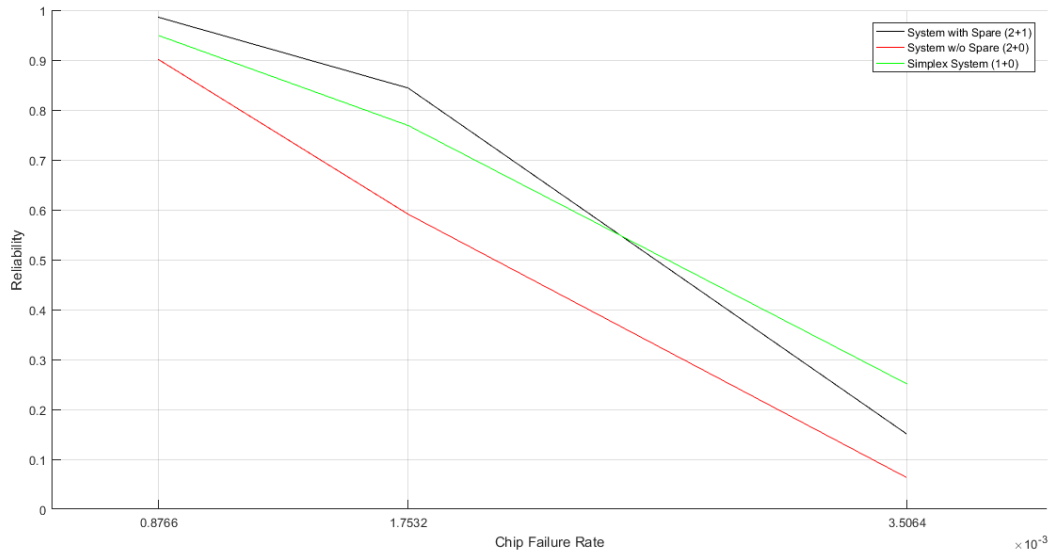


Figure 1 Comparison of the System with Simplex System

The Simplex System (1+0) is equal to the Original System (1+0) without the I/O Module and Error Handler.

Increasing the chip failure rate, the Simplex System result more reliable than the System with one spare, due to the number of components that can fail.

In the next plot, we can see the Original System in different configurations, but we have to remember that this system requires only one Computer operational and has also the I/O module and Error Handler module.

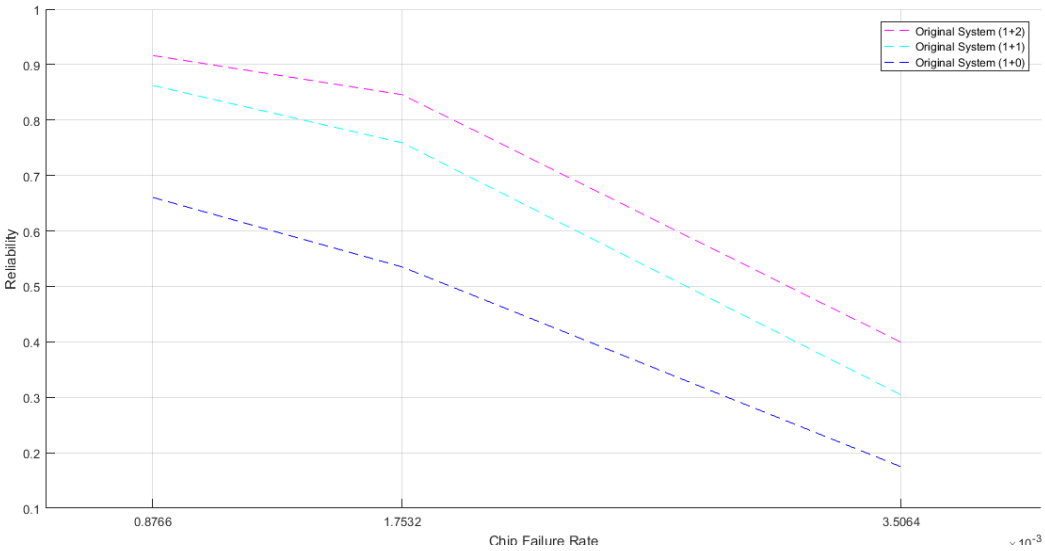


Figure 2 Comparison between the different configuration of Original System

To see the difference, of having the I/O and Error handler modules, we can compare the Simplex System and the Original System without spare, obviously our system is better in the sense of reliability because we have cut-off the two modules that are the bottle-neck in the Original System.

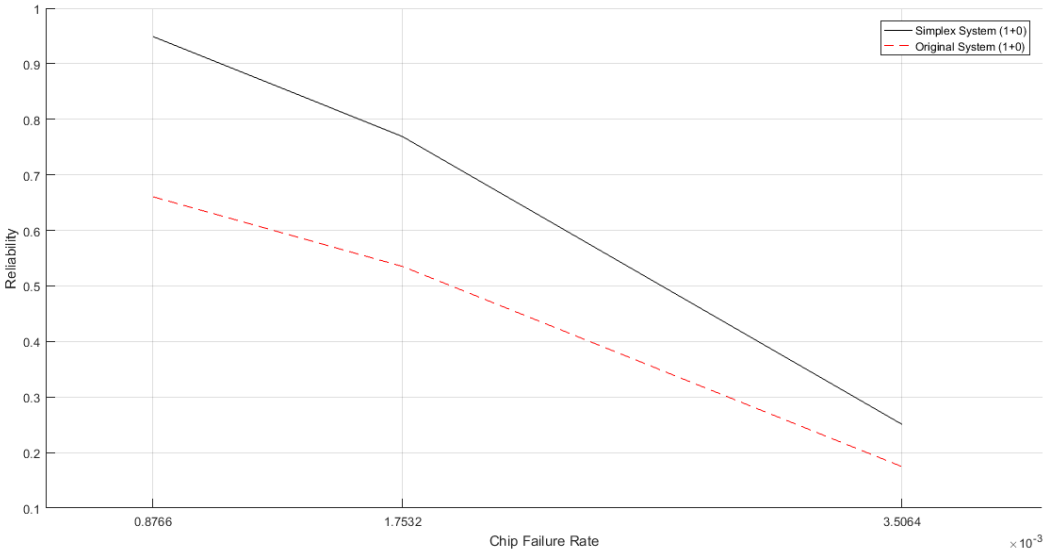


Figure 3 Comparison between Simplex System (1+0) and Original System (1+0)

Comparison of Reliability after 20 years between the System and the System with simpler Memory Modules' Output Gate for case 3.

In the Output Gate relative to the case of Computer Coverage in the SAN Memory Module we have a particular condition:

```
if ((memory_failed->Mark() == 1) && (computer_failed->Mark() < num_comp - 2))

computer_failed->Mark() = (num_comp - 1);
```

I want to study this system with a simpler condition:

```
if (computer_failed->Mark() >= (num_comp - 2))

computer_failed->Mark() = (num_comp - 1);
```

	System (2+2)	System (2+1)	System (2+0)
#States	1316072	60928	1616

Failure Rate	System (2+2)	System (2+1)	System (2+0)
0.0008766	9.872174e-001	9.855998e-001	9.007262e-001
0.0017532	9.196407e-001	8.438526e-001	5.910448e-001
0.0035064	2.400970e-001	1.501138e-001	6.291530e-002

Failure Rate	Modified System (2+2)	Modified System (2+1)	Modified System (2+0)
0.0008766	9.917785e-001	9.872174e-001	9.007262e-001
0.0017532	9.381776e-001	8.535347e-001	5.910448e-001
0.0035064	2.548638e-001	1.546859e-001	6.291530e-002

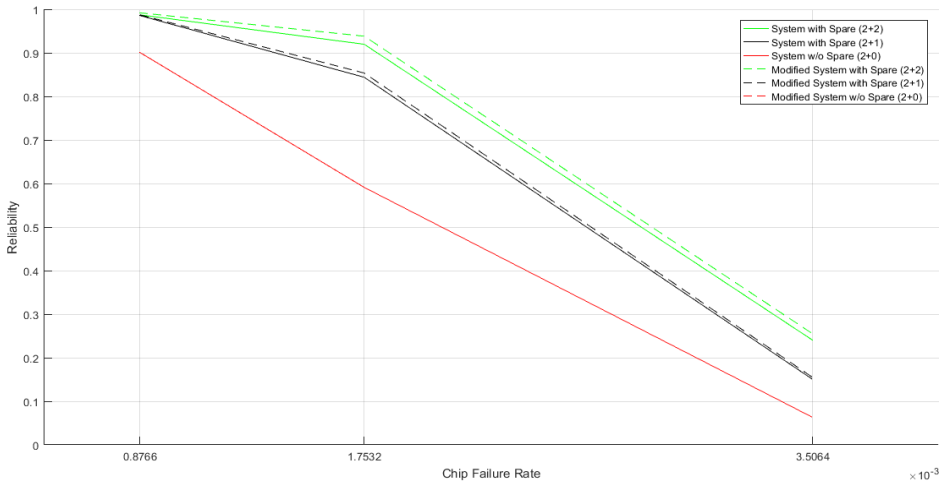


Figure 1 Comparison between System and System with simpler Output Gate

In the following plot, we can see the difference between the two condition in the sense of reliability.

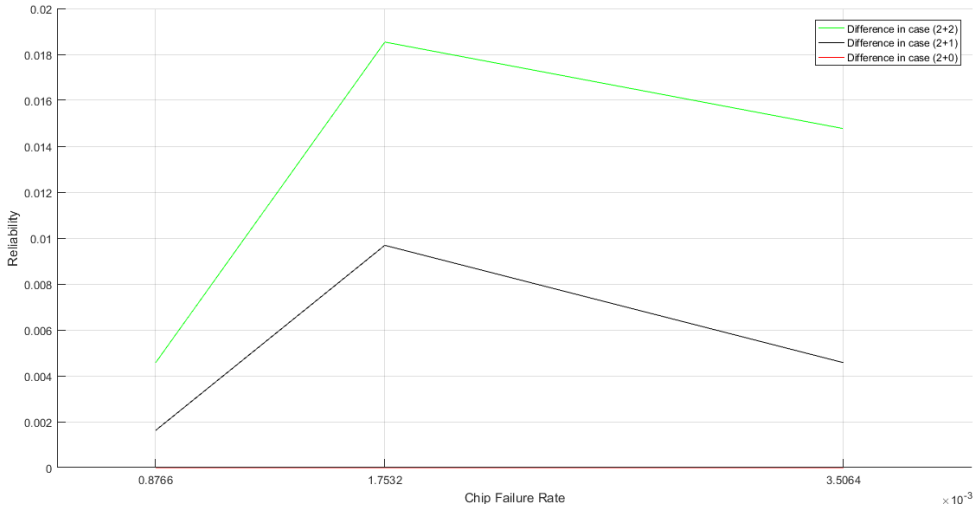


Figure 2 Differences between reliability with different Output gate of the same system