iCubeSat2015
4th Interplanetary CubeSat Workshop
Imperial College London, UK, 26-27 May 2015

Design and Implementation of Triple Modular Redundant (TMR) System on Linux-Based On-Board Computer for CubeSat

Dr. Emir Husni Angga Putra Nazmi Febrian

School of Electrical Engineering and Informatics
Institut Teknologi Bandung

Need and Objective

Need and Objective

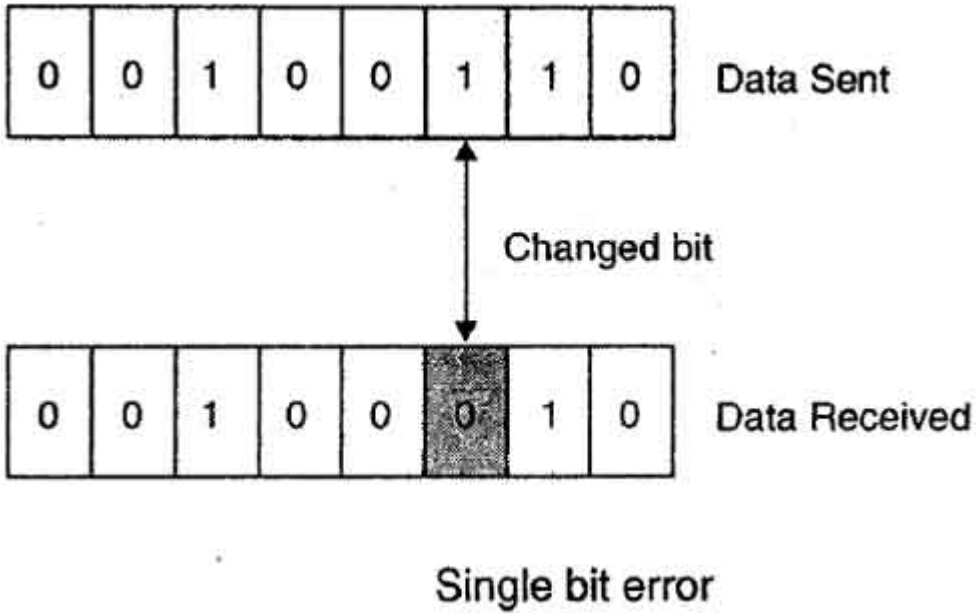
Problem Statement :
Non-Destructive & Destructive Event in Space Environment

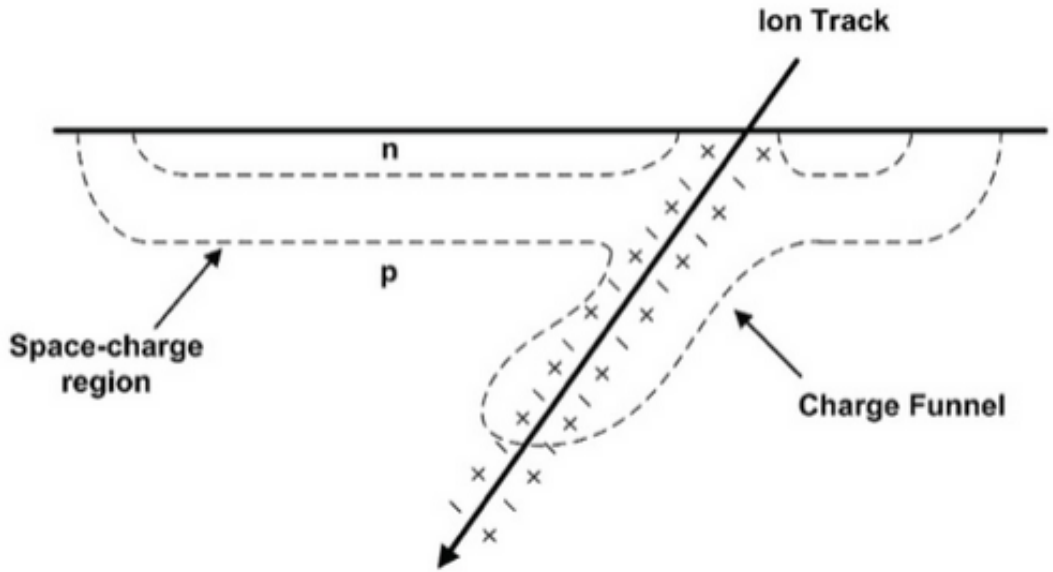
Non-destructive events

- Single Event Transient
- Single Bit Upset
- Multi Bit Upset

Destructive events

- Single Event Latch-up
- Single Event Gate Rupture (SEGR)
- Single Event Burnout





Non-Destructive Event

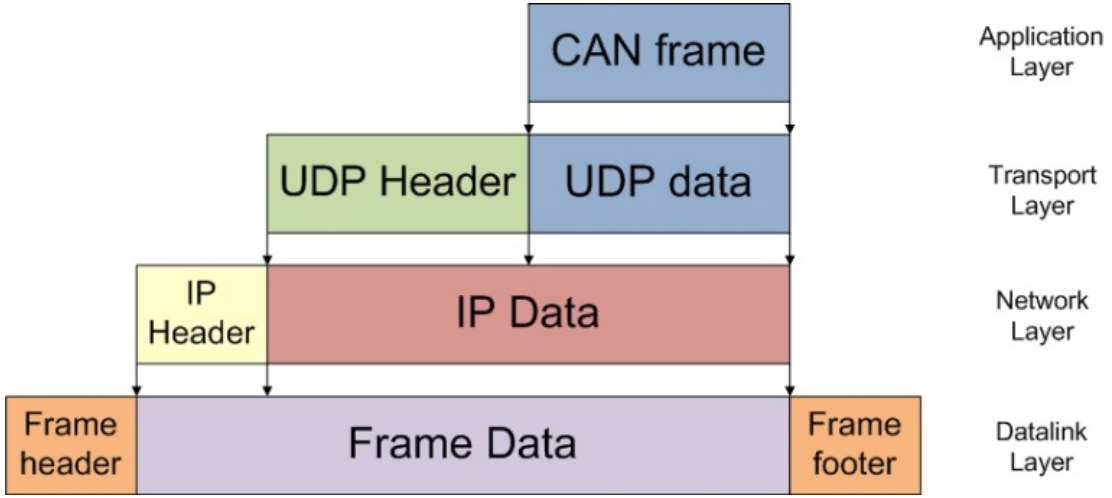
Solution : Error Correction Coding on TMR system communication

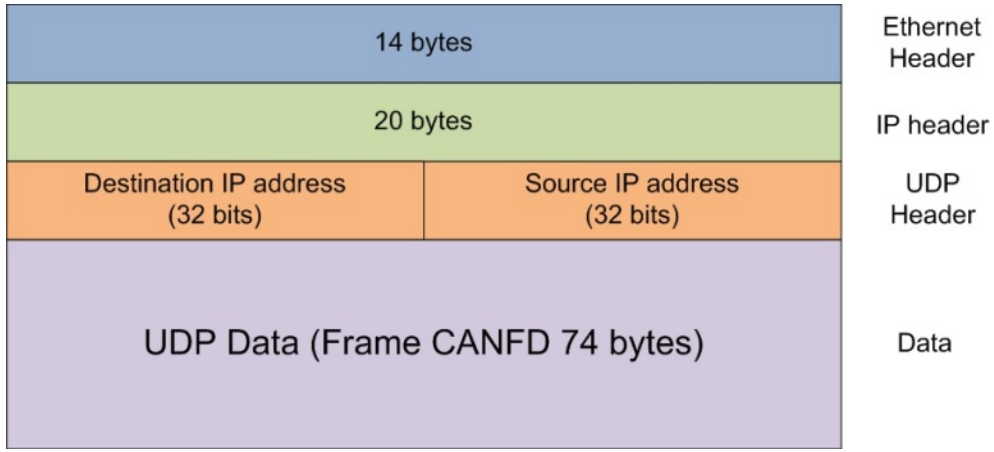
inter on-board computer (OBC)

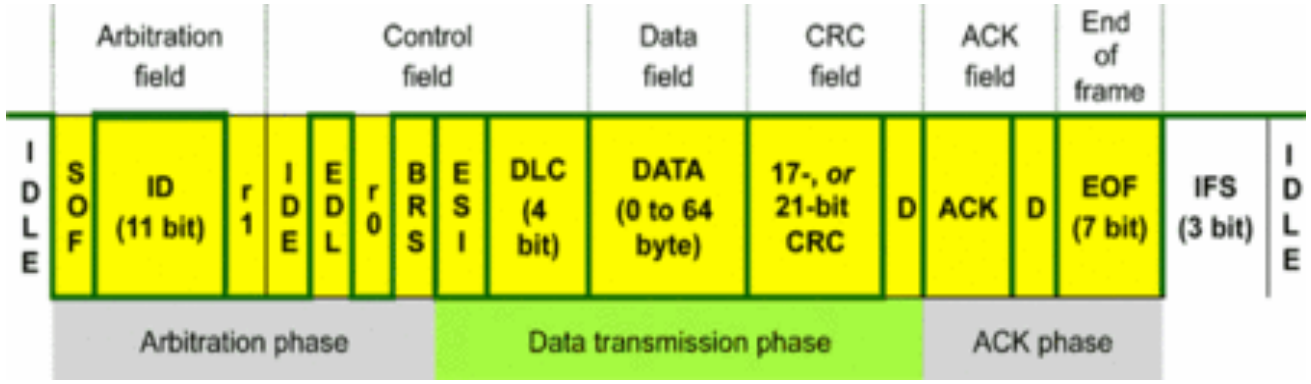
Data communication between OBC on TMR system

- Using CAN-over-IP as packet data protocol
- CAN frame implemented on UDP data

*CAN = Controller Area Network







Non-Destructive Event

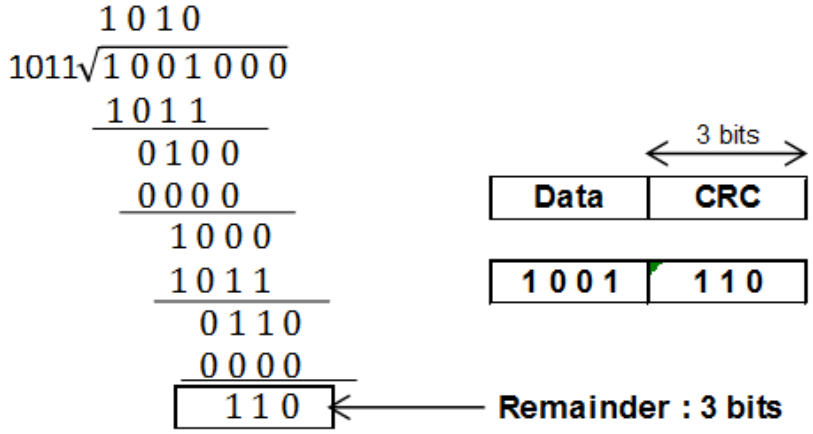
Solution : Error Correction Coding on TMR system communication

inter on-board computer

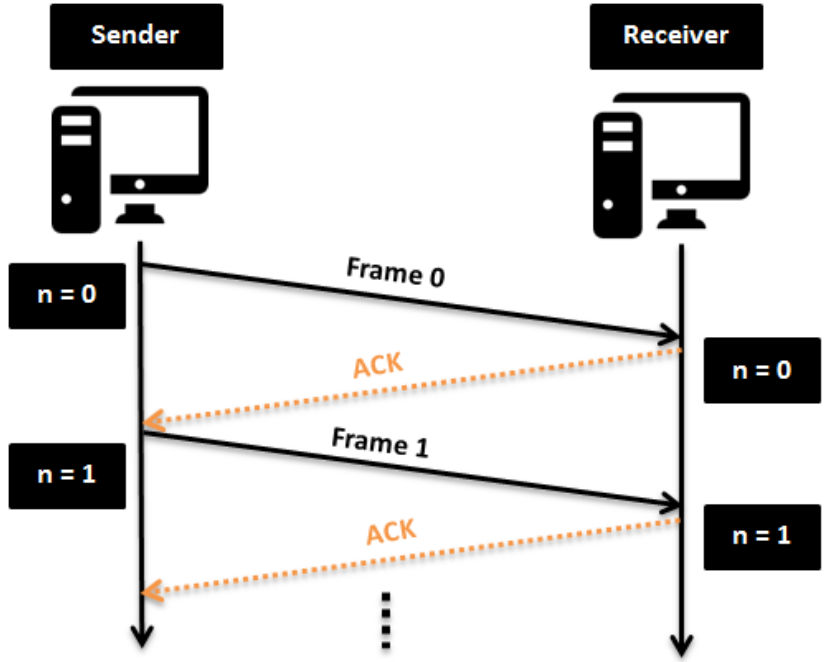
Error Correction Coding

1. Error detection
2. Error correction

1. Error detection
using CRC (Cyclic Redundancy Check)



2. Error correction
Using ARQ (Automatic Repeat Request)

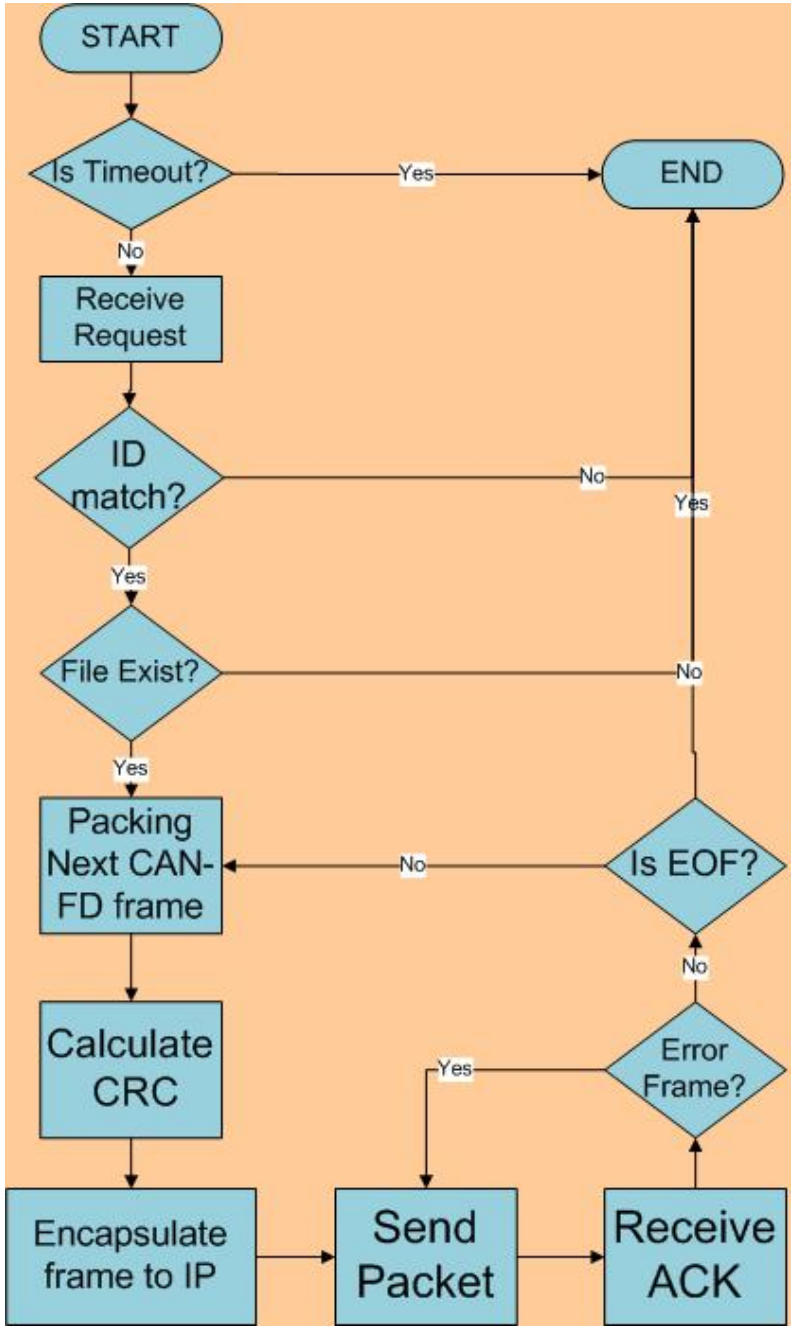


Non-Destructive Event
Error Correction Coding Implementation

Communication inter OBC

- Linux socket programming
- Using C language

SENDER SIDE

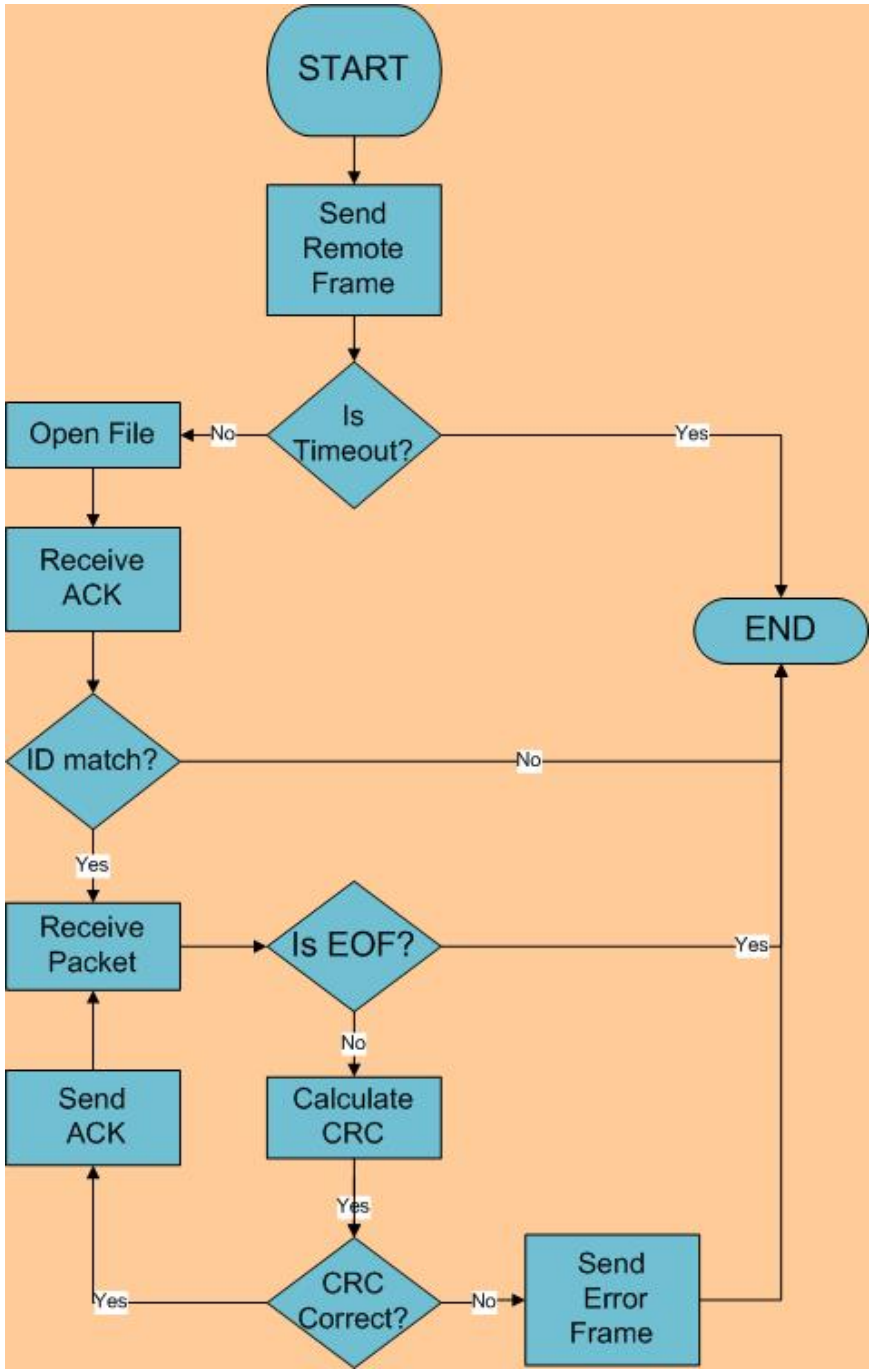


Non-Destructive Event
Error Correction Coding implementation

Communication inter OBC

- Linux socket programming
- Using C language

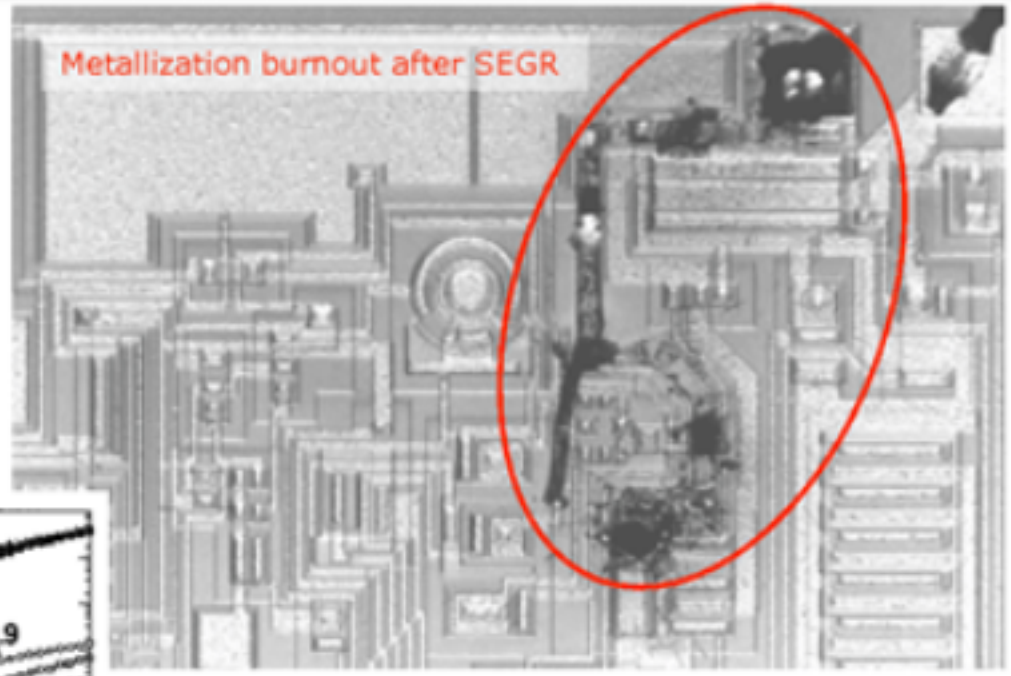
RECEIVER SIDE



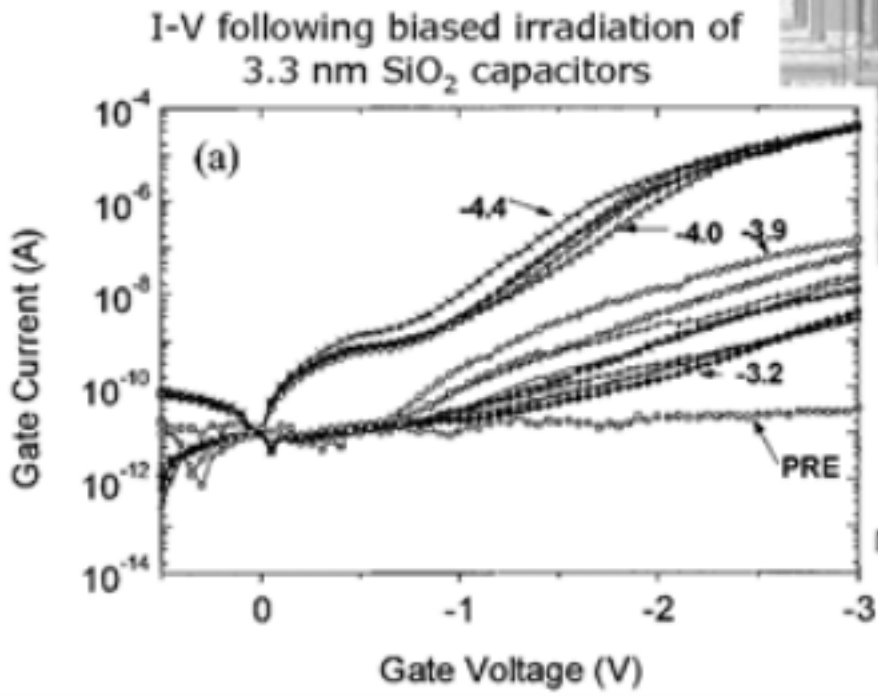
Destructive Event

Single-ion induced dielectric failure

MOSFETs, Capacitors, FG Devices



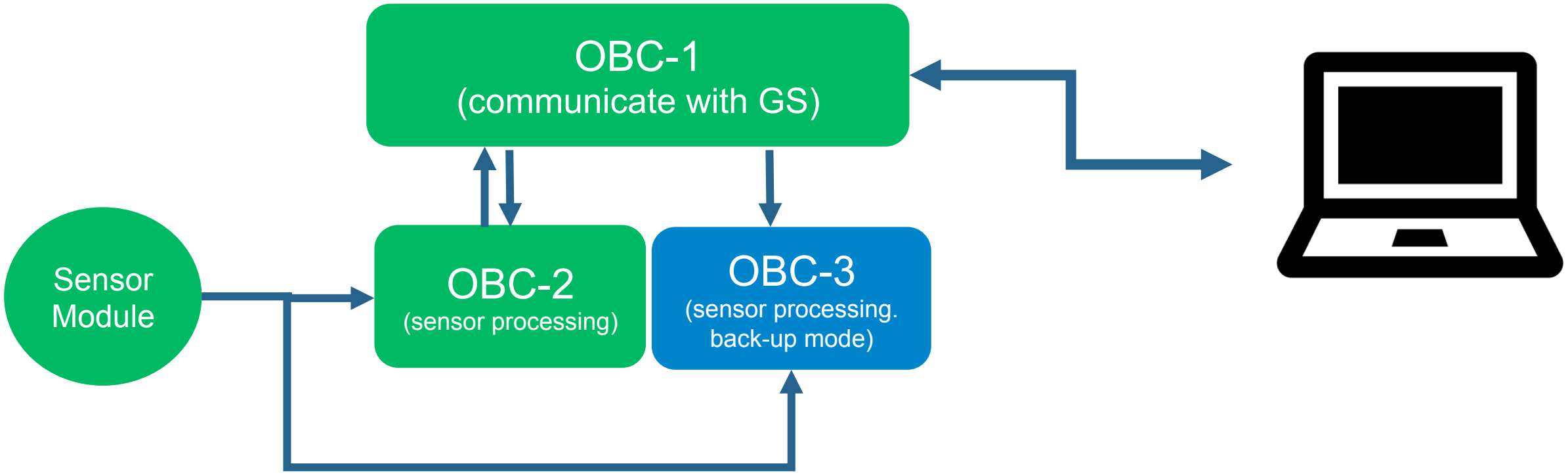
Lum, et al., IEEE TNS 51 3263 (2004)



Massengill, et al., IEEE TNS 48 1904 (2001)

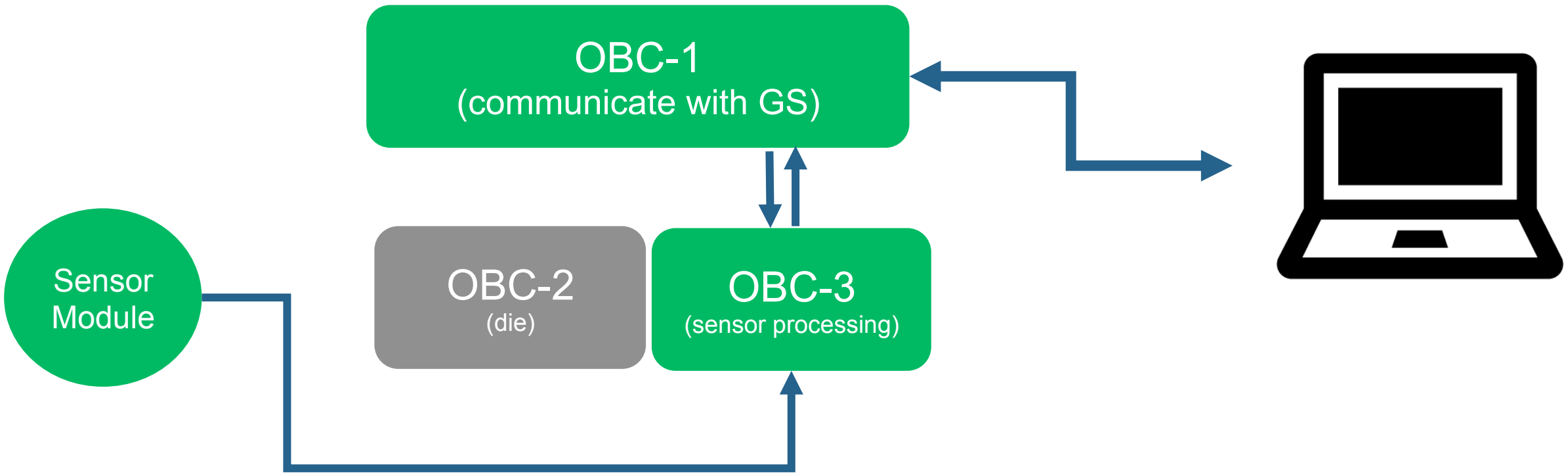
Destructive Event

Solution : Fail-over TMR system on on-board computer



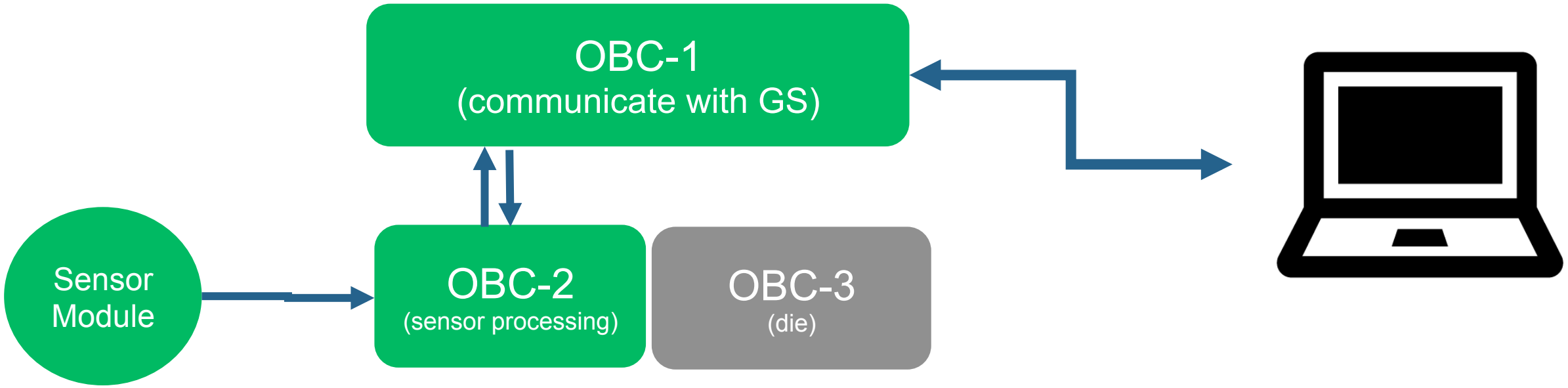
Destructive Event

Solution : Fail-over TMR system on on-board computer



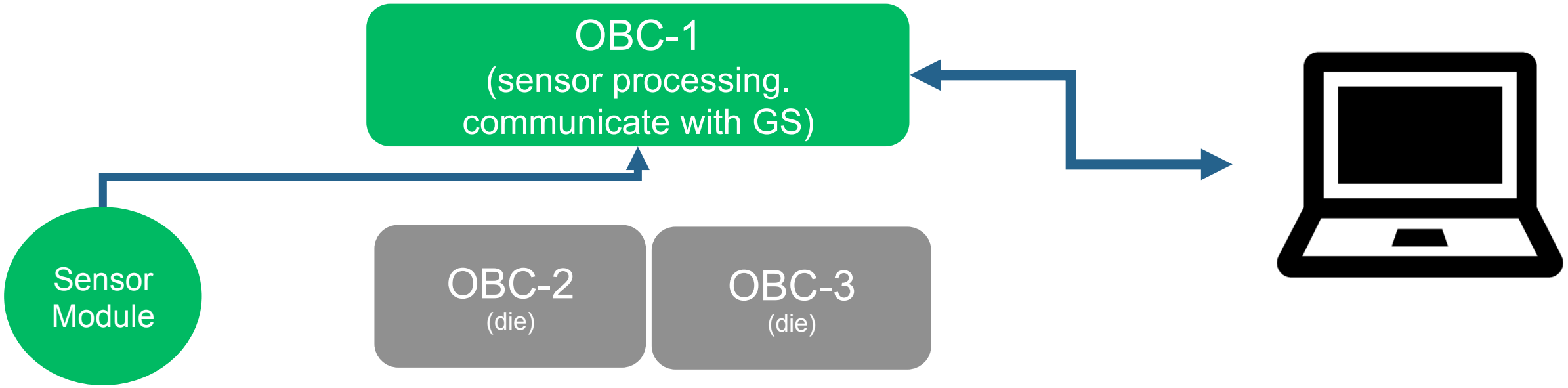
Destructive Event

Solution : Fail-over TMR system on on-board computer



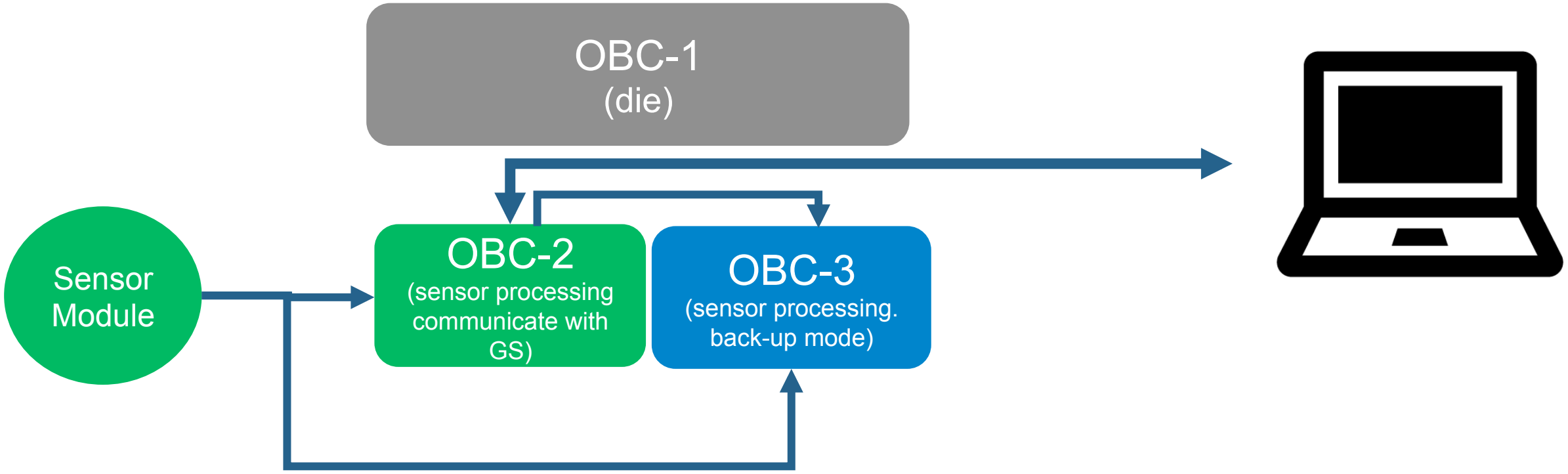
Destructive Event

Solution : Fail-over TMR system on on-board computer



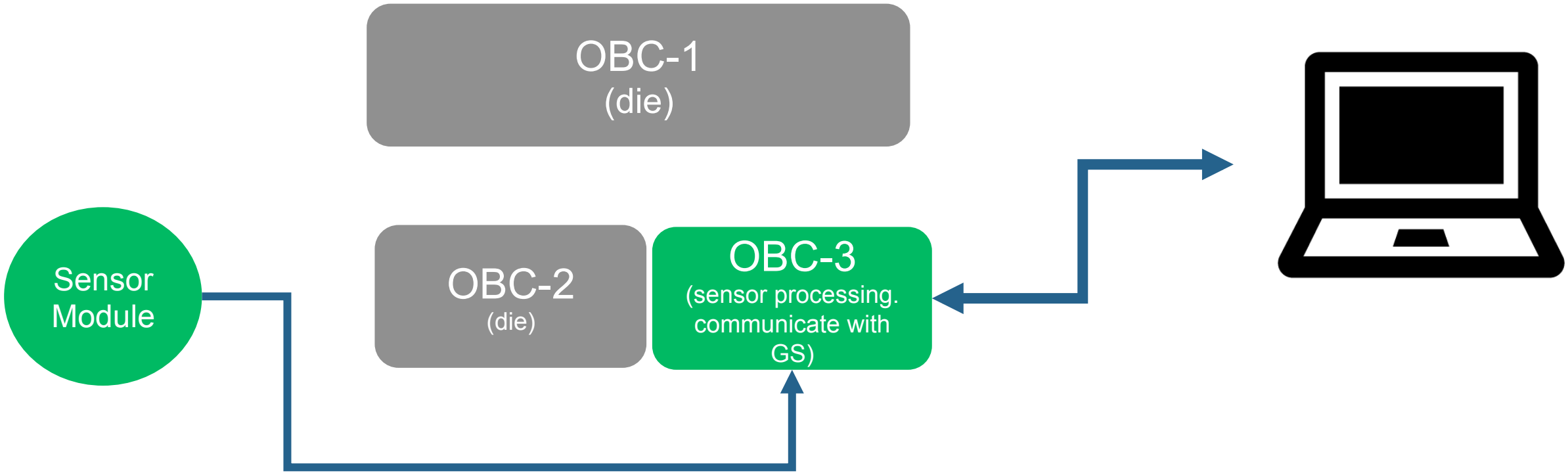
Destructive Event

Solution : Fail-over TMR system on on-board computer



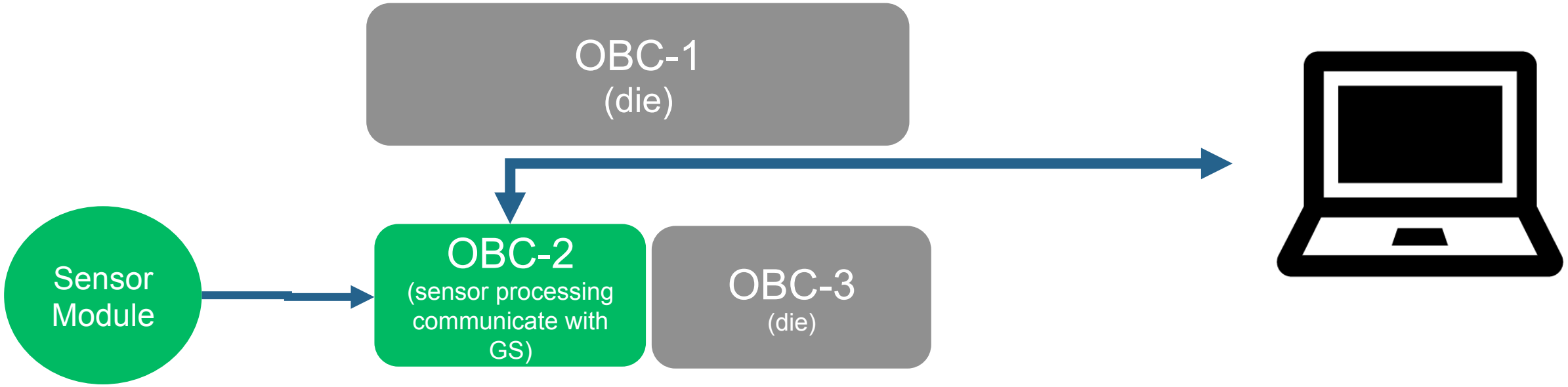
Destructive Event

Solution : Fail-over TMR system on on-board computer



Destructive Event

Solution : **Fail-over TMR system on on-board computer**



Destructive Event

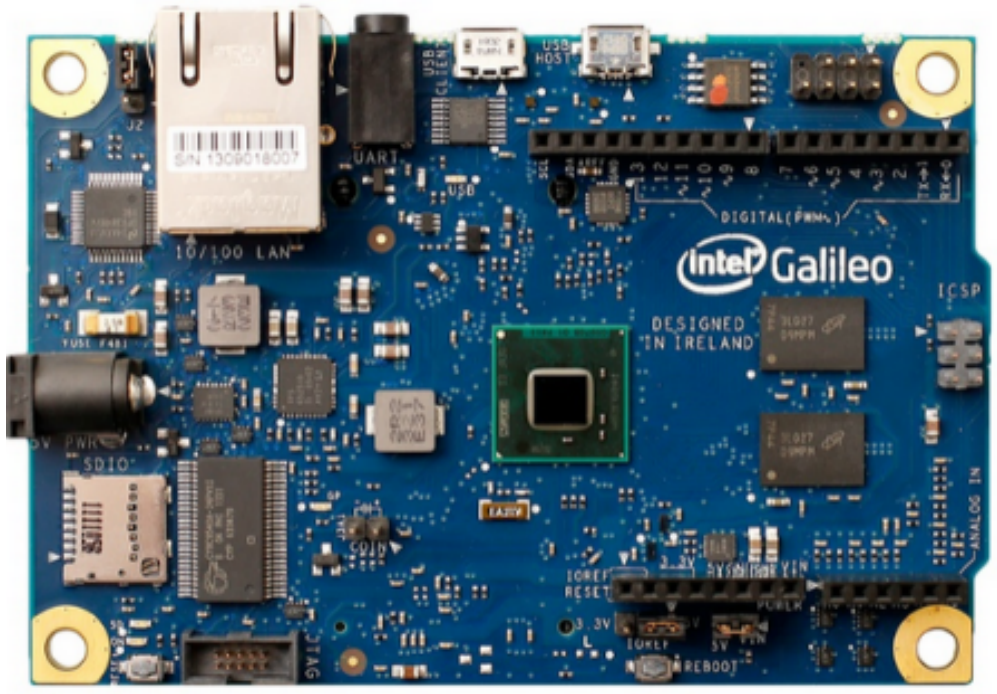
Fail-over TMR system implementation on on-board computer

TMR System

- Linux command line interface
- Using bash shell script

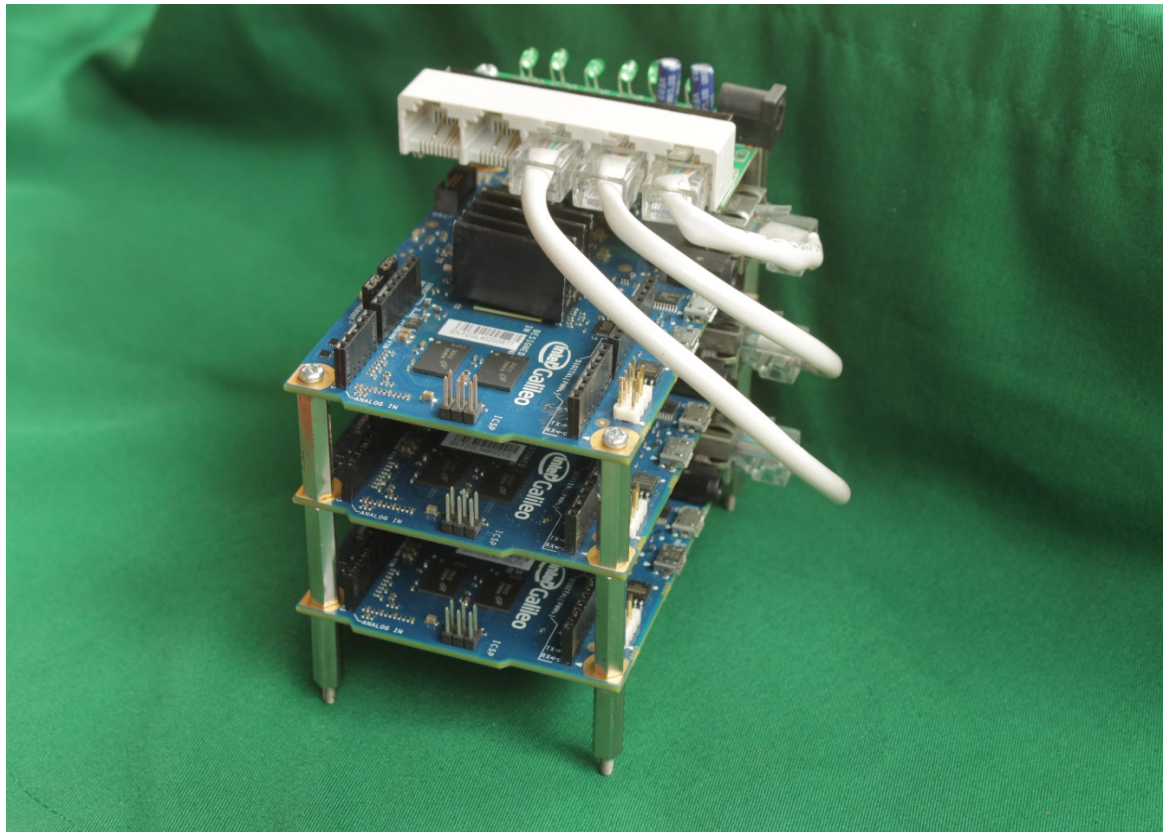
Hardware (single board computer)

- Intel Galileo Generation 1



Systems Implementation

- Each OBC connected through passive switch
- Communicate with GS through ION-DTN protocol



Specification

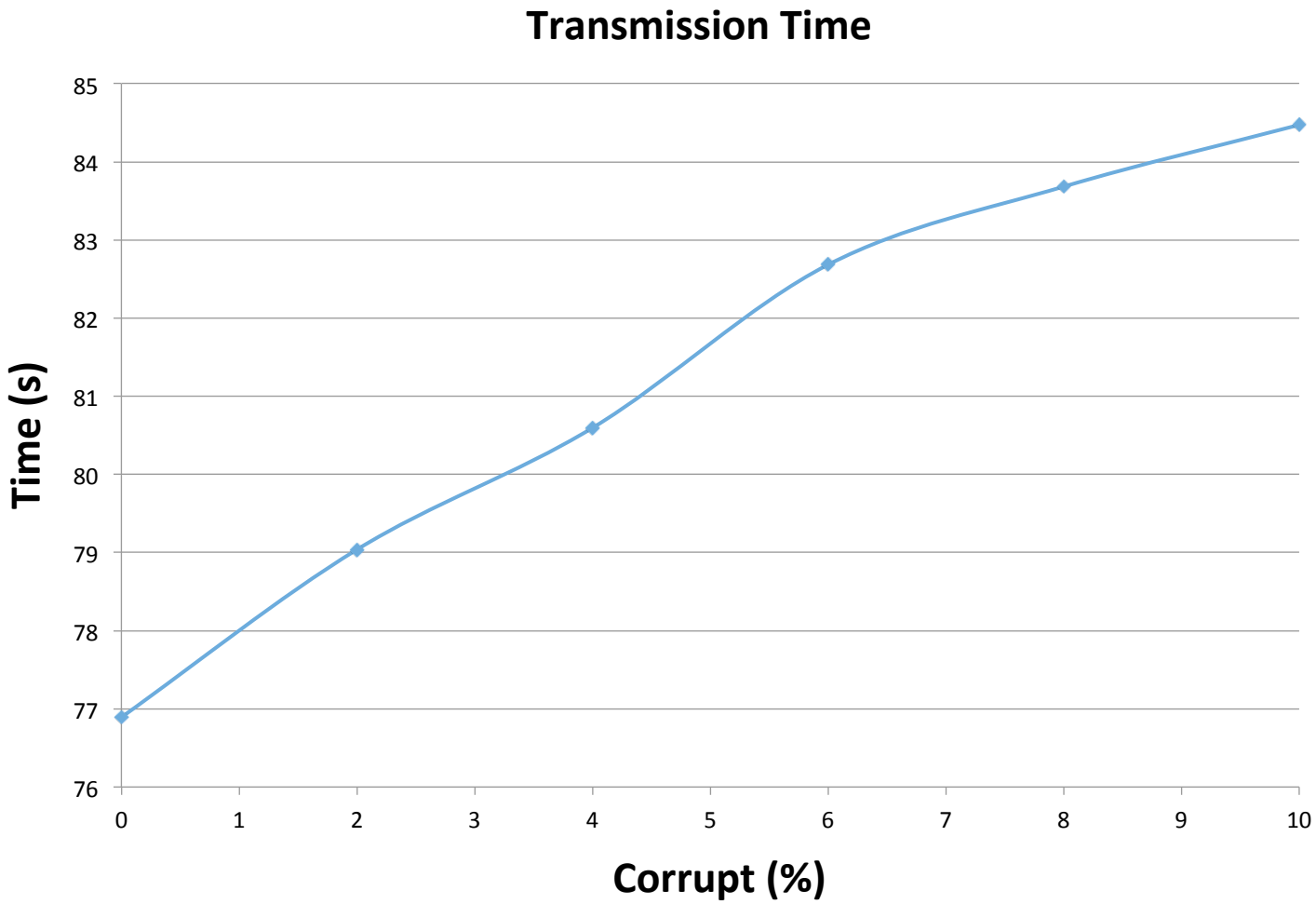
- Data corrupt resistance up to 10%
- Error control coding
- Hardware triple modular redundancy
- Remote software update

Experiment Results
Error Correction Coding

Transmission Time		
Corrupt	**Total Time**	**Average Time (s)**
0 %	1:04:04.72	76,894
2 %	1:05:51.924	79,038
4 %	1:07:09.74	80,595
6 %	1:08:54.15	82,683
8 %	1:09:44.11	83,682
10 %	1:10:23.6	84,472

This experiment is done by repeating CRC test **50 times** for each corrupt percentage to get the average time

Experiment Results
Error Correction Coding



Experiment Results
Triple Modular Redundant System

Case 1 : Un-connected from GS

	OBC-1	OBC-2	OBC-3	Request Hit	Request Miss	Total Packet Acquired (byte)
1	ACTIVE	ACTIVE	ACTIVE	13	5	6,299,264
2	ACTIVE	ACTIVE	FAIL	12	3	4,600,751
3	ACTIVE	FAIL	ACTIVE	11	10	4,600,860
4	ACTIVE	FAIL	FAIL	no request	no request	3,827,809
5	FAIL	ACTIVE	ACTIVE	no request	no request	5,219,563
6	FAIL	ACTIVE	FAIL	no request	no request	5,219,608
7	FAIL	FAIL	ACTIVE	no request	no request	3,410,318

LEO Orbit

Rotation Period
62’ sun phase + 38’ eclipse = 100 minutes

GS-connected duration estimation
≈ 10 minutes

GS-unconnected duration
100 minutes – 10 minutes ≈ 90 minutes

Assumption : All scenario happen in one rotation
Testing scenario : 90 minutes/7 scenario ≈ 13 minutes/scenario

RESULTS
Maximum packet data acquired ≈ 6.3 MB
Maximum packet data acquired from scenario #1

Experiment Results
Triple Modular Redundant System

Case 2 : Connected from GS

	OBC-1	OBC-2	OBC-3	First packet Sent Time (s)	Update Success?	Packet received in GS during process (byte)	Size of packet acquired from GS-unconnected period (First packet)	Total packet delivered (byte)
1	ACTIVE	ACTIVE	ACTIVE	15.03	yes	985,924	44.1 MB	45,085,924
2	ACTIVE	ACTIVE	FAIL	14.57	yes	985,952	44.1 MB	45,085,952
3	ACTIVE	FAIL	ACTIVE	14.81	yes	352,818	44.1 MB	44,452,818
4	ACTIVE	FAIL	FAIL	14.21	yes	981,536	44.1 MB	45,081,536
5	FAIL	ACTIVE	ACTIVE	15.03	yes	1,542,597	44.1 MB	45,642,597
6	FAIL	ACTIVE	FAIL	15.45	yes	1,542,557	44.1 MB	45,642,557
7	FAIL	FAIL	ACTIVE	15.57	yes	1,191,887	44.1 MB	45,291,887

Testing scenario : 10 minutes/scenario
Size of packet accumulated on GS-unconnected period (first packet data to be delivered to GS) are calculated from the maximum packet data acquired from **scenario #1** on **Case 1** in 90 minutes duration.

RESULTS
Software update successfully done remotely from GS
All packet data acquired from GS-unconnected period successfully delivered to GS
During GS-connected period, the system is able to acquiring and delivering data to GS

Conclusion

- By using TMR system, probability of ‘total system fail’ was significantly reduced. Sensor data is safely kept and delivered to GS, except when all OBC fail simultaneously.
- Error correction coding reduced packet data retransmission, thus increased data transmission speed and amount of data delivered.
- By using Linux operating system, internet based communication can be used as protocol for data communication inter OBC.