Intelligent Control and Automation, 2016, 7, 25-38
Published Online May 2016 in SciRes. http://www.scirp.org/journal/ica
http://dx.doi.org/10.4236/ica.2016.72004

Multi-Node Fault-Tolerant Two-Cell Real-Time S2A Network

Merna N. Abou Eita[1], Mostafa W. Hussein[1], Ahmed A. Ibrahim[1], Shereen S. Abouelazayem[1], Mennatallah A. Morsi[1], Eslam A. Moustafa[1], Hassan H. Halawa[1], Ramez M. Daoud[1], Hassanein H. Amer[1], Hany M. ElSayed[2]

[1]Electronics and Communications Engineering Department, The American University in Cairo, New Cairo, Egypt
[2]Electronics and Communications Engineering Department, Cairo University, Giza, Egypt
Email: mernanady@aucegypt.edu, mostafa_h@aucegypt.edu, ah_ibrahim@aucegypt.edu, shereen172@aucegypt.edu, menna_ali@aucegypt.edu, eslam_14@aucegypt.edu, hassan@aucegypt.edu, rdaoud@aucegypt.edu, hamer@aucegypt.edu, helsayed@ieee.org

Received 2 March 2016; accepted 8 May 2016; published 11 May 2016

Copyright © 2016 by authors and Scientific Research Publishing Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY).
http://creativecommons.org/licenses/by/4.0/
Open Access

Abstract

This paper presents a novel fault-tolerant networked control system architecture consisting of two cells working in-line. This architecture is fault-tolerant at the level of the controllers as well as the sensors. Each cell is based on the sensor-to-actuator approach and has an additional supervisor node. It is proven, via analysis as well as OMNeT++ simulations that the production line succeeds in meeting all control system requirements with no dropped or over-delayed packets. A reliability analysis is then undertaken to quantitatively estimate the increase in reliability due to the introduction of fault-tolerance.

Keywords

Fault-Tolerance, NCS, S2A, In-Line, TMR, Ethernet, OMNeT++, Reliability

1. Introduction

Traditional point-to-point control systems are currently being replaced by networks in modern manufacturing systems. These networks reduce cost by decreasing the amount of electrical wiring and also decrease maintenance costs. These Networked Control Systems (NCSs) transmit packets that have real-time constraints. These packets are small and frequent. They originate from a Sensor node (S) that samples a physical phenomenon, such as temperature, either regularly (in clock-driven systems) or when there is a system change (in event-driven systems). A packet is then sent to a controller (K) that decapsulates it, calculates the control decision, encapsu-

How to cite this paper: Abou Eita, M.N., Hussein, M.W., Ibrahim, A.A., Abouelazayem, S.S., Morsi, M.A., Moustafa, E.A., Halawa, H.H., Daoud, R.M., Amer, H.H. and ElSayed, H.M. (2016) Multi-Node Fault-Tolerant Two-Cell Real-Time S2A Network. *Intelligent Control and Automation*, **7**, 25-38. http://dx.doi.org/10.4236/ica.2016.72004

lates this decision and sends it over the network to an Actuator node (A).

Currently, the most commonly used network protocols in NCSs are CAN, PROFIBUS, Ethernet/IP, ControlNET and DeviceNET [1] [2]. However, a new trend has recently gained momentum, namely the use of Ethernet [3]-[9]. This is expected to significantly reduce equipment cost as well as installation and maintenance costs in the context of factory automation. Moreover, non-real-time traffic can then be added to the control traffic on the same network; this will enable, for instance, a maintenance engineer to remotely login to the machine while it is running.

However, it is important to remember here that the non-real-time traffic may cause the control traffic to violate system delay constraints. In [10], it was proven that a careful design of the system and the amount of traffic carried by the network, can guarantee correct system operation.

Furthermore, an NCS architecture that typically has higher performance than conventional architectures is the Sensor-to-Actuator (S2A) architecture. In this architecture, each actuator has its own controller integrated in the same node. As such control packets are transmitted directly from the sensors nodes to the integrated actuator nodes over one hop instead of over two hops as in traditional in-loop controller architectures.

Both the conventional in-loop and the S2A architectures need to be very reliable in order to reduce down time. Fault-tolerance is therefore a very important design aspect to be considered. In [11]-[15], fault-tolerance was incorporated in the in-loop architecture whether at the sensor, controller or actuator levels. In [16]-[19], fault-tolerance was also incorporated at the network fabric level. Regarding the S2A architecture, fault-tolerance was incorporated at the controller level in [12] and at the sensor level in [13]. In all these works, it was shown that the fault-tolerant system did not violate any control system requirement.

In this paper, the S2A architecture will be investigated in an in-line production scheme. Two cells will be concatenated. This is typically found in production lines where a work piece can be conditioned on several machines. It will be shown how to make the entire production fault-tolerant at the controller level first. Fault-tolerance at the sensor level will then be added and it will be shown that the system will meet all required timing constraints. Furthermore, a reliability study will be conducted to quantitatively evaluate the increase in system lifetime due to the introduction of fault-tolerance. Network fabric level fault-tolerance, which has been previously investigated in the literature [17], is beyond the scope of this work.

In [20], a control architecture, where actuator nodes are equipped with integrated control functions, was studied. This allows for a more integrated design approach of real-time fieldbus-based NCSs. It was shown in [20] how architecture and control co-design were used to achieve a successful control system implementation able to meet the control requirements.

In [21], an Ethernet-based direct Sensor-to-Actuator (S2A) architecture was proposed building up on the integrated control approach in [20]. The proposed S2A architecture was compared to a traditional in-loop control architecture, with the same number of sensor and actuator nodes, using both unmodified Fast and Gigabit Ethernet. It was shown in [21] that the proposed S2A architecture experiences a lower total end-to-end delay compared to the corresponding in-loop control architecture.

In [12], a fault-tolerant implementation of [21] was proposed where two controllers are integrated via a multiplexer into each actuator node. All sensor information is transmitted to every actuator's two controllers so that, in case of failure of one of the two controllers, the other can take over control of the actuator. The proposed fault-tolerant architecture was simulated in [12] using both Fast and Gigabit Ethernet and it was shown that the proposed architecture meets the required control deadline. However, using Gigabit Ethernet allowed for much lower control packet end-to-end delays due to the higher transmission rate.

In [13], fault-tolerance at the sensor level was added to the fault-tolerant S2A architecture in [12] through Triple Modular Redundancy (TMR). It was shown through simulations that the proposed model succeeds in meeting the required control deadline using Gigabit Ethernet but not using Fast Ethernet due to the greatly increased control traffic as a result of TMR.

An extra dimension of fault tolerance is added to an expanded NCS by in-lining two separate cells (machines) together each composed of 16 sensors, 1 supervisor and 4 integrated actuators similar to the S2A architecture in [21]. The proposed two-cell fault-tolerant architecture is presented next.

The rest of this paper is organized as follows. Section 2 describes some related works. In Section 3, the two-cell architecture is developed and its reliability modeling methodology detailed. In Section 4, the fault-tolerance of the proposed model is expanded through the application of Triple Modular Redundancy (TMR) at the sensor level. A case study is presented in Section 5 to quantify the improvements in reliability offered by the proposed models. Section 6 concludes this research.

2. Two-Cell Architecture

In-line architectures have numerous applications in many industrial processes; an example is having two separate machines operating in tandem (in-line production) with the second machine working on functions released by the first one.

In addition, by interconnecting two machines, each with its own NCS, the system as a whole is expected to become more reliable through added fault-tolerance measures. In such a fault-tolerant architecture, the second cell can take over operation of both cells in case of the occurrence of a failure in the first one.

For each individual cell, an S2A (Sensor-to-Actuator) architecture, utilizing Switched Ethernet, is employed as in [21] thus all sensor control packets are transmitted directly to all actuators every sampling period. The system sampling period is fixed to 694 µ seconds for a sampling frequency of 1440 Hz similar to [21]. Additionally, each sensor and actuator also sends a single packet to a supervisor node for monitoring purposes.

For fault-tolerance at the level of the control function, each actuator has an integrated controller (K) in addition to an extra network interface (E) as shown in **Figure 1**. During normal operation in the absence of any faults, the integrated controller (K) is responsible for taking the required control action based on the received sensor data. However, when the integrated controller (K) fails, the supervisor (Sup) node enters the cell's control loop by generating the required control action and transmitting it to the affected actuator's extra network interface (E). In **Figure 1**, Sup1 represents the supervisor in cell 1, Sup2 the supervisor in cell 2 and Ki, j controller j in cell i (i = 1,2 and j = 1 to 4).

Thus, the control traffic flows can change based on the failure state; as such, all potential failure states must be investigated individually in order to guarantee that all system deadlines are met with no control packet losses.

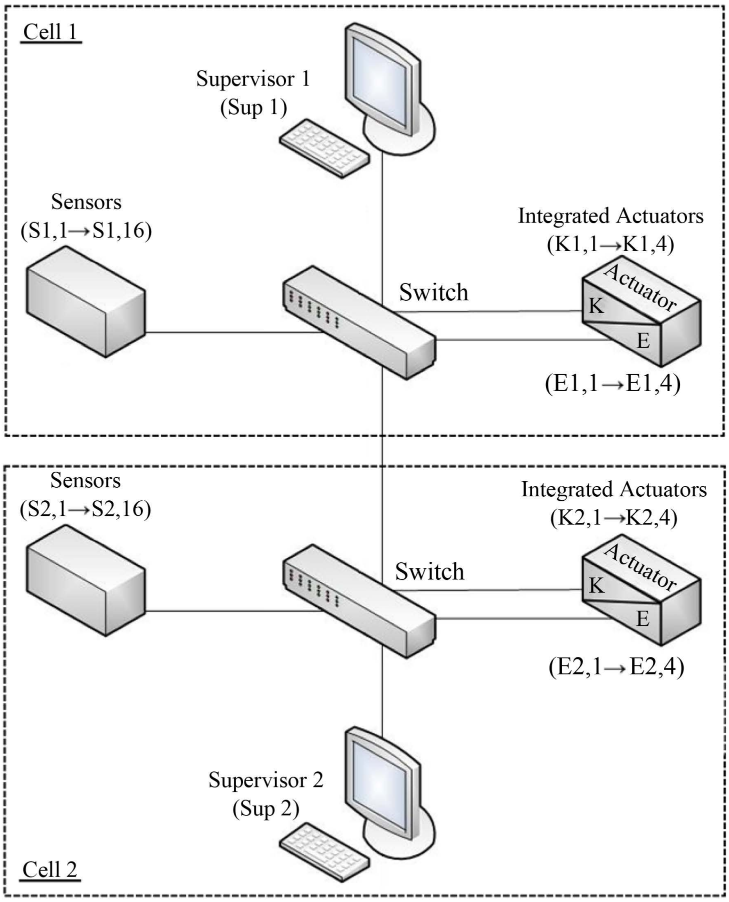

Figure 1. Simplified two-cell architecture.

2.1. Fault-Free (FF) Scenario

When all system components are operational, various packets are exchanged over the network. These control packets must be successfully transmitted, with no packet loss, within the required system deadlines in order to guarantee correct system behavior.

Sensors: All the sensors in cell one, shown in **Figure 1**, send packets to all four actuators in cell one. In addition, the sensors send their data to both cells' supervisors. Similarly, the sensors in cell two send packets to all four actuators in cell two and also send data to both cells' supervisors.

Actuators: The controllers within the actuators in cell one receive packets from the sensors in cell one. The controllers within the actuators in cell two receive packets from the sensors in cell two. Actuators from both cells send additional monitoring packets to both cells' supervisors.

Supervisors: The two supervisors receive data from all the sensors and all the actuators in both cells to allow for fault-tolerance. Finally, each supervisor sends a watchdog signal to the other supervisor every 347 microseconds (half the sampling period) to indicate to the other supervisor that it is functioning properly. This message is only 10 Bytes, as opposed to control messages, which are 100 Bytes. If a supervisor does not receive a watchdog signal from the other supervisor, the supervisor assumes the other one has failed and takes over its responsibilities. Thus, for correct operation of the proposed system, it is assumed that the supervisors are fail silent.

2.2. Fault Tolerant (FT) Scenarios

In the proposed fault-tolerant two-cell architecture, both cells can still continue operating normally even if certain components fail within the system. Next is a description of the different fault-tolerant scenarios. It is important to note that, since TMR is employed at the level of the sensor nodes, the failure of any one sensor can be tolerated by the proposed control system.

2.2.1. Supervisor in Cell One Fails

If Sup1 from **Figure 1** fails, the system will remain operational with Sup2 receiving data from all the sensors and actuators in cell one, and it will also receive data from all the sensors and actuators in cell two. The failure of the supervisor will not increase the sensor to actuator delay in either cell.

2.2.2. Supervisor in Cell Two Fails

Similar to the previous scenario but Sup2 fails instead of Sup1.

2.2.3. Controllers in Cell One Fail

If any of the controllers (or all of them as a worst case scenario) within the actuators in cell one fail (K1,1; K1,2; K1,3; K1,4 from **Figure 1**), Sup1 will detect and take over the function of the failed controllers, relaying data to the actuators through the Ethernet port (E) within each actuator. The Supervisor is able to detect any controller failure since all controllers send additional monitoring packets to the supervisors every sampling period. There will be a slight increase in the sensor to actuator delay for the affected actuators in cell one because data must go from the sensors to the supervisor and then the actuators, but in cell two the delay will not be affected because data will travel directly from the sensors to the actuators.

2.2.4. Controllers in Cell Two Fail

Similar to the previous scenario but the controllers within the actuators in cell two (K2,1; K2,2; K2,3; K2,4) fail instead, Sup2 will take over the function of the failed controllers.

2.2.5. All Controllers in Both Cells Fail

If the controllers within the actuators fail in cell one and cell two (K1,1; K1,2; K1,3; K1,4; K2,1; K2,2; K2,3; K2,4 from **Figure 1**) simultaneously (as a worst case scenario), supervisor one will take over the function of the failed controllers in cell one and supervisor two will take over the function of the failed controllers in cell two. In this situation, data must go from sensor to supervisor first before going from supervisor to the actuators. As a result, there is a slight increase in the sensor to actuator delay in both cells compared to the delay in the fault-free scenario.

2.2.6. Supervisor and Controllers in Cell One Fail

If (K1,1; K1,2; K1,3; K1,4; Sup1 from **Figure 1**) fail, Sup2 receives packets from the sensors in cell one to relay them to the actuators in cell one through the Ethernet port in each actuator, and it will also receive packets from the sensors and controllers in cell two. There is a notable increase in the sensor to actuator delay in cell one, but no increase in delay in cell two.

2.2.7. Supervisor and Controllers in Cell Two Fail

Similar to the previous scenario but (K2,1; K2,2; K2,3; K2,4; Sup2) fail instead and Sup1 takes over control of the failed cell.

2.2.8. Supervisor in Cell One Fails, Controllers in Cell Two Fail

If (K2,1; K2,2; K2,3; K2,4; Sup1 from **Figure 1**) fail, Sup2 receives packets from the sensors and controllers in cell one, but there is no increase in the sensor to actuator delay in cell one. However, in cell two, Sup2 must also undertake the function of the failed controllers within the actuators in the second cell; as a result, there is an increased delay in cell two.

2.2.9. Supervisor in Cell Two Fails, Controllers in Cell One Fail

Similar to the previous scenario but (K1,1; K1,2; K1,3; K1,4; Sup2) fail instead and Sup1 takes over control of the actuators in cell one.

2.2.10. Supervisor in Cell One Fails, Controllers in Both Cells Fail

If (K1,1; K1,2; K1,3; K1,4; K2,1; K2,2; K2,3; K2,4; Sup1 from **Figure 1**) fail, Sup2 must undertake the function of the failed controllers in both cells. However, because cell two is much closer to supervisor two, the end-to-end delay in cell one is much larger than the delay in cell two.

2.2.11. Supervisor in Cell Two Fails, Controllers in Both Cells Fail

Similar to the previous scenario but (K1,1; K1,2; K1,3; K1,4; K2,1; K2,2; K2,3; K2,4; Sup2) fail and Sup1 must instead undertake the function of the failed controllers in both cells.

2.3. Delay Analysis

Through analysis of the model, the end-to-end delay for the control packets will be calculated for both Fast Ethernet and Gigabit Ethernet. A worst-case delay analysis will be carried out on this model. As previously mentioned, one of the restrictions on the proposed model is that the control action must be taken within 694 µs. Therefore, it is crucial for the worst-case delay to not exceed this limit in order to ensure correct control operation. The following analysis will focus on the last packet transmitted from the final sensor node; this represents the worst-case scenario because all the previously sent packets are queued before this particular packet, hence it will require the largest amount of time to be transmitted over the network. In addition, processing delay is not taken into account because previous work has shown it is so small compared to the rest of the delays and as such it can be considered negligible [22]. Thus, the amount of time required for the transmission of a single packet over a particular link is given by:

$$D_{\text{packet}} = D_{\text{transmission}} + D_{\text{propagation}} \tag{1}$$

The total end-to-end delay for the worst-case packet flow is given by:

$$D_{\text{total}} = \text{Total Number of Packets Transmitted Sequentially} \times D_{\text{packet}} \tag{2}$$

The link transmission delay ($D_{\text{transmission}}$) is the amount of time required for all of the packet's bits to be transmitted onto the link. It depends on the packet length L (bits) and link transmission rate R (bps) [23].

$$D_{\text{transmission}} = L/R \tag{3}$$

The length of the packet is fixed to 100 Bytes at the application layer; however, additional packet and frame header overhead (approximately 58 Bytes) must be taken into consideration. All the links are Gigabit Ethernet in one scenario and Fast Ethernet in the second scenario, therefore:

$$\text{Fast Ethernet}: D_{\text{transmission}} = (158 \times 8) / (10)^8 = 12.64\ \mu\text{s} \tag{4}$$

$$\text{Gigabit Ethernet}: D_{\text{transmission}} = (158 \times 8) / (10)^9 = 1.264\ \mu\text{s} \tag{5}$$

The propagation delay ($D_{\text{propagation}}$) is the time taken for the packet to travel from the sender to the receiver; it depends on the link length *d* (m) and the propagation speed *s* (m/s) [23].

$$D_{\text{propagation}} = d/s \tag{6}$$

The length between each node and the switch is $d = 1.5$ m and the transmission speed in the Ethernet links is $s = 2 \times 10^8$ m/s.

$$D_{\text{propagation}} = 1.5 / \left(2 \times 10^8\right) = 7.5\ \text{ns} \tag{7}$$

Hence, the total end-to-end delay is given by:

$$\text{Delay}_{\text{Total}}\ (\text{in}\ \mu\text{s}) = \left(\left(\text{number of packets in worst case queue}\right) \times \left(D_{\text{transmission}} + D_{\text{propogation}}\right)\right) \tag{8}$$

Using the delays obtained above as constants and substituting with the appropriate values in (8), the sensor to actuator delays for all possible system states (including fault-free and fault-tolerant scenarios) is calculated next.

2.3.1. Fault-Free (FF) Scenario

For the fault-free scenario, the number of packets in the worst case queue is 20 packets (16 packets from the sensors to the switch and 4 packets from the switch to the actuators) following the same analysis methodology as in [12]. Thus, the total end-to-end delays for Fast and Gigabit Ethernet can be obtained as follows:

$$\text{Fast Ethernet Cell 1}: D_{\text{Total}} = \left(20 \times (12.64 + 0.0075)\right) = 252.95\ \mu\text{s} \tag{9}$$

$$\text{Gigabit Ethernet Cell 1}\ D_{\text{Total}} = \left(20 \times (1.264 + 0.0075)\right) = 25.43\ \mu\text{s} \tag{10}$$

$$\text{Fast Ethernet Cell 2}: D_{\text{Total}} = \left(20 \times (12.64 + 0.0075)\right) = 252.95\ \mu\text{s} \tag{11}$$

$$\text{Gigabit Ethernet Cell 2}: D_{\text{Total}} = \left(20 \times (1.264 + 0.0075)\right) = 25.43\ \mu\text{s} \tag{12}$$

2.3.2. Fault-Tolerant (FT) Scenarios

Following the same delay calculation methodology, **Table 1** summarizes the delay analysis for all aforementioned Fault-Tolerant (FT) scenarios.

Table 1. Summary of the oritical end-to-end delays for all scenarios.

Scenario	Cell 1			Cell 2		
	Packets	Fast Ethernet delay (µs)	Gigabit Ethernet delay (µs)	Packets	Fast Ethernet delay (µs)	Gigabit Ethernet delay (µs)
FF	20	252.95	25.43	20	252.95	25.43
FT-1	20	252.95	25.43	20	252.95	25.43
FT-2	20	252.95	25.43	20	252.95	25.43
FT-3	22	278.245	27.973	20	252.95	25.43
FT-4	20	252.95	25.43	22	278.245	27.973
FT-5	22	278.245	27.973	22	278.245	27.973
FT-6	24	303.54	30.516	20	252.95	25.43
FT-7	20	252.95	25.43	24	303.54	30.516
FT-8	20	252.95	25.43	22	278.245	27.973
FT-9	22	278.245	27.973	20	252.95	25.43
FT-10	44	556.49	55.946	22	278.245	27.973
FT-11	22	278.245	27.973	44	556.49	55.946

2.4. Simulation vs. Analytical Results

The proposed two-cell fault-tolerant architecture was simulated on OMNeT++ [24] in the fault-free scenario as well as under all outlined failure scenarios. For all simulated scenarios, all control packets were transmitted successfully with no packet losses. Additionally, all observed control packet delays were less than the required system deadline.

In **Figure 2**, the maximum end-to-end delay for the control packets transmitted from the 16 sensors to the 4 actuators in cell 2 is shown in the absence of any failures. The x-axis represents the Simulation Time (seconds) and the y-axis shows the End-to-end Delay (in seconds). The end-to-end delays from the simulation include packet transmission, propagation, queuing, encapsulation and decapsulation delays.

The observed end-to-end delays were deterministic due to the periodic nature of the control traffic combined with the use of Switched Ethernet. In all simulated scenarios, the observed percentage error between the simulated and analytical delay results did not exceed 5% as shown in **Table 2**.

2.5. Reliability Modeling

The reliability of the proposed two-cell architecture will be calculated next. Two values will be calculated: the Control Function Reliability (CFR) and the Node Reliability (NR).

2.5.1. Control Function Reliability (CFR)

CFR(*t*) focuses only on the components where the control function is executed, *i.e.*, the supervisors (Sup1 and Sup2) and the four controllers connected to the four actuators in both cells (Ki, j, i = 1,2 and j = 1 to 4). It will be assumed that, if the system loses its observability (both supervisors fail), this will be considered as a system failure even if both cells are still operational. There is one controller in each actuator and there are four actuators in each cell; in addition, there is a supervisor in each cell, which means the control function for the two-cell system is based on 10 components.

In order to calculate *CFR*(*t*), all the different situations are analyzed and the failure states are identified. There are 210 situations to consider since there are 10 components involved in the Control Function. A small scale

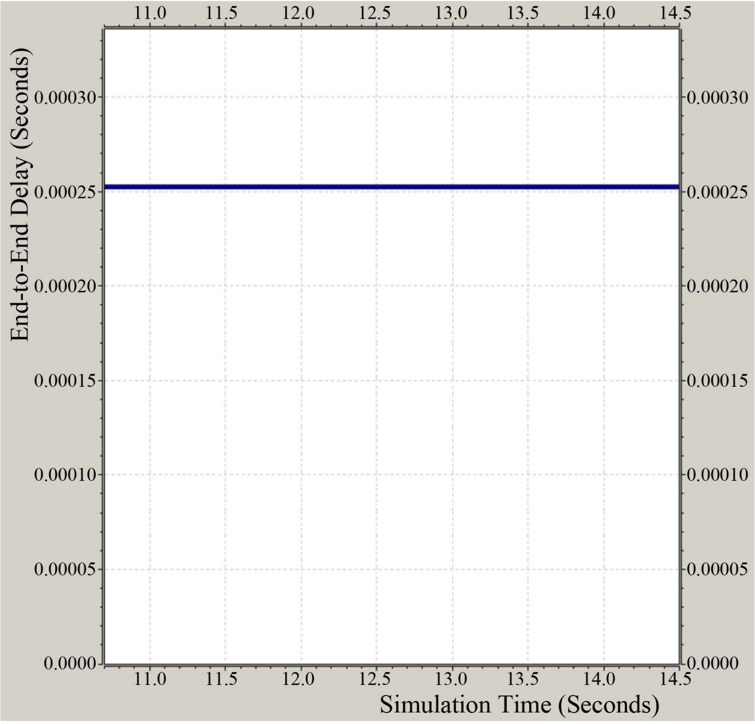

Figure 2. Maximum end-to-end delay at the actuators of cell 2 (fault-free scenario).

Table 2. Summary of percentage error (simulation vs. theoretical delays) for all scenarios.

Scenario	Cell 1		Cell 2	
	Fast Ethernet % error	Gigabit Ethernet % error	Fast Ethernet % error	Gigabit Ethernet % error
FF	4.77	4.31	4.77	4.31
FT-1	4.77	4.31	4.77	4.31
FT-2	4.77	4.31	4.77	4.31
FT-3	4.77	4.35	4.77	4.31
FT-4	4.77	4.31	4.77	4.35
FT-5	4.77	4.35	4.77	4.35
FT-6	4.78	4.39	4.77	4.31
FT-7	4.77	4.31	4.78	4.39
FT-8	4.77	4.31	4.77	4.35
FT-9	4.77	4.35	4.77	4.31
FT-10	2.55	0.812	4.77	4.35
FT-11	4.77	4.35	2.55	0.812

model is first analyzed, using Algorithm I of **Figure 3**, with one controller and one supervisor per cell.

It was observed that all failure states have one feature in common: in each failure state, both supervisors are in a failure state, regardless of the state of the controllers. The rationale behind this finding is as follows: Assuming the Ethernet port of the actuators is always operational, the controllers Ki, j will not cause a system failure because, even if they fail, the supervisors can take over their function. This is clear from the 11 scenarios described in Section 3.2. However, when both supervisors fail at the same time, the system loses its observability which is considered to be a system failure as mentioned above (even if all the controllers are working). Hence, *CFR*(*t*) for the two-cell system is:

$$CFR(t) = 1 - \left(1 - R_{sup1}\right) \times \left(1 - R_{sup2}\right) \tag{13}$$

The reliability $CFR_{sim}(t)$ is that of the simplex system (two cells without any fault-tolerance). Note that:

$$CFR_{sim}(t) = \left[\left(R_{sup}\right) \times \left(R_k\right)^4\right]^2 \tag{14}$$

In the above equation, it is assumed that both supervisors have the same failure rate. The same is assumed for all controllers connected to the actuators. The reliability of any of the 8 controllers in both cells is R_k.

2.5.2. Node Reliability (NR)

As mentioned above, another perspective would include the Ethernet ports of both sensors and actuators in the reliability calculations (in addition to both supervisors and the 4 controllers connected to the actuators in each cell), *i.e.*, all nodes connected to the network fabric. As a result, across both cells, the components consist of 16 sensor Ethernet ports in each cell, 1 supervisor in each cell and 4 controllers as well as 4 actuator Ethernet ports for each cell; in other words, a total of 25 components for each cell and a total of 50 components for the two-cell system in total.

Since there are 50 components to analyze, 250 situations have to be studied. As for *CFR*(*t*), a small scale model is first analyzed following Algorithm I with one controller, one sensor Ethernet port, one supervisor and one controller Ethernet port for each cell, resulting in a total of 8 components, or 28 = 256 situations.

Of the 256 situations, precisely 27 are operational and the remaining 219 situations are failure states. By observing the 27 states in which the system is operational, the following observations can be made.

Sensors: If any of the sensor Ethernet ports fail for any reason in either cell, the system fails.

Controllers: If the controllers connected to the actuators fail, the system will continue to work on the condition that one of the supervisors is working and that the corresponding Ethernet ports in the actuators are working as well.

```
For each possible state
  Build the network
  Remove failed nodes
  Verify control network operation
  If correct operation
    System is operational
  Else
    System failure
Record System State Vector and
Operational State
```

Figure 3. Algorithm I. Exhaustive analysis.

Supervisors: At least one out of the two supervisors must be working at all times. If both supervisors fail at the same time, the observability of the system is lost which is assumed to cause a system failure.

Actuator Ethernet Ports: The system can continue working normally even if the Ethernet ports in the actuators are down, but only if the controllers attached to the corresponding actuators are working. For each actuator, its controller and Ethernet port form a 1-of-2 system. If a controller fails inside the actuator and the Ethernet port inside the same actuator fails at the same time, the system fails. Hence, $NR(t)$ can be calculated as follows:

$$NR(t) = \left[R_s\right]^{32} \times \left[1-\left(1-R_{Sup1}\right)\times\left(1-R_{Sup2}\right)\right] \times \left[1-\left(1-R_k\right)\times\left(1-R_a\right)\right]^8 \tag{15}$$

where R_s is the reliability of the sensor's Ethernet port, R_a is the reliability of the actuator's Ethernet port and R_k is the reliability of the controller attached to the actuator. Note that $NR_{sim}(t)$ is the reliability of the simplex system from a nodes point of view.

$$NR_{sim}(t) = \left[\left(R_{Sup}\right)\times\left(R_k\right)^4 \times \left(R_s\right)^{16}\right]^2 \tag{16}$$

It is again assumed that both supervisors have the same failure rate. All sensors also have the same failure rate as do all controllers.

3. Two-Cell Architecture with TMR

Although connecting two cells together is expected to increase system reliability (whether $CFR(t)$ or $NR(t)$), there is still the problem of single points of failure, *i.e.*, the sensors. If any one sensor out of the thirty two sensors fails, the whole system will fail. In [13] (as mentioned in Section 2), it was shown that applying Triple Modular Redundancy (TMR) at the sensor level was feasible and that it increased system reliability as expected.

The same approach is going to be implemented next; TMR is applied to all 32 sensors (for the two cells). A sensor will only fail if at least two of its three modules fail. The application of TMR results in 48 sensors in cell one and 48 sensors in cell two, giving a total of 96 sensors in the system. And while the increase from 32 sensors to 96 sensors means an increase in costs, the system is expected to become much more reliable. Such an expensive but extremely reliable architecture is appropriate for applications in sensitive environments such as the nuclear industry or the space industry where reliability is the most important factor in system design.

On the other hand, such an increase in the number of sensors will significantly increase network traffic. Note that any control packet must not have an end-to-end delay greater than the 694 µs system control deadline. Clearly, the fault-free and fault-tolerant scenarios will be identical to those described in Sections 3.1 and 3.2. However, the delays are expected to increase significantly due to the extra traffic generated by 96 sensors instead of 32.

3.1. Fault-Free (FF) and Fault-Tolerant (FT) Scenarios

In addition to the fault-free scenario, this system must meet the required control deadline under all the fault-tolerant scenarios stated in Section 3.2. Furthermore, as long as two out of every three sensors are working, the system will tolerate the failure, even if one sensor fails during mission time.

Although the sensor to actuator delays experienced in the proposed model in Section 3 were under the 694 microsecond limit, there is an expected increase in the delays for the proposed TMR model because of the large amount of extra traffic due to TMR. In addition to meeting the required control delay deadline, the proposed TMR model must guarantee zero control packet loss.

3.2. Delay Analysis (with TMR)

Calculations for TMR applied to two cells are obtained in the same manner depicted in Section 3.3 for two cells connected to each other without TMR. The only difference is an increase in the number of packets in the system. Furthermore, calculations (and OMNeT++ simulations) were conducted only using Gigabit Ethernet links because it was shown in [13] that using fast Ethernet causes the delay to exceed the 694 µs deadline when applied to cells with TMR sensors. Below is the analytical sensor to actuator delay calculations using equation (8) for the fault-free and the eleven fault-tolerant scenarios.

3.2.1. Fault-Free (FF) Scenario

$$\text{Cell 1: } D_{\text{Total}} = \left(52\times\left(1.264+0.0075\right)\right) = 66.118\ \mu\text{s} \qquad (17)$$

$$\text{Cell 2: } D_{\text{Total}} = \left(52\times\left(1.264+0.0075\right)\right) = 66.118\ \mu\text{s} \qquad (18)$$

3.2.2. Fault-Tolerant (FT) Scenarios

Following the same delay calculation methodology, **Table 3** summarizes the delay analysis for all aforementioned Fault-Tolerant (FT) scenarios.

3.3. Simulation vs. Analytical Results (with TMR)

The proposed two cell fault-tolerant architecture, after applying TMR on the sensor nodes, was simulated on OMNeT++ [24] in the fault-free scenario as well as under all outlined failure scenarios. For all simulated scenarios, all control packets were transmitted successfully with no packet losses. Additionally, all observed control packet

Table 3. Summary of analytival delays and percentage error (simulation vs. theoretical) for all TMR scenarios.

Scenario	Cell 1			Cell 2		
	Packets	Gigabit Ethernet delay (µs)	% Error	Packets	Gigabit Ethernet delay (µs)	% Error
FF	52	66.118	4.28	52	66.118	4.28
FT-1	52	66.118	4.28	52	66.118	4.28
FT-2	52	66.118	4.28	52	66.118	4.28
FT-3	54	68.661	4.29	52	66.118	4.28
FT-4	52	66.118	4.28	54	68.661	4.29
FT-5	54	68.661	4.29	54	68.661	4.29
FT-6	56	71.204	4.31	52	66.118	4.28
FT-7	52	66.118	6.08	56	71.204	4.31
FT-8	52	66.118	4.29	54	68.661	6.03
FT-9	54	68.661	4.29	52	66.118	4.29
FT-10	108	137.322	3.39	54	68.661	4.29
FT-11	54	68.661	4.29	108	137.322	3.39

delays were less than the required system deadline. In all simulated scenarios, the highest observed percentage error between the simulated and analytical delay results was 6.08% as shown in **Table 3**.

3.4. Reliability Modeling

3.4.1. Control Function Reliability (CFR)

From a control function point of view, which only focuses on the supervisors and the controllers within the actuators, the reliability equation does not change from the two cell system without TMR, even with the addition of the extra sensors because the sensors are not taken into consideration in this reliability model. Hence, the reliability equations are still the same as in (13) and (14).

3.4.2. Node Reliability (NR)

System reliability can also be calculated by taking into account the sensors and the Ethernet ports within the actuators, in addition to the supervisors and the controllers within the actuators. In this case, the equation will only differ in the sensor block, but the rest of the equation from (15) will remain the same. Instead of the reliability of the sensors being R_s^{32}, it will be $\left[3R_s^2 - 2R_s^3\right]^{32}$ [25] [26], making the system much more reliable during mission time. As such, the reliability equation is:

$$NR_{TMR}(t) = \left[3R_s^2 - 2R_s^3\right]^{32} \times \left[1-\left(1-R_{Sup1}\right)\times\left(1-R_{Sup2}\right)\right]\times\left[1-\left(1-R_k\right)\times\left(1-R_a\right)\right]^8. \quad (19)$$

4. Case Study

For the two proposed fault-tolerant architectures, a case study was conducted to quantify overall system reliability compared to a corresponding simplex system. An exponential Time To Failure (TTF) is assumed with time measured in days. **Table 4** summarizes the assumed case study parameters.

Based on the case study parameters assumed in **Table 4**, the CFR and NR for the proposed system will be compared to a corresponding system with no TMR as well as a simplex system.

4.1. Control Function Reliability (CFR)

The CFR for the two proposed architectures is shown, and compared to the reliability of a corresponding simplex system with no fault-tolerance, in **Figure 4**.

It can be seen that, from a Control Function perspective, there is no difference in overall system reliability between the two proposed fault-tolerant architectures (with and without TMR). However, in both cases, a significant improvement in CFR can be seen compared to the corresponding simplex architecture.

4.2. Node Reliability (NR)

Figure 5 illustrates the NR for the three studied system architectures.

It can be seen that, when Node Reliability is taken into account, the proposed fault-tolerant architecture with TMR shows significant improvement in reliability compared to that without TMR as well as the simplex architecture. This is due to the large number of sensor nodes (32) utilized across the architecture's two cells. Without TMR, each of these sensor nodes is a single point of failure for the entire system.

Table 4. Summary of assumed reliability modeling parameters.

Parameter	Value
λ_{sensor}	1/365 (days^{-1})
λ_a	1/365 (days^{-1})
$\lambda_{\text{supervisor}}$	1/180 (days^{-1})
$\lambda_{\text{controller}}$	1/90 (days^{-1})
Coverage (C)	1

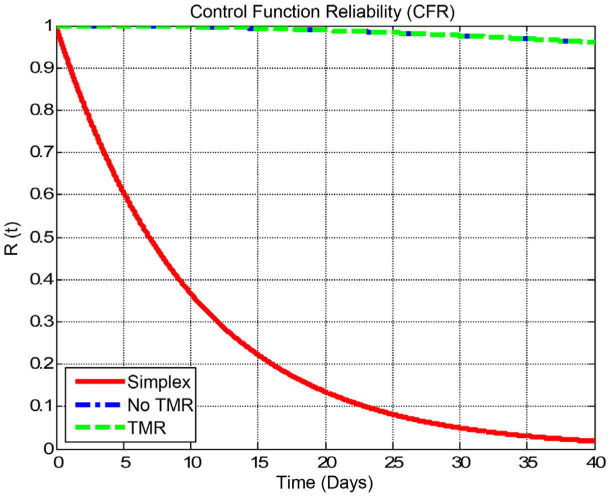

Figure 4. Control Function Reliability (CFR) over time.

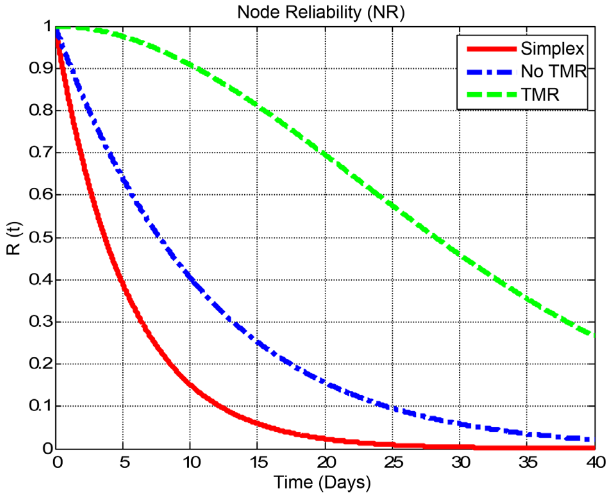

Figure 5. Node Reliability (NR) over time.

5. Conclusions

Fault-tolerant design is essential for a robust Networked Control System (NCS) with a high reliability and a long operational lifetime. With the increasing complexity of NCSs consisting of a large number of nodes such as sensors, controllers and actuators, the probability of the occurrence of any single failure increases significantly. Without fault-tolerance, the occurrence of a single fault in any one node can lead to the failure of the entire control system resulting in lengthy downtimes and consequently significant production losses.

In this paper, the architecture of a two-cell fault-tolerant NCS is developed on-top-of both Unmodified Fast and Gigabit Switched Ethernet. The proposed architecture models a production line composed of two identical machines each based on a Sensor-to-Actuator (S2A) control architecture.

Fault-tolerance is first applied at the controller level across both cells. An extra network interface is added to each actuator node in addition to the integrated controller node. In case of failure of the integrated controller, a supervisor node becomes part of the control loop and sends the required control action to the affected actuator's added network card. Under all possible failure scenarios, it was shown that the proposed fault-tolerant architecture fulfils the required control system deadline with zero dropped or over-delayed packets under both Fast and Gigabit Ethernet (both analytically and through OMNeT++ simulations).

Additionally, the fault-tolerance of the proposed architecture was expanded to the level of the sensor nodes through Triple Modular Redundancy (TMR). The application of TMR at the sensor nodes led to a significant increase in the number of control packets transmitted across the proposed architecture making Fast Ethernet unsuitable for meeting the required control system deadline. It was shown that the proposed fault-tolerant architecture fulfils the required control system deadline with zero dropped or over-delayed packets using Gigabit Ethernet under all possible failure scenarios.

Two reliability modeling methodologies were illustrated to quantify the achievable improvement in lifetime compared to a corresponding simplex architecture with no fault-tolerance: Control Function Reliability (CFR) and Node Reliability (NR). CFR only considers the probability of failure of the integrated controllers and supervisors while NR also takes into account the probability of failure of the sensor Ethernet ports and the added integrated network interfaces. A case study was carried out for a typical industrial system. It was shown that, for both modeling methodologies, the proposed fault-tolerant architectures significantly improve overall system reliability.

References

[1] ODVA, Volume 1: CIP Common. http://www.odva.org/10_2/03_events/03_ethernet-homepage.htm

[2] ODVA, Volume 2: EtherNet/IP Adaptation on CIP. http://www.odva.org/10_2/03_events/03_ethernet-homepage.htm

[3] IEEE 802.3 Std.

[4] IEC 61784-1/2. www.iec.ch

[5] Pedreiras, P., Almeida, L. and Gai, P. (2002) The FTT-Ethernet Protocol: Merging Flexibility, Timeliness and Efficiency. *Proceedings of the IEEE Euromicro Conference on Real-Time Systems ECRTS*, Vienna, June 2002, 134-142. http://dx.doi.org/10.1109/emrts.2002.1019195

[6] Decotignie, J.D. (2005) Ethernet-Based Real-Time and Industrial Communications. *Proceedings of the IEEE*, **93**, 1102-1117. http://dx.doi.org/10.1109/JPROC.2005.849721

[7] Marsal, G. (2006) Evaluation of Time Performances of Ethernet-Based Automation Systems by Simulation of High-Level Petri Nets. PhD Thesis, Ecole Normale Superieure De Cachan, December 2006.

[8] Felser, M. (2005) Real-Time Ethernet-Industry Prospective. *Proceedings of the IEEE*, **93**, 1118-1129. http://dx.doi.org/10.1109/JPROC.2005.849720

[9] Skeie, T., Johannessen, S. and Brunner, C. (2002) Ethernet in Substation Automation. *IEEE Control Systems*, **22**, 43-51. http://dx.doi.org/10.1109/MCS.2002.1003998

[10] Daoud, R., Elsayed, H., Amer, H. and Eid, S. (2003) Performance of Fast and Gigabit Ethernet in Networked Control Systems. *Proceedings of the* 46*th IEEE Midwest Symposium on Circuits and Systems MWSCAS*, Cairo, December 2003, 505-508. http://dx.doi.org/10.1109/MWSCAS.2003.1562328

[11] Daoud, R., Amer, H. and ElSayed, H. (2005) Fault-Tolerant Two-Level Pyramid Networked Control Systems. *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Catania, 19-22 September 2005, 974. http://dx.doi.org/10.1109/etfa.2005.1612629

[12] Moustafa, E., Halawa, H., Daoud, R. and Amer, H. (2014) Evaluating the Performance of Fault-Tolerant S2A vs. In-Loop Controller Models for Ethernet-Based NCS. *Intelligent Control and Automation*, **5**, 81-90. http://dx.doi.org/10.4236/ica.2014.52009

[13] Abouelazayem, S., Ibrahim, A., Morsi, M., Abou Eita, M., Hussein, M., Moustafa, E., Halawa, H., Daoud, R., Amer, H. and Elsayed, H. (2014) TMR Sensors for Reliable S2A Architectures. *Proceedings of the International IEEE Conference on Ultra Modern Telecommunications and Control Systems*, ICUMT, St. Petersburg, October 2014, 624-630. http://dx.doi.org/10.1109/icumt.2014.7002173

[14] Thomsen, J. and Blanke, M. (2006) Fault-Tolerant Actuator System for Electrical Steering of Vehicles. *Proceedings of the* 32*nd Annual Conference of the IEEE Industrial Electronics Society IECON*, Paris, 6-10 November 2006, 3597-3602. http://dx.doi.org/10.1109/iecon.2006.347693

[15] Gamer, T., Oriol, M. and Wahler, M. (2014) Increasing Efficiency of M-out-of-N Redundancy. *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Barcelona, 16-19 September 2014, 1-8. http://dx.doi.org/10.1109/etfa.2014.7005105

[16] Proenza, J., Barranco, M., Rodriguez-Navas, G., Gessner, D., Guardiola, F. and Almeida, L. (2012) The Design of the CANbids Architecture. *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Krakow, 17-21 September 2012, 1-8. http://dx.doi.org/10.1109/etfa.2012.6489646

[17] Halawa, H., Hilal, Y., Aziz, G., Alfi, C., Refaat, T., Daoud, R., Amer, H. and ElSayed, H. (2014) Network Fabric Redundancy in NCS. *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Barcelona, September 2014.

[18] Barranco, M., Pozo, F. and Proenza, J. (2014) A Model for Quantifying the Reliability of Highly-Reliable Distributed Systems Based on Fieldbus Replicated Buses. *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Barcelona, 16-19 September 2014, 1-8. http://dx.doi.org/10.1109/etfa.2014.7005236

[19] Kehrer, S., Kleineberg, O. and Heffernan, D. (2014) A Comparison of Fault-Tolerance Concepts for IEEE 802.1 Time Sensitive Networks (TSN). *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Barcelona, 16-19 September 2014, 1-8. http://dx.doi.org/10.1109/etfa.2014.7005200

[20] Marti, P., Fuertes, J.M. and Fohler, G. (2001) An Integrated Approach to Realtime Distributed Control Systems over Fieldbuses. *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Antipes/Juan les Pins, October 2001.

[21] Moustafa, E., Halawa, H., Daoud, R. and Amer, H. (2013) Sensor Actuator Ethernet-Based Networked Control Systems. *Proceedings of the International IEEE Conference on Sciences & Computer Engineering STA*, Sousse, 20-22 December 2013, 530-534. http://dx.doi.org/10.1109/sta.2013.6783183

[22] Seno, L., Vitturi, S. and Tramarin, F. (2011) Experimental Evaluation of the Service Time for Industrial Hybrid (Wired/Wireless) Networks under Non-Ideal Environmental Conditions. *Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation ETFA*, Toulouse, 5-9 September 2011, 1-8. http://dx.doi.org/10.1109/etfa.2011.6059005

[23] Kurose, J.F. and Ross, K.W. (2000) Computer Networking: A Top-Down Approach Featuring the Internet. Addison Wesley Publishing Company.

[24] Official Site For OMNeT++. www.omnetpp.org

[25] Trivedi, K.S. (2002) Probability and Statistics with Reliability, Queuing, and Computer Science Applications. Wiley, NY.

[26] Siewiorek, D.P. and Swarz, R.S. (1998) Reliable Computer Systems—Design and Evaluation. A K Peters, Natick.