Redundant System Basic Concepts

Updated Oct 2, 2020

Introduction

Redundancy is a common approach to improve the reliability and availability of a system. Adding redundancy increases the cost and complexity of a system design and with the high reliability of modern electrical and mechanical components, many applications do not need redundancy in order to be successful. However, if the cost of failure is high enough, redundancy may be an attractive option.

This paper provides a basic background into the types of redundancy that can be built into a system and explains how to calculate the effect of redundancy on system reliability. National Instruments controllers provide the flexibility to create a variety of redundant architectures.

Models of Redundancy

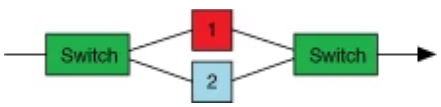
While there are various methods, techniques, and terminologies for implementing redundancy, the following models represent the more common ones used in industry. The three main models described in this paper are *Standby Redundancy, N Modular Redundancy,* and *1:N Redundancy*.

Standby Redundancy

Standby redundancy, also known as *Backup Redundancy* is when you have an identical secondary unit to back up the primary unit. The secondary unit typically does not monitor the system, but is there just as a spare. The standby unit is not usually kept in sync with the primary unit, so it must reconcile its input and output signals on takeover of the Device Under Control (DUC). This approach does lend itself to give a “bump” on transfer, meaning the secondary may send control signals to the DUC that are not in sync with the last control signals that came from the primary unit.

You also need a third party to be the watchdog, which monitors the system to decide when a switchover condition is met and command the system to switch control to the standby unit and a voter, which is the component that decides when to switch over and which unit is given control of the DUC. The system cost increase for this type of redundancy is usually about 2X or less depending on your software development costs. In Standby redundancy there are two basic types, *Cold Standby* and *Hot Standby.*

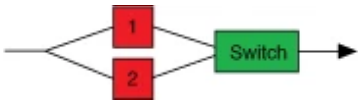
Cold Standby



In cold standby, the secondary unit is powered off, thus preserving the reliability of the unit. The drawback of this design is that the downtime is greater than in hot standby, because you have to power up the standby unit and bring it online into

a known state. This makes it more challenging to reconcile synchronization issues, but do to the length of the time it takes to bring the standby unit on line, you will usually suffer a big bump on switchover.

Hot Standby



In hot standby, the secondary unit is powered up and can optionally monitor the DUC. If you use the secondary unit as the watchdog and/or voter to decide when to switch over, you can eliminate the need for a third party to this job. This design does not preserve the reliability of the standby unit as well as the cold standby design. However, it shortens the downtime, which in turn increases the availability of the system.

Some flavors of *Hot Standby* are similar to *Dual Modular Redundancy (DMR)* or *Parallel Redundancy*, which are covered in the next section. These naming conventions are commonly interchanged. The main difference between *Hot Standby* and *DMR* is how tightly the primary and the secondary are synchronized. DMR completely synchronizes the primary and secondary units. However, some literature loosely uses the term DMR for any redundancy model that use two units.

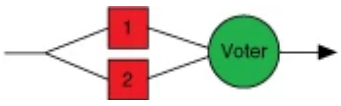
N Modular Redundancy

N Modular Redundancy, also known as Parallel Redundancy, refers to the approach of having multiply units running in parallel. All units are highly synchronized and receive the same input information at the same time. Their output values are then compared and a voter decides which output values should be used. This model easily provides bumpless switchovers.

This model typically has faster switchover times than Hot Standby models, thus the system availability is very high, but because all the units are powered up and actively engaged with the DUC, the system is at more risk of encountering a common mode failure across all the units. There are techniques to help avoid this risk that are mentioned later in this paper.

Deciding which unit is correct can be challenging if you only have two units. Sometimes you just have to choose which one you are going to trust the most and it can get complicated. If you have more than two units the problem is simpler, usually the majority wins or the two that agree win. In N Modular Redundancy, there are three main typologies: *Dual Modular Redundancy, Triple Modular Redundancy,* and *Quadruple Redundancy*.

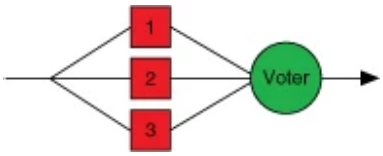
Dual Modular Redundancy



Dual Modular Redundancy (DMR) uses two functional equivalent units, thus either can control the DUC. The most challenging aspect of DMR is determining when to switch over to the secondary unit. Because both units are monitoring the application, you have to decide what to do if they disagree. You either need to create a tiebreaker vote or simply designate the secondary unit as the default winner, assuming it is more trustworthy than the primary unit. You may be able to trust the secondary unit more if the primary is normally in control and if regular diagnostics are run on the secondary to help insure its reliability, but this is very application specific.

The average cost increase of a DMR system is about twice that of a non-redundant system, factoring in the cost of the additional hardware and the extra software development time.

Triple Modular Redundancy



Triple Modular Redundancy (TMR) uses three functionally equivalent units to provide redundant backup. This approach is very common in aerospace applications where the cost of failure is extremely high.

TMR is more reliable than DMR due to two main aspects. The most obvious reason is that you now have two “standby” units instead of just one. The other reason is that in TMR, you commonly see a technique called diversity platforms or diversity programming applied. In this technique, you would use different software or hardware platforms on your redundant systems to prevent common mode failure.

For example, units 1 and 2 could be designed using one hardware platform such as CompactRIO, and unit 3 would use PXI or Compact FieldPoint. You can apply the same concept to software by using LabVIEW for two of the systems and LabWindows/CVI for the other. You may even have entirely different development teams working separately on the different the systems as well. Every system is designed to solve the same problem using the same common requirements.

The voter decides which unit will actively control the application. With TMR, the decision of which system to trust is made democratically and the majority rules. If you get three different answers, the voter must decide which system to trust or shut down the entire system. Thus, the switchover decision is straightforward and fast. The drawback of this approach is cost; TMR can raise the cost of your system by at least 3X.

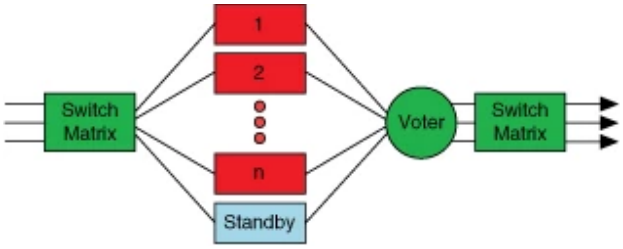
Quadruple Modular Redundancy

Quadruple Modular Redundancy (QMR) is fundamentally similar to TMR but using four units instead of three to increase the reliability. The obvious drawback is the 4X increase in system cost.

1:N Redundancy

1:N is a design technique used where you have a single backup for multiple systems and this backup is able to function in the place of any single one of the active systems. This technique offers redundancy at a much lower cost than the other models by using one standby unit for several primary units.

This approach only works well when the primary units all have very similar functions, thus allowing the standby to back up any of the primary units if one of them fails.



Other drawbacks of this approach are the added complexity of deciding when to switch and of a switch matrix that can reroute the signals correctly and efficiently.

Redundancy Improves Reliability

Reliability is defined as the probability of not failing in a particular environment for a particular mission time. Reliability is a statistical probability and there are no absolutes or guarantees. The goal is to increase the odds of success as much as you can within reason.

The following equation is the probability equation most commonly used in industry to calculate reliability. This equation assumes that you have a constant failure rate (λ).

$R(t) = e^{-\lambda t}$, where:

R(t) is the probability of success

t is the mission time, or the time the system must execute without an outage

λ is the constant failure rate over time (*N* failures/hour)

1/λ is the Mean Time To Failure (MTTF)

Usually, the only factor that you can have an impact on is the failure rate (λ). The environment is determined by the nature of the application itself. You cannot normally change the mission time, unless you can work in planned maintenance at strategic times. Therefore, you can best influence the reliability of your system through prudent part selection and design practices.

Basic math demonstrates how redundant system design practices can improve your system reliability:

R is the probability of success and **F** is the probability of failure. Based on mathematical set theory, all applications are either successful or not successful (a failure). Thus the sum of the two logical states is unity.

Therefore we can state that:

R + F = 1

Because most companies tend to track failures more than successes, you can turn the equation around to calculate reliability:

R = 1 - F

The following example uses a 10% probability of failure:

R = 1 - 0.1 = 0.90

The probability of success for a given system would be 90%, or 90 out of 100 should succeed.

To use this concept to calculate the reliability of a redundant system, use the following equation:

$$R = 1 - (F_1)(F_2)$$

If systems 1 and 2 are redundant systems, meaning that one system can functionally back up the other, then F_1 is the probability of failure of system 1 and F_2 is the probability of failure of system 2.

If we leverage redundancy, the probability of success for a given system would be 99%, or 99 out of 100 should succeed:

R = 1 - (0.1)(0.1) = 0.99

Another way to calculate reliability is to take the probability equation and instead solve for the mean time to failure (**MTTF**) of the system:

$$R(t) = e^{-\lambda t} = e^{-t/MTTF}$$

Solving for **MTTF**:

$$MTTF = -\left(t / ln\,[R(t)]\right)$$

For example, if your application had a mission time of 24 hours a day, 7 days a week, for one year (24/7/365) and you experienced a success rate of 90%:

MTTF = - (1 yr / *ln* [.90]) = 9.49 years

If you add redundancy, then the success rate increases to approximately 99% for the same mission time:

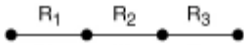
MTTF = - (1 yr / *ln* [.99]) = 99.50 years

These equations effectively demonstrate the vast improvement in reliability that redundancy can bring to any system.

Levels of Redundancy

There are many situations where it might be more beneficial to employ redundancy only to a less reliable component of the system.

The following model depicts a system that has three dependent components in series. If one component fails, the entire system fails.



R_1 = Component 1 reliability

R_2 = Component 2 reliability

R_3 = Component 3 reliability

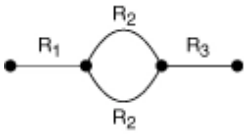
You can calculate the reliability of the entire system by multiplying the reliability of each of the components together.

$R_{system} = (R_1)\ (R_2)\ (R_3)$

For an example, if **R_1** = .98, **R_2** = .85, and **R_3** = .97, then the reliability of the system would be:

$R_{system} = (.98)\ (.85)\ (.97) = .81$

If you back up just the least reliable component of the system, **R_2**, with redundancy, the model now looks like this:



Remember, since:

R = 1 - F

And for a redundant component:

$R = 1 - (F_1)(F_2)$

The reliability equation of the system now looks like this:

$R_{system} = (R_1)\ (1-(F_2)(F_2))\ (R_3)$

If you recalculate the example for R_1 = .98, R_2 = .85, and R_3 = .97, the reliability of the system is now:

R_{system} = (.98) (1 - (.15) (.15)) (.97) = .93

We have improved the reliability of the system by 12 percentage points by implementing redundancy for only one component.

There are many levels at which you can implement this strategy and it is application specific. You have to understand which components are most likely to fail. It may be at the sensor, the cabling, the device, the chassis, the power supply, or any number of other components of the system.

Redundancy Improves Availability

Availability is the percentage of time that a system is up and running for a particular mission time.

Availability = Uptime/(Uptime + Downtime)

If you have a mission time of 24/7 for six months and have no downtime, then you would have 100% availability. If you have one day of downtime for that same mission, then the availability of the system becomes 99.4%.

When you are developing strategies for improving availability, you must first accept the reality that you will have to deal with failures occasionally. So the focus in designing any system for high availability is to reduce downtime and make the repair time as short as possible.

Without redundancy, the system downtime depends on how quickly you can:

1. Detect the failure.
2. Diagnose the problem.
3. Repair or replace the failed part of the system.
4. Return the system to full operational status.

For hardware failures, the best solution is to replace the bad component or system. Depending on the accessibility and availability of spares, replacement could take anywhere from a few minutes to several days. If you encountered a software failure, you may only need to reboot the system to temporarily repair it. However, rebooting could take from a few seconds to multiple hours, depending on the complexity of the system.

With redundancy, the system downtime is dependent on how fast you can detect a failure and switch over to the backup unit. This is easily under one second and many systems can achieve sub-millisecond downtimes. Redundancy can therefore improve the availability of your system by several orders of magnitude.

For example, consider a system that needs to run 24/7 for one year. If it experiences one hour of downtime during the

mission, the availability would be 99.9% (or *3 nines* in availability-speak.) If you use redundancy and the downtime is one second, the availability would be 99.99999% or *7 nines*.

Mission Time	Total Downtime	Availability
24/7 for 1 year	1.2 months	90.0%
24/7 for 1 year	3.7 days	99.0%
24/7 for 1 year	8.8 hours	99.9%
24/7 for 1 year	5.3 minutes	99.999%
24/7 for 1 year	3.2 seconds	99.99999%

Another important concept to remember is that the switchover times for redundancy are commonly so fast that the system is not noticeably affected by the downtime. Thus, for all practical purposes, the system never experienced an outage, thus achieving 100% availability.

Summary

Redundancy can greatly improve the reliability and availabilitly of your control and/or monitoring system. Most applications do not need redundancy to be successful, but if the cost of failure is high enough you may need redundancy. You will need to assess your application and choose the redundancy model that best meets your needs. With National Instruments products, you can implement any of the redundant models mentioned above.

For more information about using National Instruments product in applications that require high reliability and or high availability, contact your local sales engineer.