Interlocks

Safety Systems

David Cass Tyler©

PO Box 1026
Willard, NM 87063

David.Cass.Tyler@gmail.com

12/18/2010

Interlock systems provide the important ability to render a system "safe" and to prevent it from posing a threat to people and equipment. Secondarily, interlock systems provide information that it is safe to operate a system.

Table of Contents

Abstract 3
How the Interlock Systems Hardware Works 3
Using Physical Switches as Inputs 3
Using Relays as Inputs 4
Using Relay Coils as Outputs 4
Software Interlocks 4
Fail-Safe Design Considerations 4
How the Interlock Systems Software Works 4
Operator Arm/Abort Panel 5
Operator Computer Interface 5
Summary 6

Abstract

Interlock systems provide the important ability to render a system "safe" and to prevent it from posing a threat to people and equipment. Secondarily, interlock systems provide confirmation that it *is* safe to operate a system.

How the Interlock Systems Hardware Works

Think of an interlock loop as a series of switches that must *all* be closed (on) to close the interlocks. If any of the switches are turned off, the interlocks are open. Suppose that one of the switches is a magnetic switch on a personnel access door. If someone enters, the interlock circuit would be opened indicating that the system is not safe to operate.

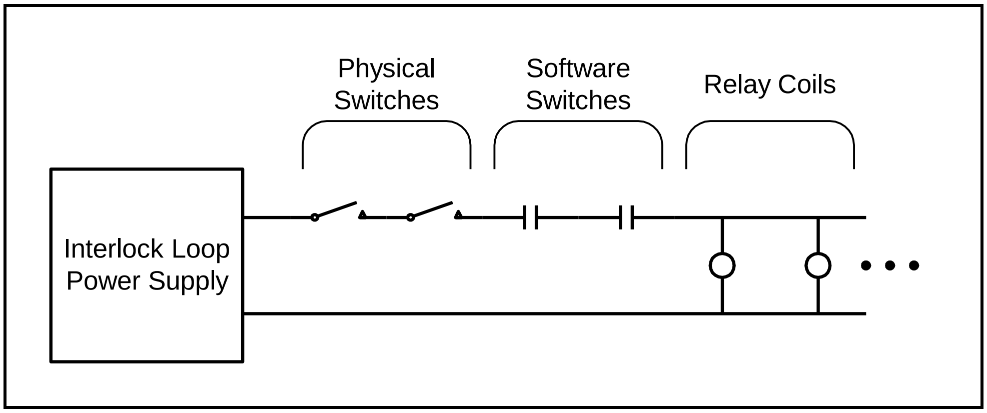


Figure 1 - An Example Interlock System

Pictured above is an example interlock system. The interlock loop shows a couple of physical switches and a couple of relays that may be software activated. When all of these are closed, the circuit is closed and power is supplied to the relay coils. These relay coils can then provide power to other systems – "Interlock Ok" signals, relays, solid state relays, etc. You might provide the power to hold normally closed safety dump circuits open, for instance. Thus, if the interlocks are broken, the relays that provide power to the dump circuits open, power is lost and the dump circuits automatically drop, thereby shorting out any residual high voltage and rendering the system safe. Industrial Laser systems, for instance, make extensive use of interlock systems to shut-down dangerous power circuits and high intensity laser beams that might harm people if access panels or opened or beams are fired, etc.

Using Physical Switches as Inputs

The physical switches shown in Figure 1 above can include magnetic switches on personnel access doors or doors on equipment racks that might expose equipment to RF and electromagnetic pulses, etc.

Using Relays as Inputs

If we use relays to proffer our interlock “vote” we can sense the activation leg using a Digital Input or an indicator lamp. This allows us to programmatically and/or visually determine if a particular input is present or not. This makes the interlock state easier and quicker to determine. We can use a Digital Output from an embedded controller to activate a relay and thereby give the controller a participating “vote” in the interlock scheme. Loss of communication with the Operator Control GUI might be considered important enough to trip the interlocks, as an example. We can also use “normally closed” relays so that we can trip the interlocks using motion sensors, etc.

Using Relay Coils as Outputs

We can use the completed interlock circuit to drive Relay Coils that provide Low Voltage outputs, High Voltage outputs, A/C outputs and virtually any other kind of interruptible output that we desire. We can provide the power to raise a normally closed Dump relay as an example. If the interlocks break, we lose activation power and the Dumps close. Thus, they automatically fail in a “safe” condition. We can do the same thing to trip normally open relays that connect 208 V 3 phase power to motors, power supplies, conveyor belts, etc. We can also power an Interlock Sense loop so that the embedded controllers can sense the loss of interlocks and shut things down.

Software Interlocks

There are efforts underway to define standards for Real-Time Ethernet based, networked, software Interlocks. Utilizing the new Quality of Service standards, it is possible to implement Near Real-Time, i.e. sub-second, software interlocks. While copper has the distinct advantage of working at the speed of light, software interlocks can implement an interlock system between multiple UAVs, across continent wide systems or for other systems where copper would be infeasible.

Fail-Safe Design Considerations

Safety Systems should be designed to “fail safe” – i.e. high voltages should be turned off, motors should be turned off, conveyor belts should be stopped, etc. Interlock signal lines should be “active high” so that if signal lines get run over or cut, the interlocks will break. Every effort should be made to assure positive feedback that the system has been “safed”. If you are unable to verify that the system is in a safe condition there should be an obvious indication such as a blinking red light. The job of the interlock system is to make the system safe and operators are staking their lives that the system truly is safe. Don’t let them down.

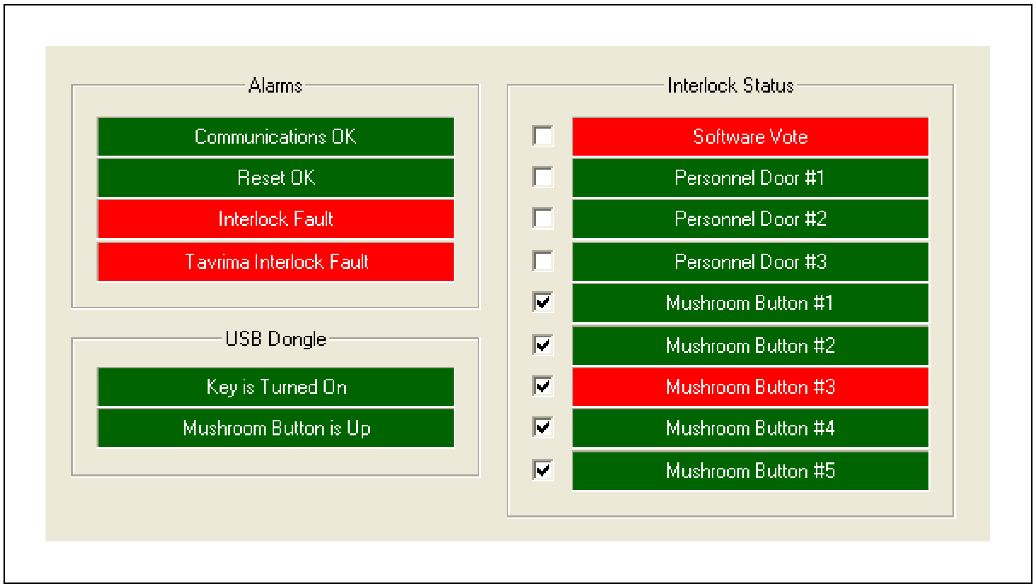
How the Interlock Systems Software Works

The embedded controllers can sense an “Interlock Ok” signal, break out of operational control loops and stop any process that might be potentially harmful, sometimes before it even starts. Most importantly, sensing an interlock fault allows you to put the system into a “maximum safe” state. Loss of interlock warrants taking the maximum safety precautions and presuming that human life is in danger. Operations such as turning on motors or starting conveyor belts should not be undertaken unless the interlocks are satisfied.

Operator Arm/Abort Panel

A nice operator panel would contain a key switch, a magnetic reset button, a mushroom button and a green LED. The key switch allows the operator to turn the interlocks off and pocket the key so that they can enter dangerous areas with no fear that someone might arm the system. There are key switches that trap the key when the system is on so that it must be turned to off to remove the key. The magnetic reset button acts like the on button on a table saw. It allows you to arm the system, but if the interlocks are broken, it will not re-arm after the interlocks are restored until the arm switch is pressed again. The “Big Red Mushroom Button” is a well understood shutdown mechanism. Once pushed in, it must be popped out before you can re-arm the interlocks. The green LED provides a positive indication that the system has been armed. Other features, such a klaxon that sounds 5 seconds before final arm and as a flashing red emergency lamp to alert that a test is in progress, can be incorporated to make the system more useful.

Operator Computer Interface



Alarms

🟩 Communications OK
🟩 Reset OK
🟥 Interlock Fault
🟥 Tavrima Interlock Fault

USB Dongle

🟩 Key is Turned On
🟩 Mushroom Button is Up

Interlock Status

☐ 🟥 Software Vote
☐ 🟩 Personnel Door #1
☐ 🟩 Personnel Door #2
☐ 🟩 Personnel Door #3
☑ 🟩 Mushroom Button #1
☑ 🟩 Mushroom Button #2
☑ 🟥 Mushroom Button #3
☑ 🟩 Mushroom Button #4
☑ 🟩 Mushroom Button #5

Figure 2 - Interlocks Display

The biggest headache with interlock systems is trying to figure out why they won’t arm. You can spend a lot of time running around checking access switches, etc. if the system doesn’t arm immediately. This is less of an issue if you provide an operator interface that displays the state of the interlock system. Figure 2, above, goes green when the interlocks are fully satisfied and problem areas are displayed in red.

As displayed, it indicates that mushroom button 3 is pushed in and that the embedded controller is not yet ready to proceed. Once both these conditions are cleared, the system can be re-armed by pushing the magnetic arm switch.

The system in Figure 2 uses a hardware panel with the key, the mushroom button and the arm switch implemented in a USB plug-in. This was done because the operator is physically removed from the embedded control so it was infeasible to run copper lines to the embedded controllers. Instead, the hardware panel plugs into a USB port, at the laptop that runs the Operator Interface. The Operator GUI monitors the hardware panel and communicates, via Ethernet, with an embedded interlock controller that resides with the test hardware. The embedded interlock controller is what monitors the states of the Personnel Doors and the Mushroom Buttons and it proffers a “software vote”. Among other considerations, the software vote monitors continuous “keep alive” connection with the Operator Interface and, via it, the USB hardware panel.

This system also talks to two interlock loops. The checked elements, in this case the mushroom buttons, will cause both a normal upset and also a “Tavrima” upset. This allows normal operations to be interrupted without discharging the Tavrima if a personnel door is opened and with discharging the Tavrimas if a mushroom button is pushed.

Summary

Interlock systems can range between very simple and very complex. All of them share a single goal – to protect humans and equipment from injury or damage. All of these systems should be designed to fail in a safe manner so that once the interlock system has tripped the system being protected is safe to approach. If the system cannot be confirmed to be safe, warning must be given so that human life is not endangered.