

SCS System Redundancy

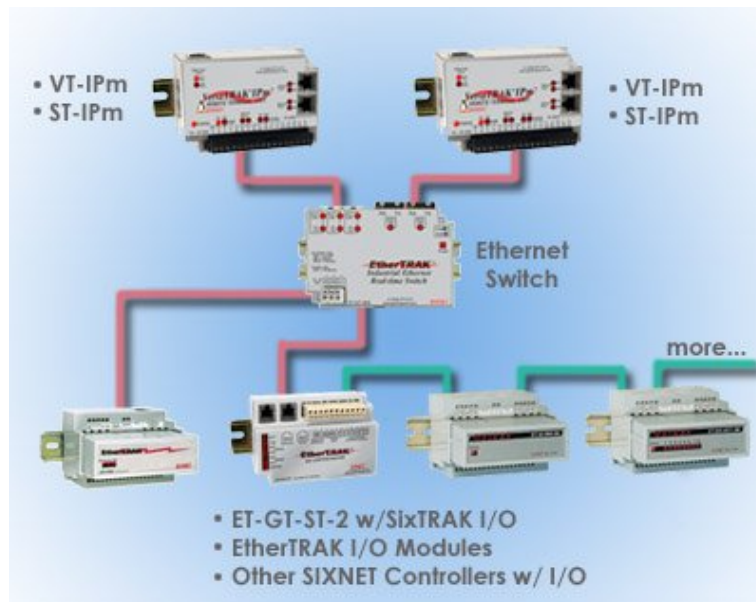
Abstract: This technical note describes techniques for configuring redundant process control systems employing SIXNET IPm controllers, distributed SixTRAK I/O stations and redundant Ethernet networks. Suggestions for enhancing system reliability and saving valuable engineering time are presented.

Suggested SIXNET Hardware / Architecture

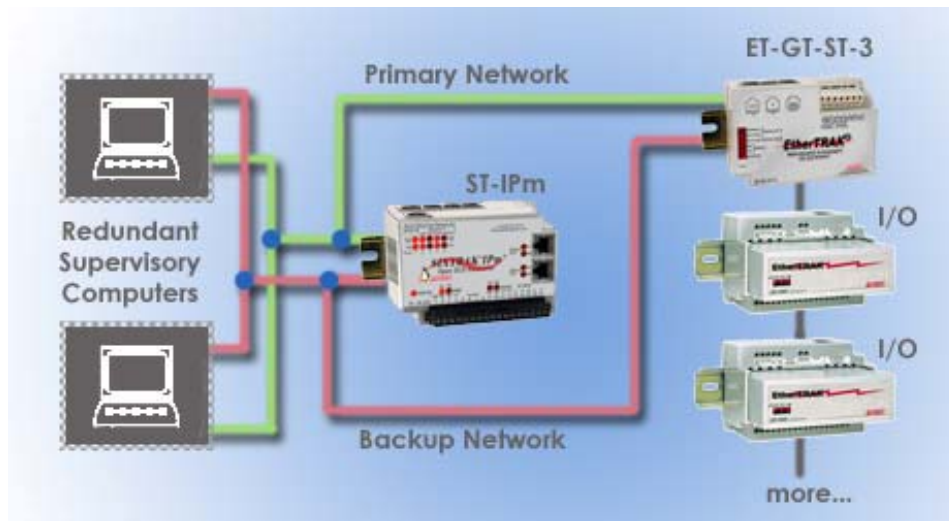
SIXNET provides the robust hardware needed in high reliability applications. SIXNET modular building blocks connect seamlessly to create fully integrated, high performance automation systems.

- SixTRAK IPm (ST-IPm-####) DCS Process Controllers
- VersaTRAK IPm (VT-IPm-####) Advanced RTUs
- EtherTRAK I/O and Ethernet-connected SixTRAK process quality I/O
- SIXNET real-time industrial Ethernet switches
- SIXNET managed switches control Ethernet “rings”

The distributed I/O can be any combination of EtherTRAK and/or RemoteTRAK I/O modules, or SixTRAK I/O connected to a SIXNET controller or I/O gateway. Please note that SixTRAK I/O connected directly to either redundant IPm controller will not have the benefit of the redundant processors. This technical note shows the preferred system architecture for redundant systems.

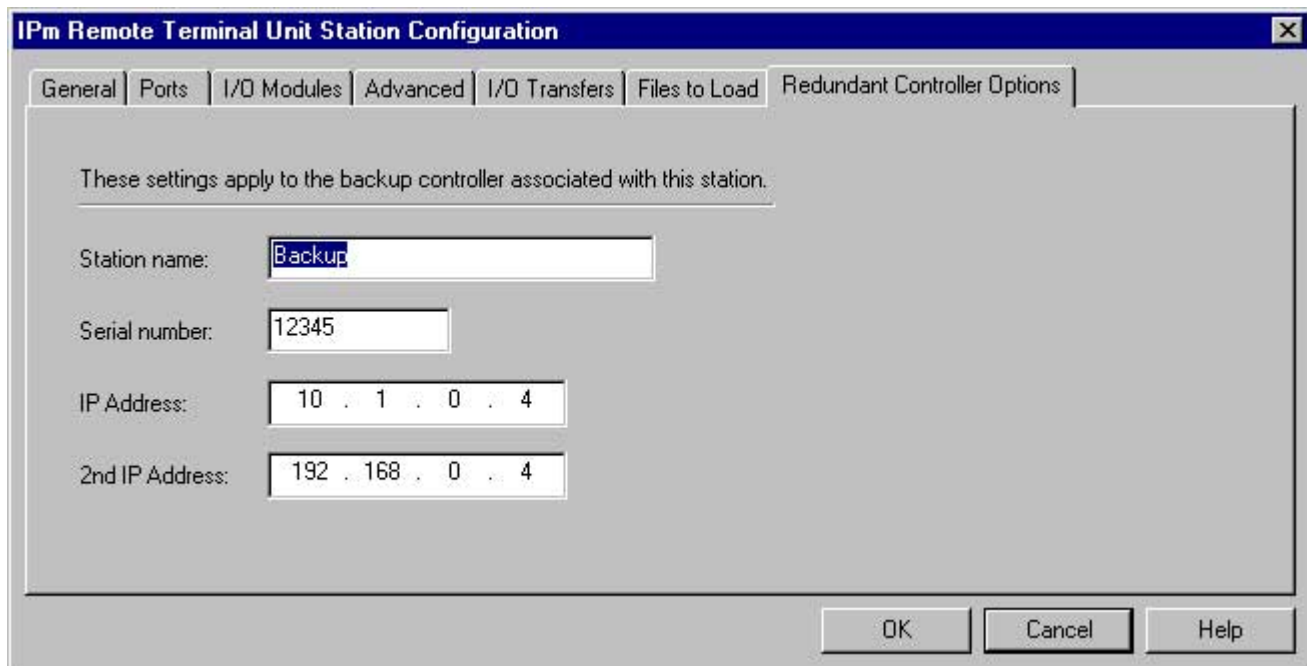


If redundant Ethernet communication paths between a controller and the distributed I/O stations are desired, then the use of SixTRAK IPm DCS controllers and ET-GT-ST-3, Redundant I/O gateways, are recommended. These devices have dual independent Ethernet ports for this purpose.



Configuring Redundant Controllers

The SIXNET I/O Tool Kit can automatically create and maintain the configuration for redundant VT-IPM and ST-IPM controllers. In the “General” (main) configuration window, check the box, “Automatically configure a backup controller”. Click the “Redundant Controller Options” tab and supply the station name, serial number and IP address(s) for the backup controller. (By default, the next available IP address that is higher than the IP address of the primary controller will be assigned.)



The I/O Tool Kit software will automatically create the configuration for the backup controller. Each time a modified configuration is saved, the configuration for the backup controller will automatically be updated as well. Each time the configuration is loaded into the primary controller, the Tool Kit will also load a fresh configuration into the backup controller. Since the “Load All” feature updates the ISaGRAF project in each controller, the application instructions will be synchronized as well. Advice is given later in this document for writing a single ISaGRAF program that is suitable for running in redundant controllers. The configuration of both controllers will be identically maintained, except as follows.

1. The station name and IP address of each controller will be unique. (Please note that the station number and tag names in each controller will be identical.)
2. A provision in the I/O Tool Kit permits the addition of unique I/O transfers in each controller. This is especially helpful in transferring “current status” I/O between the two processors.
3. The I/O Tool Kit will write a file: 'etc/stacfg/main_standby' into each controller. The two controllers will each have a distinct data value written into this file. This difference is used by the ISaGRAF program to distinguish the two controllers.

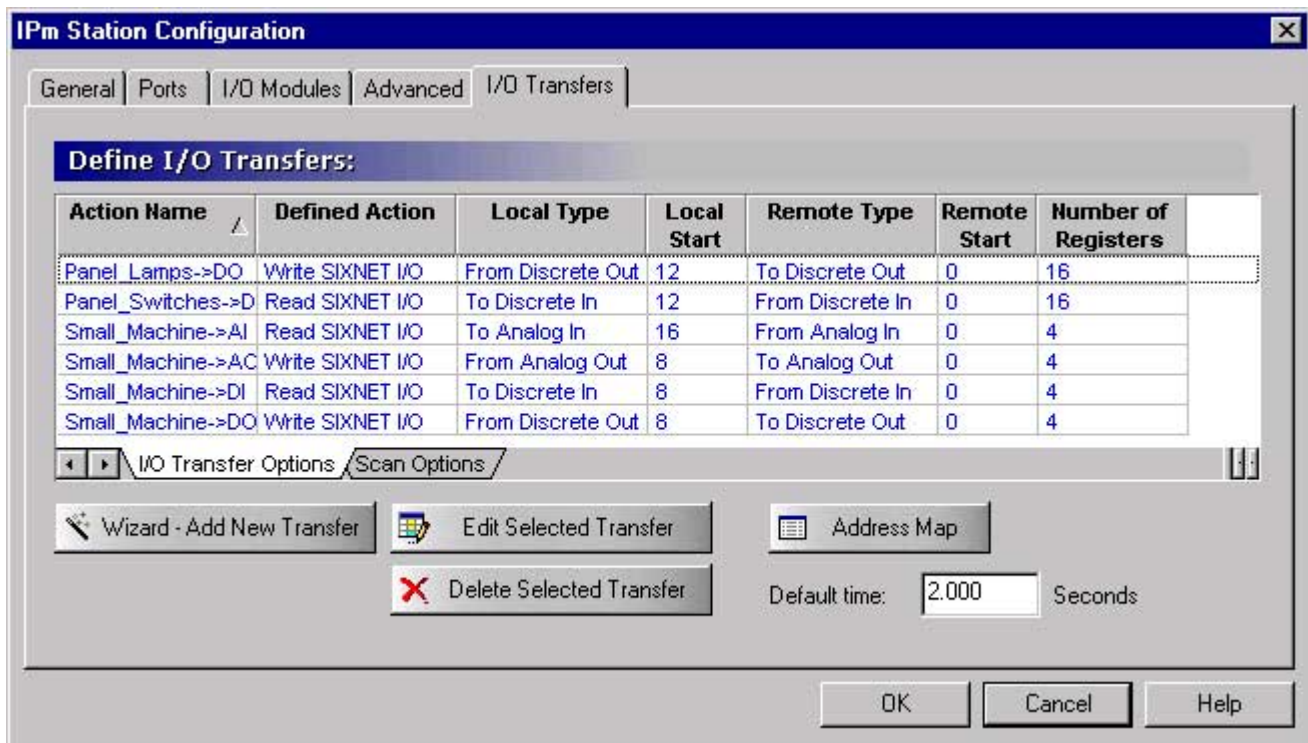
Here are the values written by the Tool Kit into the 'etc/stacfg/main_standby' file:

```

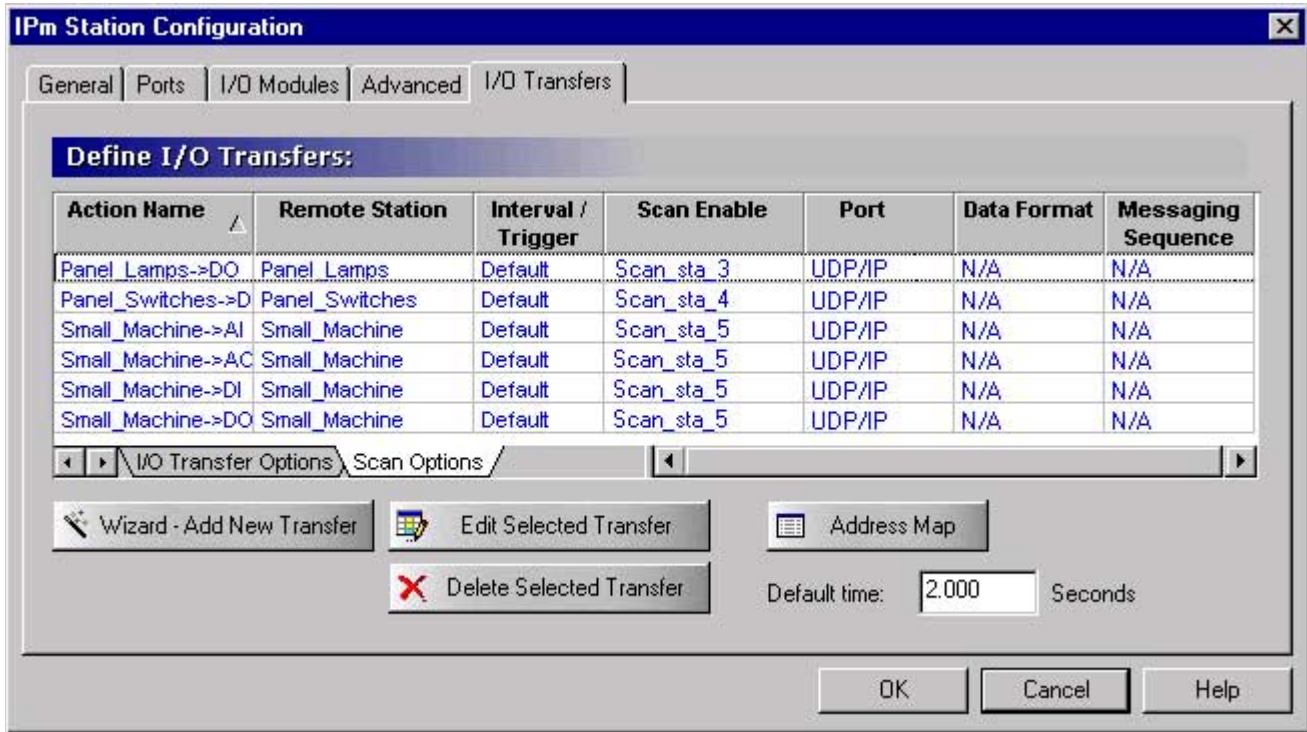
Primary controller    1
Backup controller    2
  
```

I/O Transfers and Switching Control of Outputs

Two or more IPm-based controllers are configured with “I/O Transfers” to read and/or write the distributed I/O. There is typically one I/O Transfer configured for each block of I/O in a distributed station or module.



When communicating over Ethernet, it is commonplace to allow I/O Transfers that read inputs or read outputs to run continuously in each controller. Reading outputs is a valuable way to keep standby processors in synchronization to the process. On the other hand, I/O Transfers that write outputs must be regulated in such a way that only one controller is writing outputs at any given time. This is where “Scan Enable” flags are used.



Typically, the states of virtual output registers are transferred between the two processors to enable the backup controller to stay in synchronization with the main (active) processor. Discrete, analog, long integer, and floating point registers may be used as needed. Outputs are always used for this purpose because they can be both read and written as required. The SIXNET ISaGRAF run-time in each controller reads output status at the beginning of each logic scan and will therefore know the actual status of the registers transferred from the opposing processor. The I/O Tool Kit configuration software provides a provision for declaring I/O transfers unique to the main (primary) and backup (secondary) processor in an automatically created redundant processor pair. This is important because the configuration of both controllers is by intention, identical in all other respects.

Scan Enable Flags for Control of Outputs

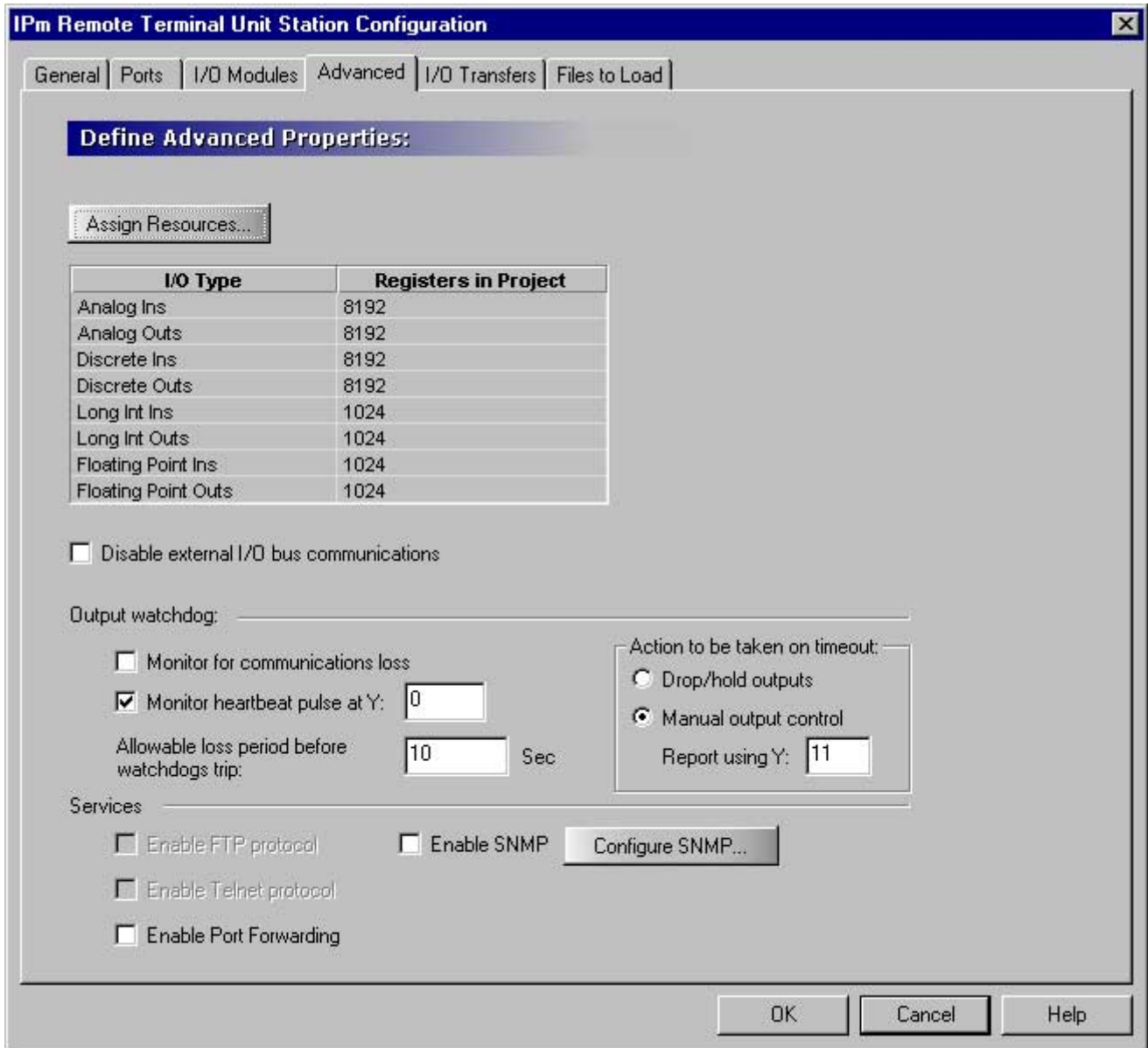
Each I/O Transfer may be enabled or disabled through its own "Scan Enable" flag. These flags are typically virtual discrete output registers and may be controlled by logic of your own design through ISaGRAF programming. This flexibility allows the system designer to base switchover decisions on watchdogs, heartbeats, various status signals or alternative means. SIXNET provides a sample ISaGRAF program for management of the switchover of output control between two redundant processors. This program named: “main_standby” should be placed near the top of the Begin section of the ISaGRAF project that is loaded into both controllers.

Only the processor that is in control should have its outputs enabled to write. When a processor is active, the reading of outputs may be disabled at your option to reduce polling time.

Detecting Processor Failure -- Watchdogs & Heartbeats

Two SIXNET IPm-based controllers are easily setup as primary and secondary stations, with automatic transfer of “output control” from one unit to the other if a controller fails. To monitor health, each controller first generates a “heartbeat pulse” (pulsing physical or virtual discrete output) from within its program. (This can be as simple as a “Blink” function in an ISaGRAF program.) The station then uses an I/O Transfer to pass the heartbeat pulse to the other controller. Each controller uses its own “Heartbeat Loss” watchdog feature to monitor the heartbeat pulse generated by the opposite controller. If the primary unit’s heartbeat stops pulsing, the secondary controller takes over control of the outputs in the system, and vice versa. The heartbeat loss watchdog can be configured to turn on a discrete output register if the heartbeat fails. This “Report using Y” output can be used directly as a Scan Enable for the I/O Transfers that write the distributed outputs in the system.

Note: The Heartbeat Loss feature has a minimum allowable loss period of one second before it sets the “Report using Y” output TRUE. Faster heartbeat monitoring can be achieved by using a timer in an ISaGRAF program, instead of using the Heartbeat Loss feature.



Status Modules and Flags

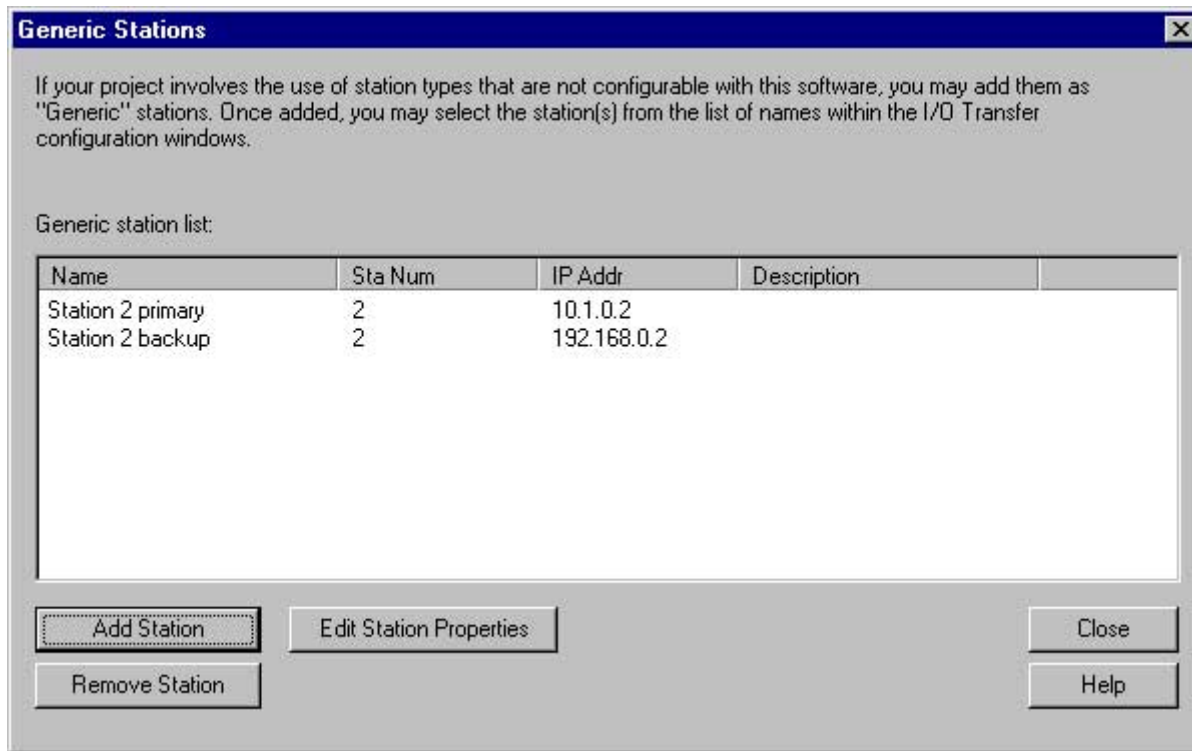
“Station Status” modules can be added to an IPm controller’s configuration. These virtual modules contain discrete inputs that reflect the communication health of the distributed stations or modules in your system. Once I/O Transfer communication to a distributed station or module is successful, it’s corresponding Station Status discrete register will be set TRUE (1) and will remain that way. If subsequent I/O Transfer communication to the distributed station or module fails, the corresponding Station status register will be set FALSE (0). By monitoring these registers in your ISaGRAF program or Windows applications, you can take appropriate action if a distributed station or module goes offline.

Note: Distributed stations or modules will be reported in their last known state if communication is halted with a Scan Enable flag.

Redundant Communication Paths to the Distributed I/O

Two independent communication paths may be used to establish redundant communications. Although dual Ethernet networks offer the highest system performance, RS485 connections are sometimes employed as a low cost backup communications means. Generally, a switchover to the backup path is initiated if the “Station Status” flag of the primary communication path drops off (0).

When configuring Ethernet I/O Transfers to a distributed station, it is typical to enter the station’s number or choose the station name from the dropdown list provided. These two methods of distributed station identification permit a single destination Ethernet IP address to be specified for all Ethernet I/O Transfers to the distributed station. When redundant Ethernet communication paths are used, you must have the ability to configure multiple IP addresses for the same distributed station number. The Generic Stations feature allows you to freely create distributed station definitions, each with a unique IP address but the same station number. You can then choose the generic stations as appropriate in your I/O Transfer configurations. Generic Stations also allow you to have a separate Station Status flag for each communication path to the distributed station.



Follow these steps to create a backup communication path:

1. Define a “Generic Station” that the backup path will communicate with. The Generic Station is essentially a clone of the distributed station, but with a different name. Optionally, you can specify a unique IP address for each generic station. If you are using a SixTRAK IPm, be sure that the IP addresses of the two Ethernet ports are located on different subnets. Ethernet 1 defaults to 10.1.0.1 and Ethernet 2 defaults to 192.168.0.1. Refer to the SIXNET I/O Tool Kit help for more information on Generic Stations and Ethernet addressing.

-
2. Add I/O Transfers to the Generic Station(s), but choose the backup communication path. This can be the second Ethernet port or it can be a serial port. Finally, define Scan Enable bits as appropriate for these backup I/O Transfers.
 3. Assign a Station Status bit for each Generic Station. These bits will be available from the dropdown list of remote stations in the Station Status configuration window of the SIXNET I/O Tool Kit.

Reading Tags into Supervisory Systems

It is easy to read I/O registers or "Tags" from both the primary and secondary (backup) controller into a supervisory SCADA computer. This feature enables the system to read the backup system and verify its readiness. Since the two controllers use identical tag names, a method of distinguishing the tags from each controller is desired. This is accomplished by assigned a station prefix to the tag names of the I/O in each station. The Station Prefix feature may be enabled in the I/O Tool Kit's View menu. Assign a short prefix to each affected station. Be sure to keep the number of characters in the resulting tag name within the maximum tag name limitation of your SCADA system.

Special SIXNET ISaGRAF Features For Redundant Support

PLCs read inputs and write outputs after each logic execution. They assume that no changes in outputs can occur by another processor. They are not intended for redundant systems.

SIXNET ISaGRAF, in addition to updating input status, also reads back actual output states at the beginning of each logic scan, in the event that some other source (most likely a redundant controller) has made a change. It is this single enhancement that is most important in developing redundant processor systems that smoothly switch over between active processors.

SIXNET ISaGRAF also permits the identical program to be loaded into multiple controllers. Identical tag names may be used in both processors. Please note that tag name prefixes can be used to distinguish the tags coming from multiple processors so that both can be referenced from upstream supervisory systems. Error checking in the SIXNET I/O Tool Kit can also be configured to permit duplicated tag names in multiple stations. Refer to the SIXNET I/O Tool Kit help for details.

ISaGRAF Programming Tips

Redundant system designers often speak about "bumpless transfer". They are referring to the desirable situation in which a backup controller can smoothly take over control without "bumping" (disturbing) the process. The key to a smooth transition is for the backup process to know the state of the process and the state of the internal and external variables associated with the process. As guidelines to obtaining this objective, the following suggestions are offered:

1. Continuously copy the current values of relevant variables (registers) from the active to the backup processor. First, be sure to assign relevant variables to external registers. It is most efficient if they are assigned to consecutive registers so that they can be moved in blocks. I/O Transfers can be used to copy these I/O blocks between processors.
2. The actual state of outputs can be read from the distributed I/O stations. It is best to reference these actual outputs instead of internal variables in your program, (whenever practical) to ensure that program decisions are made based upon current status.

-
3. Try to avoid the use of state logic, in which actions of the program are dependent on previous decisions or logic states. In batch programs, it is desirable to base actions on the step number, rather than a "next state". A register containing the step number can be copied between processors to keep the backup program in sync. This advice is not absolute. The state of a process may be known through the sharing of state variables, or logic that enables the program to figure out the state of the process. As an example of acceptable state programming, consider a simple start/stop contactor. The state of the output coil should be preserved when control transfers to the backup processor. The present state can always be determined from reading the start input, the stop input and output coil.
 4. The current value of timers, counters, and set points should be synchronized (as external I/O) between the active and backup processor.
 5. Exercise care in the operation of PID loops and other time or state dependant calculations. A PID block, for example, must not be permitted to "wind up" (excess integral term). To this end, output change rate limits, or a switch over from manual to auto control can be used to accomplish smooth transition.
 6. Review your program and ask the question, "Will this logic or calculation perform correctly if the program becomes active at any time and in any process state?"

A Final Word about Safety

Software based controllers, including SIXNET IPm units, should never be relied upon to protect personnel or to prevent consequential property damage. The use of hardware interlocks and safety switches should always be employed, in keeping with accepted practices and as may be required by applicable codes and regulations. The use of redundant controllers is not a suitable substitute for the appropriate safety practices.