4

Writing the Setup Program

In This Chapter. . . .

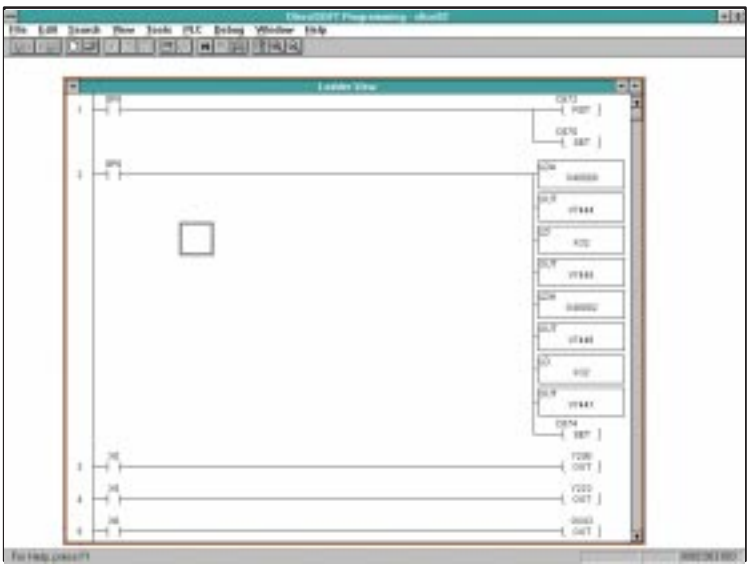
- Choosing a Programming Device
- Writing Your Slice I/O Setup
- Slave Removal
- Rejoining Slaves
- Special Relays Used for Slice I/O
- How to Use the Special Relays

Choosing a Programming Device

You can write your setup logic by using either a handheld programmer or our Windows-based ***Direct***SOFT programming software. It is generally much easier to use the software to generate the necessary setup logic. The examples that follow show the instructions in this format. Connect your computer through the CPU, and not through one of the slave units. Until you have completed the installation *and* the setup logic, you cannot communicate with the CPU via the slave unit communication ports.

To get started, enter ***Direct***SOFT and carry out the normal ***Direct***SOFT setup procedures for communicating with your DL405 CPU. If you do not know how to do this, refer to your ***Direct***SOFT Manual. Chapter 11 of your DL405 User Manual also has a very good explanation of the basic DL405 instruction set and examples of how these instructions are used for writing general ladder logic. In this chapter, we will only show you those instructions that are used to set up your Slice I/O system.

First open ***Direct***SOFT and establish a communication link with your CPU. Then enter the Edit Mode for programming. You should now be looking at a screen similar to the one shown below:



The ***Direct***SOFT window shown above depicts a program that has already been written. Of course, your programming window will be empty when you first open it. The following pages will show you how to write each part of your Slice I/O setup program.

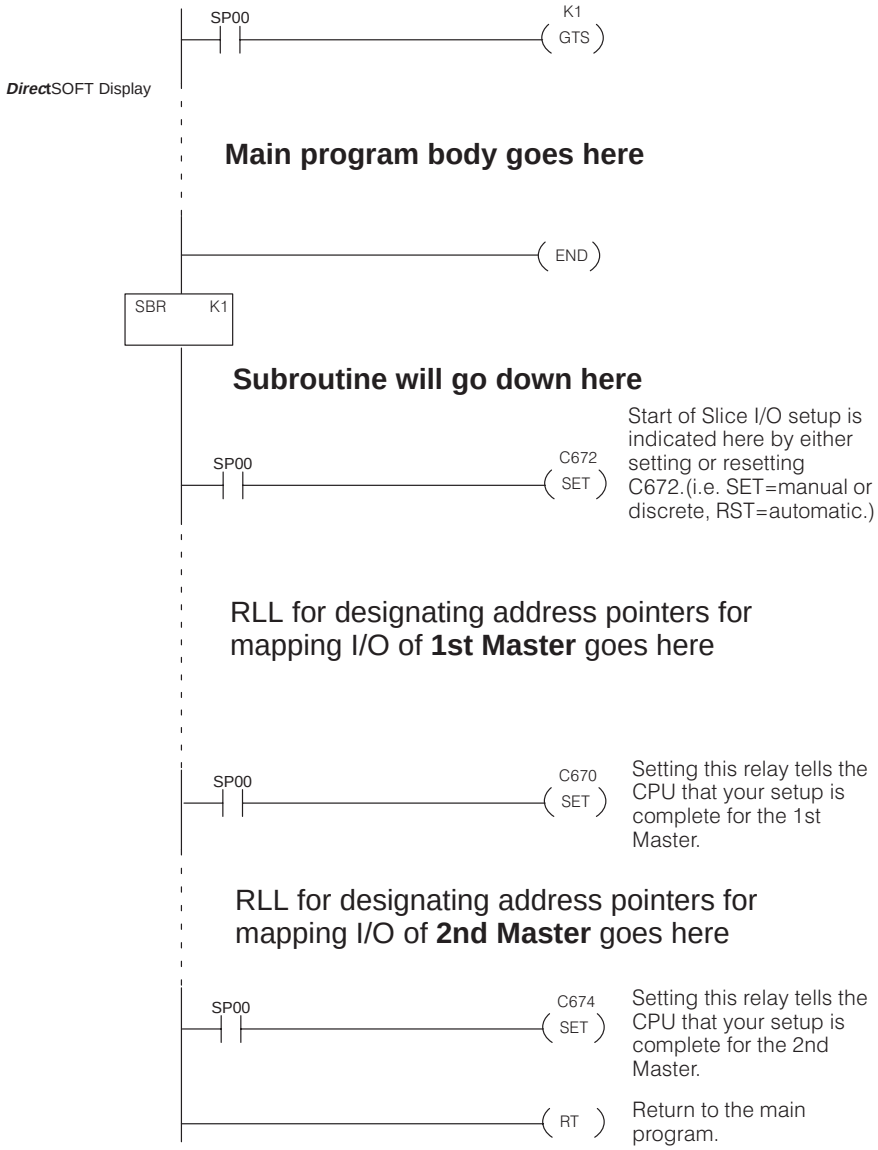
Writing Your Slice I/O Setup

Step 1: Decide How You Are Going to Execute Your Program

Is your setup logic going to be in the main program body or is it going to be in a subroutine? If you have a DL430, the decision is made for you. The DL430 does not support the subroutine instructions, so you have to put the setup logic in the main body of the program. The DL440, on the other hand, does support the subroutine instructions. The reason for using subroutines is because the setup logic only needs to be executed once. In the example below, we have suggested the use of SP00 so that the subroutine is only executed during the first scan. This means it will not impact the scan time on subsequent scans.

When you write your setup logic, it will be sandwiched in between rungs that affect the status of certain internal relays that are assigned to Slice I/O setup. These relays designate the beginning and end of your setup commands.

Sample RLL Structure for Slice I/O Setup



Step 2: Write the Setup Logic for Each Slice Master

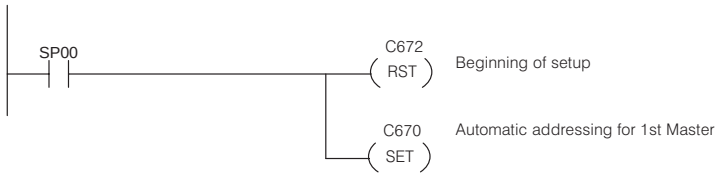
Whether you choose to write the Slice I/O setup program as a subroutine or as a part of the main program, the procedure is still the same. If you are using **automatic addressing** the process is very simple.

NOTE: You cannot use automatic addressing for both masters at the same time. If you want to use automatic addressing, you have to choose only one channel. Also make sure that the X's and Y's that are automatically assigned to the slaves are not used by the other modules in the system. Automatic addressing starts at X200 and Y200.

Automatic Addressing

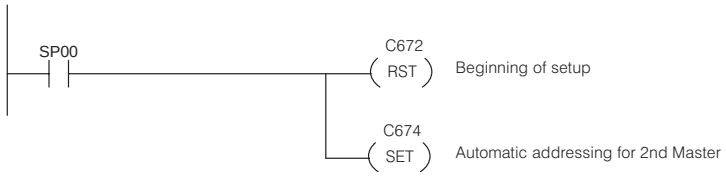
If you are using only one master module, then automatic addressing will probably be the only type of addressing you may ever need. Using *two* masters, however, produces some additional requirements. Automatic addressing can be used with either the 1st Master or the 2nd Master, but it can only be used with one of them in any given system. With automatic addressing, you do not have to assign the individual slave I/O addresses with your setup ladder logic because the CPU automatically assigns the data types (X and Y) and the respective addresses. You do, however, have to make sure that the C672 is set to zero (0) and that either C670 or C674 are set to one (1). If you are using automatic addressing with the 1st Master, then C670 must be set. If you are using automatic addressing with the 2nd Master, then C674 must be set. Switch #4 must be ON in order to use Auto Addressing.

Automatic Addressing Setup for 1st Master



The use of automatic addressing for the 2nd Master is essentially the same, except that you SET C674 instead of C670.

Automatic Addressing Setup for 2nd Master



When the CPU detects one of the above setups in your ladder logic, it will assign slave inputs starting at X200 and slave outputs starting at Y200. It will consume 16 points for the inputs and 16 points for the outputs of each slave, regardless of which type of Slice slave you are using. For example, a D4–SS–106 will consume 16 input points and 16 output points, even though the slave does not have that many I/O points available. **You may have up to 12 slaves for the corresponding master when using automatic addressing.**

How About the Other Types of Addressing?

With manual or discrete addressing, you have some additional steps. In these cases, you have to write ladder logic that tells the CPU which addresses and data types you want to use. The CPU has predefined memory locations, called pointers (V74xx), that you can use to accomplish this task. Simply use the tables in Appendix B to find the V-memory location (V40xxx) that corresponds to the data type and address that you want to use as the starting address. Then, you can use the setup logic shown in the following examples to load these V-memory addresses into the pointers that the CPU uses to determine the Slice I/O point addresses. By doing this, your setup logic merely tells the CPU where to store the slave I/O points in the CPU image register area.

Manual Addressing

With manual addressing, you may use up to 15 slaves per channel. The following example system only uses 3 slaves. We have decided to use global GX data types in this example for our inputs and outputs. If you completed worksheets for your system, simply transfer the worksheet data as shown here. Also, if you examine this setup program, you'll notice that the V40xxx addresses have been properly designated as shown in Appendix B. The table at the bottom of the page is used for finding the CPU's V74xx pointer addresses.

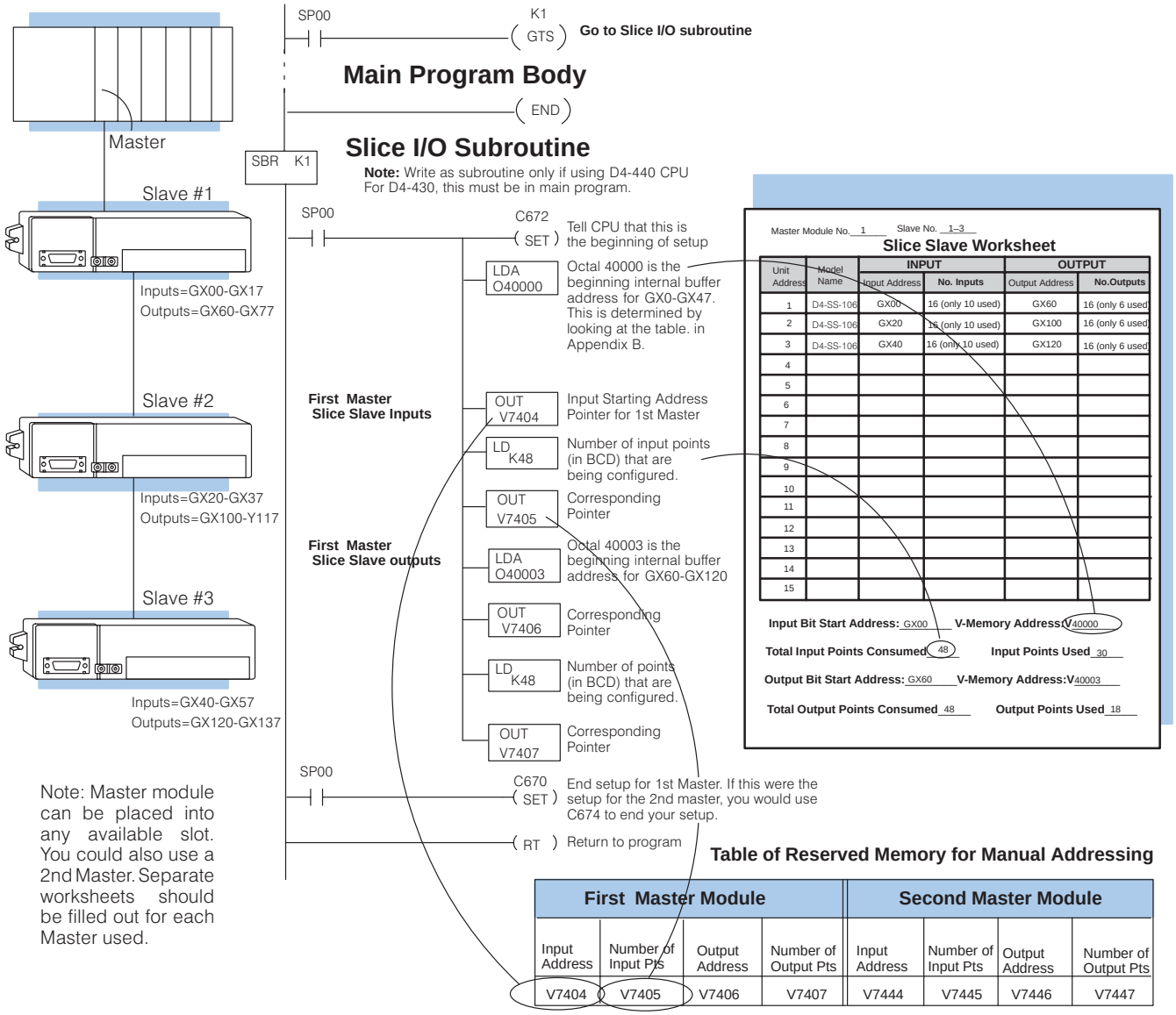


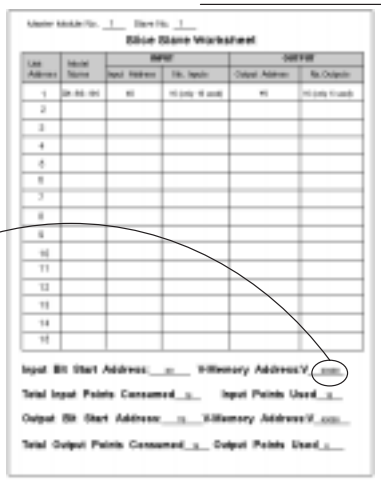
Table of Reserved Memory for Manual Addressing

First Master Module				Second Master Module			
Input Address	Number of Input Pts	Output Address	Number of Output Pts	Input Address	Number of Input Pts	Output Address	Number of Output Pts
V7404	V7405	V7406	V7407	V7444	V7445	V7446	V7447

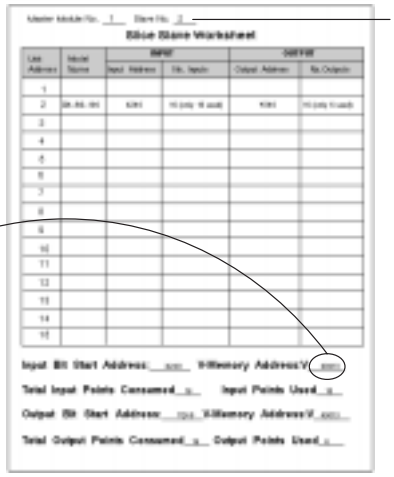
Discrete Addressing

The example shown below takes the same system shown on the previous page and uses *discrete* addressing. Notice that it uses an *expanded* reserved memory table for the CPU pointers and notice that **each slave is setup individually**. Also, the starting addresses can be out of sequence. In the example, we have used X0–X17and Y0–Y17 as the starting addresses for Slave #1 (V40400, V40500) and X240–X257 and Y240–Y257 as the starting addresses for Slave #2 (V40413, V40513). We have not shown Slave #3, but it could use any unused addresses from the X, Y, C, or GX tables, as well as be out of sequence. With this method, It's best to use separate worksheets for each slave. **You may have up to 7 slaves per master when using discrete addressing**.

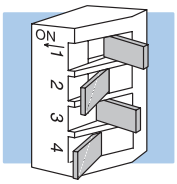
```
SP00 ──┤├──( GTS )  K1   Go to Slice I/O subroutine

Main Program Body

       ──( END )

SBR K1  Slice I/O Subroutine

Note: Write as subroutine only if using D4-440 CPU
For D4-430, this must be in main program.

SP00 ──┤├──( SET )  C672   Tell CPU that this is the beginning of setup

                 LDA O40400   Octal 40400 is the internal buffer address for X0-X17. This is determined by looking at the table. in Appendix B.

First Master Slice Slave #1 Input
                 OUT V7404    Input Starting Address Pointer for 1st Master
                 LD K16       Number of input points (in BCD) that are being configured.
                 OUT V7405    Corresponding Pointer

First Master Slice Slave #1 output
                 LDA O40500   Octal 40500 is the internal buffer address for Y0-Y17.
                 OUT V7406    Corresponding Pointer
                 LD K16       Number of points (in BCD) that are being configured.
                 OUT V7407    Corresponding Pointer

First Master Slice Slave #2 Input
                 LDA O40413   Internal buffer address for X240-X257.
                 OUT V7410    Memory pointer
                 LD K16       Number of input points (in BCD) that are being configured.
                 OUT V7411    Memory pointer

First Master Slice Slave #2 output
                 LDA O40513   Internal buffer address for Y240-Y257.
                 OUT V7412    Memory pointer
                 LD K16       Number of points (in BCD) that are being configured.
                 OUT V7413    Memory pointer

Additional Slaves Continue from Here.
                 Etc. for each slave

SP00 ──┤├──( SET )  C670
End setup for 1st Master. If this were the setup for the 2nd master, you would use C674 to end your setup.

       ──( RT )   Return to program
```



Slave #1



Slave #2



Remember: You must set Pos.4 of the DIP switch to ON in order for discrete addressing to be available.

Note: Additional worksheet would be completed for Slave#3

Table of Reserved Memory for Discrete Addressing

	First Master Module				Second Master Module			
Slave	Input Address	Number of Input Pts	Output Address	Number of Output Pts	Input Address	Number of Input Pts	Output Address	Number of Output Pts
1	V7404	V7405	V7406	V7407	V7444	V7445	V7446	V7447
2	V7410	V7411	V7412	V7413	V7450	V7451	V7452	V7453
3	V7414	V7415	V7416	V7417	V7454	V7455	V7456	V7457
4	V7420	V7421	V7422	V7423	V7460	V7461	V7462	V7463
5	V7424	V7425	V7426	V7427	V7464	V7465	V7466	V7467
6	V7430	V7431	V7432	V7433	V7470	V7471	V7472	V7473
7	V7434	V7435	V7436	V7437	V7474	V7475	V7476	V7477

Slave Removal

Why Would You Use Slave Removal?

There are certain types of applications where you might want slave stations to be temporarily "logged out". Or, there may be some point in the process where you want to permanently remove one or more slaves. You may also want a slave to be disconnected when there is any sort of communications error. Of course, you do not want to disrupt anything else during the removal. This is when you need the slave removal feature.

What is It?

The slave removal feature allows you to remove a slave "on the fly", and even add it back to the system later. This can be triggered specifically in your program or it can occur upon detection of an error in the system. When slave removal is accomplished, the outputs for that slave go to zero (0) and the inputs are no longer read by the CPU.

Types of Slave Removal

You have a choice between two types of slave removal:

- **Manual Slave Removal**-–At any point in your program, you can tell the CPU to ignore the I/O points of a particular slave. There does not have to be an error to trigger this feature.
- **Automatic Slave Removal**—This mode is triggered only by the occurrence of a Slice I/O error for the slave unit designated.

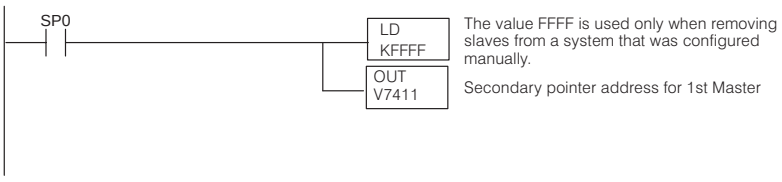
Don't confuse the use of the words "automatic" and "manual" here with our earlier reference for addressing modes. The terms here refer only to *slave removal*. For example, you can *manually* remove a slave from a system that has been *automatically* addressed. You can also automatically remove a slave from a system that has been manually addressed. With the one exception covered in the bottom paragraph, your addressing mode for your slave I/O points has nothing to do with slave removal.

How Pointer Addresses are Used for Slave Removal

The slave removal feature has "primary pointer" and "secondary pointer" setup locations. The *primary pointer address* is a V-memory assignment that is dependent on which type of slave removal is being used (manual or automatic)and the location of the master in the base (which slot). In a moment, we will show you a table of addresses so that you can determine where the primary pointers are located.

The *secondary pointer* address is always V7411 for the 1st Master and V7451 for the 2nd Master. **If you are removing slaves from a configuration that was addressed using manual addressing, the secondary pointer address must have hexadecimal FFFF written to it.** In all other cases, these addresses can have any number written to them **except** FFFF. Below is a sample segment of RLL that shows FFFF being written to the secondary pointer address of the 1st Master for a system that had its I/O points addressed manually.

Sample Logic for Writing to Secondary Pointer



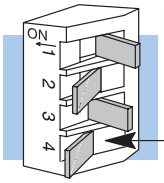
4 Steps for Using Slave Removal

Use the following steps to make use of the slave removal function:

1. Properly set the DIP switch on the rear of the master(s).
2. Determine the binary bit pattern for slave removal.
3. Determine the setup pointer for storing the bit pattern from Step 3.
4. Write the slave removal setup program.

Step 1: Setting the DIP Switch

Slave removal is only possible when you have placed Position 4 of the master module's DIP switch to ON.



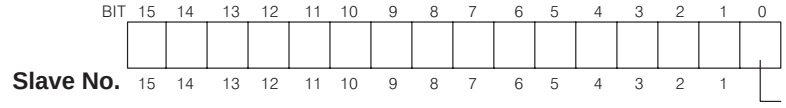
DIP switch of Master Module

Must be placed in the ON position.

Step 2: Determining the Bit Pattern for Slave Removal

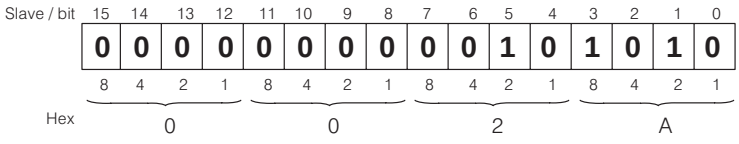
To remove a slave from the system, you set the bits in a 16-bit block according to the scheme shown below. This pattern must be converted to hex for programming.

How the Bits are Set to Designate Which Slaves to Remove



Not used for manual removal
Set this bit to 1 to automatically remove any slave that has a communication problem.

Example for removing Slaves 1, 3, and 5:



Since the Bit number is the same as the Slave number, it is easy to know which bits to set. Once you set these bits, you can convert the binary value to hex. Notice how bits 1 and 3 result in 10, which is hex A.

Hexadecimal 2A

Step 3: Determining the Setup Pointer for Storing the Bit Pattern

The table shown below gives the pointer address for setting up the slave removal. Notice that the addresses vary according to the slot occupied by the master or masters, as well as the type of removal being used.

Slot	V-memory for Manual Removal	V-memory for Automatic Removal
0	V7660	V7670
1	V7661	V7671
2	V7662	V7672
3	V7663	V7673
4	V7664	V7674
5	V7665	V7675
6	V7666	V7676
7	V7667	V7677

Example:

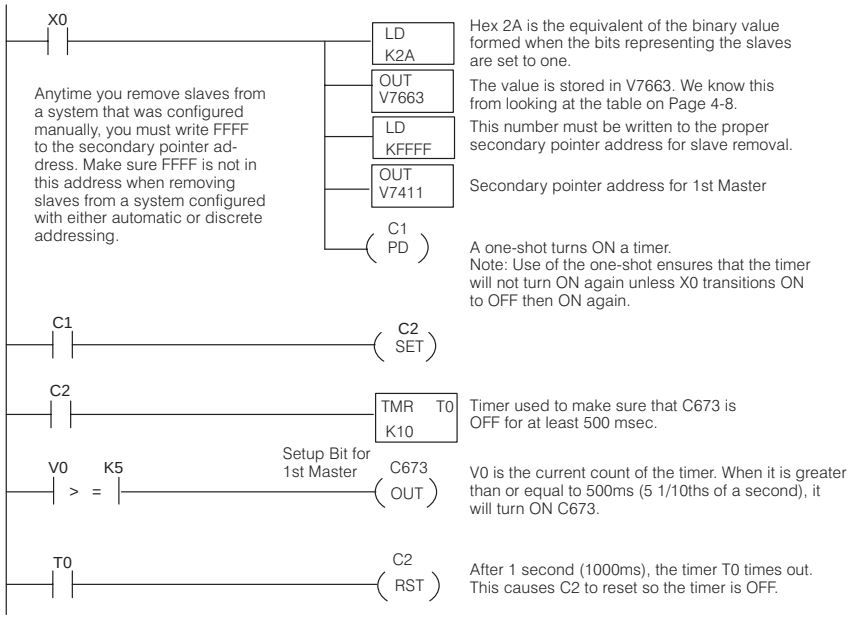
If we are using Manual slave removal and the Master is in Slot 3..

We would store the hex number representing the slave or slaves being removed in V7663.

Step 4: Write the Slave Removal Setup Program

The ladder logic is only slightly different for manual and automatic slave removal. Anytime you are using **manual slave removal**, the last few commands of the setup must transition either C673 or C677 OFF(for at least 500ms) and ON (for at least 500ms). C673 is used for the 1st Master and C677 is used for the 2nd Master. In the example below, we have used a one-shot and a timer to make sure we hold the OFF and ON states for the proper amount of time. We have decided to remove Slaves 1, 3, and 5 for the 1st Master when an ON signal is received from X0. This example configuration, by assumption, had its I/O points configured using manual addressing.

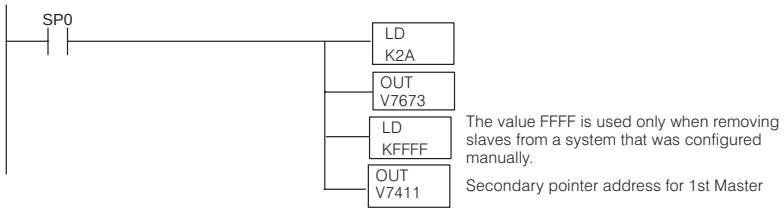
Sample Ladder Logic for Manual Slave Removal



Sample Ladder Logic for Automatic Slave Removal

Using the the same master and slaves of our example, let's take a look at how you would setup the **automatic removal of a slave**. Notice three differences:

- You use SP0 to setup the slave removal on the first scan.
- The V-memory is found on the right-hand side of the table (Page 4-8).
- There is no setup bit (such as C673 or C677) used.



NOTE: Remember, when you determine the bit pattern value for **automatic slave removal**, you have the option of merely setting Bit 0. This would indicate that you want *any* slave to drop out when it causes a communications error. If you do this, then you won't have to set each slave bit individually. In the above example, we only remove slaves 1, 3, and 5. Therefore, we decided not to use Bit 0. We instead set Bits 1, 3, and 5 which resulted in the value HEX 2A.

Rejoining Slaves

What is It?

After removing a slave, usually the application will call for the slave to be brought back on-line with the system.

How is It Done?

In the case of automatic slave removal, the rejoining of the slave or slaves is automatic. That is, as soon as the communications error is cleared, the removed slave or slaves will be brought back on-line. You don't have to write any logic.

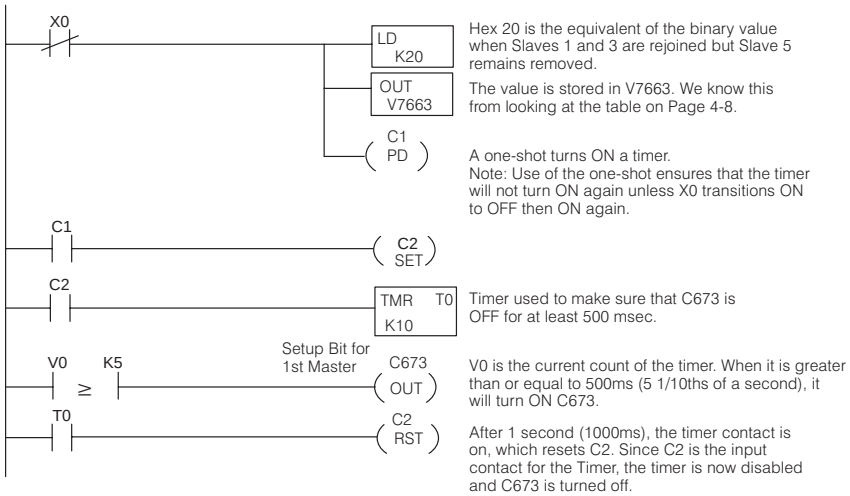
In contrast to this, when slaves have been manually removed from the system, you must write special ladder logic in order to bring them back on-line. There are two steps for doing this:

1. Change the bit pattern in the primary pointer address so that zeros (0) are in every bit position where you want a slave rejoined. Leave 1's in the bit positions where you have slaves removed that you wish to remain removed.
2. Transition the setup bit (C673 or C677) from OFF (at least 500ms) to ON (at least 500ms).

NOTE: The rejoining process causes the CPU to look at the bit pattern in the primary pointer address and REJOIN any slave that has a corresponding bit that is 0, and REMOVE any slave that has a corresponding bit that is set to 1. For example, if you write a zero to bit 3 in order to rejoin Slave 3, but you have bits 6 and 7 with ones stored at the time you transition the setup bit (C673 or C677); then, Slave 3 will be rejoined but Slaves 6 and 7 will be removed. If you don't want any slaves removed when you rejoin one or more slaves, then make sure that all 0's are written to the primary pointer address.

Example of Rejoining a Slave

Here's an example of rejoining Slaves 1 and 3 to a Slice I/O configuration where Slaves 1, 3, and 5 were previously removed. This means the bit pattern would be hex 20 because Bit 5 would still be a 1 and all the other bits would be 0's.



Special Relays Used for Slice I/O

The Slice I/O system has several relays that are used with your system. Some of these relays can be used in RLL routines that will detect and solve errors as a troubleshooting tool. In some cases (i.e. C700, C720, C710,C730), you can use ***Direct***SOFT to look in corresponding V-memory addresses for more information on the error. The following table lists all of the special relays assigned for Slice I/O.

Function of Relay	First Master Relay (s)	Second Master Relay (s)	Description
End of Setup	C670	C674	When set, these relays signify the end of the setup for all addressing modes.
Clear I/O on Error (Automatic Slave Removal)	C671	C675	These two relays are for determining whether you want the remote input points to be set to zero when an error occurs, or whether you want to freeze the current input status. If the relay is set, all the input points are cleared when an error occurs.
Beginning of Setup	C672	C672	When used in your ladder logic, this relay indicates that you are beginning your setup of the addressing for a Slice I/O system. If this relay is set to 1, the CPU knows to use manual or discrete addressing. If it is reset to 0, the CPU knows to use automatic addressing.
Activate Removal or Rejoining of Slaves	C673	C677	When transitioned from OFF to ON these relays will either remove or rejoin slaves depending on what is stored in the primary pointer address.
Communication Error	C700	C720	Automatically set by the CPU when there has been a communication error. Check the individual bits at V7700 to find out if the 1st Master or any of its slaves are responsible. Check the bits at V7701 to find out if the 2nd Master or any of its slaves are responsible. A 1 in bit 0 of either V-memory location means the master has been setup wrong (i.e. baud rate does not match its slaves). A 1 in any of the other bits indicates that there is either no response from the corresponding slave or the slave has failed a data test.
Mapping O.K.	C710	C730	Check the individual bits at V7702 to find out which slaves of the 1st Master have been mapped properly. Check V7703 for the mapping of the 2nd Master's slaves. If correct, there is a 1 in each bit position where there is an active slave.

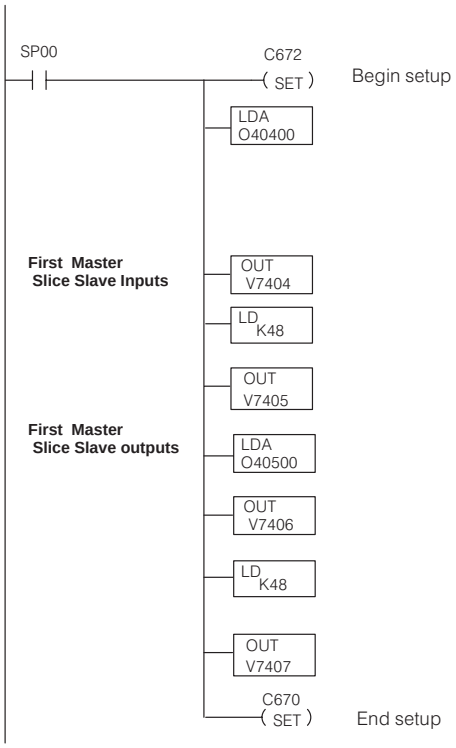
How to Use the Special Relays

C672/C670/C674

Here are some example uses of these relays and an added explanation for each of the relays discussed on the previous page:

These are setup flags for **marking the beginning and end** of your ladder logic that sets up your Slice I/O configuration. C672 marks beginning of all addressing logic. C670 is for ending setup for the 1st Master and C674 for the 2nd.

Example: Begin/End Setup for Manual Addressing of 1st Master



C671/C675 I/O Status On Error

C671 is assigned to the 1st Master. C675 is assigned to the 2nd Master. When any master can't talk to one or more of its slaves, the "link" LED will come on to indicate that there is a problem. The system will stop updating the remote I/O status in the CPU for that slave unit. You have several options at that point. One such option is either to **freeze the last known input status** that is in the CPU's memory image area, or to **write a zero to each point**. If these flags are OFF when the error occurs, all inputs will be zeroed.

Example:

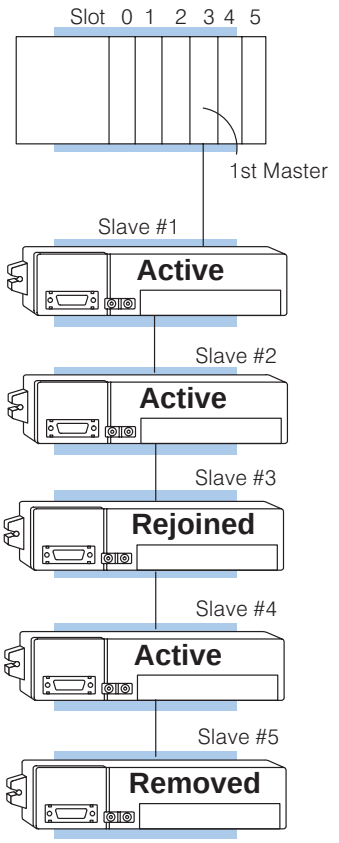
After power up, anytime a remote I/O error occurs for the 2nd Master, the input status will be frozen for the slave that has caused the error.

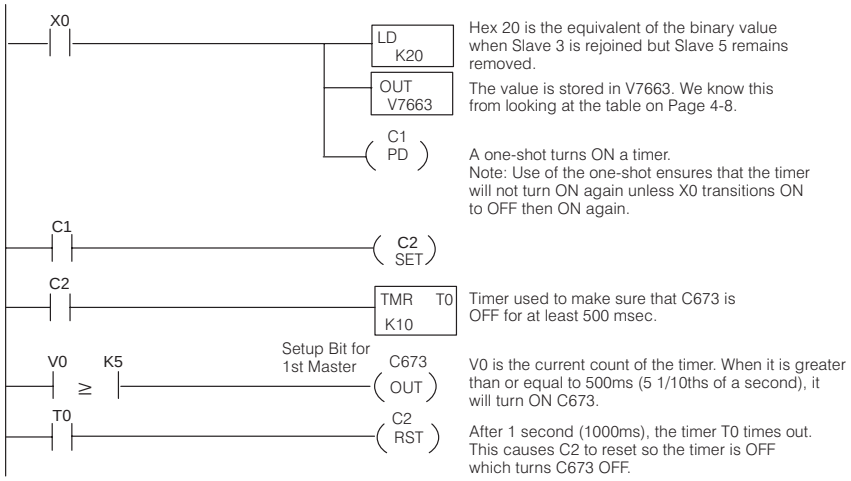
C673/C677 Activate Removal or Rejoining of Slaves

C673 is assigned to the 1st Master, and C677 to the 2nd. These relays have to be transitioned from OFF to ON in order to activate a setup written for removal and rejoining of slaves. They must be OFF for at least 500ms and ON for at least 500ms in order for the transition to be effective. In the example below, we are rejoining Slave 3 but Slave 5 remains removed. In this example, we are showing the 1st Master in slot 3 and I/O assignments had been made previously using manual addressing (ladder logic not shown here).

Example:

The diagram below shows the status after program execution.





V7663 (Status before the above is executed)

BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0
Slave No.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	

V7663 (Status after the above is executed by transitioning C673)

BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Slave No.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	

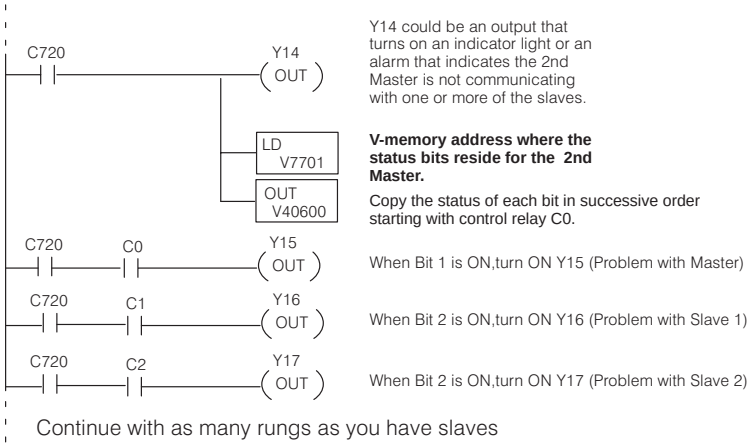
Note: Zero's in any of the bit positions mean that you want a slave to remain active if it is active or you want the slave rejoined if inactive. One's in any of the bit positions means that you want a slave to be removed if it is active or you want a slave to remain removed if already removed.

C700/C720 Locate Communications Error

These relays will be set when there is a **communications error** between the respective master and a slave or slaves assigned to the relay number. C700 is for the 1st Master and C720 is for the 2nd Master. In addition to these control relays, there are also V-memory locations that can be used to help pinpoint the error. V7700 is assigned to the 1st Master and V7701 is assigned to the 2nd Master.To specifically identify whether the problem is with the master or with one of its slaves, you can have your logic check specific bits in the corresponding V-memory.

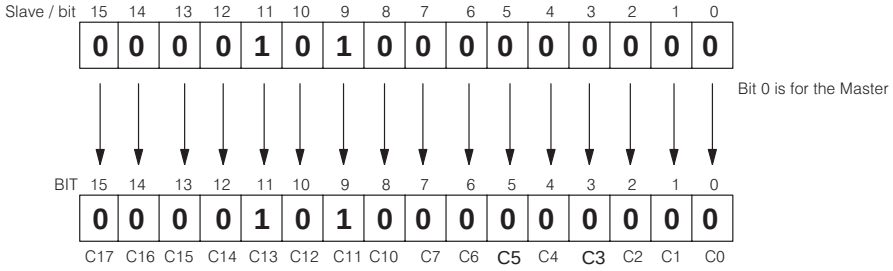
One easy way to do this is to load the contents of the V-memory location into the accumulator and then copy it to one of the V-memory locations that is assigned to control relays that are available for general use. Then, you can use these individual control relays inside of your ladder logic program to help pinpoint the error. In the following example, we used the charts in Appendix B to determine the V-memory address for C0-C17 (V40600). We loaded V7701, which is the communication error location for the 2nd Master, and then copied it to V40600.

Example:



Bit 0 is used to indicate a problem with the master, so the first control relay that contains slave information is C1. Also, notice how the control relays do not match up with the slave number after bit 7. This is because the control relays are numbered in octal, not decimal. For example, you'll notice that slave 9 is represented by C11.

V7701 – 2nd master with communication errors at Slaves 11 and 9



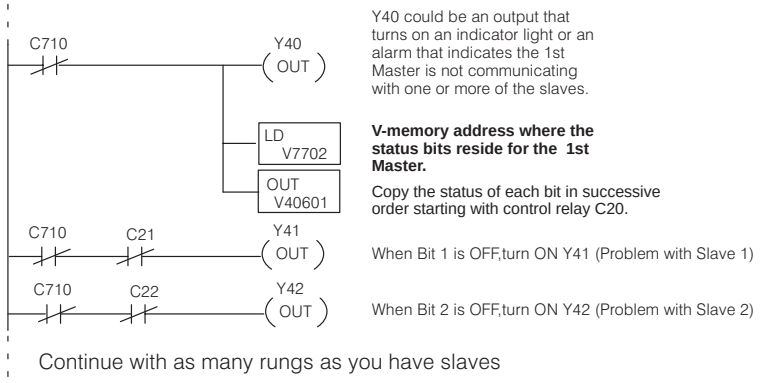
V40600 – Control Relays C0–C17

C710 and C730 Mapping O.K.

C710 is assigned to the 1st Master. C730 is assigned to the 2nd Master. If set, these flags indicate that the I/O points have been properly mapped. If they are off, then it indicates that a setup problem exists. In addition to these control relays, there are also V-memory locations that can be used to help pinpoint the error. V7702 is assigned to the 1st Master and V7703 is assigned to the 2nd Master. To specifically identify the location of the setup error, you can have your logic check specific bits in the corresponding V-memory.

One easy way to do this is to load the contents of the V-memory location into the accumulator and then copy it to one of the V-memory locations that is assigned to control relays that are available for general use. Then, you can use these individual control relays inside of your ladder logic program to help pinpoint the error. In the following example, we used the charts in Appendix B to determine the V-memory address for C20-C37 (V40601). We loaded V7702, which is the communication error location for the 1st Master, and then copied it to V40601.

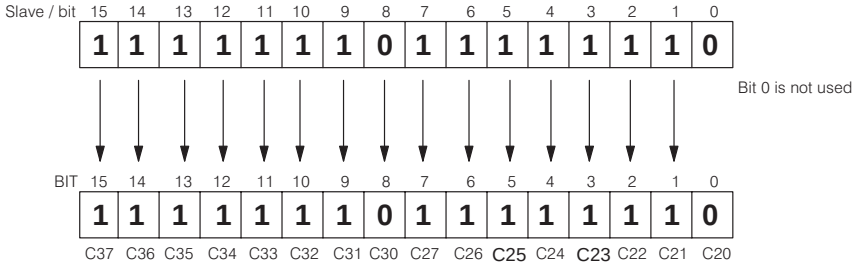
Example:



Note: C20 is not used here because the first bit does not mean anything for the mapping check.

Since bit 0 is not used, the first control relay that contains slave information is C21. Also, notice how the control relays relate to the slave number. You should remember that control relays are numbered in octal, not decimal. For example, you'll notice that slave 8 is represented by C30 in this example.

V7702 – 1st master showing that everything is O.K. except Slave 8 has not been mapped properly. (Remember, the bit is off when a problem exists.)



V40601 – Control Relays C20–C37

The only addressing mode that allows mapping of each individual slave is discrete addressing. This is how individual slaves can be mapped improperly and result in the error bit status shown above.