MaxPak® III High Speed Link Module

M/N 57C424

Industrial

CONTROLS

Instruction Manual J2-3010

September, 1991

RELIANCE ELECTRIC

The information in this user's manual is subject to change without notice. Reliance Electric and its affiliates assumes no responsibility for errors that may appear in this user's manual.

DANGER

ONLY QUALIFIED ELECTRICAL PERSONNEL FAMILIAR WITH THE CONSTRUCTION AND OPERATION OF THIS EQUIPMENT AND THE HAZARDS INVOLVED SHOULD INSTALL, ADJUST, OPERATE, AND/OR SERVICE THIS EQUIPMENT. READ AND UNDERSTAND THIS INSTRUCTION MANUAL AND APPROPRIATE MAXPAK III AND AUTOMAX DCS MANUALS IN THEIR ENTIRETY BEFORE PROCEEDING. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN SEVERE BODILY INJURY OR LOSS OF LIFE.

WARNING

INSERTING OR REMOVING THIS MODULE OR ITS CONNECTING CABLES MAY RESULT IN UNEXPECTED MACHINE MOVEMENT. TURN OFF POWER BEFORE INSERTING OR REMOVING THE MODULE OR ITS CONNECTING CABLES. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

Table of Contents

Appendices

List of Figures

List of Tables

1.0 INTRODUCTION

The products described in this instruction manual are manufactured and distributed by Reliance® Electric Industrial Company.

Note: This instruction manual assumes the following regarding user knowledge:

1. The user has complete familiarity with the MaxPak III drive, and has read the applicable documentation provided with the drive.
2. The user has complete familiarity with the AutoMax DCS system, and has read the applicable documentation provided with the system.

The High Speed Link (HSL) module is a point-to-point real time control Input/Output (I/O) channel (or communications link) that provides communication between the MaxPak III Digital D-C Drive (MaxPak III) and the AutoMax Distributed Control System (DCS) Industrial Controller. See Figure 1-1.

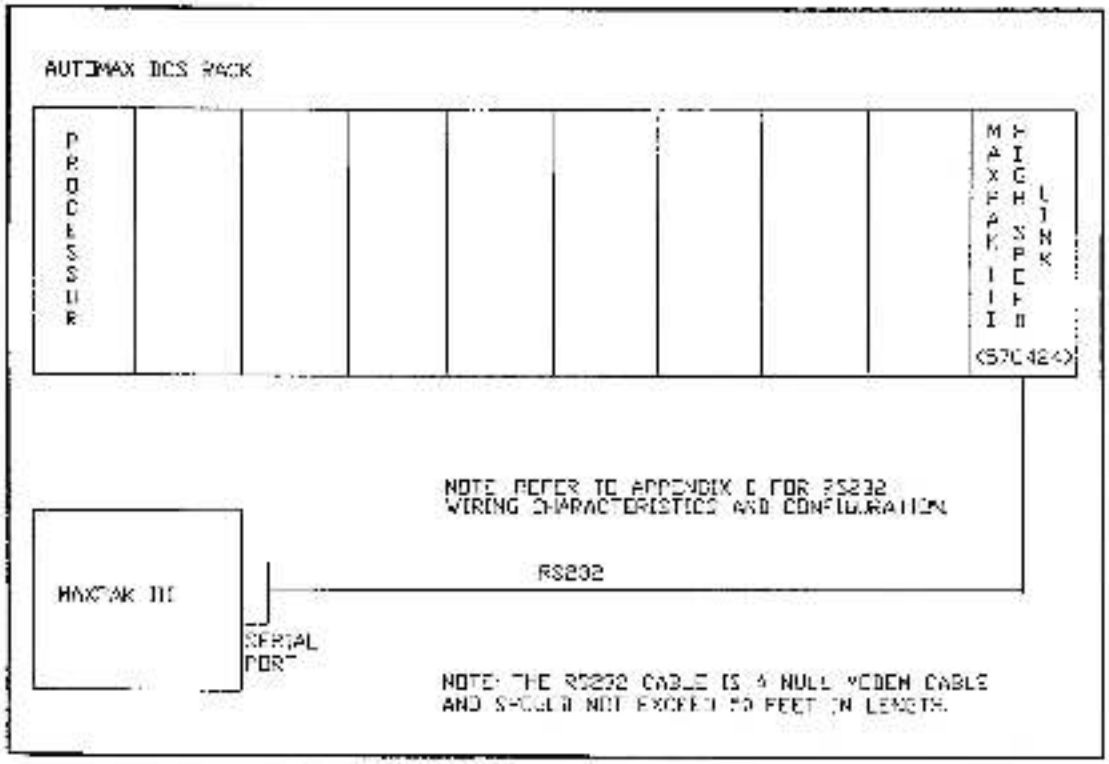


Figure 1-1 – MaxPak III High Speed Link System

The HSL module can be used in applications where the AutoMax system closes an outer regulation loop and then passes information to an inner loop on the MaxPak III drive (see section 4.5 of this manual for program examples). Hence, the MaxPak III High Speed Link system provides a standard hardware configuration for a variety of applications.

HSL Module

The HSL module is designed to fit into the AutoMax Multibus rack. Its dual port memory (see Section 2.2.1) passes status, sequencing, and regulation information along the link via an RS-232 medium at 41.6 Kbaud.

Status, sequencing, and regulation information are passed along the link from the HSL module's dual port memory map (see section 4.2 of this manual) using a customized message protocol. This is a master-slave communication with the HSL module being the master and the MaxPak III drive being the slave except for on-line update messages. The HSL module initiates the transmit messages for the three basic commands for a message to be sent to the MaxPak III drive. These three basic commands are:

- Configuration request
- Change mode request: go to on-line or go to off-line
- Update request: off-line update or on-line update

These commands are to be issued by the user or the configuration program writing the appropriate value to command register (Register 30). (See section 2.2.1.2 of this manual for further details.)

The configuration registers (40 thru 45, and 70 thru 81) are to be written by the user or the application program prior to issuing the configuration request to Register 30. These registers determine the size of the messages to/from the MaxPak III, the update message scan time and the update message timeout. (See section 4.4 of this manual for the minimum HSL module and MaxPak III drive configuration.)

MaxPak III

Variables to be sent to/received from the MaxPak III drive are assigned dual port addresses on the HSL module and are assigned using the IODEF statement (Reference Instruction Manual J-3675 for structuring IODEF statements) in an AutoMax Configuration Task. Moreover, the variables to be used in the AutoMax Configuration Task must exist in the MaxPak III drive's configuration. (Reference section 4.5 of this manual for program examples.)

There are a total of 512 integer registers (HSL_OUT_INT_* and HSL_IN_INT_*); 256 for each type that can be communicated between the AutoMax controller and the MaxPak III drive. Simarly, there are a total of 512 boolean registers (HSL_OUT_BOOL_* and HSL_IN_BOOL_*); 256 for each type. However, the number and type of variables are completely configurable by the user via the HSL module configuration registers. Note that the constraints on the number of variables are limited according to the number of variables in the MaxPak III drive's configuration.

The MaxPak III drive transmits acknowledge messages to the HSL module for the three basic commands described previously. The acknowledge message formats correspond to the three basic commands.

1.1 Additional Information

You must become familiar with the instruction manuals which describe your system configuration. This may include, but is not limited to, the following:

- J-3649 DCS 5000/AutoMax Configuration Task Instruction Manual
- J-3630 AutoMax Programming Executive Instruction Manual
- J-3675 AutoMax Enhanced Basic Language Instruction Manual
- J-3676 AutoMax Control Block Language Instruction Manual
- J-3677 AutoMax Ladder Logic Language Instruction Manual
- D2-3203 MaxPak III Drive Version 6 Instruction Manual

1.2 Related Hardware and Software

- MaxPak III Digital D-C Drive
- MaxPak III Digital D-C Drive Version 6.1A or Higher Operating Software
- RS-232 Serial Cable
- HSL Module (M/N 57C424)
- AutoMax DCS

The HSL Module must use the MaxPak III Digital D-C Drive Version 6.1A or higher software for proper operation. If the drive was shipped from the factory without a custom ordered or Engineering Sales ordered configuration, the Version 6.1A software should contain a default HSL assignment register map in the MaxPak III drive's configuration file (see Appendix A).

It is recommended to upload the configuration file and save to disk for reference prior to making changes to the existing configuration file. If overwritten, the existing configuration file will be lost.

1.3 Conventions

- Bit 15 is the most significant bit in a word. This applies to both the HSL Module and the MaxPak III drive references to bit positions in this document.
- While the MaxPak III drive is Off-line (SYS_ONLINE@ = OFF), the user has access to all drive input variables with access levels of 0 through 5 which are defined in the HSL map. Output variables are always accessible.
- While the MaxPak III drive is On-line (SYS_ONLINE@ = ON), the user has access to drive input variables with access levels of 0 through 2 which are defined in the HSL map.
- One (1) tick is equivalent to 0.5 ms for the AutoMax DCS message scan period. Note that this product employs variable tick rates at 0.5 ms/tick.

The remainder of this manual describes the functions and specifications of the module. It also includes a detailed overview of installation and troubleshooting procedures, as well as examples of configuration.

2.0 MECHANICAL/FUNCTIONAL DESCRIPTION

The following sections describe the mechanical and functional characteristics of the HSL module.

2.1 Mechanical Description

The HSL module is an electronic module (printed circuit assembly) that plugs into the AutoMax DCS rack. This printed circuit assembly has an RS-232 port for communicating with the MaxPak III Digital D-C Drive.

The faceplate of the module contains a 7-segment LED, a green LED, two thumbwheel switches, and one electrical connector. See Figure 2-1.

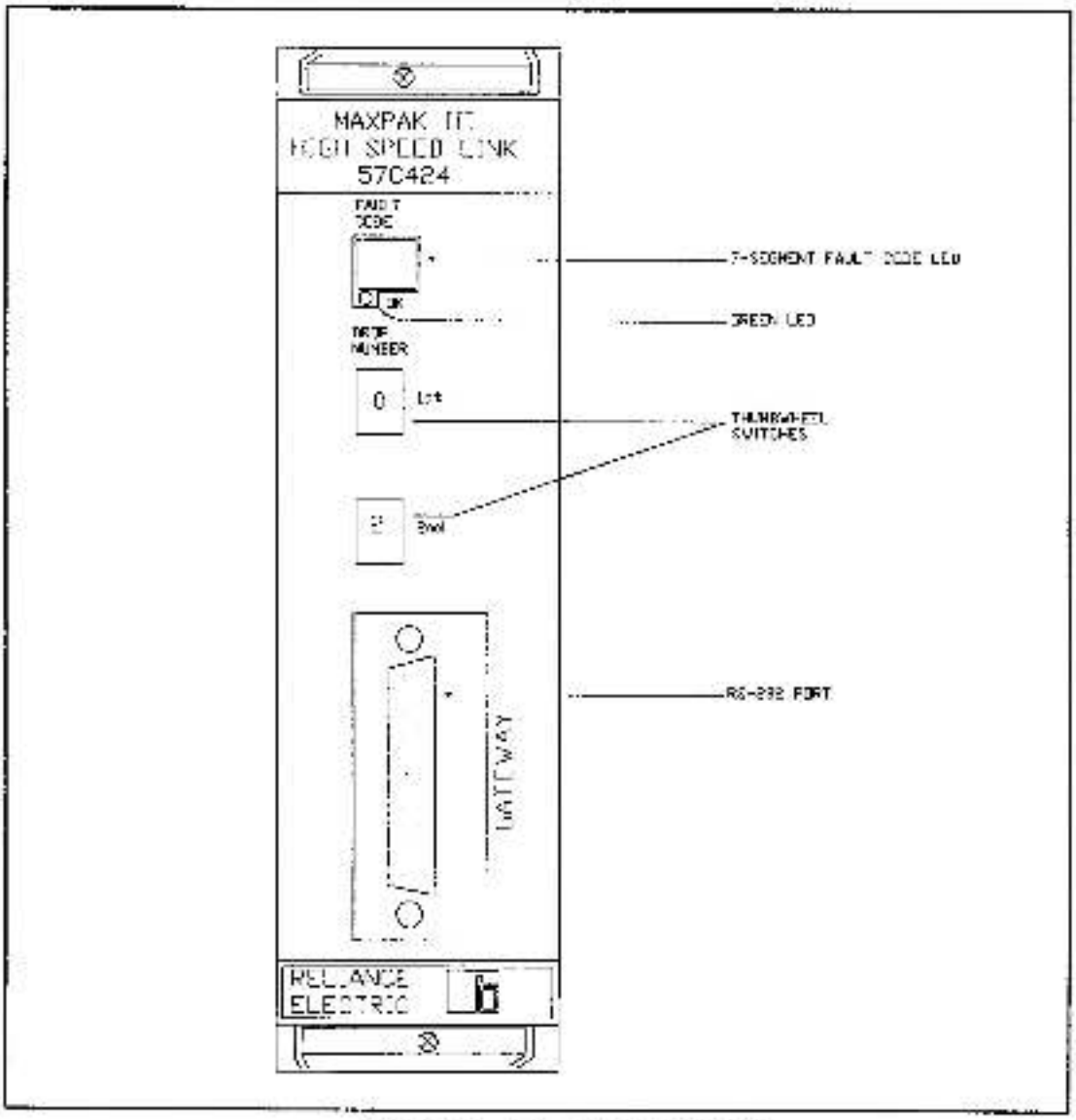


Figure 2-1 – HSL Module Faceplate

The 7-segment LED is labeled "FAULT CODE". This 7-segment LED is used to display card status. Refer to Appendix B for the 7-segment LED Display Codes.

A green LED, labeled "OK", is used to indicate whether the HSL module is functioning properly.

Two thumbwheel switches (labeled "1ST" and "2ND") are present but are used only for factory burn-in testing.

One electrical connector labeled "GATEWAY" is used for transmitting and receiving data to/from a MaxPak III Digital D-C Drive.

This module is also equipped with card ejectors to allow easy removal from the AutoMax DCS rack. Hold down screws are provided to secure the module in the rack.

The printed circuit board assembly has a rugged plastic housing to protect both the component and foil sides. The rear edge connector is shrouded by an enclosure to further protect the card from handling damage.

2.2 Functional Description

The remainder of this chapter covers the following topics:

- 57C424 HSL Card Dual Port Register Map
- MaxPak III High Speed Link (HSL) Variables
- Sequence of Events
- Messages
- Timeout
- Tach Loss and Overspeed
- Data Integrity

2.2.1 57C424 HSL Card Dual Port Register Map

The 57C424 HSL Card Dual Port Register Map consists of Control Registers, (Configuration Registers and Communication/Status Registers) and Data Registers (Receive and Transmit). The following outlines the Read/Write (R/W) function of the registers. An explanation of each register follows in the succeeding paragraphs.

Note: Not all the registers shown in the dual port register map are contiguous (Registers not specified in this map are not accessible. Any access attempts will result in a bus error.)

Serial Link Statistics

REGISTER 12:	Device Number	(R)
REGISTER 13:	Keyswitch State	(R)
REGISTER 14:	Messages Received	(R)
REGISTER 15:	Receive Timeouts	(R)
REGISTER 16:	Checksum Errors	(R)
REGISTER 17:	Overrun Errors	(R)
REGISTER 18:	Framing Errors	(R)
REGISTER 19:	Messages Transmitted	(R)
REGISTER 20:	Parity Errors	(R)

Link Status

REGISTER 4:	Link Active	(R)
REGISTER 26:	Communications Error Flags	(R)
REGISTER 27:	Configuration Error Flags	(R)
REGISTER 28:	MaxPak III Status Byte	(R)
REGISTER 29:	Transmit Active	(R)

Commands

REGISTER 0:	Status & Control Register 0	(R/W)
REGISTER 30:	COMMAND/STATUS CHANGE	(R/W)
REGISTER 31:	Communication ERROR reset	(R/W)

MaxPak III Variable Map Data

REGISTER 40:	Total input On-Line integer Registers	(R)
REGISTER 41:	Total input Off-Line integer Registers	(R)
REGISTER 42:	Total input On-Line boolean bits	(R)
REGISTER 43:	Total input Off-Line boolean bits	(R)
REGISTER 44:	Total output integer Registers	(R)
REGISTER 45:	Total output boolean bits	(R)

Link Device Data

REGISTER 60:	Fatal Error #	(R)
REGISTER 62:	HSL Comm Card ASCII ID 'HS'	(R)
REGISTER 63:	HSL Comm Card ASCII ID 'L'	(R)
REGISTER 64:	MaxPak III Version Number	(R)

Data to MaxPak III Configuration

REGISTER 70:	Number of integer registers to transmit to the MaxPak III	(R/W)
REGISTER 71:	Number of boolean bits to transmit to the MaxPak III	(R/W)

Data from MaxPak III Configuration

REGISTER 75: Number of integer registers to receive from MaxPak III ..... (R/W)
REGISTER 76: Number of boolean bits to receive from MaxPak III ..... (R/W)

Link configuration

REGISTER 80: Maximum receive TIMEOUT (msec) ..... (R/W)
REGISTER 81: Speed loop time period (ticks: 1 = 0.5msec) ..... (R/W)

Input integer registers (from MaxPak III drive)

REGISTER 100: Data Receive Integer Register 0 ..... (R)
REGISTER 101: Data Receive Integer Register 1 ..... (R)
REGISTER 102: Data Receive Integer Register 2 ..... (R)
•
•
REGISTER 354: Data Receive Integer Register 254 ..... (R)
REGISTER 355: Data Receive Integer Register 255 ..... (R)

Input booleans (from MaxPak III drive)

REGISTER 400: Data Receive Packed boolean register #0 ..... (R)
REGISTER 401: Data Receive Packed boolean register #1 ..... (R)
•
•
REGISTER 415: Data Receive Packed boolean register #15 ..... (R)

Output registers (to MaxPak III drive)

REGISTER 1100: Data Transmit Integer Register 0 ..... (R/W)
REGISTER 1101: Data Transmit Integer Register 1 ..... (R/W)
REGISTER 1102: Data Transmit Integer Register 2 ..... (R/W)
•
•
REGISTER 1354: Data Transmit Integer Register 254 ..... (R/W)
REGISTER 1355: Data Transmit Integer Register 255 ..... (R/W)

Output booleans (to MaxPak III drive)

REGISTER 1400: Data Transmit Packed boolean register #0 ..... (R/W)
REGISTER 1401: Data Transmit Packed boolean register #1 ..... (R/W)
•
•
REGISTER 1415: Data Transmit Packed boolean register #15 ..... (R/W)

2.2.1.1 Data Registers

The data registers are divided into two parts. Receive Data Registers - from the MaxPak III drive to the HSL module and Transmit Data Registers – from the HSL module to the MaxPak III drive. The registers from 100 to 1415 are assigned for the data registers.

2.2.1.1.1 Receive Data Registers

REGISTERS 100–101 – These are the "high priority" integer data receive registers 0-1. The HSL module will generate a Multibus interrupt (if so enabled) immediately upon receipt and validation of these registers. These registers are guaranteed to be sent by the MaxPak III drive every L2_SCAN_PERIOD%. They are the first registers transmitted in each message and the first registers received. The rest of the received message will still be in transit when these registers are updated (see sections 2.2.4.1.2, and 4.1 for details). Immediately following the update, the HSL module asserts the multibus interrupt, if enabled. (See APPENDIX C for details on Multibus interrupt enable and disable.) Then the application would normally read Registers 100 and 101. In this way, the application may be synchronized to the MaxPak III drive.

REGISTERS 102 to 355 – These are the remaining data registers that may be used to receive miscellaneous integer data from the MaxPak III drive (receive registers 2 through 255). Registers 102 to 355 are not guaranteed to be received every L2_SCAN_PERIOD%. Depending on the number of registers being sent, it may take several scans to send all the data.

REGISTER 400 - Data Receive Boolean bits, register #0. This register is guaranteed to be sent every L2_SCAN_PERIOD%, but is not high priority (i.e. it does not generate a Multibus interrupt). Thus, it is normal receive data that must be verified separately and its value may take longer to appear in this register after being received than the high priority data. If less than 16 bits are to be sent to the HSL module (REGISTER 76), only those bits requested will be updated.

REGISTERS 401 to 415 – These Boolean bit registers are for miscellaneous boolean data from the MaxPak III drive. They are not guaranteed to be received every L2_SCAN_PERIOD%. Depending on the number of registers being sent, it may take several L2_SCAN_PERIOD% to send all the data. If the number of boolean bits to be received by the HSL module is $\leq$ 16 (REGISTER 76), these registers are not updated.

2.2.1.1.2 Transmit Data Registers

REGISTER 1100 – High priority transmit register. The contents of this register are written to the associated MaxPak III variable immediately upon receipt and validation, without waiting for receipt of the remainder of the update message. The AutoMax DCS Processor will typically write the current minor loop reference (or some other high priority datum) to be passed to the MaxPak III drive in this register.

REGISTERS 1101 to 1355 – These are the remaining data registers that may be used to transmit miscellaneous data to MaxPak III HSL integer input registers 1 through 255. The time to send all these registers will vary depending on the total number of registers to be sent and the message scan period (ticks) (see Register 81 in Section 2.2.1.2.2). Several registers are sent each message.

REGISTER 1400 – The data in this register is defined to be booleans, or bit flags, to be set/reset by the application as it wishes. These 16 boolean bits will be sent to the MaxPak III every time the AutoMax DCS application program issues a transmit request. Examples of use include SEQ_DRIVE_EN@, SEQ_JOG@, SEQ_RUN@, SEQ_STOP_ILIM@, etc.

REGISTER 1401 to 1415 – These are the remaining registers that will be used to transmit miscellaneous boolean data to the MaxPak III drive. If the number of boolean bits to send is $\leq$ 16 (REGISTER 71), these registers will not be used. The time to send these registers will vary depending on the total number of registers (16 booleans/register) to be sent and the # of ticks in the speed loop task (REGISTER 81).

2.2.1.2 Control Registers

The Control Registers are generally divided into two parts. One is the communication/status related control registers (Registers 0 thru 31). The other is the configuration related registers (Registers 40 thru 45 and 70 thru 81).

2.2.1.2.1 Communication/Status Registers

REGISTER 0 – (Interrupt Status & Control REGISTER 0). This register is used to support AutoMax DCS hardware for Multibus interrupt.

This register is specified in an IODEF statement (see the Reliance AutoMax Basic Programming Reference, I/M #J-3600-1, for a description of the IODEF statement). The name assigned in the IODEF is used in defining a hardware event on the AutoMax DCS processor card. This feature is used to synchronize the AutoMax DCS processor card with feedback being sent by the MaxPak III drive (i.e. Speed Feedback). See APPENDIX C for REGISTER 0 layout.

(Note: An AutoMax DCS 57C430A processor (not DCS 5000) or later processor must be used if interrupt capability is desired.)

REGISTER 4 – (Link active) Bit 0 is set (1) if the HSL module is receiving messages and a valid configuration has been acknowledged by the MaxPak III. Bit 0 is reset (0) if the HSL module is not receiving messages. Also, a 'C' will appear on the 7 segment led if the HSL module is NOT receiving messages.

REGISTER 13 – (Keyswitch state) Value representing the state of the keyswitch on the AutoMax processor board. 0, > 4: CARD IN BURN-IN OVEN, 1:NORMAL (MEMORY PROTECT), 2: SETUP, 3: PROGRAM. Program mode is monitored constantly and will reflect a keyswitch change anytime.

REGS 14–20 – Counters for communication statistics. These values are updated when appropriate by the HSL module.

REGISTER 26 - This register latches the five types of communication errors that can occur. Note: The five errors defined for Register 26 are first fault errors. In other words, when any one of the errors occur it is latched and its respective bit is set in Register 26 (along with bit 15). No further errors can be latched until this error is cleared (See Register 31). This allows the user to determine which error occurred first.

BIT 0 (Receive TIMEOUT Error) - Bit 0 is set if the AutoMax DCS does not receive a message within the timeout period specified by REGISTER 80. When set, it will remain set until the AutoMax DCS application issues an error reset, REGISTER 31. (See section 2.2.1.1 for details). If the link is not active (Register 4 = 0), an error reset will have no effect. Also, if Register 26 is non-zero, no messages will be transmitted to the MaxPak III drive.

BIT 1 (Transmit Command OVERLAP Error) - Bit 1 is set if the AutoMax Processor writes a value to REGISTER 30 (the transmit message command) before the transmission of the previous message is complete. When set, it will remain set until the AutoMax DCS application issues an error reset, REGISTER 31 (See section 2.2.1.1 for details).

BIT 2 (Receive OVERLAP Error) - Bit 2 is set if the HSL module receives more than 2 new messages before it is finished processing the last received message. When set, it will remain set until the AutoMax DCS application issues an error reset, REGISTER 31 (See section 2.2.1.1 for details).

BIT 3 (Message Not Received Error) - Bit 3 is set if a configure, a status change (Off-Line to On-Line and vice-versa) or an Off-Line update is sent to the MaxPak III drive and no acknowledgement is received. When set, it will remain set until the AutoMax DCS application issues an error reset via REGISTER 31. If the MaxPak III drive is not running (SEQ_ARM_ACTIVE@ = OFF) and the HSL module is waiting on an acknowledge message from the MaxPak III drive, a timeout will not set this error latch immediately. The HSL module will send two retries of the same command message. If no acknowledgement before the third message is received, the timeout latch will be set.

BIT 4 (Invalid Operation) - HSL module attempted an invalid operation while the MaxPak III armature was active. Invalid operations include:

a) attempting to send a go off-line command
b) attempting to send a configure command

When set, it will remain set until the AutoMax DCS application issues an error reset, REGISTER 31.

BIT 15 (Global Communication Error) - This bit is set when any of the other Register 26 bits are set. (The logical 'OR' of Register 26 bits 0 to 14.)

REGISTER 27 - Configuration error flags. These flags will reflect the dynamic state of configuration status. Below is a list of errors that could occur and the bits that correspond to them:

B0 - Number of On-Line integer registers to be received from the MaxPak III drive is out of range of the MaxPak III drive's defined integer map (i.e. REGISTER 75 is too big).

B1 = Number of On-Line boolean bits to be received from the MaxPak III is out of range of the MaxPak III drive's defined booleans map (i.e. REGISTER 76 is too big).

B2 = MaxPak III drive and HSL module software revisions are incompatible.

B3 = Number of On-Line integer registers to be sent to the MaxPak III is out of range of the MaxPak III drive's defined integer register map (i.e. REGISTER 70 is too big).

B4 = Number of On-Line boolean registers to be sent to the MaxPak III is out of range of the MaxPak III drive's defined booleans map (i.e. REGISTER 71 is too big).

B5 = One or more configuration registers (70 to 81) is out of range from the default limits (see register descriptions for default limits).

B6 - B14 = Currently not defined

B15 = Global Configuration Error bit. This bit is set if any of the other Register 27 bits are set. (The logical 'OR' of Register 27 bits 0 - 14.)

When B15 is set, no messages will be sent from the HSL module to the MaxPak III drive. Once the user corrects his configuration error, this bit will be reset and message transmission may commence. It is up to the application to control the message sequence to restart the link. The HSL module will not modify values in REGISTERS 70 - 81 so that the user may see which REGISTER is in error.

REGISTER 28 - (MaxPak III Status Byte) This register is used to store booleans describing the current state of the MaxPak III drive. All messages received from the MaxPak III drive include this data automatically. Defined bits are as follows:

B7 = State of SEQ_ARM_ACTIVE@
B6 = State of SYS_SHUTDOWN@
B5 = State of SYS_INIT_FAIL@

Note: If bit 5 or bit 6 of Register 28 is non-zero, the drive is off-line.

REGISTER 29 - (Transmit active) When bit 0 of this register is set to 1, the HSL module is sending data to the MaxPak III drive. This register may be tested before a TRANSMIT COMMAND is given (via REGISTER 30) in order to prevent transmit OVERLAP errors.

Note: There is a maximum 500 uSec processing delay in recognizing the command in Register 30. Therefore, the transmit active flag goes true after a maximum of 500 uSec with respect to the command in Register 30.

REGISTER 30 - (COMMAND/STATUS CHANGE) This register is used for several purposes outlined below. Commands are issued by the user (or application task) writing the appropriate value to this register as follows:

2 = Go Off-line
3 = Go On-line
4 = Reconfigure
128 = Update Request (send message)

Note: All write access to this register **must be 16 bits wide.** For example, this sequence is correct:

```
IODEF CMD_REG%[SLOT=5,REGISTER=30]
  •
  •
CMD_REG%=4
```

whereas, this is not:

```
IODEF RECONFIGURE@[SLOT=5,REGISTER=30,BIT=2]
  •
  •
RECONFIGURE@=ON
```

Command 2 - This instructs the MaxPak III drive to go off-line.

Command 3 - This instructs the MaxPak III drive to go on-line.

Command 4 - This command is issued after the user makes a change to any of the configuration REGISTERS 70, 71, 75, 76, 80, or 81. Issuing this command after the change to these registers inform the MaxPak III of the change by sending it a message to that effect. NOTE: This can only be done while the MaxPak III is not running (SEQ_ARM_ACTIVE@ = OFF).

Command 128 - Update request. When this command is issued and no bits are set in REGISTER 26, REGISTER 27 or REGISTER 29, the HSL module will transmit the integers in dual port starting with data REGISTER 1100 and the booleans in dual port starting with data REGISTER 1400. Immediately after starting the transmit process, the HSL module will clear the command register, but this does not mean that the transfer of data is complete. For that information, monitor TRANSMIT ACTIVE REGISTER 29 for a true to false transition.

When a command is detected, and no communication and/or configuration errors are present (REGISTERS 26 AND 27 respectively) the HSL module immediately clears the command register and sets transmit active (REGISTER 29 Bit 0). After that, the message will be sent. This permits the HSL module to detect an overlap condition which may occur when transmit active is set and a new command is issued.

The command register should be checked prior to writing a value to it to insure that no previous request is pending. The command register is checked by performing a logical "AND" of its contents with Hexidecimal 0086. If the result is non-zero, a new request should not be written

REGISTER 31 - (ERROR RESET COMMAND) A false-to-true transition will clear all communication errors with the exception of a timeout. If, and only if, the link active bit is set (REGISTER 4 bit 0), error reset will also clear the timeout error. REGISTER 31 bit 0 must be held high for at least 50ms to insure the reset is seen by the HSL module. The HSL module will not write to this bit. Control of its value is given solely to the AutoMax processor card.

2.2.1.2.2 *Configuration Registers*

Configuration registers are used to tailor the HSL to the users needs. Each register controls a specific aspect of the communication. Each register must be verified to insure its value is acceptable. Default values for registers 70, 71, 75, 76, 80, and 81 are initialized by the HSL module at powerup or on a "Stop All" command. Some variables can be verified locally, others must be checked remotely.

Configuration is accomplished by writing the correct values to registers 70 – 81 and then writing a 4 to Register 30 (provided Registers 26 and 29 read 0).

The HSL may be configured when the MaxPak III drive is on-line or off-line. If the MaxPak III drive is on-line, the armature must not be active (SEQ_ARM_ACTIVE@ = OFF).

If the user tries to configure the MaxPak III drive while on-line and the armature is active, a configuration error results and is shown in REGISTER 27 bit 4. A command to go off-line will also cause this error. See REGISTER 27 for information on the Configuration Errors Register.

Both the MaxPak III drive and HSL module verify the configuration requested against their own internal variables to decide whether the new configuration is valid or not. If the configuration selected on one side conflicts with the configuration on the other side, a configuration error results and is shown in REGISTER 27. See REGISTER 27 for information on the Configuration Errors Register.

For instance, if the user wants to set the number of integer registers to transmit to the MaxPak III drive, the user places the value in REGISTER 70 and checks the MaxPak III status to insure it is not running (i.e. SEQ_ARM_ACTIVE@ = OFF, see REGISTER 28). If it is, the user must stop the drive with an update command before a configure is allowed.

After checking the status again to insure the MaxPak III drive is not running, the user would then send the configure command. If the MaxPak III drive determines if more integers have been requested than are defined in its HSL module register map, a configuration error results and is shown in REGISTER 27. See REGISTER 27 for information on the Configuration Errors Register.

REGISTER 40 – (Total input on-line integer registers) This value represents the number of input integers the MaxPak III drive has defined in its HSL register map which have an access level of 2 or less. It is provided so the user may read this register and determine the largest value permitted in REGISTER 70 that will guarantee a successful configure IF THE MaxPak III DRIVE IS ON-LINE.

REGISTER 41 – (Total input off-line integer registers) This value represents the number of input integers the MaxPak III drive has defined in its HSL register map which have an access level of 3 or more. It is provided so the user may read this register, add its value to the value read in REGISTER 41 (on-line integers) and determine the largest value permitted in REGISTER 70 that will guarantee a successful configure IF THE MaxPak III DRIVE IS OFF-LINE.

REGISTER 42 – (Total input on-line boolean bits) This value represents the number of input boolean bits the MaxPak III drive has defined in its HSL register map which have an access level of 2 or less. It is provided so the user may read this register and determine the largest value permitted in REGISTER 71 that will guarantee a successful configure IF THE MaxPak III DRIVE IS ON-LINE.

REGISTER 43 – (Total input off-line boolean bits) This value represents the number of input boolean bits the MaxPak III drive has defined in its HSL register map which have an access level of 3 or more. It is provided so the user may read this register add its value to the value read in REGISTER 42 (on-line booleans) and determine the largest value permitted in REGISTER 71 that will guarantee a successful configure IF THE MaxPak III DRIVE IS OFF-LINE.

REGISTER 44 – (Total output integer registers) This value represents the number of output integers the MaxPak III drive has defined in its HSL register map. It is provided so the user may read this register and determine the largest value permitted in REGISTER 75 that will guarantee a successful configure. The value in this register is the maximum value allowed in REGISTER 75 regardless of the current state of the MaxPak III drive (on or off line).

REGISTER 45 – (Total output boolean bits) This value represents the number of output booleans the MaxPak III drive has defined in its HSL register map. It is provided so the user may read this register and determine the largest value permitted in REGISTER 76 that will guarantee a successful configure. The value in this register is the maximum value allowed in REGISTER 76 regardless of the current state of the MaxPak III drive (on or off line).

REGISTER 60 – (Fatal Error #) A Reliance Service register. Contains specific fatal error codes used in board diagnostics, repair, and replacement.

REGISTER 62 & 63 – (HSL Comm Card ASCII ID 'HSL ') Card Ascii ID.

REGISTER 64 – Contains the software revisions of the HSL module. This data is sent to the MaxPak III drive when the link is established. This verifies that the software revision of the MaxPak III drive and the software revision of the HSL module are compatible.

REGISTER 70 – (Number of integer registers to transmit to the MaxPak III drive). This register may be changed at any time, although the change will not take effect until a configure command is issued with the drive stopped (SEQ_ARM_ACTIVE@ = OFF). The number specified in REGISTER 70 includes the high priority register (Register 1100). For example, if an application wanted to send REGISTERS 1100 – 1109 then REGISTER 70 should contain 10. If the range is exceeded, the value will remain as the application has configured it (out of range), and REGISTER 27 bits 15 and bit 3 or 5 will be set to 1 to indicate a configuration error.

Note: No other messages with the exception of a configure may be sent when REGISTER 27 is non-zero.

DEFAULT VALUE = 2. MAX. RANGE = 2 to 256.

Note: The actual range is limited by the MaxPak III drive variable map.

REGISTER 71 - (Number of boolean bits to transmit to the MaxPak III drive.) The user may change this register any time. The user must insure SEQ_ARM_ACTIVE@ is OFF and then send a configure command to the MaxPak III drive before changes in this register will have any effect. If the range is exceeded, the value will remain as the application has configured it (out of range), no messages will be sent to the MaxPak III drive, and REGISTER 27 bits 15 and bit 4 or 5 will be set to 1 to indicate a configuration error.

DEFAULT VALUE = 2. MAX. RANGE = 2 to 256.

Note: The actual range is limited by the MaxPak III drive variable map.

REGISTER 75 – (Number of On-Line integer registers to receive from the MaxPak III). This register may be changed at any time, although the change will not take effect until a configure command is issued with the drive stopped (SEQ_ARM_ACTIVE@ = OFF). Use COMMAND REGISTER 30 to send commands to the MaxPak III drive. The number specified in REGISTER 75 includes the two high priority registers (100 and 101). For example, if an application wanted to receive REGISTERS 100 - 109, then REGISTER 75 should contain a 10. If the range is exceeded, the value will remain as the application has configured it (out of range), no messages will be sent to the MaxPak III drive, and REGISTER 27 bits 15 and bit 0 or 5 will be set to 1 to indicate a configuration error.

DEFAULT VALUE = 2. RANGE = 2 to 256.

Note: The actual range is limited by the MaxPak III drive variable map.

REGISTER 76 – (Number of On-Line boolean registers to receive from the MaxPak III drive). The user must insure SEQ_ARM_ACTIVE@ is OFF and then send a configure command to the MaxPak III drive before changes in this register will have any effect. If the range is exceeded, the value will remain as the application has configured it (out of range), no messages will be sent to the MaxPak III drive, and REGISTER 27 bits 15 and bit 1 or 5 will be set to 1 to indicate a configuration error.

DEFAULT VALUE = 2. MAX. RANGE = 2 to 256.

Note: The actual range is limited by the MaxPak III drive variable map.

REGISTER 80 – (Maximum receive TIMEOUT value) Specifies the maximum time (in milliseconds) from the last valid message that the HSL module will wait for a message from the MaxPak III drive before a timeout error will occur. The user must insure SEQ_ARM_ACTIVE@ is OFF and then send a configure command to the MaxPak III drive before changes in this register will have any effect. Use COMMAND REGISTER 30 to send it. If the valid range is exceeded, the value will remain as was originally configured, no messages will be sent to the MaxPak III drive, and REGISTER 27 bits 15 and 5 will be set to 1 to indicate a configuration error.

DEFAULT VALUE = 45msec. (2 x L2_SCAN_PERIOD% + 5msec)
RANGE = 10 to 100 msec.

REGISTER 81 – (Message Scan period) Specifies the tick interval between update requests (e.g. speed loop ticks). Note: 1 tick = 0.5ms. The user must insure SEQ_ARM_ACTIVE@ is false and then send a configure command to the MaxPak III drive before changes in this register will have any effect. If the range is exceeded, the value will remain as the application has configured it (out of range), no messages will be sent to the MaxPak III drive, and REGISTER 27 bit 15 and possibly bit 5 will be set to 1 to indicate a configuration error.

DEFAULT VALUE = 11.
RANGE = 11 to 198 ticks (5.5 msec to 99 msec).

2.2.2 MaxPak III High Speed Link (HSL) Variables

HSL_TIMEOUT_T%

Description:	HSL receive message timeout time period in msec
Type:	Input Variable
Retentive:	Yes
Access Level:	5
Value Range:	10 to 250
Scale:	1 = 1 millisecond
Typical Value:	54 = (2 x AutoMax DCS Scan Period) + 10
Comments:	Maximum time between valid messages received by the MaxPak III drive from the 57C424 (HSL module) after a connection between them has been established. If this time is exceeded, HSL_TIMEOUT_E@ is set to ON. The timeout function cannot be disabled when the MaxPak III drive is on-line.

HSL_ENABLE@

Description:	HSL communications channel enable configuration switch.
Type:	Input Variable
Retentive:	Yes
Access Level:	4
Value Range:	ON, OFF
Typical Value:	OFF
Comments:	If ON during a MaxPak III drive power-up or OFF-LINE to ON-LINE transition, the HSL communications channel is installed as the controller of the MaxPak III drive onboard serial port (provided MB_ENABLE@ = OFF). When OFF, the HSL communications channel is not installed.

HSL_XMITD_MSGS%

Description:	Number of HSL messages transmitted by the MaxPak III drive
Type:	Output Variable
Retentive:	No
Access Level:	0
Value Range:	-32768 to 32767
Scale:	N/A
Typical Value:	-32768 to 32767
Comments:	This variable is treated as an unsigned number by the MaxPak III drive. It therefore starts at zero (0), counts to 32767, rolls over to -32768, and continues counting towards zero (0). Its primary use is for diagnostics.

HSL_RECVD_MSGS%

Description:	Number of HSL messages received by the MaxPak III drive
Type:	Output Variable
Retentive:	No
Access Level:	0
Value Range:	-32768 to 32767
Scale:	N/A
Typical Value:	-32768 to 32767
Comments:	This variable is treated as an unsigned number by the MaxPak III drive. It therefore starts at zero (0), counts to 32767, rolls over to -32768, and continues counting towards zero (0). Its primary use is for diagnostics.

HSL_CHKSUM_ERRS%

Description:	Number of HSL message checksum errors
Type:	Output Variable
Retentive:	No
Access Level:	0
Value Range:	-32768 to 32767
Scale:	N/A
Typical Value:	-32768 to 32767
Comments:	A checksum error indicates a corrupted message has been received. The contents of a message with a checksum error are discarded. This variable is treated as an unsigned number by the MaxPak III drive. It therefore starts at zero (0), counts to 32767, rolls over to -32768, and continues counting towards zero (0). Its primary use is for diagnostics.

HSL_TIMEOUT_E@

Description:	HSL receive message timeout error status
Type:	Output variable
Retentive:	No
Access Level:	0
Value Range:	ON or OFF
Typical Value:	OFF
Comments:	ON if the MaxPak III drive detects a receive message timeout condition. A receive message timeout condition occurs when the MaxPak III drive does not receive a new update message within the receive timeout period (HSL_TIMEOUT_T% msec) since the last update message was received. This condition forces a re-connection between the HSL module and the MaxPak III drive. Once the re-connection has been established, HSL_TIMEOUT_E@ is set to OFF.

HSL_XOVRLAP_E@

Description:	HSL transmit message overlap error status
Type:	Output variable
Retentive:	No
Access Level:	0
Value Range:	ON or OFF
Typical Value:	OFF
Comments:	ON if the MaxPak III drive is in a transmit overlap condition, OFF otherwise. A transmit message overlap condition occurs when the MaxPak III drive is currently sending a message and the L2_SCAN_PERIOD% expires, causing the MaxPak III drive to attempt to send another message.

HSL_ROVRLAP_E@

Description:	Dynamic Receive Overlap Error Detection Flag
Type:	Output variable
Retentive:	No
Access Level:	0
Value Range:	ON or OFF
Typical Value:	OFF
Comments:	ON if the MaxPak III drive detects a receive message overlap condition. A receive message overlap condition occurs when the MaxPak III drive receives a message before the previous received message has been processed. This condition forces a reconnection between the HSL module and the MaxPak III drive. Once the re-connection has been established, HSL_ROVRLAP_E@ is set OFF.

FLT_HSL_COMM_L@

Variable Description:	HSL Communications Fault Latch
Type:	Output variable
Retentive:	No
Access Level:	0
Value Range:	ON or OFF
Scale:	Boolean
Typical Value:	OFF
Comments:	Latched (ON) when any of HSL_TIMEOUT_E@, HSL_ROVRLAP_E@, or HSL_XOVRLAP_E@, become ON while the MaxPak III drive is On-Line. Unlatched (OFF) by a drive fault reset command (An off-to-on transition of FLT_RESET@). This variable is also contained in bit 10 of the variable FLT_LATCHES%.

HSL_ERROR_FLAGS%

Description:	HSL communications error flags
Type:	Output Variable
Retentive:	No
Access Level:	0
Value Range:	0 to 7
Scale:	N/A
Typical Value:	0 (no errors)
Comments:	Contains the variables: HSL_TIMEOUT_E@ in bit 0 HSL_ROVRLAP_E@ in bit 1 HSL_XOVRLAP_E@ in bit 2 Bits 3 – 15 are not used and will be 0.

2.2.3 Sequence of Events

The HSL module can transmit many variables. One variable from AutoMax to the MaxPak III drive has high priority and 2 variables from the MaxPak III drive to the AutoMax have high priority. The high priority variables are typically used by current reference and sometimes speed feedback. The following diagram shows "TYPICAL" MaxPak III-toAutoMax connections.

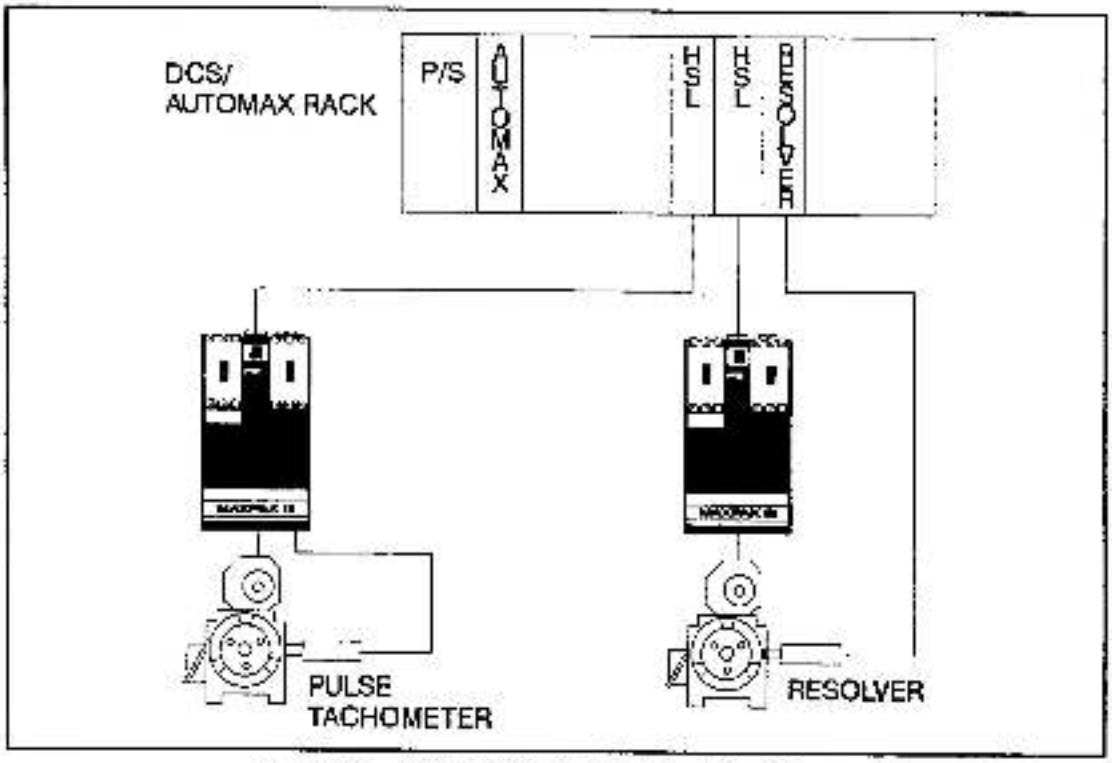


Figure 2-2 – Typical MaxPak III-to-AutoMax Connections

The "TYPICAL" sequence of events is as follows:

If speed feedback is connected directly to the MaxPak III drive, the AutoMax DCS processor runs the Speed Loop, generates a current reference, and sends the current reference via HSL module to the MaxPak III drive. The MaxPak III drive uses this reference for input to the Current Minor Loop. Speed feedback is then sent back to the AutoMax DCS from the MaxPak III drive. Upon receipt of speed feedback, the HSL module will interrupt the AutoMax thereby starting its speed loop. In this case, the MaxPak III drive has both speed feedback and armature voltage locally, and may act immediately if a problem is detected (i.e., tachloss and overspeed).

Reference section 2.2.6 of this instruction manual for details on tach loss and overspeed.

2.2.4 MESSAGES

Many messages may be sent to the MaxPak III drive and are controlled by command REGISTER 30 in the AutoMax DCS. When the AutoMax DCS task is ready to send data, it must command the HSL module via dual-port REGISTER 30 to begin sending the message.

The MaxPak III drive continually sends a new message to the AutoMax DCS at the end of each MaxPak III L2_SCAN_PERIOD%. L2_SCAN_PERIOD% is a user configurable variable typically equal to 20 mSec.

The number of registers that are sent from each end of the link is user configurable at both ends, independently. The message size is controlled by the driver software, but is directly related to the HSL module's message scan period (e.g. speed loop ticks in Register 81) and MaxPak III drive's L2_SCAN_PERIOD% value.

2.2.4.1 Message Control (L2_SCAN_PERIOD%)

The messages sent from the AutoMax DCS to MaxPak III drive and from the MaxPak III drive to AutoMax DCS are not necessarily synchronized.

Update time (scan time) to transmit variables from the MaxPak III drive to the HSL module is based on the time set into the MaxPak III drive variable L2_SCAN_PERIOD% (10 – 20 milliseconds). (Note: A value of 5 msec for L2_SCAN_PERIOD% is not supported for use with the HSL.) Programmable timeout periods on both the MaxPak III drive and HSL module are employed to assure valid data and link integrity.

2.2.4.1.1 *AutoMax DCS to MaxPak III Information Update Rate Calculations*

WARNING

THE USER MUST INSURE THAT ALL APPLICATION CRITICAL PARAMETERS (VARIABLES) ARE UPDATED SUFFICIENTLY OFTEN FOR SAFE AND PROPER OPERATION (E.G SPEED FEEDBACK, VOLTAGE FEEDBACK, AND E-STOP). CRITICAL PARAMETERS MUST BE GIVEN HIGH PRIORITY OR FIXED REGISTER ASSIGNMENTS AND UPDATE TIMES MUST BE CALCULATED. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

The time required for the HSL module to send a message to the MaxPak III drive is determined by the message type. An update command will be recognized within 600 usec of the command being written to Register 30. A request to place the MaxPak III drive on-line/off-line or a configure request will take longer (maximum of 11 ms).

Each update command causes a portion of the output integer map (Registers 1100 - 1355) and a portion of the output boolean map (Register 1400-1415) to be sent to the MaxPak III drive. Some of these variables are "fixed", meaning that they will be sent each time an update request is issued. The "high priority" register is included in this category. Other variables are "multiplexed", meaning that they will be distributed over several update messages and thus may take longer to transmit. Once all multiplexed variables have been sent, the cycle repeats, starting with the current value of the first multiplexed variable. The number of fixed integers and booleans, and the total number of registers per update message are determined by the tick count written in Register 81. Fixed integers are contiguous, starting with Register 1100 and fixed booleans are contiguous, starting with Register 1400. A table detailing message composition for varying tick rates and calculations for determining information update rates follow (also see Section 4.5).

Optional synchronization of the AutoMax DCS speed loop task to the MaxPak III L2_SCAN_PERIOD% is provided via Multibus Interrupt. The interrupt occurs when the AutoMax DCS has received the two high priority integer registers from the MaxPak III drive.

Table 2-1 – AutoMax DCS –> MaxPak III Drive Message Composition

Reg 81 (TICKS)	(1) Max # Fixed Integer (MFI) Registers	(2) Max # Fixed Boolean (MFB) Registers	Max # of Registers per Message (MRPM)
11–21	2	1	4
22–32	6	2	10
33–43	9	3	17
44–54	18	3	28
55–65	18	3	38
66–76	18	3	46
77–87	21	3	55
88–98	29	4	64
99–109	37	4	72
110–120	45	5	81
121–131	54	5	90
132–142	63	6	100
143–153	71	6	108
154–164	79	7	117
165–175	87	7	125
176–186	95	8	134
187–197	104	8	143
198 ONLY	111	10	152

(1) Includes 1 high priority integer

(2) Each boolean register includes up to 16 boolean variables

DEFINITIONS:

Ticks = Scan Loop Ticks (Reg 81) (See preceding table)
Total Integers (TI) = (Reg 70)
Total Packed Booleans (TB) = (Reg 71)/16 (rounded up)

Fixed Integers Per Message (FIPM) = lesser of: 1) TI or, 2) MFI

Fixed Booleans Per Message (FBPM) = lesser of: 1) TB or, 2) MFB

Multiplexed Registers Per Message (MuRPM) = lessor of: 1) 31 or, 2) MRPM-FIPM-FBPM

Note: If TI $\leq$ MFI and TB $\leq$ MFB, then there are no multiplexed variables (i.e. all variables are sent every scan).

*Integer Messages Per Cycle (IMPC) = $\frac{\text{TI-FIPM}}{\text{MuRPM}}$

*Boolean Messages Per Cycle (BMPC) = $\frac{\text{TB-FBPM}}{\text{MuRPM}}$

# of Messages Per Cycle (MPC) = IMPC + BMPC

Note: "Cycle" refers to the number of messages required to send all requested registers, both integer and boolean.

* The result of these calculations are rounded up.

INFORMATION UPDATE INTERVAL:

Fixed integers and booleans = TICKS (Register 81) x 0.5 ms
Multiplexed integers and booleans = MPC x TICKS x 0.5 ms

DCS -> MP3 UPDATE EXAMPLE

Given:

Reg 70 = 20
Reg 71 = 25
Reg 81 = 20

From Table:

MFI = 2
MFB = 1
MRPM = 4

Calculations:

TI = (Reg 70) = 20
TB = (Reg 71)/16 = 25/16 = 2 (rounded up)
FIPM = MFI = 2 (MFI < TI)
FBPM = MFB = 1 (MFB < TB)
IMPC = (20-2)/(4-2-1) = 18/1 = 18
BMPC = (2-1)/(4-2-1) = 1/1 = 1
MPC = 18 + 1 = 19

Information Update Interval:

Fixed Integers & Booleans: TICKS x 0.5 ms = 20 x 0.5 ms = 10 ms
Multiplexed Integers & Booleans: MPC x TICKS x 0.5 ms = 19 x 20 x 0.5 ms = 190 ms

2.2.4.1.2 MaxPak III to AutoMax DCS Information Update Rate Calculations

WARNING

THE USER MUST INSURE THAT ALL APPLICATION CRITICAL PARAMETERS (VARIABLES) ARE UPDATED SUFFICIENTLY OFTEN FOR SAFE AND PROPER OPERATION (E.G SPEED FEEDBACK, VOLTAGE FEEDBACK, AND E-STOP). CRITICAL PARAMETERS MUST BE GIVEN HIGH PRIORITY OR FIXED REGISTER ASSIGNMENTS AND UPDATE TIMES MUST BE CALCULATED. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

The MaxPak III drive sends a new message to the AutoMax DCS at the end of each MaxPak III L2_SCAN_PERIOD%. The L2_SCAN_PERIOD% and the multibus interrupt capability of the HSL module allows, but does not require, synchronizing MaxPak III drive physical inputs (e.g. speed feedback) in the MaxPak III drive with a task (e.g. speed loop) in an AutoMax DCS processor.

(Note: An AutoMax (not DCS 5000) and a 57C430A processor must be used if interrupt capability is desired.)

The number of registers sent by the MaxPak III drive each L2_SCAN_PERIOD% varies with the value of L2_SCAN_PERIOD%. A certain number of "fixed" integer and boolean registers will be sent each L2_SCAN_PERIOD%, while the remaining ("multiplexed") integer and boolean registers will be distributed over several messages and thus may require several L2_SCAN_PERIODs for transmission. (The "high priority" registers are included in the category of fixed registers.) When all "multiplexed" variables have been sent, the cycle repeats with the first multiplexed variable.

A table showing message composition for varying tick rates and calculations for determining information update rates follows (see also Section 4.5):

Table 2-2 – MaxPak III Drive –> AutoMax DCS Message Composition

L2_SCAN_PERIOD%	(1) Max # of Fixed Integers (MFI)	(2) Max # of Fixed Packed Booleans (MFB)	Max # of Registers per Message (MRPM)
20 mSec (default)	16	3	23
15 mSec	7	2	13
10 mSec	2	2	5

(1) Includes 2 high priority integers

(2) Each packed boolean includes up to 16 boolean variables

Note: 5 msec L2_SCAN_PERIOD% is not supported for use with the HSL option.

DEFINITIONS:

Total Integers (TI) = (Reg 75)
Total Packed Booleans (TB) = (Reg 76)/16 (rounded up)
Fixed Integers Per Message (FIPM) = lesser of: 1) TI or, 2) MFI
Fixed Booleans Per Message (FBPM) = lesser of: 1) TB or, 2) MFB
Multiplexed Registers Per Message (MuRPM) = MRPM-FIPM-FBPM

$$*\text{Integer Messages Per Cycle (IMPC)} = \frac{TI - FIPM}{MuRPM}$$

$$*\text{Boolean Messages Per Cycle (BMPC)} = \frac{TB - FBPM}{MuRPM}$$

Note: "Cycle" refers to the number of messages required to send all requested registers, both integer and boolean.

* The result of these calculations are rounded up.

# of Messages Per Cycle (MPC) = IMPC + BMPC

INFORMATION UPDATE INTERVAL:

Fixed integers and booleans = L2_SCAN_PERIOD%
Multiplexed integers and booleans = MPC x L2_SCAN_PERIOD%

MP3 -> DCS MESSAGE UPDATE RATE EXAMPLE

Given:

Reg 75 = 40
Reg 76 = 50
L2_SCAN_PERIOD% = 20

From Table:

MFI = 16
MFB = 3
MRPM = 23

Calculations:

TI = (Reg 75) = 40
TB = (Reg 76)/16 = 50/16 = 4 (rounded up)
FIPM = 16 (MFI < TI)
FBPM = MFB = 3 (MFB < TB)
MuRPM = 23-16-3 = 4
IMPC = (40-16)/(4) = 24/4 = 6
BMPC = (4-3)/(4) = 1/4 = 1 (rounded up)
MPC = 6 + 1 = 7

Information Update Interval:

Fixed integers and booleans: L2_SCAN_PERIOD% = 20 ms
Multiplexed integers and booleans: MPC x L2_SCAN_PERIOD%
= 7 x 20ms
= 140 ms

2.2.5 Timeout

On power up, a connect message is sent to the MaxPak III drive. The timeout period will not begin in the HSL module until a valid connect acknowledge message has been received from the MaxPak III drive. The timeout on the MaxPak III drive will not start until after the first valid On-Line message has been received from the HSL module. Each VALID message received resets the local receive message timeout timer. Timeout detection may not be disabled by the user, although it is automatically disabled when the MaxPak III drive is Off-Line to account for longer response time delays. The MaxPak III drive will also disable its timeouts if it gets a connect/configure message or an Off-Line command.

2.2.6 Tach Loss and Overspeed

2.2.6.1 AutoMax Rack Receiving Speed Feedback

If speed feedback is connected directly to the AutoMax DCS, the AutoMax DCS sends the MaxPak III drive current reference (CML_REF%) after running the speed loop. It must also send speed feedback to the MaxPak III drive for calculating the tach loss and overspeed functions.

Note: The MaxPak III drive should always be used to determine tach loss and overspeed regardless of where speed feedback is connected in the system.

In addition to the MaxPak III drive performing the tach loss and overspeed functions, it is highly recommended that these functions are also detected in the AutoMax DCS software and that current reference is clamped to zero when a fault is detected. Clamping the current reference after a tach loss and preventing high current references from being sent to the drive may reduce the amount of overspeed prior to drive shutdown. In any case, the user is responsible to assure a safe shutdown in any tach loss or overspeed condition.

WARNING

SPEED FEEDBACK MUST BE BROUGHT BACK TO THE MAXPAK III DRIVE TO THE APPROPRIATE NODE FOR PERFORMING THE TACH LOSS AND OVERSPEED FUNCTIONS. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

The speed feedback value should be written to **any fixed register** (typically, HSL_IN_INT_001) in the HSL module and accordingly, the configuration should be reconfigured. This will allow the speed feedback to be sent to the MaxPak III drive. For example:

When speed feedback is connected to the AutoMax DCS rack, incorporate the following syntax:

AutoMax:

IODEF RPM_FDBK% [SLOT=x, REGISTER=1101]

Note: RPM_FDBK% represents speed feedback.

MaxPak III Drive:

HSL_IN_INT_001 = L2_FEEDBACK%
[L2_FEEDBACK%] = L2_FEEDBACK%

WARNING

WHEN FEEDBACK IS BROUGHT DIRECTLY INTO THE AUTOMAX DCS, THERE IS A TRANSPORT TIME DELAY ASSOCIATED WITH BRINGING THE FEEDBACK TO THE MAXPAK III DRIVE AND TO REACT TO THE CONDITION. THIS DELAY IS DIRECTLY RELATED TO THE L2_SCAN_PERIOD% AND THE AUTOMAX DCS MESSAGE SCAN PERIOD (REGISTER 81). THE USER IS RESPONSIBLE FOR CALCULATING TIME DELAYS TO ASSURE THAT THE DRIVE IS SHUTDOWN PRIOR TO THE MOTOR AND LOAD REACHING AN UNSAFE SPEED DURING A FAULT CONDITION. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

In the event of a tach loss or overspeed condition, the user must calculate the speed that may be reached when accelerating beyond the overspeed threshold at maximum torque with minimum connected inertia until power is removed from the motor. This speed must be less than the MSS (Maximum Safe Speed) of the system.

The MSS is the lowest motor shaft speed at which any component of the rotating subsystem reaches its own published maximum mechanical safe speed. For example, if a motor and gear reducer is to be driven, then the system must be protected for the lesser of the motor MSS and the reducer (input shaft) MSS.

To calculate the transport time delay (time duration from when an overspeed or tach loss condition is detected to when the MaxPak III drive is shutdown), use the following formula:

Time Delay = (DCS speed loop scan time x 2)ms + (MaxPak III L2_SCAN_PERIOD% x 1.5)ms + 8ms

For example:

For a 22 ms speed loop scan time and a 20 ms L2_SCAN_PERIOD% in the MaxPak III DRIVE, the time will be:

(22 x 2)ms + (20 x 1.5)ms + 8ms = 44ms + 30ms + 8ms = 82ms

2.2.6.2 MaxPak III Drive Receiving Speed Feedback

When speed feedback is connected directly to the MaxPak III drive, incorporate the following syntax:

MaxPak III Drive:

[L2_FEEDBACK%] = the "variable" in the drive which represents the processed speed feedback either from a pulse tach or an analog tach (typically, PT_SPEED_FB%).

2.2.7 Data/Link Integrity

The programmable timeout periods on both the MaxPak III drive and the HSL module are employed to assure valid data and link integrity.

A two byte start sequence in the message along with 2 separate checksum evaluations provide the software data integrity. Additionally, odd parity is used to detect one bit failures within any one particular byte. No data is written to its destination until it is validated by at least one checksum and there are no hardware (parity or framing) transmission errors.

3.0 INSTALLATION

This section describes how to install and replace the High Speed Link (HSL) module. Also described is the RS-232 connection between the HSL module and the MaxPak III. Reference Appendix D for RS-232 Wiring Characteristics and Appendix E for Technical Specifications.

DANGER

THE USER IS RESPONSIBLE FOR CONFORMING WITH THE NATIONAL ELECTRICAL CODE AND ALL OTHER APPLICABLE LOCAL CODES. WIRING PRACTICES, GROUNDING, DISCONNECTS, AND OVER-CURRENT PROTECTION ARE OF PARTICULAR IMPORTANCE. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN SEVERE BODILY INJURY OR LOSS OF LIFE.

DANGER

ONLY QUALIFIED ELECTRICAL PERSONNEL FAMILIAR WITH THE CONSTRUCTION AND OPERATION OF THIS EQUIPMENT AND THE HAZARDS INVOLVED SHOULD INSTALL, ADJUST, OPERATE, AND/OR SERVICE THIS EQUIPMENT. READ AND UNDERSTAND THIS INSTRUCTION MANUAL AND APPROPRIATE MAXPAK III AND AUTOMAX DCS MANUALS IN THEIR ENTIRETY BEFORE PROCEEDING. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN SEVERE BODILY INJURY OR LOSS OF LIFE.

WARNING

INSERTING OR REMOVING THIS MODULE OR ITS CONNECTING CABLES MAY RESULT IN UNEXPECTED MACHINE MOVEMENT. TURN OFF POWER BEFORE INSERTING OR REMOVING THE MODULE OR ITS CONNECTING CABLES. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

3.1 Wiring

The installation of wiring should conform to the NEC and all applicable local codes. To reduce the possibility of electrical noise interfering with the control system, exercise care when installing wiring between the module and the external hardware. For detailed recommendations, refer to IEEE 518.

3.2 Initial Installation

Use the following procedure to install the HSL module (M/N 57C424).

1. Stop any application programs that are running in the AutoMax DCS.
2. Turn off all power to the MaxPak III drive.
3. Turn off power to the AutoMax DCS system.
4. Install the HSL module in an empty slot of the AutoMax DCS Card Rack (which matches the slot defined in the configuration).

Note: The MaxPak III RS-232 port uses a 25-pin female "D" shell type connector.

5. Connect the RS-232 cable from the 25-pin female "D" shell type connector of the MaxPak III to the HSL Module connector.
6. Turn on power to the AutoMax system.
7. Turn on power to the MaxPak III drive.
8. Enable the High Speed Link option on the MaxPak III (HSL_ENABLE@ = ON, MB_ENABLE@ = OFF). To enable the HSL option without having to modify MB_ENABLE@ or HSL_ENABLE@, perform the following instructions:
   a. At the end of MaxPak III system initialization (after 'MPDn' diag sequence), hold down the Increment key (Up Arrow) and Enter key (E) and then release the keys while 'HSL' is being displayed on the On-Board Keyboard Display.

Note: This method of channel selection will set MB_ENABLE@, HSL_ENABLE@, and COM_PORT_USE% to the values indicated in the following table.

Table 3-1 – MaxPak III Comm Port HSL Channel Selection

CHANNEL SELECTION	MB_ENABLE@	HSL_ENABLE@	COM_PORT_USE%
'HSL'	OFF	ON	0

9. Verify that the hardware has been installed correctly.

   Verify that the HSL module and the MaxPak III are communicating via the link as follows (this check assumes that the 57C424 and MaxPak III are connected with an RS-232 cable as configured in Appendix D), HSL is enabled on the MaxPak III (HSL_ENABLE@ = ON, MB_ENABLE@ = OFF), and the AutoMax DCS application controlling the HSL has not yet been started:
   a. Verify that the 'C' on the HSL module's 7-segment fault code LED has disappeared.
   b. Monitor the variable HSL_RECVD_MSGS% on the MaxPak III On-board Keyboard Display (OKD) or by using the MaxPak III Enhanced Monitor Program, or Terminal Programmer, or Handheld Terminal. This variable will have a value $\geq$ 1 if the link has been established between the MaxPak III and the HSL module.
   c. Monitor the variable HSL_XMITD_MSGS% on the MaxPak III OKD or by using the MaxPak III Enhanced Monitor Program, or Terminal Programmer, or a Handheld Terminal. This variable should be continuously incrementing if the link has been established. (Note: This variable will rollover from 32767 to –32768). (The OKD latches the value of the displayed variable. To determine whether a variable is changing, the "E" key must be used to display its starting value, then pressed twice more to display its subsequent value.) Also, examine register 14 on the HSL module (number of messages received by the HSL module) using the AutoMax DCS I/O monitor. Verify that this number is continuously incrementing.

3.3 Module Replacement

Use the following procedure to replace the module.

1. Stop any application programs that are running in the AutoMax DCS.
2. Turn off power to the MaxPak III connected to the HSL module.
3. Turn off power to the AutoMax DCS system.
4. Disconnect the RS-232 cable from the GATEWAY connector on the faceplate of the HSL module.
5. Remove the HSL module from the slot in the AutoMax DCS rack.
6. Place the new module in the slot vacated by the defective module.
7. Connect the RS-232 cable to the GATEWAY connector on the faceplate of the module.
8. Turn on power to the AutoMax DCS system.
9. Turn on power to the MaxPak III drive.
10. Verify that the hardware has been installed correctly.

Verify that the HSL module and the MaxPak III are communicating via the link as follows (this check assumes that the HSL module and MaxPak III are connected with an RS-232 cable as configured in Appendix D), HSL is enabled on the MaxPak III (HSL_ENABLE@ = ON, MB_ENABLE@ = OFF), and the AutoMax DCS application controlling the HSL has not yet been started:

1. Verify that the 'C' on the HSL module's 7-segment fault code LED has disappeared.
2. Monitor the variable HSL_RECVD_MSGS% on the MaxPak III On-board Keyboard Display (OKD) or by using the MaxPak III Enhanced Monitor Program, or Terminal Programmer, or Handheld Terminal. This variable will have a value $\geq$ 1 if the link has been established between the MaxPak III and the HSL module.
3. Monitor the variable HSL_XMITD_MSGS% on the MaxPak III OKD or by using the MaxPak III Enhanced Monitor Program, or Terminal Programmer, or a Handheld Terminal. This variable should be continuously incrementing if the link has been established. (Note: This variable will rollover from 32767 to -32768). (The OKD latches the value of the displayed variable. To determine whether a variable is changing, the "E" key must be used to display its starting value, then pressed twice more to display its subsequent value) Also, examine register 14 on the HSL module (number of messages received by the HSL module) using the AutoMax DCS I/O monitor. Verify that this number is continuously incrementing.

4.0 PROGRAMMING

This section describes how data is organized in the HSL module and MaxPak III drive download requirements. It also provides examples of how that module is accessed by application programs.

The programmer must include limits in the application software to ensure that the data sent to the MaxPak III is always in the allowable range. If appropriate limits are not included, the data is clamped and an error is logged in the MaxPak III drive.

Refer to J-3649 Instruction Manual for AutoMax DCS systems using Version 2.1 or earlier Program Executives to write configuration tasks.

Refer to J-3684 Instruction Manual for AutoMax DCS systems using Version 3.0 or later Program Executives to write configuration tasks.

Programming Notes:

If using the HSL Default Register Map (see Appendix A), all variables up to the last one used must have an accurate and acceptable value. This is typically accomplished by an initialization task running in the AutoMax rack.

WARNING

ALL HSL INPUT VARIABLES MUST BE INITIALIZED TO APPROPRIATE VALUES PRIOR TO SENDING UPDATE MESSAGES. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

The values of the Configuration Registers (70, 71, 75, 76, 80, 81) may be incorporated in the initialization task.

If L2_EXECUTE_EN@ or ML_EXECUTE_EN@ is to be used (= ON) then a L2_SCAN_PERIOD% equal to 15 or 20 is recommended. A MaxPak III CPU OVERLOAD may occur if the L2_SCAN_PERIOD% is less than 15.

Constraints:

All writes to Register 30 (CMD_REG%) must be an integer, not boolean.

CMD_REG% must be the last defined common in a control block task.

If CNF_ERRS% = 0, any command may be written to the CMD_REG% provided the following conditions are met:

- CMD_REG% logically "ANDed" with 086h = 0
- TX_ACTIVE@ (Register 31, bit 0) = 0
- LINK_ACTIVE@ (Register 4, bit 0) = 1
- COMM_ERRS% (Register 26) = 0

Note: The CMD_REG% must be checked for 0 prior to checking TX_ACTIVE@.

If CNF_ERRS% < > 0, only a configuration request may be written to the CMD_REG%. This may be written at any time although it will not be acted upon until LINK_ACTIVE@ = TRUE and COMM_ERRS% = 0.

If a COMM_ERR causes a CONFIGURE request to be latched in the CMD_REG (Register 30), the CONFIGURE message will be sent as soon as the COMM_ERR is reset. If it causes any other request to be latched, the CMD_REG will be cleared as soon as the COMM_ERR is reset and the request will be ignored.

4.1 MaxPak III HSL Registers

MaxPak III HSL Registers are classified into input/output and boolean/integer registers. They may be assigned to any MaxPak III variable. Registers will contain the values of the MaxPak III variables (input or output) to which they are assigned. MaxPak III input variables assigned to input registers will receive their values from the input registers.

The MaxPak III drive may be either On-Line or Off-Line. When the MaxPak III is On-Line, only the number of output variables requested by the HSL module's REGISTER 75, and REGISTER 76 will be sent to the HSL module. In other words, if REGISTERS 75 and 76 contain 10 and 5 respectively, HSL_OUT_INT_000 through HSL_OUT_INT_009 and HSL_OUT_BOOL_000 through HSL_OUT_BOOL_004 are sent respectively.

If the MaxPak III drive is Off-Line, all output registers defined in the HSL register map are sent regardless of what is specified in REGISTER 75 and REGISTER 76.

The number of input variables sent to the MaxPak III drive is always determined by the contents of Registers 70 and 71, regardless of whether the drive is On-line or Off-line.

HSL registers are assigned within the MaxPak III prior to connection to the HSL link by using the Download program in the MaxPak III HOSTIF mode. (Refer to the MaxPak III drive instruction manual). Example download statements are:

```
1000        HSL_IN_INT_000 = CML_REF%
1100        HSL_IN_BOOL_000 = SEQ_DRIVE_EN@
.           .           .
2000        HSL_OUT_INT_000 = PT_SPEED_FB%
2100        HSL_OUT_BOOL_000 = SEQ_ARM_ACTIVE@
```

4.2 High Speed Link (HSL) Configuration Map

The following diagram illustrates the HSL Hardware Configuration. The MaxPak III drive contains the HSL register assignments (HSL_OUT_* and HSL_IN_*) as defined in the download configuration file. Remember, the HSL default configuration file shipped with the drive (see Appendix A) can be used or a custom configuration file may be created. The HSL registers (ex. HSL_OUT_INT_000) are assigned MaxPak III drive variables as required. The corresponding registers in the HSL module (ex. REG 100) are assigned AutoMax DCS variables as required using IODEF statements in the AutoMax DCS. This creates a direct mapping between Maxpak III drive and AutoMax variables.

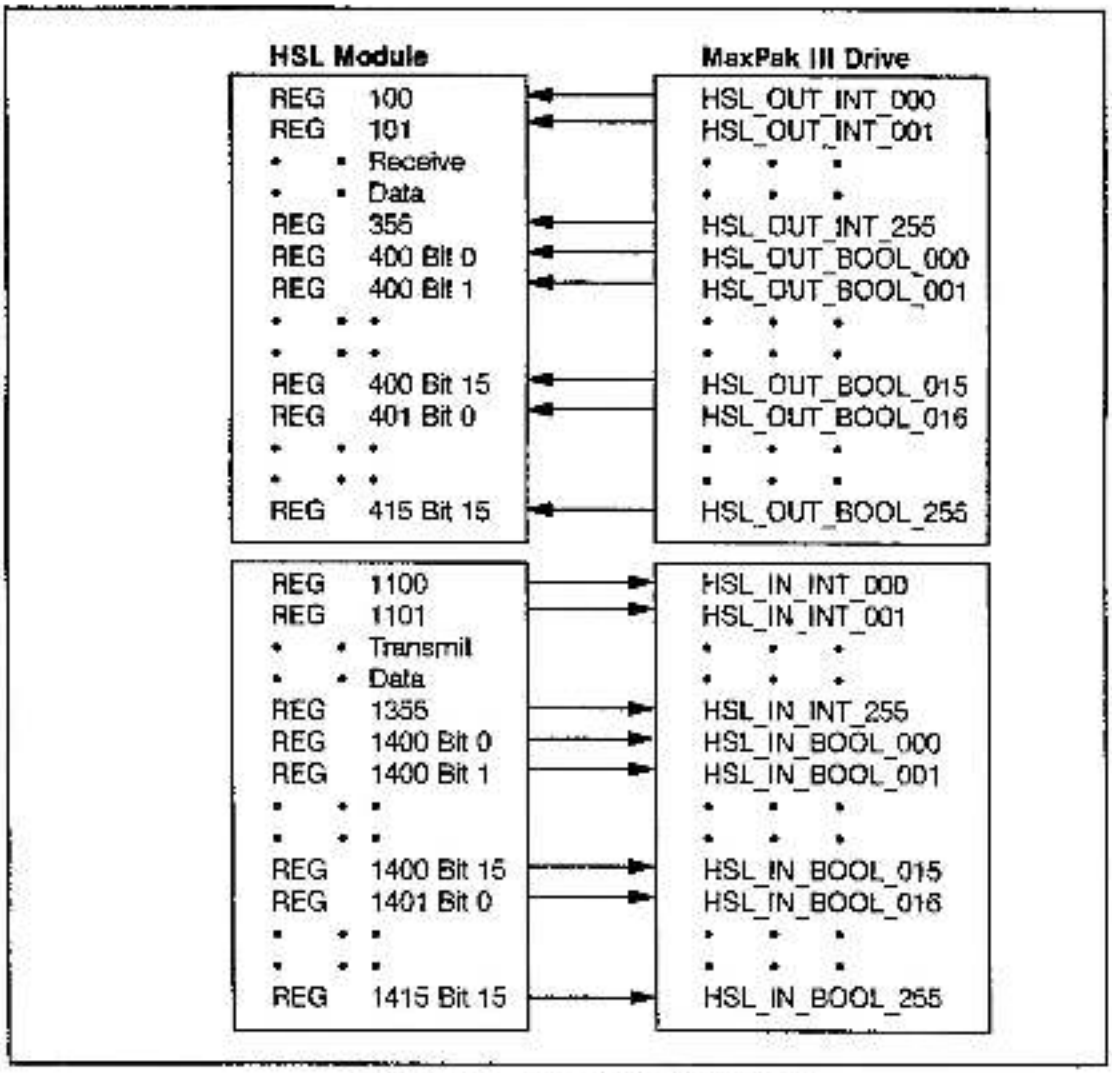


Figure 4.1 – HSL Configuration Map

4.3 Download Requirements

The MaxPak III drive download utility verifies that source configuration syntax conforms to the following HSL syntax requirements as each statement is read during the verify stage.

4.3.1 Statement Syntax

<line number> SPACE <HSL register name> SPACE = SPACE <MP3 variable name>

Where: SPACE means 1 or more required space character or tab.

<line number>
- Decimal value from 1 to 65535 inclusive.
- Must begin in column 1.

<HSL register name>

'HSL_IN_INT_nnn'
Integer input registers
where: nnn = register number. Range: 0–255.
Leading zeros permitted but not required.

'HSL_OUT_INT_nnn'
Integer output registers
where: nnn = register number. Range: 0–255.
Leading zeros permitted but not required.

'HSL_IN_BOOL_nnn'
Boolean input registers
where: nnn = register number. Range: 0–255.
Leading zeros permitted but not required.

'HSL_OUT_BOOL_nnn'
Boolean output registers
where: nnn = register number. Range: 0–255.
Leading zeros permitted but not required.

<variable_name>
- Alphanumeric or ('_', '@', '%') < = 16 characters in length.
- First character must be an alpha character.
- Last character must be % or @ indicating integer or boolean variable type respectively. The variable type must match the HSL register type i.e. HSL_IN_INT_nnn and HSL_OUT_INT_nnn registers must be assigned to Integer variables. HSL_IN_BOOL_nnn and HSL_OUT_BOOL_nnn must be assigned to Boolean variables.

4.3.2 Statement Position Within Source Configuration File

HSL register assignments can occur anywhere in the MaxPak III drive configuration source; they can be grouped together or interspersed throughout the file.

HSL register assignments can occur in any order. That is, they do not have to appear in ascending or descending register order and can be mixed.

For Example, the following source lines are valid;

```
010  HSL_IN_INT_002 = L2_REFA%
015  ML_CMP_O_EXE_EN@ = ON
020  HSL_OUT_BOOL_001 = SEQ_ARM_ACTIVE@
030  HSL_IN_INT_000 = L2_REFB%
040  [L2_REFB%] = L2_REFB%
050  HSL_IN_BOOL_001 = FLT_RESET@
```

However, it is highly recommended that HSL register assignments be grouped together and in ascending numeric order. For example:

```
010  HSL_IN_INT_000 = L2_REFA%
011  HSL_IN_INT_001 = L2_REFB%
     •
     •
     remaining HSL_IN_INT assignments
     •
     •
020  HSL_OUT_INT_000 = CML_REF%
021  HSL_OUT_INT_001 = L2_REFC%
     •
     •
     remaining HSL_OUT_INT assignments
030  HSL_IN_BOOL_000 = FLT_RESET@
031  HSL_IN_BOOL_001 = MEM_RESTORE@
     •
     •
     remaining HSL_IN_BOOL assignments
     •
     •
040  HSL_OUT_BOOL_000 = SEQ_ARM_ACTIVE@
041  HSL_OUT_BOOL_001 = L2_REFA_ENABLE@
     •
     •
     remaining HSL_OUT_BOOL assignments
     •
     •
```

4.3.3 HSL Table Requirements

This outlines the interdependencies among HSL register assignments as a whole. HSL assignments are dependent on each other.

The following constraints apply:

A maximum of 1024 HSL register assignments is available. The maximum number of assignments for any one HSL type (integer input, integer output, boolean input, and boolean output) is 256.

HSL assignments of each of the four types must be contiguous (within each type) and start with register 0. The assignments can occur within the source file in any order but, having read them all, download will check for continuity from Register 0.

Duplicate variables name assignments within any 1 HSL type (same variable name) are not permitted. for example, the following is not permitted.

```
010 HSL_IN_INT_001 = L2_REFA%
    •
    •
050 HSL_IN_INT_009 = L2_REFA%
```

However, assigning the same input variable to both an input and an output register is permitted. For example, the following is permitted.

```
010  HSL_IN_INT_001 = L2_REFA%
020  HSL_OUT_INT_001 = L2_REFA%
030  HSL_IN_BOOL_005 = L2_REFA_ENABLE@
040  HSL_OUT_BOOL_005 = L2_REFA_ENABLE@
```

Duplicate register assignments within any 1 HSL type (same register) are not permitted. For example, the following is not permitted.

```
010 HSL_IN_INT_001 = L2_REFA%
      •
      •
050 HSL_IN_INT_001 = L2_REFB%
```

There are also requirements specific to the individual types. See Sections 4.3.5 through 4.3.8 for details.

Note: The MaxPak III drive download utility verifies that the source configuration conforms to HSL table requirements at the end of the verify stage and before the download stage. The exception being the detection of ONLINE (Access Level < 3) and OFFLINE (Access Level > 2) constraints; these can only be detected during the download stage.

Appendix G (Verify Errors) and Appendix H (Download Errors) list and describe the errors that could occur during verify and download.

Following the verify stage, an account of the HSL registers is displayed to the screen and to the log file (if specified). This summary appears as follows:

HIGH SPEED LINK (HSL) REGISTER ASSIGNMENT STATISTICS

Boolean Input	**section count (HSL_IN_BOOL_*) :nnn**
Boolean Output	**section count (HSL_OUT_BOOL_*) :nnn**
Integer Input	**section count (HSL_IN_INT_*) :nnn**
Integer Output	**section count (HSL_OUT_INT_*) :nnn**

Where nnn is the number of registers assigned in the section.

4.3.4 MaxPak III Drive Requirements

In addition to the Statement Syntax Requirements, Statement Position Within Source Configuration File, and HSL Table Requirements, the MaxPak III qualifies the HSL assignments according to the following requirements.

The variables assigned to the HSL registers must exist in the MaxPak III.

The variables assigned to HSL_IN_INT_000, HSL_IN_INT_001, and HSL_IN_BOOL_000 must be ONLINE (Access Level < 3).

Once the first OFFLINE (Access Level > 2) variable has been assigned, all remaining registers must be assigned to OFFLINE variables.

Only MaxPak III drive integer input variables can be assigned to HSL_IN_INT_ registers.

Only MaxPak III drive integer variables (input or output) can be assigned to HSL_OUT_INT_ registers.

Only MaxPak III drive boolean input variables can be assigned to HSL_IN_BOOL_ registers.

Only MaxPak III drive boolean variables (input or output) can be assigned to HSL_OUT_BOOL_ registers.

If one or more HSL register assignments are downloaded to the MaxPak III drive then the entire HSL register map in the MaxPak III drive is cleared and the new assignments (in the downloading configuration) are added.

4.3.5 Integer Input Registers

Specific download requirements for Integer Input Registers are:

- There must be at least 2 integer inputs; HSL_IN_INT_000 and HSL_IN_INT_001. If one or both of these registers are not used for the desired application, they must have valid values. COM_NULL_IN% may be assigned to either of these registers and/or a variable that will not be used in the application.
- These first two must be ONLINE (Access Level < 3) variables.
- The first OFFLINE (Access Level > 2) variable must be after the last ONLINE variable. This implies that once the first OFFLINE variable has been assigned to register nnn, then all remaining HSL registers (within the integer input type) nnn + 1 to 255 can only be assigned to OFFLINE variables.

HSL_IN_INT_000: This register receives its value from the HSL module REGISTER 1100. This is a high priority AutoMax DCS to MaxPak III data transfer value via HSL module REGISTER 1100. This register is normally used to receive CURRENT REFERENCE (CML_REF%).

HSL_IN_INT_001 - HSL_IN_INT_255: These registers receive their values from HSL module REGISTERS 1101 – 1355. They may be used to receive any user defined information from the AutoMax DCS. HSL_IN_INT_001 is received every time the HSL module gets a transmit request. All others may take several requests to be sent.

4.3.6 Boolean Input Registers

Specific download requirements for Boolean Input Registers are:

- There must be at least two boolean inputs; HSL_IN_BOOL_000 and HSL_IN_BOOL_001. If one or both of these registers are not used for the desired application, they must have valid values. COM_NULL_IN@ may be assigned to either of these registers and/or a variable that will not be used in the application.
- These first two must be ONLINE (Access Level < 3) variables.
- The first OFFLINE (Access Level > 2) variable must be after the last ONLINE variable. This implies that once the first OFFLINE variable has been assigned to register nnn, then all remaining HSL registers (within the boolean input type) nnn + 1 to 255 can only be assigned to OFFLINE variables.

HSL_IN_BOOL_000 - HSL_IN_BOOL_255: Contain the bit booleans from the AutoMax DCS via dual port REGISTERS 1400 to 1415. Each HSL module register received contains a bit packed word of 16 booleans. The MaxPak III drive will not use all the booleans received in the last register if the number of booleans to be sent to the MaxPak III drive (REGISTER 71) is not a multiple of 16. HSL_IN_BOOL_000 - 015 are received from the HSL module every message. All others may take several messages to receive.

4.3.7 Integer Output Registers

Specific download requirement is as follows:

- There must be at least two integer outputs; HSL_OUT_INT_000 and HSL_OUT_INT_001. If one or both of these registers are not used for the desired application, they must have valid values. COM_NULL_OUT% may be assigned to either of these registers and/or a variable that will not be used in the application.

HSL_OUT_INT_000 - 001: High priority data transfer values from MaxPak III drive to AutoMax DCS REGISTER 100-101. If speed feedback is connected directly to the drive, one of these registers would normally be used to transmit speed feedback (PT_SPEED_FDBK%) to the AutoMax DCS.

HSL_OUT_INT_002 - HSL_OUT_INT_255: Each variable represents one integer sent to the AutoMax DCS dual port REGISTERS, starting with REGISTER 102 and continuing to REGISTER 355. These registers may be used to send any user defined information from the MaxPak III drive to the AutoMax DCS.

4.3.8 Boolean Output Registers

Specific download requirement is as follows:

- There must be at least two Boolean outputs; HSL_OUT_BOOL_000 and HSL_OUT_BOOL_001. If one or both of these registers are not used for the desired application, they must have valid values. COM_NULL_OUT@ may be assigned to either of these registers and/or a variable that will not be used in the application.

HSL_OUT_BOOL_000 - HSL_OUT_BOOL_255: Each variable represents 1 bit in the bit packed word sent to the AutoMax DCS dual port REGISTERS 400 to 415. These bits may point to any pertinent boolean data in the MaxPak III drive that the user wants to send to the AutoMax DCS. HSL_OUT_BOOL_000 - HSL_OUT_BOOL_015 are guaranteed to be received by the HSL module every L2_SCAN_PERIOD%. The remaining booleans may take several messages to transmit.

4.4 Minimum System Configuration

The HSL module is configured using AutoMax DCS. A minimum system configuration is as follows: AutoMax DCS

AutoMax DCS

REGISTER 70 = 2 ->	# integers to transmit to MaxPak III drive
REGISTER 71 = 2 ->	# boolean bits to transmit to MaxPak III drive
REGISTER 75 = 2 ->	# integers to receive from the MaxPak III drive
REGISTER 76 = 2 ->	# boolean bits to receive from the MaxPak III drive
REGISTER 80 = (2 x MaxPak III L2_SCAN_PERIOD%) + 5 ->	Timeout (in msec) between MaxPak III drive messages
REGISTER 81 = 11 ->	# of ticks between update requests (e.g. speed loop tick interval). Note: 1 tick = 0.5 ms

MaxPak III Drive

HSL_TIMEOUT_T% =
(2 * DCS/AutoMax Scan Period) + 10

HSL_ENABLE@ = ON

4.5 High Speed Link Configuration Program Examples

Two High Speed Link configuration program examples are provided in this section. The first example uses the High Speed Link default configuration file (See Appendix A. If the drive was shipped from the factory without a custom ordered or Engineering Sales ordered configuration, the Version 6.1A software should contain a default HSL assignment register map in the MaxPak III drive's configuration file.) The second example uses a custom configuration file and the Multibus interrupt capability. Reference Appendix K for a summarized functional overview of the HSL.

Note: It is recommended to upload the configuration file and save to disk for reference prior to making changes to the existing configuration file. If overwritten, the existing configuration file will be lost.

4.5.1 Example #1

Problem:

Run 22 ms speed loop in the AutoMax DCS and transmit current reference to a MaxPak III (no field weakening). Speed feedback via resolver card in an Automax rack. Must also be able to tune CML and FLD from the Automax. Since speed loop will be run in the AutoMax, Loop 2 can be shut off (L2_EXECUTE_EN@ = OFF). The Machine Logic, Ratio Detector and Loop 3 can also be shut off (ML_EXECUTE_EN@, RD_EXECUTE_EN@, L3_EXECUTE_EN@) since they will not be used.

MaxPak III Drive Configuration:

Use the MaxPak III drive default HSL register map (Reference Appendix A in this manual). Configure the MaxPak III drive for:

HSL_ENABLE@ = ON
HSL_TIMEOUT_T% = 54 (2 * speed loop scan period) + 10
L2_EXECUTE_EN@ = OFF
ML_EXECUTE_EN@ = OFF
RD_EXECUTE_EN@ = OFF
L3_EXECUTE_EN@ = OFF
L2_SCAN_PERIOD% = 20
[L2_FEEDBACK%] = L2_FEEDBACK%

Note: In addition, each HSL integer or boolean having a corresponding node must be assigned to that node (e.g. [CML_REF%] = CML_REF%)

AutoMax Configuration:

The AutoMax must be able to control the following MaxPak III drive input integers:

CML_REF%	FLD_I_REF%
CML_ADAPT_GAIN%	FLD_ECON_REF%
CML_CC_THR%	FLD_ECON_DLY_T%
CML_FB_GAIN%	FLD_IREF_LIM_HI%
CML_PI_KP%	FLD_IREF_LIM_LO%
CML_PI_WLD%	FLD_I_FB_MUL%
CML_RATE_LIM%	FLD_I_PI_KP%
L2_FEEDBACK%	FLD_I_PI_WLD%

Since the last of these is mapped to input integer 032 (HSL_IN_INT_032), the AutoMax must control input integers 000 through 032, for a total of 33 input integers. This includes a block of integers from 002 (OUT_SI_0_ANA_0%) through 019 (ML_GAIN_1_IN%) with the exception of 005 (L2_FEEDBACK%) that will not be used. These integers are "don't care's", since we are not using L2, ML or the SI (Signal Interface) card. Note that L2_FEEDBACK% is essential so that the drive can detect tach loss and overspeed.

The AutoMax must also be able to control the following MaxPak III drive input booleans:

SEQ_DRIVE_EN@
SEQ_RUN@
FLT_RESET@

Since the last of these is mapped to input boolean 005 (HSL_IN_BOOL_005), the AutoMax must control input booleans 000 through 005, for a total of 6 input booleans. The 3 unused booleans (SEQ_JOG@, SEQ_STOP_ILIM@, and FLD_ECON_EN@) must all be turned OFF. Since this is the default state of HSL variables, the AutoMax will not be affected.

The AutoMax needs to monitor the following MaxPak III drive output integers:

CML_REF_SJ%
CML_ERROR_SJ%
CML_FB_SJ%
FLD_DELTA%
FLD_I_FB_SJ%

Since the last of these is integer output 008 (HSL_OUT_INT_008), the AutoMax must be configured for 9 output integers.

The AutoMax needs to monitor the following MaxPak III drive output booleans:

FLD_OK@
SEQ_ARM_PERM@
SEQ_FLD_PERM@
SEQ_RUNNING@
IN_RUNPERM@
FLT_DRIVE_FAULT@

Since the last of these is boolean output 023 (HSL_OUT_BOOL_023), the AutoMax must be configured for 24 output booleans.

Configuration Task

A minimum configuration task for this example is listed in paragraphs to follow. The minimum configuration task is written under the assumption that the processor card resides in Slot 0 and the HSL module in Slot 2. Note that this listing shows only those definitions relating to the HSL; a real configuration task will require other definitions as well.

Note that in line 1150, the MaxPak III drive variable FLT_DRIVE_FAULT@ is IODEF'd as DRIVE_FAULT@. This name change facilitates the use of this variable in ladder tasks, which do not support the longer name.

The application tasks listed herein are described later in this section.

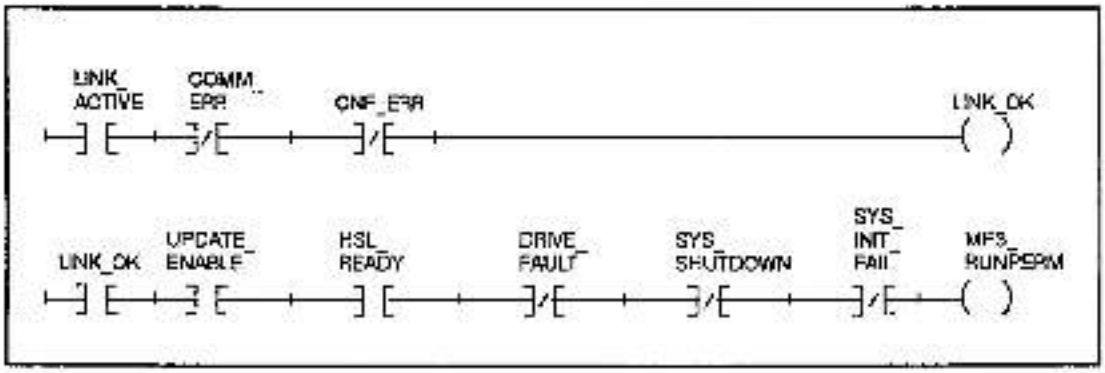
```
10   TASK !LOCK[   TYPE = PC,       PRIORITY =  6, SLOT = 0] :! interlock logic
20   TAsK SPEED[   TYPE = CONTROL,  PRIORITY =  4, SLOT = 0] :! speed loop
30   TASK HSL_RUN[ TYPE = BASIC,    PRIORITY =  5, SLOT = 0] :! link restart task
40   TASK HSL_INIT[ TYPE = BASIC,   PRIORITY = 10, SLOT = 0] :! initialization tasK
```

```
99    ! command/status registers:
100   IODEF LINK_ACTIVE@[          SLOT=2, REGISTER= 4, BIT= 0]
110   IODEF COMM_ERRS%[            SLOT=2, REGISTER=26]
120   IODEF COMM_ERR@[             SLOT=2, REGISTER=26, BIT=15]
130   IODEF CNF_ERRS%[             SLOT=2, REGISTER=27]
140   IODEF CNF_ERR@[              SLOT=2, REGISTER=27, BIT=15]
150   IODEF MP3_STATUS%[           SLOT=2, REGISTER=28]
160   IODEF SYS_INIT_FAIL@[        SLOT=2, REGISTER=28, BIT= 5]
170   IODEF SYS_SHUTDOWN@[         SLOT=2, REGISTER=28, BIT= 6]
180   IODEF TX_ACTIVE@[            SLOT=2, REGISTER=29, BIT= 0]
190   IODEF CMD_REG%[              SLOT=2, REGISTER=30]
200   IODEF COMM_RESET@[           SLOT=2, REGISTER=31, BIT= 0]
210   IODEF INPUT_INTS%[           SLOT=2, REGISTER=70]
220   IODEF INPUT_BOOLS%[          SLOT=2, REGISTER=71]
230   IODEF OUTPUT_INTS%[          SLOT=2, REGISTER=75]
240   IODEF OUTPUT_BOOLS%[         SLOT=2, REGISTER=76]
250   IODEF TIMEOUT%[              SLOT=2, REGISTER=80]
260   IODEF TICKS%[                SLOT=2, REGISTER=81]

399   ! input integers (to mp3):
400   IODEF CML_REF%[              SLOT=2, REGISTER=1100]
410   IODEF FLD_I_REF%[            SLOT=2, REGISTER=1101]
420   IODEF L2_FEEDBACK%[          SLOT=2, REGISTER=1105]
430   IODEF CML_ADAPT_GAIN%[       SLOT=2, REGISTER=1120]
440   IODEF CML_CC_THR%[           SLOT=2, REGISTER=1121]
450   IODEF CML_FB_GAIN%[          SLOT=2, REGISTER=1122]
460   IODEF CML_PI_KP%[            SLOT=2, REGISTER=1123]
470   IODEF CML_PI_WLD%[           SLOT=2, REGISTER=1124]
480   IODEF CML_RATE_LIM%[         SLOT=2, REGISTER=1125]
490   IODEF FLD_ECON_REF%[         SLOT=2, REGISTER=1126]
500   IODEF FLD_ECON_DLY_T%[       SLOT=2, REGISTER=1127]
510   IODEF FLD_IREF_LIM_HI%[      SLOT=2, REGISTER=1128]
520   IODEF FLD_IREF_LIM_LO%[      SLOT=2, REGISTER=1129]
530   IODEF FLD_I_FB_MUL%[         SLOT=2, REGISTER=1130]
540   IODEF FLD_I_PI_KP%[          SLOT=2, REGISTER=1131]
550   IODEF FLD_I_PI_WLD%[         SLOT=2, REGISTER=1132]

699   ! input booleans (to mp3):
700   IODEF SEQ_DRIVE_EN@[         SLOT=2, REGISTER=1400, BIT= 0]
710   IODEF SEQ_RUN@[              SLOT=2, REGISTER=1400, BIT= 2]
720   IODEF FLT_RESET@[            SLOT=2, REGISTER=1400, BIT= 5]

899   ! output integers (from mp3):
900   IODEF CML_REF_SJ%[           SLOT=2, REGISTER= 102]
910   IODEF CML_ERROR_SJ%[         SLOT=2, REGISTER= 103]
920   IODEF CML_FB_SJ%[            SLOT=2, REGISTER= 104]
930   IODEF FLD_DELTA%[            SLOT=2, REGISTER= 107]
940   IODEF FLD_I_FB_SJ%[          SLOT=2, REGISTER= 108]

1099  ! output booleans (from mp3):
1100  IODEF FLD_OK@[               SLOT=2, REGISTER= 400, BIT= 2]
1110  IODEF SEQ_ARM_PERM@[         SLOT=2, REGISTER= 400, BIT= 9]
1120  IODEF SEQ_FLD_PERM@[         SLOT=2, REGISTER= 400, BIT=10]
1130  IODEF SEQ_RUNNING@[          SLOT=2, REGISTER= 400, BIT=13]
1140  IODEF IN_RUNPERM@[           SLOT=2, REGISTER= 401, BIT= 5]
1150  IODEF DRIVE_FAULT@[          SLOT=2, REGISTER=401, BIT= 7] !FLT_DRIVE_FAULT@
```

```
1999 ! Miscellaneous
2000 MEMDEF UPDATE_ENABLE@     :! speed loop updates enabled status
2010 MEMDEF INITIALIZED@       :! hsl input variable initialization status
2020 MEMDEF LINK_OK@           :! hsl link operational status
2030 MEMDEF HSL_RESET@         :! user link reset request
2040 MEMDEF HSL_READY@         :! link ready for real-time data transfer
2050 MEMDEF MP3_RUNPERM@       :! drive run permissive status
2060 MEMDEF DRVRR@             :! drive run relay
2070 MEMDEF RUN@               :! user run request
```

INTERLOCK LOGIC (TASK ILOCK.PC):

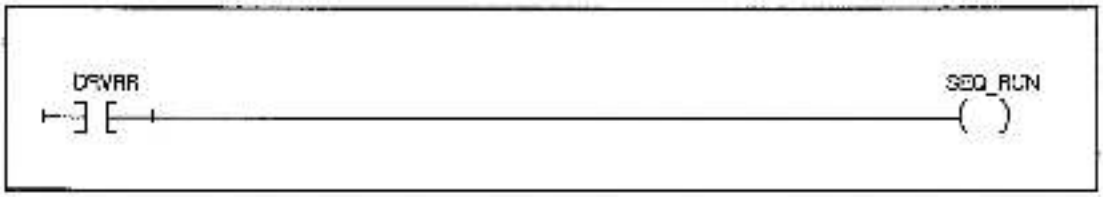
The PC task which controls interlocking should include the following sequences. (The sequences shown here are universal and may be used for any HSL application assuming that the appropriate definitions have been included in the configuration task as illustrated previously).



LINK_OK@ will be active whenever the physical HSL link is active, a valid configuration is in place and all latched errors are cleared. This boolean is used by the speed loop and the link restart task.

MP3_RUNPERM@ should be included in the run permissive string for the MaxPak III drive. For example:

DRVRR will typically be used to enable execution of the speed loop blocks (except for the update request block as described under SPEED LOOP later in this section) and to enable the MaxPak III drive sequence run input (SEQ_RUN@) as shown below:



SPEED LOOP (TASK SPEED.BLK):

The speed loop task must start every 22 milliseconds (Forty-four 0.5 millisecond ticks) and generate both CML_REF% and speed feedback (L2_FEEDBACK%). The latter are calculated based on data read from the resolver. The speed loop must also execute a block which causes an update request (decimal 128) to be written to CMD_REG% (command register 30), provided UPDATE_ENABLE@ has been asserted by the HSL_RUN task and LINK_OK@ has been asserted by the interlock logic. This can be accomplished in two blocks as shown (assume that UPDATE@ has been defined as a local variable):

```
•
•
•
2000 CALL AND(      INPUT1      = LINK_OK@,          &
                    INPUT2      = UPDATE_ENABLE@,    &
                    OUTPUT      = UPDATE@)

2010 CALL SWITCH(   INPUT1      = 128,               &
                    INPUT2      = CMD_REG%,          &
                    SELECT      = UPDATE@,           &
                    OUTPUT      = CMD_REG%)
•
•
•
```

To insure that the update request is written to the command register only after CML_REF% and L2_FEEDBACK% have been updated, the CMD_REG% MUST be the last register written. This is accomplished by making CMD_REG% the last defined COMMON as follows:

```
•
•
•
100  COMMON UPDATE_ENABLE@
•
•
•
210  COMMON LINK_OK@
220  COMMON CMD_REG%              ! must be last defined common

230  LOCAL UPDATE@
•
•
•
```

Finally, on completion the speed loop must start the link restart task (HSL_RUN), which should be synchronized to it for proper transmission timing. This is accomplished by defining an event indicating that the speed loop is done and setting that event just prior to exit as shown:

```
•
•
•
500  EVENT NAME = SL_DONE              ! speed loop done
•
•
•
4000 SET SL_DONE
4010 END
```

LINK RESTART (TASK HSL_RUN.BAS):

This task is responsible for preparing the HSL link and the MaxPak III drive for real-time data transfer. This will happen after a Start All and each time the common variable HSL_RESET@ is asserted. HSL_RESET@ will remain ON while the task executes then will automatically be shut OFF on completion (approximately 5 to 8 seconds after start). If the task has executed successfully, the common variable HSL_READY@ will be set ON at this time, otherwise it will remain OFF. HSL_READY@ is used by the interlock logic to generate MP3_RUNPERM@, the MaxPak III drive run permissive indicator.

Early in its execution, this task resets the variable INITIALIZED@ to OFF, which in turn causes the initialization task (HSL_INIT) to set all HSL input variables to their appropriate values. When the initialization task successfully completes, it sets INITIALIZED@ back ON. The link restart task checks INITIALIZED@ later in execution prior to enabling update requests. If INITIALIZED@ is still OFF, the process is aborted and update requests remain disabled.

WARNING

ALL HSL INPUT VARIABLES MUST BE INITIALIZED TO APPROPRIATE VALUES PRIOR TO SENDING UPDATE MESSAGES. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

The task is listed below. It may be used in any HSL application which includes configuration, speed loop, initialization and interlock tasks adhering to the guidelines detailed throughout this discussion. The only application-dependent modifications required are to the dual-port configuration register values (registers 70-81) in lines 2000 to 2050. NOTE, HOWEVER, THAT THIS TASK DOES NOT TAKE THE DRIVE OFFLINE AND THEREFORE DOES NOT ALLOW THE TRANSMISSION OF ANY OFFLINE VARIABLES. (If for any reason it should become necessary to initialize offline variables, the procedure in Appendix J can be followed, regardless of whether HSL_RUN.BAS is installed or not.)

```
00099! commons:
00100  COMMON LINK_OK@, LINK_ACTIVE@, COMM_ERR@, CNF_ERR@
00110  COMMON CMD_REG%, TX_ACTIVE@, COMM_RESET@
00120  COMMON INPUT_INTS%, INPUT_BOOLS%
00125  COMMON OUTPUT_INTS%, OUTPUT_BOOLS%
00130  COMMON TICKS%, TIMEOUT%
00140  COMMON UPDATE_ENABLE@, INITIALIZED@
00150  COMMON HSL_RESET@, HSL_READY@
00160  COMMON FLT_RESET@, DRIVE_FAULT@

00500  EVENT NAME = SL_DONE   :! speed loop done indicator

00799  ! set HSL_RESET@ ON after "Start All" only
00800  HSL_RESET@ = ON

00999  ! program loop: wait for HSL_RESET@ to go ON:
01000  WAIT ON SL_DONE
01010  IF NOT HSL_RESET@ THEN 1000
```

```
01099  ! reset requested - shut off all outputs:
01100  UPDATE_ENABLE@ = OFF   :! stop update message transmissions
01110  HSL_READY@ = OFF
01120  INITIALIZED@ = OFF     :! start initialization task
01130  WAIT ON SL_DONE        :! allow last update message to        &
                                 complete (if present)

01199  ! wait for link to go active:
01200  IF NOT LINK_ACTIVE@ THEN DELAY 10 TICKS: GOTO 1200

01299  ! clear comm errors if they exist;
01300  IF NOT COMM_ERR@ THEN 2000
01310  COMM_RESET@ = OFF      :! insure that up-transition is detected
01320  DELAY 10 TICKS
01330  COMM_RESET@ = ON
01340  DELAY 100 TICKS        :! allow CMD_REG% to clear if request &
                                 was latched

01350  COMM_RESET@ = OFF
01360  IF COMM_ERR@ THEN 5000 :! if comm error remains, abort

01999  ! configure link:
02000  INPUT_INTS% = 33       :! the values in lines 2000 - 2050
02010  INPUT_BOOLS% = 6       :! will vary from one application
02020  OUTPUT_INTS% = 9       :! to the next.
02030  OUTPUT_BOOLS% = 24
02040  TICKS% = 44            :! 44 * 0.5 ms = 22 ms scan loop
02050  TIMEOUT% = 45          :! (2 * L2_SCAN_PERIOD%) + 5

02060  CMD_REG% = 4           :! issue configure request
02070  DELAY 100 TICKS        :! allow configure to complete

02999  ! abort if link failure or initialization failure:
03000  IF NOT LINK_OK@ THEN 5000
03010  IF NOT INITIALIZED@ THEN 5000

03099  ! send all hsl-mapped variables to the MaxPak III:
03100  UPDATE_ENABLE@ = ON    :! start speed loop sending updates
03110  FLT_RESET@ = OFF       :! make sure FLT_RESET@ is OFF
03120  DELAY 3 SECONDS        :! allow time for all variables        &
                                 to reach MaxPak III

03130  IF NOT LINK_OK@ THEN 5000

03199  ! reset drive fault if present:
03200  IF NOT DRIVE_FAULT@ THEN 4000
03210  FLT_RESET@ = ON
03220  DELAY 3 SECONDS        :! allow time for FLT_RESET@ to        &
                                 reach MaxPak III

03230  FLT_RESET@ = OFF
03240  IF NOT LINK_OK@ THEN 5000

03999  ! successful - enable HSL:
04000  HSL_READY@ = ON

04999  ! clear HSL_RESET@ and repeat:
05000  HSL_RESET@ = OFF
05010  GOTO 1000

09999  END
```

INITIALIZATION TASK (TASK HSL_INIT.BAS):

The initialization task writes appropriate values to all HSL input variables. It does this each time the link restart task resets the variable INITIALIZED@ to OFF. On completion, the initialization task sets INITIALIZED@ back ON.

WARNING

ALL HSL INPUT VARIABLES MUST BE INITIALIZED TO APPROPRIATE VALUES PRIOR TO SENDING UPDATE MESSAGES. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

```
100     COMMON INITIALIZED@
109     ! hsl input variables:
110     COMMON CML_REF%, FLD_I_REF%, L2_FEEDBACK%
120     COMMON CML_ADAPT_GAIN%, CML_CC_THR%, CML_FB_GAIN%
130     COMMON CML_PI_KP%, CML_PI_WLD%, CML_RATE_LIM%
140     COMMON FLD_ECON_REF%, FLD_ECON_DLY_T%
150     COMMON FLD_IREF_LIM_HI%, FLD_IREF_LIM_LO%
160     COMMON FLD_I_FB_MUL%, FLD_I_PI_KP%, FLD_I_PI_WLD%
170     COMMON SEQ_DRIVE_EN@, FLT_RESET@, SEQ_RUN@

0999    ! wait for INITIALIZED@ to go OFF:
1000    IF INITIALIZED@ THEN DELAY 10 TICKS: GOTO 1000

1099    ! initialize all hsl input variables: 1100CML_REF% = 0
*       *
*       *
1280    SEQ_RUN@ = OFF

1999    ! set INITIALIZED@ ON and repeat:
2000    INITIALIZED@ = ON
2010    GOTO 1000

9999    END
```

Determining AutoMax DCS to MaxPak III Drive Information Update Rate:

At 44 ticks of 0.5 milliseconds, the update rate for fixed integers and booleans being sent to the MaxPak III drive is 44 x 0.5 = 22 milliseconds. Since at 44 ticks there are 18 fixed integer registers and 3 fixed boolean registers (48 boolean variables), the input integers up to ML_GAIN_0_IN% (HSL_IN_INT_017) are fixed as are all booleans. Thus all of these variables will be updated every 22 milliseconds.

The "high priority" register (HSL_IN_INT_000, CML_REF%) is a special case in that the MaxPak III drive will use the value of this register immediately upon receipt without waiting to receive the remaining fixed registers.

The information update interval for the remaining integers can be determined as follows:

From Table 2-1:

MFI = 18
MFB = 3
MRPM = 28

TI = 33 (Register 75 = 33)
TB = 6/16 = 1 (rounded up) (Register 76 = 6)

FIPM = MFI = 18 (MFI < TI)
FBPM = TB = 1 (TB < MFB)
MuRPM = 28-18-1 = 9 (9<31)
IMPC = (33-18)/(9) = 15/9 = 2 (rounded up)
BMPC = (1-1)/(9) = 0

MPC = 2 + 0 = 2

Information Update Interval = 2 * 44 * 0.5 = 44 milliseconds

Thus the input integers from L2_REFA% through FLD_I_PI_WLD% will be updated every 44 milliseconds.

Determining MaxPak III Drive to AutoMax DCS Information Update Rate:

At L2_SCAN_PERIOD% = 20, there are 16 fixed integer registers and 3 fixed boolean registers (48 boolean variables). Since the total number of output integers is only 9 and the total number of output booleans is only 24, all booleans and integers will be fixed (there will be no multiplexing). Thus, all output integers and booleans will be updated every L2_SCAN_PERIOD% (20 milliseconds).

Since Multibus interrupts are not enabled, there will be no functional difference between the "high priority" registers and the other fixed registers (see Example #2 in this manual for a Multibus driven application).

4.5.2 Example #2

Problem:

Same as example 1, but this time use the multibus interrupt capability (speed feedback directly to the drive via pulse tach) and a custom HSL register map.

Since the speed loop will be triggered off the multibus interrupt, CML_REF% will be sent to the drive each time an interrupt occurs. This time interval is determined by the L2_SCAN_PERIOD% of the drive, resulting in a speed loop scan period of 20 milliseconds.

MaxPak III Drive Configuration:

Configure the MaxPak III drive for:

HSL_ENABLE@ = ON
HSL_TIMEOUT_T% = 50 [(2 * speed loop scan period) + 10]
L2_EXECUTE_EN@ = OFF
ML_EXECUTE_EN@ = OFF
RD_EXECUTE_EN@ = OFF
L3_EXECUTE_EN@ = OFF
L2_SCAN_PERIOD% = 20
[L2_FEEDBACK%] = PT_SPEED_FB%

```
HSL_IN_INT_000      = CML_REF%
HSL_IN_INT_001      = FLD_I_REF%
HSL_IN_INT_002      = CML_ADAPT_GAIN%
HSL_IN_INT_003      = CML_CC_THR%
HSL_IN_INT_004      = CML_FB_GAIN%
HSL_IN_INT_005      = CML_PI_KP%
HSL_IN_INT_006      = CML_PI_WLD%
HSL_IN_INT_007      = CML_RATE_LIM%
HSL_IN_INT_008      = FLD_ECON_REF%
HSL_IN_INT_009      = FLD_ECON_DLY_T%
HSL_IN_INT_010      = FLD_IREF_LIM_HI%
HSL_IN_INT_011      = FLD_IREF_LIM_LO%
HSL_IN_INT_012      = FLD_I_FB_MUL%
HSL_IN_INT_013      = FLD_I_PI_KP%
HSL_IN_INT_014      = FLD_I_PI_WLD%

HSL_IN_BOOL_000     = SEQ_DRIVE_EN@
HSL_IN_BOOL_001     = SEQ_RUN@
HSL_IN_BOOL_002     = FLT_RESET@

HSL_OUT_INT_000     = PT_SPEED_FB%
HSL_OUT_INT_001     = CML_REF_SJ%
HSL_OUT_INT_002     = CML_ERROR_SJ%
HSL_OUT_INT_003     = CML_FB_SJ%
HSL_OUT_INT_004     = FLD_DELTA%
HSL_OUT_INT_005     = FLD_I_FB_SJ%

HSL_OUT_BOOL_000 = FLD_OK@
HSL_OUT_BOOL_001 = SEQ_ARM_PERM@
HSL_OUT_BOOL_002 = SEQ_FLD_PERM@
HSL_OUT_BOOL_003 = SEQ_RUNNING@
HSL_OUT_BOOL_004 = IN_RUNPERM@
HSL_OUT_BOOL_005 = FLT_DRIVE_FAULT@
```

Note: In addition, each HSL integer or boolean having a corresponding node must be assigned to that node (e.g. [CML_REF%] = CML_REF%)

Note that the HSL_TIMEOUT_T% has been changed to reflect the new interrupt-based speed loop scan period. The only other differences from the previous example, other than the addition of custom HSL definitions, is the mapping of [L2_FEEDBACK%] to PT_SPEED_FB% (speed feedback now comes directly from the pulse tach) and the assignment of PT_SPEED_FB% to an HSL "high-priority" output integer (to be used by the AutoMax speed loop).

Configuration Task:

A minimum configuration task for this example is listed below. The task is written under the assumption that the processor card resides in Slot 0 and the HSL module in Slot 2. Note that this listing shows only those definitions relating to the HSL; a real configuration task will require other definitions as well.

Note that in line 1150, the MaxPak III drive variable FLT_DRIVE_FAULT@ is IODEF'd as DRIVE_FAULT@. This name change facilitates the use of this variable in ladder tasks, which do not support the longer name.

The application tasks listed herein are described later in this section.

```
10   TASK ILOCK[    TYPE = PC,         PRIORITY = 6, SLOT = 0]  :! interlock logic
20   TASK SPEED[    TYPE = CONTROL,    PRIORITY = 4, SLOT = 0]  :! speed loop
30   TASK HSL_RUN[  TYPE = BASIC,      PRIORITY = 5, SLOT = 0]  :! link restart task
40   TASK HSL_INIT[ TYPE = BASIC,      PRIORITY = 10, SLOT = 0] :! initialization task

99   ! command/status registers:
100  IODEF HSL_ISCR%[                  SLOT = 2, REGISTER = 0]
110  IODEF LINK_ACTIVE@[               SLOT = 2, REGISTER = 4, BIT = 0]
120  IODEF COMM_ERRS%[                 SLOT = 2, REGISTER = 26]
130  IODEF COMM_ERR@[                  SLOT = 2, REGISTER = 26, BIT = 15]
140  IODEF CNF_ERRS%[                  SLOT = 2, REGISTER = 27]
150  IODEF CNF_ERR@[                   SLOT = 2, REGISTER = 27, BIT = 15]
160  IODEF MP3_STATUS%[                SLOT = 2, REGISTER = 28]
170  IODEF SYS_INIT_FAIL@[             SLOT = 2, REGISTER = 28, BIT = 5]
180  IODEF SYS_SHUTDOWN@[              SLOT = 2, REGISTER = 28, BIT = 6]
190  IODEF TX_ACTIVE@[                 SLOT = 2, REGISTER = 29, BIT = 0]
200  IODEF CMD_REG%[                   SLOT = 2, REGISTER = 30]
210  IODEF COMM_RESET@[                SLOT = 2, REGISTER = 31, BIT = 0]
220  IODEF INPUT_INTS%[                SLOT = 2, REGISTER = 70]
230  IODEF INPUT_BOOLS%[               SLOT = 2, REGISTER = 71]
240  IODEF OUTPUT_INTS%[               SLOT = 2, REGISTER = 75]
250  IODEF OUTPUT_BOOLS%[              SLOT = 2, REGISTER = 76]
260  IODEF TIMEOUT%[                   SLOT = 2, REGISTER = 80]
270  IODEF TICKS%[                     SLOT = 2, REGISTER = 81]

399  ! input integers (to mp3):
400  IODEF CML_REF%[                   SLOT = 2, REGISTER = 1100]
410  IODEF FLD_I_REF%[                 SLOT = 2, REGISTER = 1101]
420  IODEF CML_ADAPT_GAIN%[            SLOT = 2, REGISTER = 1102]
430  IODEF CML_CC_THR%[                SLOT = 2, REGISTER = 1103]
440  IODEF CML_FB_GAIN%[               SLOT = 2, REGISTER = 1104]
450  IODEF CML_PI_KP%[                 SLOT = 2, REGISTER = 1105]
460  IODEF CML_PI_WLD%[                SLOT = 2, REGISTER = 1106]
470  IODEF CML_RATE_LIM%[              SLOT = 2, REGISTER = 1107]
480  IODEF FLD_ECON_REF%[              SLOT = 2, REGISTER = 1108]
490  IODEF FLD_ECON_DLY_T%[            SLOT = 2, REGISTER = 1109]
500  IODEF FLD_IREF_LIM_HI%[           SLOT = 2, REGISTER = 1110]
510  IODEF FLD_IREF_LIM_LO%[           SLOT = 2, REGISTER = 1111]
520  IODEF FLD_I_FB_MUL%[              SLOT = 2, REGISTER = 1112]
530  IODEF FLD_I_PI_KP%[               SLOT = 2, REGISTER = 1113]
540  IODEF FLD_I_PI_WLD%[              SLOT = 2, REGISTER = 1114]

699  ! input booleans (to mp3):
700  IODEF SEQ_DRIVE_EN@[              SLOT = 2, REGISTER = 1400, BIT = 0]
710  IODEF SEQ_RUN@[                   SLOT = 2, REGISTER = 1400, BIT = 1]
720  IODEF FLT_RESET@[                 SLOT = 2, REGISTER = 1400, BIT = 2]

899  ! output integers (from mp3):
900  IODEF PT_SPEED_FB%[               SLOT = 2, REGISTER = 100]
910  IODEF CML_REF_SJ%[                SLOT = 2, REGISTER = 101]
920  IODEF CML_ERROR_SJ%[              SLOT = 2, REGISTER = 102]
930  IODEF CML_FB_SJ%[                 SLOT = 2, REGISTER = 103]
940  IODEF FLD_DELTA%[                 SLOT = 2, REGISTER = 104]
950  IODEF FLD_I_FB_SJ%[               SLOT = 2, REGISTER = 105]

1099 !output booleans (from mp3):
```

```
1100 IODEF FLD_OK@[              SLOT=2, REGISTER= 400, BIT= 0]
1110 IODEF SEQ_ARM_PERM@[        SLOT=2, REGISTER= 400, BIT= 1]
1120 IODEF SEQ_FLD_PERM@[        SLOT=2, REGISTER= 400, BIT= 2]
1130 IODEF SEQ_RUNNING@[         SLOT=2, REGISTER= 400, BIT= 3]
1140 IODEF IN_RUNPERM@[          SLOT=2, REGISTER= 400, BIT= 4]
1150 IODEF DRIVE_FAULT@[         SLOT=2, REGISTER= 400, BIT= 5] ! FLT_DRIVE_FAULT@

1999 ! Miscellaneous
2000 MEMDEF UPDATE_ENABLE@       ! speed loop updates enabled status
2010 MEMDEF INITIALIZED@         ! hsl input variable initialization status
2020 MEMDEF LINK_OK@             ! hsl link operational status
2030 MEMDEF HSL_RESET@           ! user link reset request
2040 MEMDEF HSL_READY@           ! link ready for real-time data transfer
2050 MEMDEF MP3_RUNPERM@         ! drive run permissive status
2060 MEMDEF DRVRR@               ! drive run relay
2070 MEMDEF RUN@                 ! user run request
```

Note that this configuration task deletes L2_FEEDBACK% as an input integer and adds PT_SPEED_FB% as an output integer. Register numbers and bit numbers have also been changed to correspond to the new MaxPak III drive register map. In addition, the variable HSL_ISCR% has been added and assigned to Register 0 of the HSL module. This is the interrupt status and control register via which the HSL module will interrupt the AutoMax processor.

INTERLOCK LOGIC (TASK ILOCK.PC):

The interlock logic will be identical to that of the first example.

SPEED LOOP (TASK SPEED.BLK):

The speed loop for this configuration is identical to that of the previous example except that speed feedback is taken from the variable PT_SPEED_FB% as read from the HSL module (as opposed to the previous example where speed feedback was calculated from resolver input and written to the HSL module). Also the speed loop is driven off the interrupt from the HSL module. This is accomplished by defining a hardware event (called START_TASK in this example) and using it to trigger the scan loop as follows:

```
01000  EVENT NAME = START_TASK, INTERRUPT_STATUS=HSL_ISCR%,  &
                    TIMEOUT=DISABLED
*
*
*
02000  CALL SCAN LOOP( TICKS=4, EVENT=START_TASK )
*
*
*
```

Note that hardware timeout detection is disabled since timeouts are automatically detected in software by the HSL protocol. Also note that the SCAN LOOP block must still define a tick interval, which should be chosen to be as close as possible to the MaxPak III drive L2_SCAN_PERIOD%.

LINK RESTART (TASK HSL_RUN.BAS):

The link restart task is identical to that of the previous except for the configuration register values in lines 2000 - 2050:

```
02000 INPUT_INTS% = 15
02010 INPUT_BOOLS% = 3
02020 OUTPUT_INTS% = 6
02030 OUTPUT_BOOLS% = 6
02040 TICKS% = 40                    :! 40 * 0.5 ms = 20 ms scan loop
02050 TIMEOUT% = 45                  :! (2 * L2_SCAN_PERIOD%) + 5
```

INITIALIZATION (TASK HSL_INIT.BAS):

The initialization task for this example is identical to that of the previous except that L2_FEEDBACK%, which is no longer an HSL input variable, is omitted.

Determining AutoMax DCS to MaxPak III Drive Information Update Rate:

By defining a custom HSL register map in the MaxPak III drive, the number of input integers has been reduced to 15 and input booleans to 3. However, since the TICKS count written to Register 81 has decreased to 40, the number of registers sent per scan has also been reduced. At 40 ms, up to 9 fixed integer registers and 3 fixed boolean registers (48 boolean variables) are allowed. This means that the first 9 integer registers (CML_REF% through FLD_ECON_REF%) and all boolean variables will be sent every 20 milliseconds (since this is the period of the hardware event driving the speed loop). The "high priority" register (HSL_IN_INT_000) is a special case in that the MaxPak III drive will use this register immediately upon receipt without waiting to receive the remaining fixed registers. The information update interval for the remaining 4 integer registers is calculated as follows:

From Table 2-1:

MFI = 9
MFB = 3
MRPM = 17

TI = 15 (Register 75 = 15)
TB = 3/16 = 1 (rounded up) (Register 76 = 3)

FIPM = MFI = 9 (MFI < TI)
FBPM = TB = 1 (TB < MFB)
MuRPM = 17-9-1 = 7 (7 < 31)
IMPC = (15-9)/(7) = 6/7 = 1 (rounded up)
BMPC = (1-1)/(7) = 0

MPC = 1 + 0 = 1

Information Update Interval = 1 * 40 * 0.5 = 20 milliseconds

Thus, the input integers from FLD_ECON_DLY_T% through FLD_I_PI_WLD% will be updated every 20 milliseconds. Note that in this case, the update interval for multiplexed integers has been reduced to equal that of fixed integers. This demonstrates a potential advantage of using a custom HSL register map over the default map: it often will reduce the information update interval of multiplexed variables.

Determining MaxPak II Drive to AutoMax DCS Information Update Rate:

As in the case of input variables, use of the custom HSL register map has also reduced the number of output variables. Therefore, since there were no multiplexed output registers in the previous example, there will be none in this case. Thus, all output integers and booleans will be updated every L2_SCAN_PERIOD% (20 milliseconds).

However, since Multibus interrupts are enabled in this example, the "high priority" registers (HSL_OUT_INT_000 and HSL_OUT_INT_001, PT_SPEED_FB% and CML_REF_SJ%) have special significance. They will cause a Multibus interrupt to be generated immediately upon their receipt. This allows these registers to be promptly processed by the speed loop without having to wait for the remaining fixed registers to be received.

5.0 MESSAGE COMMUNICATION ERRORS

There are four possible message communication errors, all of which the HSL module will recognize and take appropriate action on. They are: message verification errors, transmit overlap errors, receive overlap errors, timeout errors, message not received errors, and invalid operation errors. The first four of these error conditions are also recognized by the MaxPak III drive.

5.1 Message Verification Errors

A message verification error is detected if either of the two separate checksums do not verify. Also, the transceiver hardware on each end of the link will verify that there are no framing, overrun, or parity errors that occurred within each message. If either software or hardware errors are detected by either side of the link, the message will be marked as invalid and ignored.

5.2 Transmit Overlap Errors

A communication OVERLAP occurs when the transmitting device attempts to send a new message prior to completion of the previous message transfer.

5.2.1 AutoMax DCS to MaxPak III Drive Transmit Overlap

Overlap is detected by the HSL module when the AutoMax processor board sets dual port REGISTER 30 before it is finished transmitting the preceding message. If this occurs, the following will result:

a) The OVERLAP error bit flag will be set in dual port REGISTER 26 bit 1.

b) The message currently being transmitted is halted and no more transmissions may commence. This will eventually cause the MaxPak III drive to exceed its timeout period and perform a fault stop sequence to stop the armature.

This is a latched fault condition. To clear this fault condition:

a) The communication error reset, REGISTER 31, must be asserted for a minimum time period of 50 milliseconds.

Note: The error reset command will only be effective for an off to on transition.

Overlap may be prevented by insuring that the TRANSMIT ACTIVE bit flag in dual port REGISTER 29 is set and Register 30 is empty, before writing to REGISTER 30 for a TRANSMIT REQUEST.

An OVERLAP error does not affect messages from the MaxPak III drive to the AutoMax DCS. The HSL module will continue to accept and process received messages.

Error REGISTER 26 is provided for use by the applications engineer in debugging. Typically, during task development, the HSL module will shut the drive down. This bit, along with other information, is used to tell the engineer why the drive was shutdown (i.e. because the HSL module stopped transmitting due to an OVERLAP error).

Also, by detecting the OVERLAP in the HSL module, an application running a speed loop will not have to check for overlap. The task will simply set the write request bit in REGISTER 30 and continue. A ladder logic task will typically be running looking at the error bits such as OVERLAP, TIMEOUT, and CONFIG errors.

WARNING

THE USER MUST PROVIDE A TASK TO SHUT DOWN THE ENTIRE SYSTEM IN AN ORDERLY FASHION IN THE ADVENT OF A FAULT ON THE HSL CARD. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

The OVERLAP error will normally not be an error displayed on a user console.

Lastly, after the applications tasks have been debugged, the OVERLAP error normally should not occur.

5.2.2 MaxPak III Drive to AutoMax DCS Transmit Overlap

Under normal operation this error should never occur. However, if the MaxPak III drive detects an overlap condition, an OVERLAP fault results and the following action is taken:

a) An error is logged and displayed to the on-board display.

b) A drive fault stop sequence is initiated to stop the armature.

c) The MaxPak III drive will continue to transmit information to the AutoMax DCS as normal, so the AutoMax DCS may interrogate the error conditions. Any message to be sent prior to completion of previous messages will not be sent. This permits every other transmission attempt to be sent, assuming a message will complete within 2 scans.

Overlap is a latched fault condition which is treated much the same as a timeout fault. To clear this fault condition:

a) A drive fault reset (FLT_RESET@) must be performed.

5.3 Receive Overlap Errors

A Receive overlap error occurs when 2 or more new messages are received before an older message may be processed. In this case, there is no place to put the new message.

5.3.1 AutoMax DCS Receive Overlap

If a receive overlap occurs on the AutoMax DCS, the following will result:

a) The receive overlap error bit flag will be set in dual port REGISTER 26 bit 2.

b) No more messages may be transmitted to the MaxPak III drive. This will eventually cause the MaxPak III to exceed its timeout period and perform a fault stop sequence to stop the armature. The MaxPak III drive will stop sending updates to the HSL module, thus causing the HSL module to exceed its timeout period.

This is a latched fault condition. To clear this fault condition:

a) The communication error reset, REGISTER 31, must be asserted for a minimum time period 50 milliseconds.

Note: The error reset command will only be effective for an off to on transition.

5.3.2 MaxPak III Drive Receive Overlap

If a receive overlap occurs on the MaxPak III drive, the following will result:

a) An error is logged and displayed on the on-board display.

b) A drive fault stop sequence is initiated to stop the armature.

c) The MaxPak III drive stops transmitting updates and waits for a valid connect.

Overlap is a latched fault condition which is treated much the same as a timeout fault. To clear this fault condition:

a) A drive fault reset (FLT_RESET@) must be performed.

5.4 Timeout

The maximum time allowed between the receipt of any two consecutive valid messages is configurable at both ends of the link, independently.

5.4.1 AutoMax to MaxPak III Drive Timeout [(2 x AutoMax DCS Scan Period) + 10]

If the MaxPak III drive does not receive a valid message within the configured timeout period after the last valid message, a TIMEOUT fault results. If this occurs:

a) An error is logged and displayed on the on-board display.

b) A drive fault stop sequence is initiated to stop the armature.

c) HSL_TIMEOUT_E@ is set/reset to reflect the dynamic state of the timeout condition. If it exists, HSL_TIMEOUT_E@ will be set. After a valid connect has been received, HSL_TIMEOUT_E@ will be reset.

d) The MaxPak III drive stops transmitting updates and waits for a valid connect.

This is a latched fault condition. To clear this fault condition:

a) The link must be re-established such that a valid data message is received by the AutoMax DCS.

b) A drive fault reset (FLT_RESET@) must be performed. If a drive fault reset is performed before the link is re-established, the fault will be immediately latched again.

5.4.2 MaxPak III to AutoMax Timeout [(2 x MaxPak III's L2_SCAN_PERIOD%) + 5]

If the HSL module does not receive a valid message within the configured timeout period after the last valid message, a TIMEOUT fault results. If this occurs:

a) A TIMEOUT error flag will be set in dual port REGISTER 26 bit 0

b) The message currently being transmitted is halted and no more transmissions may commence. This will eventually cause the MaxPak III drive to exceed its timeout period and perform a fault stop sequence to stop the armature.

c) LINK ACTIVE, REGISTER 4 bit 0, will be reset and the 'C' will be displayed on the 7-segment LED.

This is a latched fault condition. To clear this fault condition:

a) The link must be re-established such that a valid data message is received from the MaxPak III drive. NOTE: receiving a valid message will set LINK ACTIVE (REGISTER 4 bit 0, TRUE).

b) The communication error reset, REGISTER 31, must be asserted for a minimum time period of 50 milliseconds. If the fault reset is performed while the link is not active, the reset will have no effect.

Note: The error reset command will only be effective for an off to on transition.

5.5 Message Not Received Errors

A message not received error occurs when a configure, a status change (Off-Line to On-Line and vice-versa) or an Off-Line update is sent to the MaxPak III drive and no acknowledgement is received.

If a message not received error occurs, the following will result:

a) The message not received error bit flag will be set in dual port register 26 bit 3.

b) No more messages may be transmitted or received to/from the MaxPak III drive. This will eventually cause the MaxPak III drive to exceed its timeout period and perform a fault stop sequence to stop the armature.

This is a latched fault condition. To clear this fault condition:

a) The communication error reset, REGISTER 31, must be asserted for a minimum time period of 50 mSec.

Note: The error reset command will only be effective for an off to on transition.

5.6 Invalid Operation Errors

If the user attempts certain operations while the armature is active, an invalid operation error occurs. Invalid operations include:

a) attempting to send a request to go Off-Line

b) attempting to send a configure request

If an invalid operation error occurs, the following will result:

a) The invalid operation error will be set in dual port register 26 bit 4.

No more messages may be transmitted or received to/from the MaxPak III drive. This will eventually cause the MaxPak III drive to exceed its timeout period and perform a fault stop sequence to stop the armature.

This is a latched fault condition. To clear this fault condition:

a) The communication error reset, REGISTER 31, must be asserted for a minimum time period of 50 milliseconds.

Note: The error reset command will only be effective for an off to on transition.

6.0 CONFIGURATION ERRORS

A configuration error occurs when one or more configuration registers (70–81) contain invalid values. As a result, only additional configuration/connect messages may be sent to the MaxPak III drive. The configuration error will be latched in Register 27 (see Section 2.2.1.2.1 for details). To correct the configuration error, place the correct values in Registers 70–81 and write a 4 to Register 30. Once the user corrects the configuration error, message transmission may be resumed, assuming the application software has logic to restart the link on HSL module time-out. The HSL module will not modify values in REGISTER 70 – 81 so the user may see which REGISTERS are in error.

Note: Receiving a configuration command from the HSL module will cause the MaxPak III drive to disable its timeout function.

Note: If a communication error is simultaneously latched in Register 26, the error must be reset before re-configuring. Refer to Section 2.2.1.2.1.

7.0 TROUBLESHOOTING

This section explains how to troubleshoot the High Speed Link (HSL) module (57C424 Comm Card). If the problem cannot be corrected by following the instructions in this section, the module is not user-serviceable.

DANGER

ONLY QUALIFIED ELECTRICAL PERSONNEL FAMILIAR WITH THE CONSTRUCTION AND OPERATION OF THIS EQUIPMENT AND THE HAZARDS INVOLVED SHOULD INSTALL, ADJUST, OPERATE, AND/OR SERVICE THIS EQUIPMENT. READ AND UNDERSTAND THIS INSTRUCTION MANUAL AND APPROPRIATE MAXPAK III AND AUTOMAX DCS MANUALS IN THEIR ENTIRETY BEFORE PROCEEDING. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN SEVERE BODILY INJURY OR LOSS OF LIFE.

WARNING

INSERTING OR REMOVING THIS MODULE OR ITS CONNECTING CABLES MAY RESULT IN UNEXPECTED MACHINE MOVEMENT. TURN OFF POWER BEFORE INSERTING OR REMOVING THE MODULE OR ITS CONNECTING CABLES. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

SYMPTOM	ACTION
HSL MODULE "OK" LED IS OFF	Check indicators on Processor connected to HSL module: 1.) "OK" LED is ON 2.) The "BAT. OK" LED is ON 3.) BUS ERROR (LED Code is not 31, or 50-58) Check indicators on AutoMax DCS power supply: 1.) "POWER ON" LED is ON 2.) "P/S READY" LED is ON 3.) "SYSTEM READY" LED is ON 4.) "BLOWN FUSE" LED is OFF If OK, proceed to ACTION for the following SYMPTOM in this table (7-SEGMENT LED CODE: 0-7, 9, b, d, .3, .4). If not OK, refer to AutoMax DCS manuals for isolation procedures: J-3650 (Processor Module M/N 57C430) and J-3670 (AUTOMAX Power Supply Module and Racks: M/N 57C491 Power Supply, M/N 57C331 16-Slot Rack, M/N 57C332 10-Slot Rack).

7-SEGMENT LED CODE: 0–7, 9, b, d, .3, .4 (See Appendix B)	Turn Off power to the AutoMax DCS. Reseat card in rack. Verify continuity of RS-232 cable. See Appendix D for configuration. Reseat cables at both ends of link. Turn On power to the AutoMax DCS. Reset the AutoMax DCS system by cycling power off and then on. If code is still present, replace the HSL module.
7-SEGMENT LED CODE: C (See Appendix B)	Verify that the RS-232 cable is connected at both ends of the link. Verify continuity of RS-232 Cable. See Appendix D for wiring characteristics. Verify the High Speed Link option is enabled on the MaxPak III. (HSL_ENABLE@ = ON, MB_ENABLE@ = OFF). Attempt to reconfigure link to default parameters (Registers 70–81), insure Register 26 = 0 and then set Register 30 to 4 (Configuration request).
7-SEGMENT LED CODE: .r (See Appendix B)	1. Correct software revisions as appropriate. 2A. Verify that HSL module's Register 26 is set to 0; If not, perform an error reset to HSL module's Register 31 (force a 0-to-1 transition). Thereafter, set HSL module's Register 30 to 4 to send a configure message. -OR- 2B. An alternate method is to cycle power on the AutoMax DCS system.

High Speed Link appears to have been established but the AutoMax DCS application is not running properly	Verify HSL module is in the correct slot as defined in the IODEF'S of the HSL module registers. If not, place the HSL module in the proper slot as defined in the IODEF of the HSL module registers or redefine IODEF of the HSL module registers to be in accordance with the present slot position of the HSL module. Verify the HSL has been properly configured (Registers 70–81) to the requirements of the application. (Note: The HSL will auto-connect with a default configuration but this may not be sufficient for the needs of the specific application.) If not, reconfigure (see Section 2.2.1.2.2). Check if any communications or configuration errors exist (Register 26 BIT 15 = 1; or Register 27 < > 0). If communication or configuration errors are present, reset the error(s) by cycling power on the AutoMax rack. Use Register 31 to reset error and then reconfigure. See Section 6.0 for Configuration Errors and Section 5.0 for Communication Errors. Check indicators on Processor connected to HSL module: 1.) "OK" LED is ON 2.) The "BAT. OK" LED is ON 3.) BUS ERROR (LED Code is not 31, or 50–58) If not OK, refer to AutoMax manual J-3650 (Processor Module M/N 57C430) for isolation procedures. Check the task(s) which are using the High Speed Link. Verify all task(s) are running. Do any tasks have run time errors? If so, refer to AutoMax Programming Executive Instruction manual, J-3684. Check the MaxPak III OKD error log. Is the HSL Error Fault Latch set (FLT_HSL_COMM_E@ = ON) ? If so, attempt a drive FLT_RESET@. If the condition does not clear, locate and remove the error condition as described in Section 5.0.

Application starts to run but then gets a HSL_COMM_ERR (Time-out Error and/or Message Overlap Error) on the MAXPAK III	TIMEOUT ERRORS: MaxPak III OKD error log (ELOG) displays HSL_COMM_ERR/SPECIFIC=1. Verify both ends of RS-232 cable is connected to the MaxPak III and the HSL module in the AutoMax DCS rack. Is the HSL module timeout set to the proper value in the MaxPak III configuration (HSL_TIMEOUT_T%)? Check to see if the AutoMax DCS speed loop task is running with the proper period (CALL SCAN_LOOP). Verify that the AutoMax application commands an HSL update within the HSL_TIMEOUT_T% (refer to Sections 4.4 and 5.4.1 for details). Increase the value of HSL_TIMEOUT_T% if necessary. Note: Once an error occurs on the MaxPak III, the error reset FLT_RESET@ must be toggled to clear the latched error condition thus allowing the armature to become active when all permissives are ON. MESSAGE OVERLAP ERRORS: MaxPak III OKD error log (ELOG) displays HSL_COMM_ERR/SPECIFIC=2 –or– HSL_COMM_ERR/SPECIFIC=4. Increase L2_SCAN_PERIOD% if it is less than 20. Verify that unused MaxPak III machine logic blocks are disabled. Verify that MaxPak III features that are not required are disabled (e.g. RD_EXECUTE_EN@ = OFF, L3_EXECUTE_EN@ = OFF, L2_EXECUTE_EN@ = OFF, ML_EXECUTE_EN@ = OFF if not needed).

APPENDIX A
Default HSL Register Map Assignments

REGISTER		VARIABLE
HSL_IN_INT_000	=	CML_REF%
HSL_IN_INT_001	=	FLD_I_REF%
HSL_IN_INT_002	=	OUT_SI_0_ANA_0%
HSL_IN_INT_003	=	OUT_SI_0_ANA_1%
HSL_IN_INT_004	=	OUT_SI_0_FREQ_0%
HSL_IN_INT_005	=	L2_FEEDBACK%
HSL_IN_INT_006	=	L2_REFA%
HSL_IN_INT_007	=	L2_REFA_MUL%
HSL_IN_INT_008	=	L2_RA_DRAW_GAIN%
HSL_IN_INT_009	=	L2_REFB%
HSL_IN_INT_010	=	L2_BB_GAIN_MUL%
HSL_IN_INT_011	=	L2_REFC%
HSL_IN_INT_012	=	L2_REFD%
HSL_IN_INT_013	=	L2_RA_JERK%
HSL_IN_INT_014	=	L2_RA_ACCEL%
HSL_IN_INT_015	=	L2_RA_DECEL%
HSL_IN_INT_016	=	L2_FP_GAIN_MUL%
HSL_IN_INT_017	=	ML_GAIN_0_IN%
HSL_IN_INT_018	=	OUT_SI_0_FREQ_1%
HSL_IN_INT_019	=	ML_GAIN_1_IN%
HSL_IN_INT_020	=	CML_ADAPT_GAIN%
HSL_IN_INT_021	=	CML_CC_THR%
HSL_IN_INT_022	=	CML_FB_GAIN%
HSL_IN_INT_023	=	CML_PI_KP%
HSL_IN_INT_024	=	CML_PI_WLD%
HSL_IN_INT_025	=	CML_RATE_LIM%
HSL_IN_INT_026	=	FLD_ECON_REF%
HSL_IN_INT_027	=	FLD_ECON_DLY_T%
HSL_IN_INT_028	=	FLD_IREF_LIM_HI%
HSL_IN_INT_029	=	FLD_IREF_LIM_LO%
HSL_IN_INT_030	=	FLD_I_FB_MUL%
HSL_IN_INT_031	=	FLD_I_PI_KP%
HSL_IN_INT_032	=	FLD_I_PI_WLD%
HSL_IN_INT_033	=	FLD_V_COMP_MUL%
HSL_IN_INT_034	=	FLD_V_FB_MUL%
HSL_IN_INT_035	=	FLD_V_PI_KP%
HSL_IN_INT_036	=	FLD_V_PI_WLD%
HSL_IN_INT_037	=	FLD_V_REF%
HSL_IN_INT_038	=	PT_ANALOG_SCALE%
HSL_IN_INT_039	=	IN_SI_0_A0_OFS%
HSL_IN_INT_040	=	IN_SI_0_A0_RNG%
HSL_IN_INT_041	=	IN_SI_0_A1_OFS%
HSL_IN_INT_042	=	IN_SI_0_A1_RNG%
HSL_IN_INT_043	=	IN_SI_0_A2_OFS%
HSL_IN_INT_044	=	IN_SI_0_A2_RNG%
HSL_IN_INT_045	=	IN_SI_0_A3_OFS%
HSL_IN_INT_046	=	IN_SI_0_A3_RNG%
HSL_IN_INT_047	=	IN_SI_0_F0_OFS%

REGISTER		VARIABLE
HSL_IN_INT_048	=	IN_SI_0_F1_OFS%
HSL_IN_INT_049	=	OUT_SI_0_F0_OFS%
HSL_IN_INT_050	=	OUT_SI_0_F1_OFS%
HSL_IN_INT_051	=	OUT_SI_0_A0_OFS%
HSL_IN_INT_052	=	OUT_SI_0_A0_RNG%
HSL_IN_INT_053	=	OUT_SI_0_A1_OFS%
HSL_IN_INT_054	=	OUT_SI_0_A1_RNG%
HSL_IN_INT_055	=	SEQ_STOP_L2_THR%
HSL_IN_INT_056	=	SEQ_MX_IOFF_T%
HSL_IN_INT_057	=	SEQ_MX_MCLOSE_T%
HSL_IN_INT_058	=	SEQ_MX_MOPEN_T%
HSL_IN_INT_059	=	SEQ_MX_STOP_T%
HSL_IN_INT_060	=	ID_REVISION%
HSL_IN_INT_061	=	ML_GAIN_0_DIV%
HSL_IN_INT_062	=	ML_GAIN_0_MUL%
HSL_IN_INT_063	=	ML_GAIN_1_DIV%
HSL_IN_INT_064	=	ML_GAIN_1_MUL%
HSL_IN_INT_065	=	ML_GAIN_2_DIV%
HSL_IN_INT_066	=	ML_GAIN_2_IN%
HSL_IN_INT_067	=	ML_GAIN_2_MUL%
HSL_IN_INT_068	=	L2_RB_GAIN_DIV%
HSL_IN_INT_069	=	L2_RB_LIM_HI%
HSL_IN_INT_070	=	L2_RB_LIM_LO%
HSL_IN_INT_071	=	L2_RB_OFFSET%
HSL_IN_INT_072	=	L2_REFB_MUL%
HSL_IN_INT_073	=	L2_REFC_MUL%
HSL_IN_INT_074	=	L2_REFD_MUL%
HSL_IN_INT_075	=	L2_REF_DIV%
HSL_IN_INT_076	=	L2_REF_LIM_HI%
HSL_IN_INT_077	=	L2_REF_LIM_LO%
HSL_IN_INT_078	=	L2_VARM_OFFSET%
HSL_IN_INT_079	=	L2_RA_GAIN_MUL%
HSL_IN_INT_080	=	L2_RA_GAIN_DIV%
HSL_IN_INT_081	=	L2_RA_LIM_HI%
HSL_IN_INT_082	=	L2_RA_LIM_LO%
HSL_IN_INT_083	=	L2_RA_OFFSET%
HSL_IN_INT_084	=	L2_FB_GAIN%
HSL_IN_INT_085	=	L2_PI_KP%
HSL_IN_INT_086	=	L2_PI_LIM_HI%
HSL_IN_INT_087	=	L2_PI_LIM_LO%
HSL_IN_INT_088	=	L2_PI_WLD%
HSL_IN_INT_089	=	L2_FB_LL_INITV%
HSL_IN_INT_090	=	L2_FB_LL_RATIO%
HSL_IN_INT_091	=	L2_FB_LL_W_LOW%
HSL_IN_INT_092	=	L2_FP_LL_INITV%
HSL_IN_INT_093	=	L2_FP_LL_RATIO%
HSL_IN_INT_094	=	L2_FP_LL_W_LOW%
HSL_IN_INT_095	=	L2_PI_INIT_VALU%
HSL_IN_INT_096	=	L2_FP_LAG_INITV%
HSL_IN_INT_097	=	L2_FP_LAG_W_LAG%
HSL_IN_INT_098	=	L2_FB_GAIN_DIV%
HSL_IN_INT_099	=	L2_FB_GAIN_MUL%
HSL_IN_INT_100	=	L2_FP_GAIN_DIV%
HSL_IN_INT_101	=	JOG_REFA%

REGISTER		VARIABLE
HSL_IN_INT_102	=	JOG_RA_ACCEL%
HSL_IN_INT_103	=	JOG_RA_DECEL%
HSL_IN_INT_104	=	L2_SEL_1_INPUT0%
HSL_IN_INT_105	=	L2_SEL_1_INPUT1%
HSL_IN_INT_106	=	L2_SEL_1_INPUT2%
HSL_IN_INT_107	=	L2_SEL_1_INPUT3%
HSL_IN_INT_108	=	L2_SEL_1_DIV%
HSL_IN_INT_109	=	L2_SEL_1_I0_MUL%
HSL_IN_INT_110	=	L2_SEL_1_I1_MUL%
HSL_IN_INT_111	=	L2_SEL_1_I2_MUL%
HSL_IN_INT_112	=	L2_SEL_1_I3_MUL%
HSL_IN_INT_113	=	L2_SEL_1_LIM_HI%
HSL_IN_INT_114	=	L2_SEL_1_LIM_LO%
HSL_IN_INT_115	=	ML_SEL_1_INPUT0%
HSL_IN_INT_116	=	ML_SEL_1_INPUT1%
HSL_IN_INT_117	=	ML_SEL_1_INPUT2%
HSL_IN_INT_118	=	ML_SEL_1_INPUT3%
HSL_IN_INT_119	=	ML_SEL_1_DIV%
HSL_IN_INT_120	=	ML_SEL_1_I0_MUL%
HSL_IN_INT_121	=	ML_SEL_1_I1_MUL%
HSL_IN_INT_122	=	ML_SEL_1_I2_MUL%
HSL_IN_INT_123	=	ML_SEL_1_I3_MUL%
HSL_IN_INT_124	=	ML_SEL_1_LIM_HI%
HSL_IN_INT_125	=	ML_SEL_1_LIM_LO%
HSL_IN_INT_126	=	ML_SEL_0_INPUT0%
HSL_IN_INT_127	=	ML_SEL_0_INPUT1%
HSL_IN_INT_128	=	ML_SEL_0_INPUT2%
HSL_IN_INT_129	=	ML_SEL_0_INPUT3%
HSL_IN_INT_130	=	ML_SEL_0_DIV%
HSL_IN_INT_131	=	ML_SEL_0_I0_MUL%
HSL_IN_INT_132	=	ML_SEL_0_I1_MUL%
HSL_IN_INT_133	=	ML_SEL_0_I2_MUL%
HSL_IN_INT_134	=	ML_SEL_0_I3_MUL%
HSL_IN_INT_135	=	ML_SEL_0_LIM_HI%
HSL_IN_INT_136	=	ML_SEL_0_LIM_LO%
HSL_IN_INT_137	=	L2_SEL_0_INPUT0%
HSL_IN_INT_138	=	L2_SEL_0_INPUT1%
HSL_IN_INT_139	=	L2_SEL_0_INPUT2%
HSL_IN_INT_140	=	L2_SEL_0_INPUT3%
HSL_IN_INT_141	=	L2_SEL_0_DIV%
HSL_IN_INT_142	=	L2_SEL_0_I0_MUL%
HSL_IN_INT_143	=	L2_SEL_0_I1_MUL%
HSL_IN_INT_144	=	L2_SEL_0_I2_MUL%
HSL_IN_INT_145	=	L2_SEL_0_I3_MUL%
HSL_IN_INT_146	=	L2_SEL_0_LIM_HI%
HSL_IN_INT_147	=	L2_SEL_0_LIM_LO%
HSL_IN_INT_148	=	ML_FNC_0_INPUT%
HSL_IN_INT_149	=	ML_FNC_0_X0%
HSL_IN_INT_150	=	ML_FNC_0_X1%
HSL_IN_INT_151	=	ML_FNC_0_X2%
HSL_IN_INT_152	=	ML_FNC_0_X3%
HSL_IN_INT_153	=	ML_FNC_0_X4%
HSL_IN_INT_154	=	ML_FNC_0_X5%
HSL_IN_INT_155	=	ML_FNC_0_X6%
HSL_IN_INT_156	=	ML_FNC_0_X7%

REGISTER		VARIABLE
HSL_IN_INT_157	=	ML_FNC_0_Y0%
HSL_IN_INT_158	=	ML_FNC_0_Y1%
HSL_IN_INT_159	=	ML_FNC_0_Y2%
HSL_IN_INT_160	=	ML_FNC_0_Y3%
HSL_IN_INT_161	=	ML_FNC_0_Y4%
HSL_IN_INT_162	=	ML_FNC_0_Y5%
HSL_IN_INT_163	=	ML_FNC_0_Y6%
HSL_IN_INT_164	=	ML_FNC_0_Y7%
HSL_IN_INT_165	=	ML_CMP_0_INPUT%
HSL_IN_INT_166	=	ML_CMP_0_THR_HI%
HSL_IN_INT_167	=	ML_CMP_0_THR_LO%
HSL_IN_INT_168	=	ML_CMP_1_INPUT%
HSL_IN_INT_169	=	ML_CMP_1_THR_HI%
HSL_IN_INT_170	=	ML_CMP_1_THR_LO%
HSL_IN_INT_171	=	ML_CMP_2_INPUT%
HSL_IN_INT_172	=	ML_CMP_2_THR_HI%
HSL_IN_INT_173	=	ML_CMP_2_THR_LO%
HSL_IN_INT_174	=	ML_CMP_3_INPUT%
HSL_IN_INT_175	=	ML_CMP_3_THR_HI%
HSL_IN_INT_176	=	ML_CMP_3_THR_LO%
HSL_IN_INT_177	=	ML_CMP_4_INPUT%
HSL_IN_INT_178	=	ML_CMP_4_THR_HI%
HSL_IN_INT_179	=	ML_CMP_4_THR_LO%
HSL_IN_INT_180	=	ML_FNC_1_INPUT%
HSL_IN_INT_181	=	ML_FNC_1_X0%
HSL_IN_INT_182	=	ML_FNC_1_X1%
HSL_IN_INT_183	=	ML_FNC_1_X2%
HSL_IN_INT_184	=	ML_FNC_1_X3%
HSL_IN_INT_185	=	ML_FNC_1_X4%
HSL_IN_INT_186	=	ML_FNC_1_X5%
HSL_IN_INT_187	=	ML_FNC_1_X6%
HSL_IN_INT_188	=	ML_FNC_1_X7%
HSL_IN_INT_189	=	ML_FNC_1_Y0%
HSL_IN_INT_190	=	ML_FNC_1_Y1%
HSL_IN_INT_191	=	ML_FNC_1_Y2%
HSL_IN_INT_192	=	ML_FNC_1_Y3%
HSL_IN_INT_193	=	ML_FNC_1_Y4%
HSL_IN_INT_194	=	ML_FNC_1_Y5%
HSL_IN_INT_195	=	ML_FNC_1_Y6%
HSL_IN_INT_196	=	ML_FNC_1_Y7%
HSL_IN_INT_197	=	FLD_FI_BASERPM%
HSL_IN_INT_198	=	FLD_FI_FW1CUR%
HSL_IN_INT_199	=	FLD_FI_FW1RPM%
HSL_IN_INT_200	=	FLD_FI_FW2CUR%
HSL_IN_INT_201	=	FLD_FI_FW2RPM%
HSL_IN_INT_202	=	FLD_FI_FW3CUR%
HSL_IN_INT_203	=	FLD_FI_FW3RPM%
HSL_IN_INT_204	=	FLD_FI_FW4CUR%
HSL_IN_INT_205	=	FLD_FI_FW4RPM%
HSL_IN_INT_206	=	FLD_FI_FW5CUR%
HSL_IN_INT_207	=	FLD_FI_FW5RPM%
HSL_IN_INT_208	=	FLD_FI_RATECUR%
HSL_IN_INT_209	=	FLD_FI_SPD_MUL%
HSL_IN_INT_210	=	ML_SEL_2_INPUT0%
HSL_IN_INT_211	=	ML_SEL_2_INPUT1%

REGISTER		VARIABLE
HSL_IN_INT_212	=	ML_SEL_2_INPUT2%
HSL_IN_INT_213	=	ML_SEL_2_INPUT3%
HSL_IN_INT_214	=	ML_SEL_2_DIV%
HSL_IN_INT_215	=	ML_SEL_2_I0_MUL%
HSL_IN_INT_216	=	ML_SEL_2_I1_MUL%
HSL_IN_INT_217	=	ML_SEL_2_I2_MUL%
HSL_IN_INT_218	=	ML_SEL_2_I3_MUL%
HSL_IN_INT_219	=	ML_SEL_2_LIM_HI%
HSL_IN_INT_220	=	ML_SEL_2_LIM_LO%
HSL_IN_INT_221	=	IN_SI_0_F0_RNG%
HSL_IN_INT_222	=	IN_SI_0_F1_RNG%
HSL_IN_INT_223	=	CML_AC_FREQU%
HSL_IN_INT_224	=	FLT_OL_IFB_THR%
HSL_IN_INT_225	=	FLT_OL_KI%
HSL_IN_INT_226	=	PT_FILTER_SEL%
HSL_IN_INT_227	=	PT_RATEMULT_SEL%
HSL_IN_INT_228	=	PT_RPM_PN%
HSL_IN_INT_229	=	PT_TACH_PPR%
HSL_IN_INT_230	=	L2_SCAN_PERIOD%
HSL_IN_INT_231	=	FLT_FLC_RUN_T%
HSL_IN_INT_232	=	FLT_FL_THR%
HSL_IN_INT_233	=	FLT_FL_WLG%
HSL_IN_INT_234	=	FLT_OC_PRESET%
HSL_IN_INT_235	=	FLT_OVRSPD_THR%
HSL_IN_INT_236	=	FLT_TL_DELT_THR%
HSL_IN_INT_237	=	FLT_FLC_SCALE%
HSL_IN_INT_238	=	FLD_DELT_LIM_HI%
HSL_IN_INT_239	=	FLD_DELT_LIM_LO%
HSL_IN_INT_240	=	HSL_TIMEOUT_T%
HSL_IN_INT_241	=	SEQ_JOG_OFF_T%
HSL_IN_INT_242	=	IN_SI_0_F0_FIL%
HSL_IN_INT_243	=	IN_SI_0_F0_MAX%
HSL_IN_INT_244	=	IN_SI_0_F1_FIL%
HSL_IN_INT_245	=	IN_SI_0_F1_MAX%
HSL_IN_INT_246	=	OUT_SI_0_F0_MAX%
HSL_IN_INT_247	=	OUT_SI_0_F0_RNG%
HSL_IN_INT_248	=	OUT_SI_0_F1_MAX%
HSL_IN_INT_249	=	OUT_SI_0_F1_RNG%
HSL_IN_INT_250	=	L2_RA_SCALE%
HSL_IN_INT_251	=	JOG_RA_SCALE%
HSL_IN_INT_252	=	ID_SO_NUMBER_1%
HSL_IN_INT_253	=	ID_SO_NUMBER_2%
HSL_IN_INT_254	=	ID_SO_NUMBER_3%
HSL_IN_INT_255	=	ID_SO_NUMBER_4%
!		
HSL_OUT_INT_000	=	PT_SPEED_FB%
HSL_OUT_INT_001	=	IN_SI_0_ANA_D%
HSL_OUT_INT_002	=	CML_REF_SJ%
HSL_OUT_INT_003	=	CML_ERROR_SJ%
HSL_OUT_INT_004	=	CML_FB_SJ%
HSL_OUT_INT_005	=	ARM_DELTA%
HSL_OUT_INT_006	=	IN_VARM_FB%
HSL_OUT_INT_007	=	FLD_DELTA%
HSL_OUT_INT_008	=	FLD_I_FB_SJ%

REGISTER		VARIABLE
HSL_OUT_INT_009	=	IN_SI_D_ANA_1%
HSL_OUT_INT_010	=	IN_SI_0_ANA_2%
HSL_OUT_INT_011	=	IN_SI_0_ANA_3%
HSL_OUT_INT_012	=	IN_SI_0_FREQ_0%
HSL_OUT_INT_013	=	IN_PT_EDGE%
HSL_OUT_INT_014	=	L2_REF_SJ%
HSL_OUT_INT_015	=	L2_OUTPUT%
HSL_OUT_INT_016	=	L2_SEL_0_OUTPUT%
HSL_OUT_INT_017	=	ML_SEL_0_OUTPUT%
HSL_OUT_INT_018	=	ML_GAIN_2_OUT%
HSL_OUT_INT_019	=	SEQ_BOOL_OUTS%
HSL_OUT_INT_020	=	ARM_DELTA_AVG%
HSL_OUT_INT_021	=	SEQ_CML_ENABLE%
HSL_OUT_INT_022	=	SEQ_STOP_CAUSE%
HSL_OUT_INT_023	=	COM_FRAMING_E%
HSL_OUT_INT_024	=	COM_OVERRUN_E%
HSL_OUT_INT_025	=	COM_PARITY_E%
HSL_OUT_INT_026	=	HSL_CHKSUM_ERRS%
HSL_OUT_INT_027	=	HSL_RECVD_MSGS%
HSL_OUT_INT_028	=	HSL_XMITD_MSGS%
HSL_OUT_INT_029	=	LS_6D_DEGREES%
HSL_OUT_INT_030	=	LS_FLAGS%
HSL_OUT_INT_031	=	LS_LINE_PERIOD%
HSL_OUT_INT_032	=	LS_SOFTW_P10X%
HSL_OUT_INT_033	=	IN_ACLINE_VOLTS%
HSL_OUT_INT_034	=	IN_DCTIME%
HSL_OUT_INT_035	=	FLT_LATCHES%
HSL_OUT_INT_036	=	FLT_FLC_I_AVG%
HSL_OUT_INT_037	=	FLT_FLC_REMN_T%
HSL_OUT_INT_038	=	FLT_FLC_THR%
HSL_OUT_INT_039	=	FLT_FL_LAG_OUT%
HSL_OUT_INT_040	=	FLT_OL_IFB%
HSL_OUT_INT_041	=	FLT_OL_RISE%
HSL_OUT_INT_042	=	FLT_OL_RISE_THR%
HSL_OUT_INT_043	=	FLD_ARM_I_FB%
HSL_OUT_INT_044	=	FLD_I_ERR_SJ%
HSL_OUT_INT_045	=	IN_FLD_IFB_1%
HSL_OUT_INT_046	=	FLD_I_REF_SJ%
HSL_OUT_INT_047	=	FLD_I_REF_TP1%
HSL_OUT_INT_048	=	FLD_V_COMP%
HSL_OUT_INT_049	=	FLD_V_ERR_SJ%
HSL_OUT_INT_050	=	FLD_V_FB%
HSL_OUT_INT_051	=	FLD_V_FB_COMP%
HSL_OUT_INT_052	=	FLD_V_FB_SJ%
HSL_OUT_INT_053	=	FLD_V_OUT%
HSL_OUT_INT_054	=	FLD_DELTA_1_PH%
HSL_OUT_INT_055	=	IN_SI_0_FREQ_1%
HSL_OUT_INT_056	=	IN_SI_0_A_I5V%
HSL_OUT_INT_057	=	IN_SI_0_A_N15V%
HSL_OUT_INT_058	=	IN_SI_0_A_NREF%
HSL_OUT_INT_059	=	IN_SI_0_A_P15V%
HSL_OUT_INT_060	=	IN_SI_0_A_PREF%
HSL_OUT_INT_061	=	IN_SI_0_A_UI5V%
HSL_OUT_INT_062	=	ML_GAIN_0_OUT%

REGISTER		VARIABLE
HSL_OUT_INT_063	=	ML_GAIN_1_OUT%
HSL_OUT_INT_064	=	ML_SEL_1_OUTPUT%
HSL_OUT_INT_065	=	L2_RB_TP1%
HSL_OUT_INT_066	=	L2_RB_TP2%
HSL_OUT_INT_067	=	L2_RB_TP3%
HSL_OUT_INT_068	=	L2_RB_TP4%
HSL_OUT_INT_069	=	L2_RB_TP5%
HSL_OUT_INT_070	=	L2_RA_TP1%
HSL_OUT_INT_071	=	L2_RA_TP2%
HSL_OUT_INT_072	=	L2_RA_TP3%
HSL_OUT_INT_073	=	L2_RA_TP4%
HSL_OUT_INT_074	=	L2_RA_TP5%
HSL_OUT_INT_075	=	L2_RA_TP6%
HSL_OUT_INT_076	=	L2_RA_TP7%
HSL_OUT_INT_077	=	L2_RA_TP8%
HSL_OUT_INT_078	=	L2_RA_TP9%
HSL_OUT_INT_079	=	L2_RA_TP10%
HSL_OUT_INT_080	=	JOG_RA_TP1%
HSL_OUT_INT_081	=	L2_REF_JOG%
HSL_OUT_INT_082	=	L2_REF_SEL_OUT%
HSL_OUT_INT_083	=	L2_RDET_SCL_OUT%
HSL_OUT_INT_084	=	L2_REF_RUN%
HSL_OUT_INT_085	=	L2_REF%
HSL_OUT_INT_086	=	L2_FB_TP1%
HSL_OUT_INT_087	=	L2_FB_ABS%
HSL_OUT_INT_088	=	L2_FB_LL_OUTPUT%
HSL_OUT_INT_089	=	L2_FB_SJ%
HSL_OUT_INT_090	=	L2_ERROR_SJ%
HSL_OUT_INT_091	=	L2_FP_GAIN_OUT%
HSL_OUT_INT_092	=	L2_FP_LAG_OUT%
HSL_OUT_INT_093	=	L2_FP_LL_OUTPUT%
HSL_OUT_INT_094	=	L2_FB_LL_W_HIGH%
HSL_OUT_INT_095	=	L2_FP_LL_W_HIGH%
HSL_OUT_INT_096	=	L2_SEL_1_OUTPUT%
HSL_OUT_INT_097	=	FLD_FI_ERR_SJ%
HSL_OUT_INT_098	=	FLD_FI_OUT%
HSL_OUT_INT_099	=	FLD_FI_SPD_SJ%
HSL_OUT_INT_100	=	FLD_FI_TP1%
HSL_OUT_INT_101	=	ML_FNC_0_OUTPUT%
HSL_OUT_INT_102	=	ML_FNC_1_OUTPUT%
HSL_OUT_INT_103	=	ML_FNC_2_OUTPUT%
HSL_OUT_INT_104	=	ML_SEL_2_OUTPUT%
HSL_OUT_INT_105	=	ML_SEL_3_OUTPUT%
HSL_OUT_INT_106	=	ML_SEL_4_OUTPUT%
HSL_OUT_INT_107	=	ML_CMP_0_HI_ABS%
HSL_OUT_INT_108	=	ML_CMP_0_IN_ABS%
HSL_OUT_INT_109	=	ML_CMP_0_LO_ABS%
HSL_OUT_INT_110	=	ML_CMP_1_HI_ABS%
HSL_OUT_INT_111	=	ML_CMP_1_IN_ABS%
HSL_OUT_INT_112	=	ML_CMP_1_LO_ABS%
HSL_OUT_INT_113	=	ML_CMP_2_HI_ABS%
HSL_OUT_INT_114	=	ML_CMP_2_IN_ABS%
HSL_OUT_INT_115	=	ML_CMP_2_LO_ABS%
HSL_OUT_INT_116	=	ML_CMP_3_HI_ABS%
HSL_OUT_INT_117	=	ML_CMP_3_IN_ABS%

REGISTER		VARIABLE
HSL_OUT_INT_118	=	ML_CMP_3_LO_ABS%
HSL_OUT_INT_119	=	ML_CMP_4_HI_ABS%
HSL_OUT_INT_120	=	ML_CMP_4_IN_ABS%
HSL_OUT_INT_121	=	ML_CMP_4_LO_ABS%
HSL_OUT_INT_122	=	L3_ERROR_SJ%
HSL_OUT_INT_123	=	L3_FB_LL_W_HIGH%
HSL_OUT_INT_124	=	L3_FB_SJ%
HSL_OUT_INT_125	=	L3_FP_LL_OUTPUT%
HSL_OUT_INT_126	=	L3_FP_LL_W_HIGH%
HSL_OUT_INT_127	=	L3_OUTPUT%
HSL_OUT_INT_128	=	L3_RATE_OUT%
HSL_OUT_INT_129	=	L3_REF_SJ%
HSL_OUT_INT_130	=	RD_CAL_DIA%
HSL_OUT_INT_131	=	RD_COIL_DIA%
HSL_OUT_INT_132	=	RD_DIA_TP1%
HSL_OUT_INT_133	=	RD_DIA_TP2%
HSL_OUT_INT_134	=	RD_DIA_TP3%
HSL_OUT_INT_135	=	RD_NORM_DIA%
HSL_OUT_INT_136	=	RD_DIAMETER%
HSL_OUT_INT_137	=	RD_RATE_OUT%
HSL_OUT_INT_138	=	IN_REF_15_VOLTS%
HSL_OUT_INT_139	=	IN_IFBSUM%
HSL_OUT_INT_140	=	IN_PFGA_DIG_0_1%
HSL_OUT_INT_141	=	IN_PFGA_DIG_X_2%
HSL_OUT_INT_142	=	IN_PT_EG2SCN%
HSL_OUT_INT_143	=	IN_PT_REDGE%
HSL_OUT_INT_144	=	IN_PT_VREF%
HSL_OUT_INT_145	=	SYS_NRDY_CNTR%
HSL_OUT_INT_146	=	SYS_ACCESS_LEVL%
HSL_OUT_INT_147	=	SYS_NRDY_RESULT%
HSL_OUT_INT_148	=	SYS_PWRF_CNTR%
HSL_OUT_INT_149	=	SYS_SI_PERIPHS%
HSL_OUT_INT_150	=	SYS_SLOT_0_1_ID%
HSL_OUT_INT_151	=	SYS_SLOT_2_3_ID%
HSL_OUT_INT_152	=	SYS_SLOT_4_5_ID%
HSL_OUT_INT_153	=	SYS_SLOT_6_ID%
HSL_OUT_INT_154	=	SYS_SOFTWAR_REV%
!		
HSL_IN_BOOL_000	=	SEQ_DRIVE_EN@
HSL_IN_BOOL_001	=	SEQ_JOG@
HSL_IN_BOOL_002	=	SEQ_RUN@
HSL_IN_BOOL_003	=	SEQ_STOP_ILIM@
HSL_IN_BOOL_004	=	FLD_ECON_EN@
HSL_IN_BOOL_005	=	FLT_RESET@
HSL_IN_BOOL_006	=	OUT_AUXDIG_0@
HSL_IN_BOOL_007	=	OUT_AUXDIG_1@
HSL_IN_BOOL_008	=	OUT_SI_0_DIG_0@
HSL_IN_BOOL_009	=	OUT_SI_0_DIG_1@
HSL_IN_BOOL_010	=	OUT_SI_0_DIG_2@
HSL_IN_BOOL_011	=	OUT_SI_0_DIG_3@
HSL_IN_BOOL_012	=	L2_RA_HOLD@
HSL_IN_BOOL_013	=	L2_RB_INVERT@
HSL_IN_BOOL_014	=	L2_RB_VERN_EN@
HSL_IN_BOOL_015	=	L2_RA_INVERT@

REGISTER		VARIABLE
HSL_IN_BOOL_016	=	FLT_EXT_REQU1@
HSL_IN_BOOL_017	=	FLT_EXT_REQU2@
HSL_IN_BOOL_018	=	L2_REFA_ENABLE@
HSL_IN_BOOL_019	=	L2_REFB_ENABLE@
HSL_IN_BOOL_020	=	L2_REFC_ENABLE@
HSL_IN_BOOL_021	=	L2_REFD_ENABLE@
HSL_IN_BOOL_022	=	LOG_CLR_DISABLE@
HSL_IN_BOOL_023	=	COM_OKD_LONG_EN@
HSL_IN_BOOL_024	=	L2_FB_LL_BYPASS@
HSL_IN_BOOL_025	=	L2_FB_LL_RESET@
HSL_IN_BOOL_026	=	L2_FP_LL_BYPASS@
HSL_IN_BOOL_027	=	L2_FP_LL_RESET@
HSL_IN_BOOL_028	=	MEM_SAVE@
HSL_IN_BOOL_029	=	L2_FP_LAG_BYPAS@
HSL_IN_BOOL_030	=	L2_FP_LAG_RESET@
HSL_IN_BOOL_031	=	L2_RDET_SCL_EN@
HSL_IN_BOOL_032	=	ML_SEL_1_I0_EN@
HSL_IN_BOOL_033	=	ML_SEL_1_I1_EN@
HSL_IN_BOOL_034	=	ML_SEL_1_I2_EN@
HSL_IN_BOOL_035	=	ML_SEL_1_I3_EN@
HSL_IN_BOOL_036	=	ML_SEL_0_I0_EN@
HSL_IN_BOOL_037	=	ML_SEL_0_I1_EN@
HSL_IN_BOOL_038	=	ML_SEL_0_I2_EN@
HSL_IN_BOOL_039	=	ML_SEL_0_I3_EN@
HSL_IN_BOOL_040	=	L2_SEL_1_I0_EN@
HSL_IN_BOOL_041	=	L2_SEL_1_I1_EN@
HSL_IN_BOOL_042	=	L2_SEL_1_I2_EN@
HSL_IN_BOOL_043	=	L2_SEL_1_I3_EN@
HSL_IN_BOOL_044	=	L2_SEL_0_I0_EN@
HSL_IN_BOOL_045	=	L2_SEL_0_I1_EN@
HSL_IN_BOOL_046	=	L2_SEL_0_I2_EN@
HSL_IN_BOOL_047	=	L2_SEL_0_I3_EN@
HSL_IN_BOOL_048	=	ML_SEL_2_I0_EN@
HSL_IN_BOOL_049	=	ML_SEL_2_I1_EN@
HSL_IN_BOOL_050	=	ML_SEL_2_I2_EN@
HSL_IN_BOOL_051	=	ML_SEL_2_I3_EN@
HSL_IN_BOOL_052	=	ML_SEL_3_I0_EN@
HSL_IN_BOOL_053	=	ML_SEL_3_I1_EN@
HSL_IN_BOOL_054	=	ML_SEL_3_I2_EN@
HSL_IN_BOOL_055	=	ML_SEL_3_I3_EN@
HSL_IN_BOOL_056	=	ML_SEL_4_I0_EN@
HSL_IN_BOOL_057	=	ML_SEL_4_I1_EN@
HSL_IN_BOOL_058	=	ML_SEL_4_I2_EN@
HSL_IN_BOOL_059	=	ML_SEL_4_I3_EN@
HSL_IN_BOOL_060	=	ML_CMP_0_HI_ABS@
HSL_IN_BOOL_061	=	ML_CMP_0_IN_ABS@
HSL_IN_BOOL_062	=	ML_CMP_0_LO_ABS@
HSL_IN_BOOL_063	=	ML_CMP_1_HI_ABS@
HSL_IN_BOOL_064	=	ML_CMP_1_IN_ABS@
HSL_IN_BOOL_065	=	ML_CMP_1_LO_ABS@
HSL_IN_BOOL_066	=	ML_CMP_2_HI_ABS@
HSL_IN_BOOL_067	=	ML_CMP_2_IN_ABS@
HSL_IN_BOOL_068	=	ML_CMP_2_LO_ABS@
HSL_IN_BOOL_069	=	ML_CMP_3_HI_ABS@
HSL_IN_BOOL_070	=	ML_CMP_3_IN_ABS@

REGISTER		VARIABLE
HSL_IN_BOOL_071	=	ML_CMP_3_LO_ABS@
HSL_IN_BOOL_072	=	ML_CMP_4_HI_ABS@
HSL_IN_BOOL_073	=	ML_CMP_4_IN_ABS@
HSL_IN_BOOL_074	=	ML_CMP_4_LO_ABS@
HSL_IN_BOOL_075	=	ML_TMR_0_INPUT@
HSL_IN_BOOL_076	=	ML_TMR_1_INPUT@
HSL_IN_BOOL_077	=	ML_TMR_2_INPUT@
HSL_IN_BOOL_078	=	ML_TMR_3_INPUT@
HSL_IN_BOOL_079	=	ML_TMR_4_INPUT@
HSL_IN_BOOL_080	=	ML_STOP_2@
HSL_IN_BOOL_081	=	ML_JOG_2@
HSL_IN_BOOL_082	=	ML_RUN_2@
HSL_IN_BOOL_083	=	L3_FB_LL_BYPASS@
HSL_IN_BOOL_084	=	L3_FP_LL_BYPASS@
HSL_IN_BOOL_085	=	RD_RESET@
HSL_IN_BOOL_086	=	RD_RUN@
HSL_IN_BOOL_087	=	RD_RATE_RESET@
HSL_IN_BOOL_088	=	RD_TEST_ENABLE@
HSL_IN_BOOL_089	=	L3_FP_LL_RESET@
HSL_IN_BOOL_090	=	L3_PI_RESET@
HSL_IN_BOOL_091	=	L3_RATE_RESET@
HSL_IN_BOOL_092	=	CML_FAST_BRI_EN@
HSL_IN_BOOL_093	=	CML_S6R_EN@
HSL_IN_BOOL_094	=	FLD_1_PH_APPLIC@
HSL_IN_BOOL_095	=	PT_QUADRATUR_EN@
HSL_IN_BOOL_096	=	MEM_RESTORE@
HSL_IN_BOOL_097	=	ML_EXECUTE_EN@
HSL_IN_BOOL_098	=	L2_EXECUTE_EN@
HSL_IN_BOOL_099	=	L2_SEL_1_EXE_EN@
HSL_IN_BOOL_100	=	FLT_FLC_ENABLE@
HSL_IN_BOOL_101	=	FLT_OL_STOP_DIS@
HSL_IN_BOOL_102	=	FLD_FUNCPROC_EN@
HSL_IN_BOOL_103	=	FLT_TL_DISABLE@
HSL_IN_BOOL_104	=	SEQ_M_CONTAC_EN@
HSL_IN_BOOL_105	=	SEQ_STOP_L2_BYP@
HSL_IN_BOOL_106	=	SEQ_STOP_1_BYP@
HSL_IN_BOOL_107	=	ML_SEL_0_EXE_EN@
HSL_IN_BOOL_108	=	ML_SEL_1_EXE_EN@
HSL_IN_BOOL_109	=	L2_SEL_0_EXE_EN@
HSL_IN_BOOL_110	=	L2_FB_LL_LAG_LO@
HSL_IN_BOOL_111	=	L2_FP_LL_LAG_LO@
HSL_IN_BOOL_112	=	IN_SI_0_A0_HRES@
HSL_IN_BOOL_113	=	IN_SI_0_A1_HRES@
HSL_IN_BOOL_114	=	IN_SI_0_A2_HRES@
HSL_IN_BOOL_115	=	IN_SI_0_A3_HRES@
HSL_IN_BOOL_116	=	IN_SI_1_A0_HRES@
HSL_IN_BOOL_117	=	IN_SI_1_A1_HRES@
HSL_IN_BOOL_118	=	IN_SI_1_A2_HRES@
HSL_IN_BOOL_119	=	IN_SI_1_A3_HRES@
HSL_IN_BOOL_120	=	ML_FNC_0_EXE_EN@
HSL_IN_BOOL_121	=	ML_FNC_1_EXE_EN@
HSL_IN_BOOL_122	=	ML_FNC_2_EXE_EN@
HSL_IN_BOOL_123	=	ML_SEL_2_EXE_EN@
HSL_IN_BOOL_124	=	ML_SEL_3_EXE_EN@
HSL_IN_BOOL_125	=	ML_SEL_4_EXE_EN@
HSL_IN_BOOL_126	=	ML_CMP_0_EXE_EN@

REGISTER		VARIABLE
HSL_IN_BOOL_127	=	ML_CMP_1_EXE_EN@
HSL_IN_BOOL_128	=	ML_CMP_2_EXE_EN@
HSL_IN_BOOL_129	=	ML_CMP_3_EXE_EN@
HSL_IN_BOOL_130	=	ML_TMR_0_EXE_EN@
HSL_IN_BOOL_131	=	ML_TMR_0_EXT_EN@
HSL_IN_BOOL_132	=	ML_TMR_0_ONE_EN@
HSL_IN_BOOL_133	=	ML_TMR_1_EXE_EN@
HSL_IN_BOOL_134	=	ML_TMR_1_EXT_EN@
HSL_IN_BOOL_135	=	ML_TMR_1_ONE_EN@
HSL_IN_BOOL_136	=	ML_TMR_2_EXE_EN@
HSL_IN_BOOL_137	=	ML_TMR_2_EXT_EN@
HSL_IN_BOOL_138	=	ML_TMR_2_ONE_EN@
HSL_IN_BOOL_139	=	ML_TMR_3_EXE_EN@
HSL_IN_BOOL_140	=	ML_TMR_3_EXT_EN@
HSL_IN_BOOL_141	=	ML_TMR_3_ONE_EN@
HSL_IN_BOOL_142	=	ML_TMR_4_EXE_EN@
HSL_IN_BOOL_143	=	ML_TMR_4_EXT_EN@
HSL_IN_BOOL_144	=	ML_TMR_4_ONE_EN@
HSL_IN_BOOL_145	=	L3_EXECUTE_EN@
HSL_IN_BOOL_146	=	L3_FB_LL_LAG_LO@
HSL_IN_BOOL_147	=	L3_FP_LL_LAG_LO@
HSL_IN_BOOL_148	=	RD_EXECUTE_EN@
!		
HSL_OUT_BOOL_000	=	ARM_BRIDG_POL@
HSL_OUT_BOOL_001	=	FLD_ECON_ACTIVE@
HSL_OUT_BOOL_002	=	FLD_OK@
HSL_OUT_BOOL_003	=	FLD_PF_ENABLE@
HSL_OUT_BOOL_004	=	FLD_1_PH_BRIDGE@
HSL_OUT_BOOL_005	=	FLD_REG_ENABLE@
HSL_OUT_BOOL_006	=	FLT_FIELDLOSS@
HSL_OUT_BOOL_007	=	FLT_FLC_ACTIVE@
HSL_OUT_BOOL_008	=	FLT_OVERLOAD@
HSL_OUT_BOOL_009	=	SEQ_ARM_PERM@
HSL_OUT_BOOL_010	=	SEQ_FLD_PERM@
HSL_OUT_BOOL_011	=	SEQ_JOGGING@
HSL_OUT_BOOL_012	=	SEQ_L2_ENABLE@
HSL_OUT_BOOL_013	=	SEQ_RUNNING@
HSL_OUT_BOOL_014	=	SEQ_STOP_ILIM_L@
HSL_OUT_BOOL_015	=	SEQ_STOP_L@
HSL_OUT_BOOL_016	=	IN_AUXDIGIN_0@
HSL_OUT_BOOL_017	=	IN_AUXDIGIN_1@
HSL_OUT_BOOL_018	=	IN_AUXDIGIN_2@
HSL_OUT_BOOL_019	=	IN_AUXDIGIN_3@
HSL_OUT_BOOL_020	=	IN_AUXDIGIN_4@
HSL_OUT_BOOL_021	=	IN_RUNPERM@
HSL_OUT_BOOL_022	=	IN_M_STATUS@
HSL_OUT_BOOL_023	=	FLT_DRIVE_FAULT@
HSL_OUT_BOOL_024	=	FLT_FIELDLOSS_L@
HSL_OUT_BOOL_025	=	FLT_OVERCUR_L@
HSL_OUT_BOOL_026	=	FLT_OVERLOAD_L@
HSL_OUT_BOOL_027	=	FLT_OVERSPEED_L@
HSL_OUT_BOOL_028	=	FLT_SYNCLOSS_L@
HSL_OUT_BOOL_029	=	FLT_TACHLOSS_L@
HSL_OUT_BOOL_030	=	FLT_HSL_COMM_L@
HSL_OUT_BOOL_031	=	HSL_XOVRLAP_E@

REGISTER	VARIABLE
HSL_OUT_BOOL_032 =	IN_SI_0_DIG_0@
HSL_OUT_BOOL_033 =	IN_SI_0_DIG_1@
HSL_OUT_BOOL_034 =	IN_SI_0_DIG_2@
HSL_OUT_BOOL_035 =	IN_SI_0_DIG_3@
HSL_OUT_BOOL_036 =	IN_SI_0_DIG_4@
HSL_OUT_BOOL_037 =	FLT_EXT_REQU1_L@
HSL_OUT_BOOL_038 =	FLT_EXT_REQU2_L@
HSL_OUT_BOOL_039 =	LS_BADPERD_E@
HSL_OUT_BOOL_040 =	LS_BADPHAS_E@
HSL_OUT_BOOL_041 =	LS_LOCKED@
HSL_OUT_BOOL_042 =	LS_PHASE_SEL@
HSL_OUT_BOOL_043 =	LS_PHROT@
HSL_OUT_BOOL_044 =	LS_PHROT_E@
HSL_OUT_BOOL_045 =	LS_PHROT_VALID@
HSL_OUT_BOOL_046 =	LS_PR_ACTIVE@
HSL_OUT_BOOL_047 =	LS_PR_HALFCYC_E@
HSL_OUT_BOOL_048 =	LS_PR_PERIOD_E@
HSL_OUT_BOOL_049 =	LS_PR_PHASE_E@
HSL_OUT_BOOL_050 =	MEM_SAVE_ERROR@
HSL_OUT_BOOL_051 =	LOG_EMPTY@
HSL_OUT_BOOL_052 =	LOG_FULL@
HSL_OUT_BOOL_053 =	ML_CMP_0_OUTPUT@
HSL_OUT_BOOL_054 =	ML_CMP_1_OUTPUT@
HSL_OUT_BOOL_055 =	ML_CMP_2_OUTPUT@
HSL_OUT_BOOL_056 =	ML_CMP_3_OUTPUT@
HSL_OUT_BOOL_057 =	ML_CMP_4_OUTPUT@
HSL_OUT_BOOL_058 =	ML_TMR_0_OUTPUT@
HSL_OUT_BOOL_059 =	ML_TMR_1_OUTPUT@
HSL_OUT_BOOL_060 =	ML_TMR_2_OUTPUT@
HSL_OUT_BOOL_061 =	ML_TMR_3_OUTPUT@
HSL_OUT_BOOL_062 =	ML_TMR_4_OUTPUT@
HSL_OUT_BOOL_063 =	SYS_ONLINE@
HSL_OUT_BOOL_064 =	ML_JOG@
HSL_OUT_BOOL_065 =	ML_RUN@
HSL_OUT_BOOL_066 =	ML_STOP@

APPENDIX B
7-Segment LED Display Codes

FAULT CODE	FAULT CODE DESCRIPTION
0	CPU failed power-up diagnostic
1	EPROM failed power-up diagnostic
2	RAM failed power-up diagnostic
3	CTC failed power-up diagnostic
4	"Reliance Use Only"
5	DMA failed
6	Dual Port memory failed power-up diagnostic
7	Memory management unit failed power-up diagnostic
9	Parallel I/O port failed power-up diagnostic
A	"Reliance Use Only"
b	Watchdog failed power-up diagnostic
C	Communication line status. Displayed only when no messages are received from the MaxPak III.
d	System (backplane) watchdog failed. Board is operational but will not transmit or receive data until the watchdog is reset.
E	Power failure. This code is normally present from the time that a power failure is detected until power is lost.
o	"Reliance Use Only"
.3	CTC run time failure.
.4	SIO run time failure.
.r	Incompatible software revisions (MaxPak III - 57C424)

If at power up of the rack, any diagnostic fault code remains displayed ('0' - '9' inclusive or 'b'), the High Speed Link module (M/N 0-57C424) must be replaced.

APPENDIX C
Interrupt Status and Control Register Layout (REG 0)

This register controls the multibus interrupt line the HSL module card will set high when interrupting the AutoMax DCS Processor as well as whether Multibus interrupts are enabled or not. The register bit fields are set up as follows:

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
F								E					U	A	B

F - 1 = = > Interrupt Condition Exists on the Comm Card
0 = = > AutoMax DCS Processor has acknowledged the interrupt

E - 1 = = > AutoMax DCS Processor has enabled Multibus interrupts
0 = = > Multibus interrupts are disabled

U - 1 = = > ISCR has been allocated by a AutoMax DCS Processor
0 = = > ISCR has not been allocated

AB - = = > Values may range from 0 to 3 based on which Multibus Interrupt line the AutoMax DCS Processor wants the HSL module to use.

The AutoMax DCS application program will not interface directly with this register. It will normally define an event using the AutoMax Programming Language which is assigned to this register through an IODEF statement (There is only one ISCR that the HSL module will recognize, REGISTER 0). The AutoMax DCS operating system will then configure the interrupt status and control register specified, using the bit fields as defined above. See the following example:

Configuration Task

```
10 IODEF HSL_ISCR%(SLOT = 1,REGISTER = 0) < --- ISCR REG 0 (see above)
```

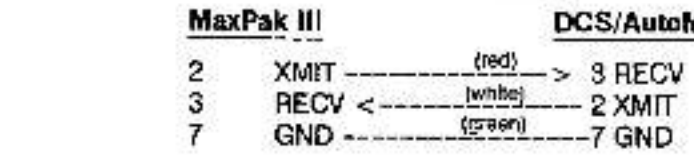
Local BASIC task

```
10 COMMON FB_INTREG%
...
...
95 EVENT NAME = SPEED_FEEDBACK,&
        INTERRUPT_STATUS = HSL_ISCR%,&
        TIMEOUT = DISABLED
...
...
255 WAIT ON SPEED_FEEDBACK
...
...
```

Bit 15 is the most significant bit in a word. This standard applies to BOTH the HSL module and the MaxPak III drive references to bit positions in this document.

APPENDIX D
RS-232 Wiring Characteristics (Part No. 610160-26R)

RS-232C Serial Wiring Characteristics



Pin 2 = MaxPak III Transmit to pin 3 HSL Module Receive.
Pin 3 = MaxPak III Receive to pin 2 HSL Module Transmit.
Pin 7 = Ground.

This cable part number is available as an assembly only. The following table describes the function, plug definition, and cable length. Note: Maximum length allowable is 10ft (3 meters) with twisted pair wire (2 twists/inch). Longer lengths can be accommodated using modems.

PART NUMBER	FUNCTION	PLUG DEFINITION	CABLE LENGTH
610160-26R	Null-Modem	Male to Male	120"

RS-232 Configuration

The RS-232 serial line is automatically configured for 41.6k baud, 8 data bits, 1 stop bit, odd parity. For configurations where a direct connection is not possible, the following equipment may be used to interface with the MaxPak III High Speed Link Module.

APPENDIX E
57C424 (HSL Module)
Techincal Specifications

Ambient Conditions

Storage Temperature	:	−30 degrees Celsius to 85 degrees Celsius
Operating Temperature	:	0 degrees Celsius to 60 degrees Celsius
Humidity Range	:	5 to non-condensing

Maximum Module Power Dissipation

13 Watts

Dimensions

Height	:	11.71 inches
Width	:	1.25 inches
Depth	:	7.375 inches

System Power Requirements

5 Volts	:	2400 mA
+ 12 Volts	:	53mA
−12 Volts	:	8mA

APPENDIX F
HSL Fatal Error Codes for the HSL Module (57C424)

These codes are "error specific" data placed in Register 60 to accompany the LED code display, which indicates the general error.

Error Specific	Description
01F0H:	A spurious interrupt in CTC channel 0 has occurred. CTC channel 0 is used to generate the baud rate and thus should never generate an interrupt. If it does, it is a fatal error and the HSL module will display a '.3' on the 7-segment LED and latch the specified error code in register 60.

Error Specific	Description
02F0H 02F1H 02F2H 02F3H	A spurious transmit interrupt on channel A has occurred. A status change interrupt on channel A has occurred. An RX (receive) interrupt on channel A has occurred. A special receive interrupt on channel A has occurred.
	The Serial Input Output chip (SIO) on the HSL module has two channels. Channel A is not used at all in HSL, therefore all of channel A interrupts are disabled. If an interrupt does occur for SIO channel A, it is a fatal error and the HSL module will display a '.4' on the 7-segment LED and latch the appropriate error code in register 60.

Error Specific	Description
03F0HA	spurious status change interrupt on channel B has occurred. Channel B of the SIO is used on the HSL module. All interrupts but one are enabled on this channel. The interrupt that is disabled is the external line status change interrupt. This interrupt occurs when a <break> is received from sending device or DCD becomes inactive (active low). For HSL, these conditions are ignored. Thus, the interrupt sensitive to these conditions is disabled. If this interrupt does occur, it is treated as an SIO fatal run time error and a '.4' is displayed on the 7-segment LED.

APPENDIX G
Verify Errors (Detected during verify stage when using DWNMP3)

These HSL errors are sent to the screen and to the log file (if requested).

These error messages are displayed as well as the source line which caused the error unless indicated otherwise.

Error #	Error	Description
234	HSL REGISTER SECTION AND VARIABLE NAME TYPES (@,%) DO NOT MATCH Action: Correct syntax error.	The HSL register type did not match the variable type (integer or boolean).
235	HSL REGISTER ASSIGNED MORE THAN ONCE IN SAME SECTION Action: Correct configuration or syntax error.	The HSL register was previously used by an HSL assignment statement in the same HSL section.
236	VARIABLE ASSIGNED TO MORE THAN ONE HSL REGISTER IN SAME SECTION Action: Correct configuration or syntax error.	The same variable name was assigned to more than one HSL register in the same HSL section.
237	HSL STATEMENTS NOT SUPPORTED BY MAXPAK III VERSION Action: Remove this HSL statement.	The version of the MaxPak III which is connected does not support HSL statements.
238	INVALID HSL REGISTER NUMBER FIELD Action: Correct syntax error.	The HSL register number field was either too long (> 3 digits) or contained a non-digit character.
239	HSL REGISTER NUMBER TOO HIGH Action: Correct syntax error.	The HSL register number field was greater than the maximum register number (255).

APPENDIX H
DownLoad Errors (Protected during the download stage)

These HSL errors are sent to the screen and to the log file (if requested).

These HSL error messages are displayed as well as the source line which caused the error unless indicated otherwise. An "*" indicates that the error should not normally occur. An "*^" indicates that the error message is displayed without a source line.

Error #	Error	Description
076	*^MP3 HAS NOT BEEN OPENED FOR HSL ASSIGNMENTS Action: Check communications.	The INIT HSL message was not received by the MaxPak III.
077	*^MP3 HAS NOT RECEIVED HSL INIT COMMAND Action: Check communications.	The download HSL open message was not received by the MP3.
078	*^MP3 HAS ALREADY RECEIVED HSL INIT COMMAND Action: Check communications.	The INIT HSL message was already received by the MaxPak III.
079	*^HSL STARTING REGISTER NUMBER IS OUT OF ORDER Action: Check communications.	An invalid HSL message starting register was received.
080	*^HSL - BAD SECTION NUMBER Action: Check communications.	An invalid HSL message section was received.
082	*^HSL - TOO MANY ASSIGNMENTS PER MESSAGE Action: Check communications.	The number of assignments contained in the HSL message was not within the valid range of 0-7 inclusive.
084	*^BAD HSL COMMAND Action: Check communications.	An invalid HSL message command was received.
085	*^THE MAXPAK III RECEIVED AN HSL MESSAGE WITH A BAD MESSAGE LENGTH Action: Check communications.	The HSL message received by the MaxPak III was not the correct size.
086	*HSL MAXIMUM SECTION ASSIGNMENT COUNT EXCEEDED Action: Check communications.	The number of HSL register assignments for this type has exceeded the maximum limit.
087	*HSL - BAD VARIABLE NAME Action: Check communications.	The variable name does not conform to MaxPak III conventions.
088	HSL - VARIABLE NOT FOUND Action. Correct configuration.	The variable name does not exist in the MaxPak III

Error #	Error	Description
089	HSL - ONLINE VARIABLE (ACCESS LEVEL < 3) EXPECTED Action: Correct configuration.	An OFFLINE variable was received but an ONLINE variable was required (input registers 000 and 001 require ONLINE variables).
090	HSL - OFFLINE VARIABLE (ACCESS LEVEL > 2) EXPECTED	An ONLINE variable was received but an OFFLINE variable was required.
091	HSL - VARIABLE TYPE DOES NOT MATCH SECTION TYPE (INPUT/OUTPUT) Action: Correct configuration.	The variable received did not match the expected HSL section (e.g. An output variable was assigned but an input was expected).

INDEX

Appendix J
Procedure for Configuring/ Re-Configuring HSL Link

This section describes a procedure by which the HSL link can be configured/ re-configured. Configuration of the HSL link can include any or all of the following:

- Changes to dual port registers 70–81
- Changes to offline HSL-mapped input variables

(Changes to online HSL-mapped input variables are handled automatically via the update request, provided all desired variables are accounted for in the values of Registers 70 and 71.)

If the link restart task (HSL_RUN.BAS) is installed as described in Section 4.6, it is important to note that any changes made to dual-port registers by the procedure below will be destroyed each time an HSL_RESET or a Start All is performed. To make these changes permanent, the appropriate values should be entered in lines 2000–2050 of HSL_RUN.BAS, then the task re-compiled. Once this has been done, changes to registers 70–81 are effected simply by setting HSL_RESET@ ON (the procedure outlined below will only be needed for making changes to offline variables).

The variable names used in the following procedure are compatible with the configuration tasks listed in the programming examples of Section 4.5. This procedure assumes that the interlock ladder logic and the speed loop enhancements described in Section 4.5 have been installed. All register references are to dual port memory on the HSL module.

1. Ensure that communications have been established (LINK_OK@ is ON).
2. Set UPDATE_ENABLE@ OFF. (Note: this may result in a timeout error)
3. Set COMM_RESET@ (register 31, bit 0) ON then OFF and make sure that COMM_ERRS% (register 26) goes to 0.
4. If the only configuration changes necessary are to dual-port registers 70–81, proceed to step 23. If changes to HSL-mapped offline input variables (or both) are required, continue below:
5. Insure that CMD_REG% (register 30) is 0 and TX_ACTIVE@ (register 29, bit 0) is OFF.
6. Set CMD_REG% (register 30) = 2.
7. Insure that MP3_STATUS% (register 28) = 0040h and COMM_ERRS% (register 26) = 0.
8. Insure that CMD_REG% (register 30) is 0 and TX_ACTIVE@ (register 29, bit 0) is OFF.
9. Set INPUT_INTS% (register 70) to the TOTAL number of input integers to be sent to the MaxPak III drive.
10. Set INPUT_BOOLS% (register 71) to the TOTAL number of input boolean variables to be sent to the MaxPak III.
11. Set CMD_REG% (register 30) = 4.
12. Insure that LINK_OK@ is ON, CMD_REG% (register 30) = 0 and that TX_ACTIVE@ (register 29, bit 0) is OFF.
13. Insure that the desired values have been written to all offline HSL-mapped input variables.
14. Set CMD_REG% (register 30) = 128.

15. Insure that LINK_OK@ is ON, CMD_REG% (register 30) = 0 and that TX_ACTIVE@ (register 29, bit 0) is OFF.
16. Set INPUT_INTS% (register 70) back to the number of online input integers to be sent to the MaxPak III.
17. Set INPUT_BOOLS% (register 71) back to the number of online input boolean variables to be sent to the MaxPak III.
18. Set CMD_REG% (register 30) = 4.
19. Insure that LINK_OK@ is ON, CMD_REG% (register 30) = 0 and that TX_ACTIVE@ (register 29, bit 0) is OFF.
20. Set CMD_REG% (register 30) = 3.
21. Wait for the MaxPak III drive to finish its diagnostic sequence and go online ("WELCOME" message displayed).
22. If HSL_RUN.BAS is not installed, or if the initialization values it produces are not acceptable, proceed to the next step. Otherwise set HSL_RESET@ ON and wait for it to go OFF (5 to 8 seconds). If HSL_READY@ comes ON at this point, the configuration process has been completed successfully.
23. Insure that LINK_OK@ is ON, MP3_STATUS% (register 28) and CMD_REG% (register 30) are 0 and that TX_ACTIVE@ (register 29, bit 0) is OFF.
24. Make any desired changes to the contents of Registers 70–81.
25. Set CMD_REG% (register 30) = 4.
26. Insure that LINK_OK@ is ON, CMD_REG% (register 30) = 0 and that TX_ACTIVE@ (register 29, bit 0) is OFF.
27. Set FLT_RESET@ ON then OFF. (This assumes that FLT_RESET@ is included in the HSL register map as an input boolean.)
28. Insure that DRIVE_FAULT@ (FLT_DRIVE_FAULT@) is OFF. (This assumes that FLT_DRIVE_FAULT@ is included in the HSL register map as an output boolean variable. Depending on configuration and multiplexing of variables, it may take several seconds for DRIVE_FAULT@ to turn off.)
29. Set UPDATE_ENABLE@ "ON" to re-enable transmission by the speed loop.

APPENDIX K
Overview of High Speed Link (HSL) functionality

The basic concept of the High Speed Link (HSL) is to transmit integer data in dual port registers 1100 - 1355 (of the 57C424) and boolean data in dual port registers 1400 - 1415 to the MaxPak III (MP3) and receive integer data from the MP3 in dual port registers 100 - 355, and receive boolean data from the MP3 in dual port registers 400 - 415. Using this data in a meaningful way is up to the applications programmer.

Typically, a user must first create a configuration task that contains the proper mapping of variable names to I/O registers using IODEF statements (see the AutoMax BASIC programming manual for definition of an IODEF statement). A typical link is described in section 4.5 of this instruction manual. Next, the user must insure that the registers defined in the configuration task correspond to the HSL variable assignments downloaded into the MaxPak III drive. For example, to send CML_REF% from an AutoMax processor to the MaxPak III drive using the HSL, the following assignments would be necessary:

```
  AutoMax Configuration File
100   IODEF CML_REF%[ SLOT = XXXX, REGISTER = 1100 ]
...
...
  MaxPak III drive configuration file
100   HSL_IN_INT_000 = CML_REF%
...
...
```

(See section 4.5 of this instruction manual for a detailed application example)

The HSL has 6 registers which are used to configure the link. Using these registers the applications programmer can define the number of registers (integer and boolean) transmitted and received by the HSL (57C424) to/from the MaxPak III drive, as well as the maximum number of registers which can be transmitted per message to the MaxPak III drive, and the timeout period for stopping the HSL on the 57C424.

For example :

Assume the user has requested to send

187 integers to the MaxPak III drive and
59 booleans to the MaxPak III drive

and has requested to receive

93 integers from the MaxPak III drive and
55 booleans from the MaxPak III drive

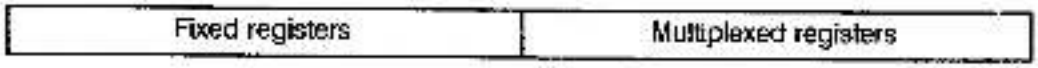
The user must therefore program the HSL dual port configuration registers as follows (and issue a "CONFIGURE" command) :

Register 70 = 187 (Number of integers to transmit to the MaxPak III drive)
Register 71 = 59 (Number of booleans to transmit to the MaxPak III drive)
Register 75 = 93 (Number of integers to receive from the MaxPak III drive)
Register 76 = 55 (Number of booleans to receive from the MaxPak III drive)
Register 80 = 54 (54 millisecond timeout period)
Register 81 = 44 (44 * .5ms = 22 ms speed loop time period; or the rate at which the appplication will be transmitting messages to the MaxPak III drive)

Using register 81, the 57C424 software can now calculate the maximum number of bytes of data which can be transmitted to the MaxPak III drive in one scan (or in 22 ms in our example). If the scan period is longer, then the largest message which can be transmitted in that time will be longer, and if the scan period between transmitting messages is shorter, then less bytes of data can be sent per message and hence the message length for that scan period would be smaller.

For each value of register 81 (range 11 - 198) the 57C424 maintains a table of information telling it how big each message can be and how to utilize the bytes in that message. In other words, how many registers will always be sent each message and how many registers will need to be multiplexed or broken into groups and transmitted across several messages.

The ideal situation would be if the HSL could send all the registers requested by the user in every message or each DCS/Automax speed loop scan (in our example 22 ms). This, however, is not possible since it physically takes longer than 22 ms to send 187 integers and 59 booleans at 41.6 Kbaud. Therefore, the HSL breaks each message into 2 parts.



HSL message format

Fixed registers

The first section is called fixed because a 'fixed' group of registers (starting with register 1100; which is always the first integer register transmitted) will be transmitted every message from the 57C424 to the MaxPak III drive once the configuration registers 70,71,75,76,80,81 have been programmed.

That means that if registers 1100 .. 1109 are the fixed registers to be transmitted, then those registers will be sent from the 57C424 dual port to the MaxPak III drive every time an "UPDATE" command is issued to the 57C424.

- High priority fixed registers

  Within the fixed section (the registers being transmitted and received every message) there is a further division into high priority and low priority registers. The data placed in high priority registers will be used by the other end of the link as soon it is received and verified by the other end of the link. These high priority registers are used before the remainder of the incoming message is received (in other words while the other data characters are still being transmitted across the serial link).

  The MaxPak III drive transmits to the 57C424 2 high priority integer registers every message regardless of the L2_SCAN_PERIOD% on the MaxPak III drive.

  The 57C424 transmits 1 high priority integer register to the MaxPak III drive every message regardless of the AutoMax DCS scan rate (register 81).

- Low priority fixed registers

  Low priority fixed registers are also guaranteed to be transmitted every message, but the entire message must be received before the data is used at the other end of the link. The number of low priority fixed registers is a function of how much time the user will allow the HSL for transmitting a message (L2_SCAN_PERIOD% for messages coming from the MaxPak III and register 81 for messages from the 57C424), thus placing an upper bound on the message length.

  The low priority registers consist of both integer and boolean registers. The number of each type of register is defined in Table 2-1 and is based on the scan period of the DCS application (Register 81).

Fixed registers		Multiplexed registers
High priority registers	**Low priority registers**	
1 integer register from 57C424 to MP3	# of int and bool registers from 57C424 to MP3 (value in Reg 81)	# of int and bool registers from 57C424 to MP3 (value in Reg 81)
2 integer registers from MP3 to 57C424	# of int and bool registers from MP3 to 57C424 determined by L2_SCAN_PERIOD%	# of int and bool registers from MP3 to 57C424 determined by L2_SCAN_PERIOD%

Multiplexed registers

This is the section of every message dedicated to multiplexing the remaining registers that would not fit into a single message. Multiplexed registers are used in order to allow up to 256 registers to be sent/received while still limiting the message length to much less than would be required to send all the data in one message.

The size of this section in each message is also determined by the L2_SCAN_PERIOD% for messages coming from the MaxPak III drive and by register 81 for messages from the 57C424.

Note that every time an UPDATE command is issued to send a message from the 57C424 to the MaxPak III drive that only ONE message is transmitted. A single UPDATE commmand does not transmit all registers requested to be sent to the MaxPak III drive (configuration registers 70,71). The proper multiplexing and grouping of registers in each message is handled by the 57C424 but the application must set the UPDATE request enough times to insure that all requested registers are transmitted. If the user is running a continuous application then all registers will be transmitted over time. However, if the user wants to send all registers only one time, he must calculate the number of update requests required, which is the reason Table 2-1 is used along with the accompanying calculations in section 2.2.4.1.1.

For our example we can calculate how long it will take to send all requested registers. First we will calculate the time to send registers to the MaxPak III drive :

Integers to send to the MaxPak III drive - 187
Booleans to send to the MaxPak III drive - 59

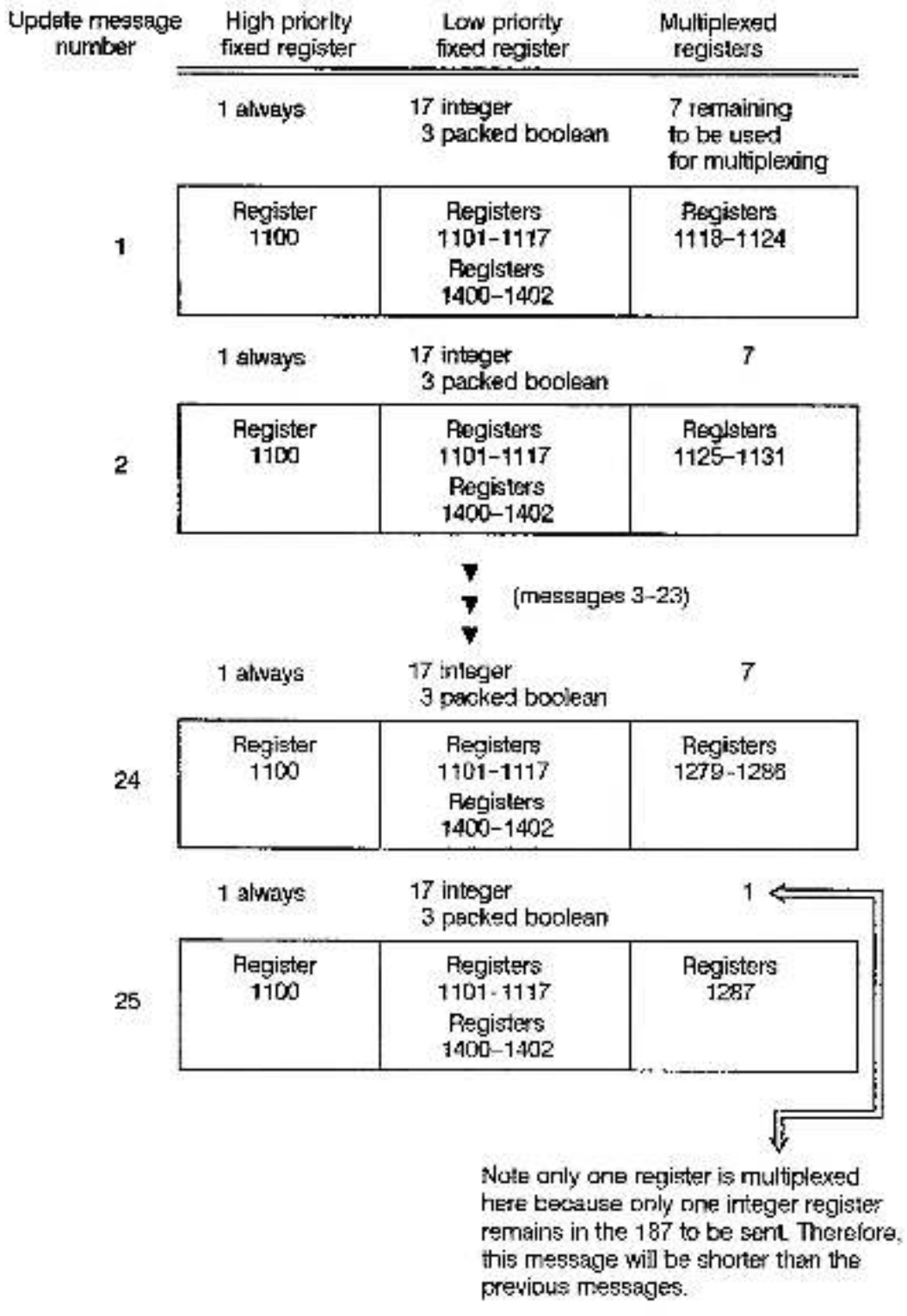
Notice the values in Table 2-1 for our setting of register 81. For a value of 44-54 (22 - 27.5 ms) the table shows the following three columns :

Max # of Fixed Integer (MFI) Registers	Max # of Fixed Packed Boolean (MFB) Registers	Max # of Registers per Message (MRPM)
18	3	28

Knowing these numbers will now help us to determine the format of each message being transmitted to the MaxPak III drive.

The maximum # of registers per message (MRPM) [28 in our our example] defines the total number of registers which can be transmitted in one message (this constraint is imposed by time - reg 81).

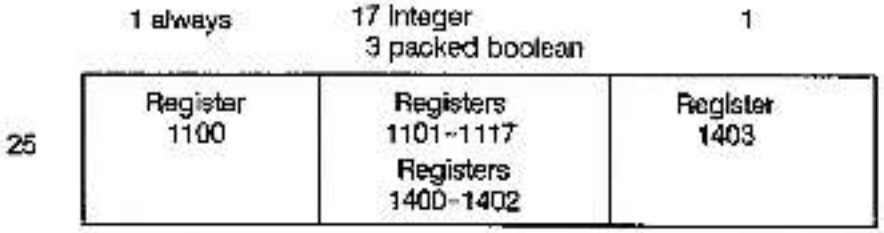
In our example 28 total registers can be transmitted. Of these 18 + 3 registers are of the fixed type (1 of which is a high priority register (from 57C424 to MaxPak III drive)). This leaves 7 registers which will make up the multiplexed register section. Therefore each message update will transmit the following registers :



Note that all the integer registers which would not fit in each message have just been multiplexed in groups of 7 (except the last message which was only one). Now the booleans which would not fit into each message are multiplexed. It should be noted

that booleans are always sent as packed 16 bit registers even though there may only be 1 bit or boolean defined in that register.

Recall the user wanted 59 booleans sent to the MaxPak III drive. Each message has room in the low priority fixed section to send 3 packed booleans (or 48 booleans = 16 * 3 registers). That means 59-48 = 11 booleans must be multiplexed. These eleven bits will pack into one 16 bit register so that only one multiplexed boolean register needs to be sent. Therefore, message update 26 has the following format :



Integer multiplexing and boolean multiplexing are not mixed in the same message. As can be seen, all the integer registers requiring multiplexing are sent and then the boolean registers requiring multiplexing are sent in separate messages.

It should be noted that it requires 26 messages to send all the 187 integers and 59 booleans requested by the user to be transmitted to the MaxPak III drive or 26 x 22 ms = 572 ms. The high priority fixed register will be transmitted every message and be used immediately upon receipt (< 3 ms after setting the update request command in register 30). The low priority fixed registers will be made available for use every 22 ms. The remaining 169 integers (187 - 1 - 17) and the remaining 11 booleans will take a maximum of 572 milliseconds to be updated.

The user need only be aware of this information to understand how long it will take to update the low priority registers.

All multiplexing of registers is handled automatically by the 57C424. The user simply loads his registers and then sets the update command in register 30 to transmit the data to the MaxPak III drive.

The example just described is for messages going from the 57C424 to the MaxPak III drive. The same scheme is used in handling registers transmitted from the MaxPak III drive to the 57C424 with two exceptions :

- The MaxPak III drive sends 2 high priority registers to the 57C424 every message.
- The table used to calculate maximum number of registers is different and is based on the scan rate of the MaxPak III (L2_SCAN_PERIOD%).

Forward To: Reliance Electric
V★S Drives
Technical Writing Group
24701 Euclid Ave.
Cleveland, OH 44117

Technical Writing Internal Use:
DIF # _____

V★S DRIVES
DOCUMENTATION IMPROVEMENT FORM

Document Number: _____

Page Number(s): _____

Comments: (Please give chapters, page numbers or specific paragraphs that the change will affect. Include markups from the document or attach additional pages if necessary.)

What will this improvement suggestion provide? _____

Originator: _____ City: _____ State: _____ ZIP: _____

Company: _____ Phone: (____) _____

Address: _____ Date: _____

Technical Writing Internal Use:

Follow-Up Action:

Writer: _____ Date: _____

Thank you for your comments . . .

RELIANCE ELECTRIC

TRAINING AND AUDIO/VISUAL PRODUCTS

Reliance Electric offers a wide variety of Industrial Training courses for electricians, electronic technicians and engineers who are responsible for the installation, repair and maintenance of production equipment and systems.

Professional quality A/V Programs are also available. These programs have been designed to provide years of efficient in-house training. Available for playback at the user's convenience, these videotape programs allow individual or groups to learn or review subjects at any time.

Printed reference materials come with all diagnostic and troubleshooting programs.

Training Courses

No.	Title
	INDUSTRIAL CONTROLS COURSES
3-10	AutoMate 15/20 Maintenance and Troubleshooting
3-11	AutoMate 30/40 Maintenance and Troubleshooting
3-14	Shark® X and XL Programming, Maintenance, and Troubleshooting
3-17	AutoMate 15/20 Application Programming
3-18	AutoMate 30/40 Application Programming
3-19	AutoMate 40 Advanced Application Programming
3-20	AutoMax™ /DCS 5000 Application Programming
3-21	Advanced AutoMax/DCS 5000 Application Programming
	DISTRIBUTED DIGITAL CONTROL SYSTEM COURSES
5-0	Maintenance and Troubleshooting AutoMax/DCS D-C Drives and Systems
5-3	Control Block and Ladder Programming of AutoMax/DCS D-C Systems
5-4	Reliance® Basic Programming Language
5-5	D-C Drives and Introduction to AutoMax/DCS
5-6	Introduction to the Use of the Reliance Programming Terminal
5-7	Programming the AutoMax/DCS Dual Axis Module
5-8	Maintenance and Troubleshooting AutoMax/DCS A-C Drives and Systems

Audio/Visual Products

Order No.	Title	Format	Price
	AutoMate 35 PROGRAMMABLE CONTROLLER PROGRAMS		
TM2204	AutoMate 35 Programmable Controller Diagnostics using the CRT	Videotape	$725
TM2205	AutoMate 35 Programmable Controller I/O Functions and Troubleshooting	Videotape	725
	NEW VIDEO TRAINING PROGRAMS		
VMBA001	Fundamentals of A-C Motors	Videotape	$495
VMBV001	Concepts of Digital Controls	Videotape	495
VWVS001	GP2000 Video Training	Videotape	495
VWVS002	HR2000 Video Training	Videotape	495

For details and prices on these courses, audio/visual products and FREE Training Schedule Brochure, HD-405, contact:

Industrial Training Department
Reliance Electric
35000 Curtis Boulevard
Eastlake, Ohio 44095

Call Toll Free:

800/321-2795 (Outside Ohio)

800/262-2688 (In Ohio)

Data or Prices subject to change without notice.

ZA-225c

RELIANCE ELECTRIC

Reliance Electric / 24703 Euclid Avenue / Cleveland, Ohio 44117 / 216-266-7000

RELIANCE ELECTRIC

Printed in U.S.A. J2-3010 September 1991