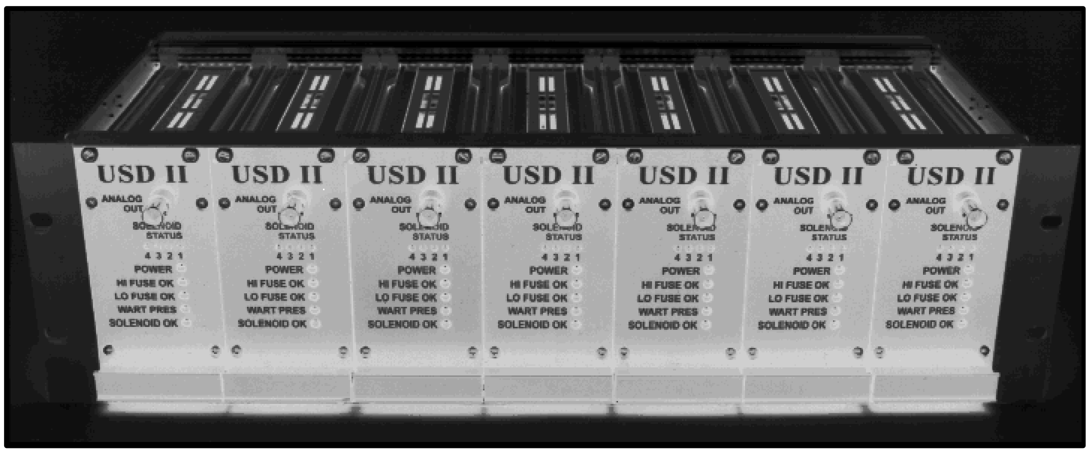
Global Electronics Ltd.

USD II
Technical Manual



Michael Leads

Section 1 - Introduction

1.1 What is the USD II and What Does It Do?

The USD II -- developed by Ford Motor Company's Advanced & Pre Program Transmission Engineering Department -- is a computer controlled solenoid driver apparatus. It allows the user to dynamically control transmission solenoids (i.e., solenoids inside a transmission, on a valve-body-test stand, or on a bench). Also, the USD II allows the user to set-up automated solenoid testing in a variety of ways.

The USD II is capable of driving each type of transmission solenoid currently in use by Ford Motor Company i.e., PWML, PWMH, ON/OFF and VFS solenoids. The USD II can house up to seven driver modules, making it capable of driving up to eight ON/OFF solenoids and a selection of five other solenoids of any type. Optionally, two units can be daisy-chained together to expand the capability to sixteen ON/OFF solenoids and up to ten other solenoid types; thus, the USD II can be configured to drive various quantities and assortments of solenoids to meet the user's specific needs.

1.2 Is the USD II Meant to Replace the Original USDB?

The USD II was designed as an upgrade to its predecessor -- the USDB -- primarily to introduce new features such as power isolation, packaging, CAN communication etc. Also some of the parts are no longer being manufactured and are harder to obtain for the USDB. Considering these facts and future needs, a decision was made to develop the USD II and offer it as an upgrade. There is an adapter harness that comes with the USD II making it compatible with any test harnesses already built-up for the USDB; also, any software currently written for the USDB will be compatible with the USD II. These updating features -- the adapter harness and the software compatibility -- makes it easy to swap one unit out for the other.

1.3 General Description and Features of the USD II

There are many significant features in the USD II. One, is that it has modular packaging. Each solenoid driver is housed in a separate plug-in module which easily slides into the front of a 19 inch mother rack. This feature allows for convenient

1.4 Updating the USD II When New Driver ICs Become Available

As new driver ICs become available, the USD II adapts by exchanging the new driver cards with the previous ones. The driver cards are easily removable and replaceable for convenient updating. Obtaining the new driver cards is done either by contracting an engineering vendor and having them designed and produced, or by designing and producing them in-house.

1.5 Future Plans for the USD II

There are future plans to add more features to the USD II through updating the software. Two features are already designed into the hardware and only require a software update to implement. One, is adding FLASH programming capability. This feature allows the user to perform software upgrades without having to remove EPROMs to program the memory, but to allow the FLASH memory to be programmed through the RS-232 port. Two, is adding expandability. This feature will allow up to eight racks to be daisy-chained together, expanding its capability to 56 driver modules.

Other plans are to establish a service support group through Global Electronics Inc. They will be responsible for supplying new units, along with being responsible for repairs, technical support, and software updates. This approach to the support issue is planned to improve turn-around-time and to reduce down-time.

1.6 About This Manual

This manual covers the necessary information to fully use and maintain the USD II; however, it assumes that the user is familiar with using a Personal Computer and with using Windows and DOS based software.

This manual features sections on the theory of operation, hardware configuration, and software protocol. Also, there is information on the two software packages developed for easily sending commands to the USD II. Furthermore, there are appendixes that expand on the items discussed along with parts lists, schematics and harness diagrams. Also included are sections on troubleshooting problems and how to perform a full functional test with calibration. This manual is written to be comprehensive so as to allow the user to easily set-up, operate and maintain the USD II.

Section 2 - System Description

2.1 Introduction to the USD II's System Components

The USD II comes equipped with a mother rack, a driver module -- or modules, depending what was ordered with the unit -- a power harness, and a solenoid harness. Optionally, there may be a solenoid rear adapter harness for those who are upgrading from the original USDB. The remaining related parts of the system are a host computer, a power supply and the operating software, which sends the commands to the USD II. Below is a description of each system component and how it relates to the system as a whole.

2.2 The Function of the Mother Rack

The main function of the mother rack is to house the driver modules, and interconnect them through the mother board; although, the mother rack has other major functions as well. It provides a means to connect and distribute power to the driver modules, it provides the circuitry for the RS-232 interface and the rack IDs -- which are three jumpers on the mother board used to identify how many racks are daisy-chained together -- and finally, to provide a means of attaching the power and solenoid harnesses.

2.3 The Function of the Driver Modules

The function of the driver modules are to provide an interface between the solenoids and the user software, which commands the solenoids. The driver modules have two circuit boards, a smart card, and a driver card. The smart cards are universal and can be used along with any USD II driver card. The smart card is the heart of the system. Its functions are to communicate to other smart cards, send commands to the driver cards, and to output LED status signals to the user. There is also a digital to analog converter chip on the smart card to provide the user with a voltage output signal that is proportional to the duty cycle sent to the solenoid. This signal can be used as an input to an oscilloscope, or any other type of equipment, to allow the user to see exactly what is being sent to the solenoid.

2.4 A Description of the Power Harness

The power harness supplies +12v and ground from the power supply to the mother board through a 20 pin Edac connector. This harness is made up of six wires, three for power and three for ground.

2.5 A Description of the Solenoid Harness

The solenoid harness carries the driving current from the mother board to the solenoids through a 56 pin Edac connector. This harness is made up of twenty four wires; twelve for solenoids; eight for transmission ID and four for ground.

2.6 A Description of the Rear Adapter Harness

The rear adapter harness converts the USD II's power and solenoid Edac connectors to the USDB's power and solenoid Amp connectors. This adapter harness is to accommodate those who are upgrading from the USDB to the USD II.

2.7 A Description of the Power Supply Requirements

The power supply needs to deliver +12 volts at 20 amps. These requirements are to meet the demands of the worst case situation while driving the solenoids. The power supply feeds power to the USD II and the solenoids, so any loading down of the power supply will cause performance problems in the USD II, this is why it is recommended to use a 20 amp power supply.

2.8 A Description of the Host Computer

The host computer can be any computer with an RS-232 output. The host computer is used to send commands to the USD II via the RS-232 serial link.

2.9 A Description of the Operating Software

Section 3 - Hardware Description

3.1 Face Plate Description

Each driver module is a subsystem component. Below is a description of the face plate features on each module and what they mean (see figure 1).

Analog Out BNC: This is an output connector that supplies an analog voltage for an oscilloscope or multimeter hook-up. This allows the user to measure the voltage delivered to the solenoid.

Solenoid Status LEDs: These LEDs show when the solenoids are activated. There are four LEDs to indicate the status of the on/off solenoids because a single on/off driver module drives four solenoids. In the case of the VFS, PWML and PWMH solenoid modules, which drive just one solenoid, only LED number one will light when the solenoid is active, indicating a good status.

Power LED: The power LED indicates there is power delivered to the module.

HI Fuse and LO Fuse OK LEDs: These two LEDs light when the fuses on the daughter cards are operational. In the case of a open fuse the corresponding LED will be off and the solenoid OK LED will also be off.

Figure 1: Front view of a driver module

3.2 Description of the Mother Board Connectors

The rear of the mother board has several connectors identified below (see Figure 2). Following is a description of each connector's function.

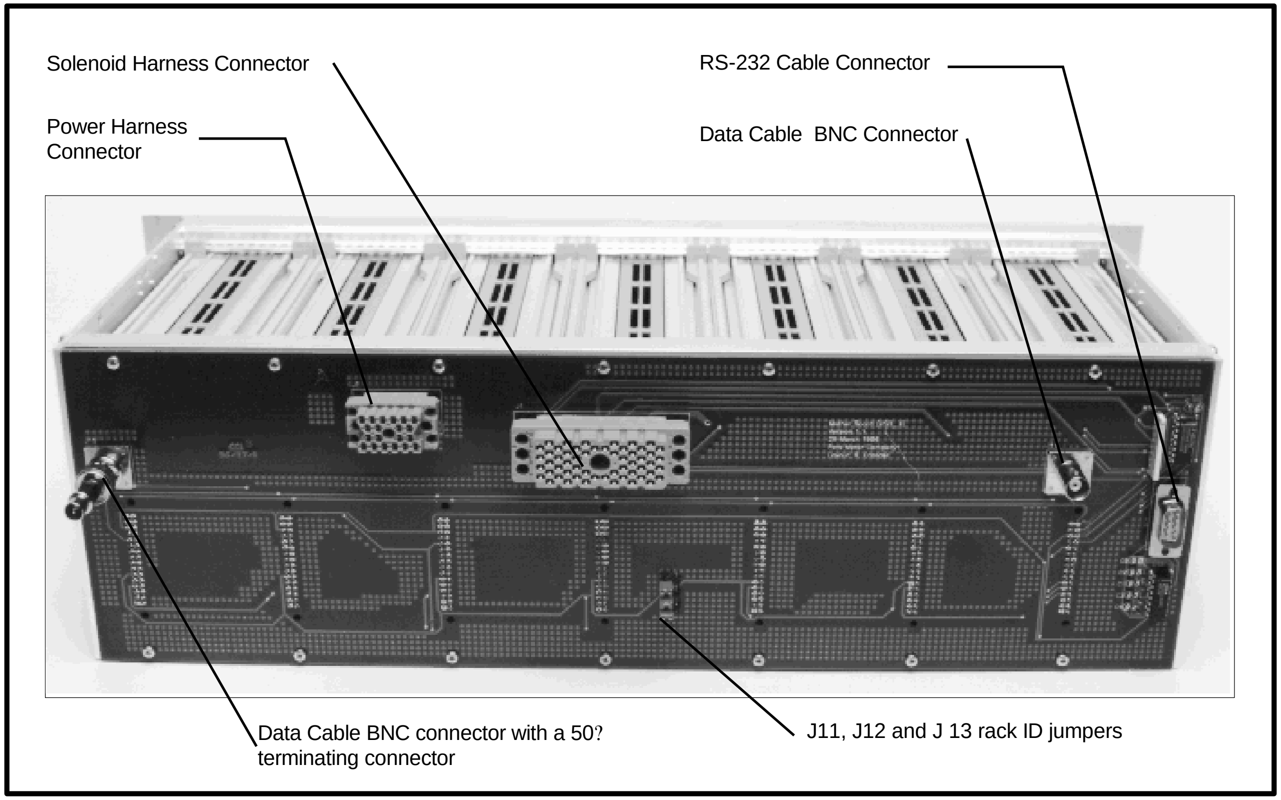


Figure 2: Rear view of the USD II, showing each connector

<u>Solenoid Harness Connector</u>: This is a 56 pin female Edac connector to connect the solenoid harness which supplies power to the solenoids.

<u>Power Harness Connector</u>: This is a 20 pin female Edac connector to

<u>50? Terminating Connector</u>: This is a BNC plug with a 50 ohm resistor across it. This is used to terminate an open BNC receptacle.

<u>Rack ID Jumpers</u>: These are jumper pins that are shorted together to give each rack a unique ID for purposes of daisy chaining.

3.3 Description of the USD II Rack

The USD II rack is a 19 inch modular system. The rack accommodates modular components for mounting in 19 inch electronics cabinets, cases and chassis. In addition it houses the mother board and connectors for the plug-in driver modules.

Section 4 - Theory of Operation

4.1 Overview of the Software Strategy

Communication between the USD II and the host computer is done by sending commands from the host computer and receiving a messages back from the USD II. A description of the communication process follows.

The host computer controls solenoids by sending commands (made up of three bytes) to the USD II, but before any commands can be sent to the driver slots, the slots must be verified. After the USD II receives commands via the RS-232 it echoes the commands back so the host computer can validate them. If the host computer detects an error in communication, the host computer will clear the command queue of that byte. Once the command is validated the USD II will send the command to the verified driver slot for execution.

The USD II communicates back to the host computer by sending messages (made up of one byte). There are two types of messages, one is a slot message and the other is a system message. Slot messages are used to send back information to the host computer regarding the slot status' such as," solenoid present OK" and "slot verified." System messages are used to send back information to the host computer regarding the USD II's system status' such as "slot not verified," and "valid function complete." As can be seen, the general way the software operates between the host computer and the USD II is by sending and receiving commands; thus, controlling the solenoids.

4.2 Smart Card's Theory of Operation

The smart card is an interface between the driver card and the host computer. It is

4.3 Driver Card's Theory of Operation

The driver cards contains Ford Motor Company's proprietary ICs. There are four types of driver ICs currently. They are the ON/OFF, the VFS, the PWML, and the PWMH. Each driver IC has its own unique signal requirements to function. The driver ICs receive their required signals from the smart cards, which in turn activates the solenoids. Below is a brief description of how each driver IC functions:

ON/OFF Driver Card The ON/OFF driver ICs require serial peripheral interface (SPI) communication. This communication involves sending a serial command to the driver IC and receiving a diagnostic word back; thus, controlling the solenoid.

VFS Driver Card The Variable Force Solenoid (VFS) driver IC requires a VFS serial data signal, which controls the average current output going to the solenoid.

PWML Driver Card The PWML driver IC requires a duty cycle signal. Control of the solenoid current is determined by the relationship between the ON time and the OFF time of the IC. So by changing the duty cycle sent to the IC, control over the solenoid current is achieved.

PWMH Driver Card The PWMH driver IC also requires a duty cycle signal. Its current output is controlled by the relationship between the ON time and the OFF time as well; thus, controlling the solenoid current.

4.4 Mother Board

The mother board had six main functions; the first, is to act as a signal buss between the driver modules; the second, is to distribute the power going to the modules and the

Section 5 - Preparation for Use

5.1 Setting-Up the USD II for Operation

To set-up the USD II for operation the following items are required: a host computer (i.e., IBM compatible, or any computer with an RS-232 output), a solenoid cable, an adapter harness (used to upgrade from the USDB), a power cable, the system software, a power supply (rated for +12V at 20 amps) and the USD II.

5.2 Procedure to Prepare the System for Use

Following are the steps to set-up the system for operation:

1. Install the system software into the host computer. (This maybe a program the user has written or one of the two programs developed by APTED.)

2. Connect the 56 pin solenoid cable, power cable and RS-232 cable into their mating connectors located on the USD II's mother board (see figure 2 from section 3 to identify the location of the connectors).

Note: The solenoid cable and the power cable are KEYED at the corners and care should be taken when inserting them. The connectors should insert and screw down easily without force.

3. Set the power supply for +12V operation and adjust the current for maximum output. With the power supply OFF, plug the opposite end of the power cable, coming from the USD II, into the power supply output connectors.

> **Note: Leave the power supply OFF until the remaining steps are completed to avoid damage to the USD II.**

4. Pug the opposite end of the RS-232 cable into the host computer and the opposite end of the solenoid cable into the transmission, valve body or test stand connector.

5. Insert the driver modules into the USD II's mother rack for the solenoid configuration needed i.e., if you need to drive four ON/OFFs, one PWML, one PWMH, and two VFS solenoids, then select the driver modules that correspond to the solenoids and insert them into the slots that relate to the attached solenoids (see the solenoid harness appendix showing how the solenoid wires relate to the slots.)

> **Note: Only slot 0 and slot 1 are configured to drive multiple ON/OFF solenoids. IF you put an ON/OFF solenoid driver module into a slot other than 1 or 0, it will only drive one ON/OFF solenoid.**

> **Note: Slot 0 must have a driver module inserted in it in order for the USD II to function i.e., if only one solenoid is required in the set-up, then it must be located in slot 0. If more than one solenoid is required in the set-up, then one of the driver modules must be in slot 0.**

5.3 Start-up Sequence

It is important to adhere to the following steps in sequence when powering up the system for operation. First, turn ON the power supply; second, boot up the host user to begin activating solenoids. To start this program, simply double click on its icon. The second package is a DOS based program; although it does require the user to be familiar with the hexadecimal commands, this program is good for troubleshooting system failures as well as driving the solenoids. To run this program, simply type the following command at the C prompt >c: wcom9. There are more details about each program in the software sections of this manual. Also there are detailed descriptions of the commands for those who want to write their own code.

5.5 Connecting Two USD IIs Together

There may be situations that require more than seven driver modules. This can be accommodated by connecting two USD IIs together, which expands the number of solenoids that can be driven. To double its capacity, connect two USD IIs together. This is done by connecting a coaxial cable between two USD IIs (see figure 3) and changing the rack IDs via the jumper pins.

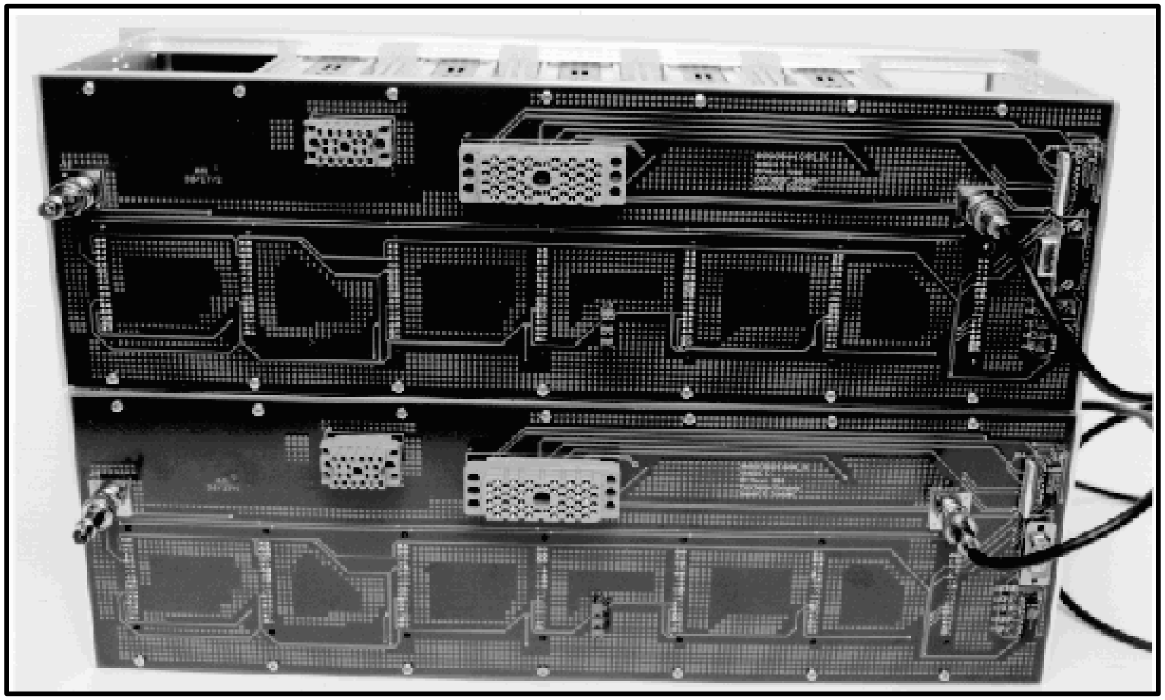


Figure 3. Pictorial showing two USD IIs daisy chained together

Section 6 - Software Protocol

6.1 Description of Commands

The host computer sends commands to the USD II. A command is used to control the USD II. Each command is made up of three bytes, except the “verify slot” command which requires a fourth byte. The categories of commands section discusses the three different types of categories, and details the three bytes required to issue a command from each category.

6.2 Description of Messages

The host computer receives messages from the USD II. A message can be a string of bytes, or just one byte, that informs the host computer of what is actually happening with the system. A message is either a system message or a slot message. A system message begins with a byte in the form 6(x) hex, or 0110 (xxxx) binary (ranging through 60 hex through 6F hex, or 0110 0000 binary through 0110 1111 binary). The system message byte section discusses the two categories of system messages. Each slot message begins with a byte of the form 5(x) hex, or 0101 (xxxx) binary (ranging through 50 hex through 5f hex, or 0101 0000 binary through 0101 1111 binary). The slot message byte section discusses the slot messages.

transmission is detected (i.e. the byte sent did not arrive at the USD II), the host computer must remove bytes from the command queue until the incorrect byte is removed. Note that this is done by sending the error command. This is described further in the command description section of this manual.

> Note: When you issue the error command, the USD II removes only the last byte placed onto the command queue. Thus, to remove an incorrect byte from the command queue, the host computer may have to send several error commands. This is illustrated in the command queue section of this manual.

> Note: Each command that is received will be placed in a command queue. The command queue will hold two 3-byte commands. This must be done so that no command is implemented until the host computer has determined that no error in transmission has occurred. If the third command is not an error command, then the first command will be implemented. See the discussion on the command queue for further details on its operation.

6.3 General Information Required to Control the Driver Boards

To standardize operation and instructions for all solenoids, all controls are based on a 10-bit percent duty cycle and all time is based on cycles.

Percent duty cycle is entered as a 10-bit number. This is shown in the equation below. Note that 000 Hex = 0%, and that 3FF Hex = 99.9%.

$$PercentDutyCycle = \frac{(10 - bitnumber) * 100}{1024} \%$$

Since PWM solenoids are based on the idea of percent duty cycle, using 10 bits to represent the duty cycle is straight forward. Note that the PWM solenoids cannot be

On/Off solenoids use the most significant four bits to control the four On/Off solenoids (B_9=Solenoid 4, B_8 =Solenoid 3, B_7=Solenoid 2, and B_6=Solenoid 1). The Sweep Voltage for an ON/OFF slot is shown in the table on the following page.

Timing for **ALL** solenoids is based on the concept of entering a number representing the number of cycles you want the solenoid to stay at a given duty cycle. The time duration of one cycle is based on the two parameters rate and frequency.

$$\text{Cycle} = \frac{15 * \text{Rate} * \text{Frequency}}{9216} \text{ milliseconds}$$

This cycle time defaults to 15 milliseconds on power up for all solenoids.

Table 1: On/Off Sweep Output Voltage

Sweep Voltage	Solenoid 4	Solenoid 3	Solenoid 2	Solenoid 1
0	Off	Off	Off	Off
.3125	Off	Off	Off	On
.6250	Off	Off	On	Off
.9375	Off	Off	On	On
1.2500	Off	On	Off	Off
1.5625	Off	On	Off	On
1.8750	Off	On	On	Off
2.1875	Off	On	On	On
2.5000	On	Off	Off	Off
2.8125	On	Off	Off	On
3.1250	On	Off	On	Off
3.4375	On	Off	On	On

Section 7 - Command Queue

7.1 Purpose of the Command queue

The purpose of the command queue is to hold commands until they have been validated. This is necessary because all error checking is done at the host computer (the USD II merely performs a local echo). To ensure that incorrect commands are not executed, they must be stored until the host computer has determined that all of the bytes sent have arrived without error to the USD II (by checking the bytes echoed back).

7.2 Operation of the Command Queue

As a byte is received it is placed in the command queue and echoed back to the host computer. The host computer must compare the byte echoed from the USD II with the byte that was sent. Thus, the host computer may be checking the third byte (the last byte) of a command as it is sending out the first byte of the next command. If an error is detected, a command error must be sent (after all three bytes of the second command have been sent). Therefore, the USD II holds two commands in its queue. If the third command is not a command error, then the USD II will execute the first

When the host computer detects that an error has occurred, it must send a command -- error.

When a command -- error -- has been received, the USD II deletes the last byte on the command queue.

This type of error checking is implemented to ensure that no incorrect commands are executed.

7.4 An Illustration of How the Command Queue Operates

It is desired to set the duty cycle of driver 5 to 1023, and the frequency of driver 1 to 60Hz. In this example, we will assume that an error occurs on the 3rd byte of the first command.

> Note: The Host Computer determines the error in the 3rd byte of the first command when sending the 1st byte of the second command. Thus, the third command would be used to tell the USD II that an error has occurred.

> Note: The -- error -- command bytes are not placed on the command queue. Also note that the -- error -- command deletes only one byte at a time.

> Note: The message #6A Hex -- valid function received -- is sent after the third control byte is echoed.

> Note: The second command must be re-sent.

0000 0001	1000 0100 1000 1010	Set frequency

7.5 A Command Queue Example for the Above Illustration

DESCRIPTION	SENT BIN	SENT HEX	RECEIVED	ECHOED	MESSAGE	QUEU BIN	QUEUE HEX
? ***POWER ON***	NONE	NONE	NONE	NONE	00H	EMPTY	EMPTY
? ***COMMAND 1 -- SET DUTY CYCLE OF DRIVER IN SLOT 5***							
CONTROL:	0000 0101	05H	0000 0101	NONE		0000 0101	05H
INSTR:	1111 0000	F0H	1111 0000	0000 0101		0000 0101 1111 0000	05H F0H
DATA:	1111 1111	FFH	1011 1111 (Error)	1111 0000		0000 0101 1111 0000 1011 1111	05H F0H BFH
? ***COMMAND 2 -- SET FREQUENCY OF DRIVER IN SLOT 1***							
CONTROL:	0000 0001	01H	0000 0001	1011 1111		0000 0101	05H

DESCRIPTION	SENT BIN	SENT HEX	RECEIVED	ECHOED	MESSAGE	QUEU BIN	QUEUE HEX
						1111 0000 1011 1111 0000 0001	F0H BFH 01H
INSTR:	1000 0100	84H	1000 0100	0000 0001		0000 0101 1111 0000 1011 1111 0000 0001 1000 0100	05H F0H BFH 01H 84H
DATA:	1000 1010	8AH	1000 1010	1000 0100		0000 0101 1111 0000 1011 1111 0000 0001 1000 0100 1000 1010	05H F0H BFH 01H 84H 8AH
? ***COMMAND -- ERROR***							
CONTROL:	0111 0000	70H	0111 0000	0000 1010		SAME	SAME
CONTROL:	0111 0000	70H	0111 0000	0111 0000		SAME	SAME

						0000 0001	01H
? ***COMMAND -- ERROR***							
CONTROL:	0111 0000	70H	0111 0000	0111 0000		SAME	SAME
CONTROL:	0111 0000	70H	0111 0000	0111 0000		SAME	SAME
CONTROL:	0111 0000	70H	0111 0000	0111 0000	6AH	0000 0101 1111 0000 1011 1111	05H F0H BFH
? ***COMMAND -- ERROR***							
CONTROL:	0111 0000	70H	0111 0000	0111 0000		SAME	SAME
CONTROL:	0111 0000	70H	0111 0000	0111 0000		SAME	SAME
CONTROL:	0111 0000	70H	0111 0000	0111 0000	6AH	0000 0101 1111 0000	05H F0H

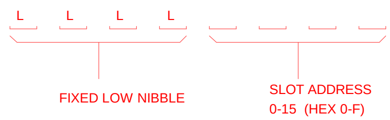
Section 8 - Categories of Commands

8.1 Command Categories

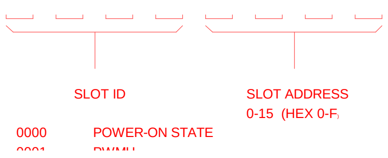
The three categories of commands for the USD II are shown on the following page. Note that this is an overview of the major categories of commands, and not the specific values of the bytes to be sent.

There are five types of bytes that make up a command. They are the **control byte**, the **instruction byte**, the **data byte**, the **verify byte**, and the **init byte**. These are shown below. Note that the verify byte is only sent following three identical control bytes of value #72 hex. All four bytes are required to implement the -- verify slot -- command. Also, note that the init byte is only sent following three identical control bytes of value #77 hex. All four bytes are required to implement the --initialize slot -- command.

INIT BYTE



VERIFY BYTE



8.2 Administrative Duties

The administrative type of command is used by the host computer to perform system functions concerning the entire USD II. For example, if the host computer wanted to know the status of all the slots, the host computer would issue the -- send slot status-- command. This would cause the USD II to send the slot status byte for each of the slots. This type of command is made up of three identical control bytes of the form #7x hex.

CONTROL CONTROL CONTROL (VERIFY)

8.3 Set Slot Parameters

The set slot parameters type of command is used by the host computer to configure the -- set parameters -- of an individual slot. The -- set parameters -- of a slot dictate how that slot will operate. The set parameters are: Set Duty Cycle; Start Duty Cycle; Stop Duty Cycle; Sweep Step Size; Sweep Step Width; End Delay; Down Count; Rate; Frequency Divider; and Mode of Operation.

This type of command is three bytes: a control byte, an instruction byte, and a data byte.

CONTROL INSTRUCTION DATA

8.4 Set Slot Action

The set slot action type of command is used by the host computer to put an individual slot into a particular mode of operation. The modes of operation are: Re-Initialize; Fixed Mode; Sweep Continuously; Sweep Continuously Between Limits; Sweep Once From Start to Stop; Go to Start, Down Count, Go to Stop (Step Mode); Return Wart ID; Sweep From Current to Stop, Increment stack; From Current, Down Count, Go to Stop, Increment Stack; and Sweep Once From Start to Stop, Increment Stack.

This type of command is three bytes: a control byte, an instruction byte, and a data byte.

Section 9 - Control Byte

9.1 Purpose of the Control Byte

The command byte is used to tell the USD II what to do. The commands are shown below, with a discussion of each on the following pages.

9.2 Bit Designation

The most significant bit (MSB = COMM.7) of the command byte is low. This permits the USD II to verify that it is a command byte.

The remaining bits of the high nibble (COMM.6, COMM.5, COMM.4) are used to indicate the command. The table below summarizes the available commands. These commands are described on the following pages.

The low nibble (COMM.0 through COMM.3) are used to indicate the address of the driver that you wish to control. Note that this permits the possibility of 16 drivers. This nibble also indicates which message is received. Also, this nibble indicates which function you wish to implement.

CCC	CONTROL	DESCRIPTION
000	Send	This command sends an instruction to the Driver located at address AAAA.
001	To All	This command sends the instruction to all slots. For this command, AAAA is ignored.
010	Not Used	
011	Not Used	

CCC	CONTROL	DESCRIPTION
100	Not Used	
101	Slot Message To Host	This action sends info to the Host Computer regarding the individual slots. Info such as wart ID and software version number are currently sent. The Host Computer should never send a byte of the form #5x Hex.
110	System Message To Host	This is reserved for messages sent from the USD II to the Host Computer. The Host Computer should never send a byte of the form #6x Hex. If the USD II receives any bytes of the form #6x Hex, then the echoed byte will be #6E Hex, indicating that there was an error in transmission.
111	Admin Duties	This command executes a function indicated by the four bits AAAA. It is possible to have up to 16 functions. To execute one of these functions, the byte must be sent three successive times. If there was an error while sending a function byte, first finish sending the three bytes (should receive an "Invalid Function" message - #68 Hex). Then, re-transmit the three function bytes.

AAAA	FUNCTION	DESCRIPTION
0000	Error	ERROR This function is used to remove a bad byte from the command queue. If an error was found during transmission, this function must be executed to remove the bad byte. Note that only the last non-function byte transmitted is removed (refer to the example in the command queue section).
0001	Flush FLUSH	COMMAND QUEUE This command will try to execute the commands remaining in the command queue. If a command cannot be dealt to the slot, then the command is thrown out, and a "General Error" message (#60 hex) is sent to the Host Computer. The process continues until the command queue is empty. Items on queue must be

CCC	CONTROL	DESCRIPTION
111	Admin Duties	CONT'D

AAAA	FUNCTION	DESCRIPTION
0010	Verify	VERIFY SLOT COMMAND This function is used to verify a slot. You must verify a slot prior to controlling it. The "verify byte" (refer to the "verify byte" section of this manual) must be sent immediately following this function. This byte tells the USD II which slot to verify, and what type of wart the Host Computer thinks is in that slot.
0011	Send Status	SEND SLOT STATUS BYTES This will send the "slot status byte" (refer to the "slot status byte" section of this manual) for 8 slots (16 slots if there is a daughter rack installed).
0100	Trans ID	SEND TRANSMISSION ID This will obtain the Transmission ID encoded in the harness, and send it to the Host Computer.
0101	Show Comm Queue	SHOW COMMAND QUEUE This will send the Host Computer the current contents of the Command Queue. This may be useful when errors occur.
0110	Control Board Version Number	CONTROL BOARD VERSION NUMBER This will send the Host Computer the software version number of the Control Board. This will be useful when assistance is required from ATDD.
0111	Initialize Slot	INITIALIZE SLOT COMMAND This is used to initialize an Individual Slot. The "init byte" must be sent immediately following this function. This byte tells the USD II which slot

CCC	**CONTROL**	**DESCRIPTION**
111	Admin Duties	CONT'D

AAAA	**FUNCTION**	**DESCRIPTION**
1000	Not Used	
1001	Not Used	
1010	Not Used	
1111	Init	INITIALIZE This function is used to initialize the USD II. This function will initialize each of the slots. Upon completion of this command, all of the solenoids will be off, and the command queue will be empty. This command is normally used at the beginning of a test.

Section 10 - Instruction Byte

10.1 Purpose of the Instruction Byte

Each solenoid is capable of 16 different instructions. Presently, with software UNIV ver 6.6, only 12 instructions are defined (codes 0-B & F). All instruction codes except F are used to enter data required for the slot. An instruction F is sent to state what action is to be taken with the data currently entered. If the data you are changing is used by the action that is currently executing, the change will occur almost immediately. For example, if you change the -- sweep step size -- while the action is -- sweep continuously --, the change will take place immediately. If you are setting up a certain sweep step size (or sweep step width, or rate, etc.) while in fixed mode, this will not be seen until the individual slot is put into a sweep mode.

10.2 Instruction Stacks

Each slot has fifty instruction stacks. Every instruction stack contains all of the set parameters for the slot. These values may be changed independently to set up different modes of operation for the slot. Note that these instruction stacks are implemented consecutively by using the appropriate stack actions below (or in any order selected by the host computer if the host computer wants to "override" and manually move the Input and output pointers.)

The host computer does not know exactly when a particular -- sweep once from start to stop -- (or any similar action) will end. The instruction stacks provide the capability of changing modes immediately following the completion of a previous mode. This way, the host computer can chain together a variety of situations, and be sure that when one stops, the next will start immediately (on the next period). Note that if the host computer overrides the input and output pointers, this will negate the timing benefits gained, because the host computer cannot know when to switch the pointers.

10.3 The Number and Size of Steps During a Sweep Function

The value of the duty cycle is composed with 10-bits (8-bits for a VFS), which allows the possibility of 1024 steps (256 for VFS). When the USD II is turned on, the step size is 1, and the rate count is 1024, and the frequency is 66Hz. A sweep action under these conditions will contain 1024 steps, each of size 1. If the rate count is set to a value other than 1024, the step sizes during a sweep will not be totally linear.

10.4 Bit Designation

IIII	INSTRUCTION	DESCRIPTION
0000 (0 HEX)	Set Duty Cycle	This instruction is used to set the **10-bit** duty cycle. The duty cycle is divided into 2^{10} = 1024 parts. The duty cycle is changed to the new value after the instruction is received (on the next cycle). NOTE: The VFS driver chip only uses the most significant **8-bit**s of the 10-bit data. NOTE: The ON/OFFs (four per slot) use the most significant **4-bit**s of the 10-bit data. The most significant bit is ON/OFF 4!!
0001 (1 HEX)	Start Duty Cycle	This will set the **10-bit** start duty cycle that is used by functions: sweep continuously between limits, sweep once from start to stop, and down count (step mode). Once set, this value will always be used unless another value is entered via this instruction, an instruction F, function 0 is entered, or the driver box is powered off and on. NOTE: Refer to function F below.
0010 (2 Hex)	Stop Duty Cycle	This will set the **10-bit** stop duty cycle that is used by functions sweep continuously between limits, sweep once from start to stop, and down count (step mode). Once set, this value will always be used unless another value is entered via this instruction, an instruction F, function 0 is entered, or the driver box is

(3 HEX)

modes. It is an **<u>8-bit</u>** value and is used by instructions: sweep continuously, sweep continuously between limits, and sweep once from start to stop. The duty cycle will be incremented by this value each cycle. This value will always be used until overwritten via this instruction, a re-initialize, or the driver box is powered off and on.

0100 Set Frequency
(4 HEX)

This sets the frequency of the duty cycle based on the formula:

$$\text{Frequency} = \frac{614{,}400}{(N * \text{RATE})}\,\text{Hz}$$

where N is the **<u>8-bit</u>** number entered with this instruction, and rate is the 16-bit number entered with Instructions 7 & 8. The number N can be any number between 4 and 255 (#FF HEX). Numbers less than 4 will be ignored. Note that using a low value for N will increase power dissipation in the driver circuit and may increase power in the solenoid. Set frequency is used by all actions except re-initialize. Once set, this value will always be used unless another value is entered via this instruction, an Instruction F -- action 0 (re-Initialize) is entered, or the driver box is powered off and on. NOTE: If rate is to be changed as well, it must be loaded first.

0101 Down Count Low
(5 HEX)

This **<u>8-bit</u>** value is only used by the action down count (step mode). This value is the lower byte of the two-byte counter used to indicate how many cycles to remain at the start duty cycle before changing to the stop duty cycle.

0110 Down Count High
(6 HEX)

This **<u>8-bit</u>** value is only used by the action down count (step mode). It represents the high byte of the two-byte counter used to indicate how many cycles to remain at the start duty cycle before changing to the stop duty cycle.

$$\text{Duration} = \frac{\text{DownCount}}{-}\,\text{sec}$$

IIII	**INSTRUCTION**	**DESCRIPTION**
0111 (7 HEX)	Rate Byte Low	This **8-bit** value is used with the rate byte high (instruction 8) to build the 16-bit value, rate. This 16-bit value is used to adjust the frequency of the duty cycle based on the formula: $$\text{Frequency} = \frac{614{,}400}{(N * RATE)} \text{Hz}$$ where rate is the 16-bit value entered by this instruction and instruction 8, and N is the 8-bit value entered from instruction 4 (Set Frequency). It is recommended that this value be an integer multiple of 1024 to ensure equal step sizes in sweeping duty cycles. Values less than 1024 will be set to 1024. NOTE: This will not change the frequency immediately. In order to have this value take effect, you must issue an Instruction 4 (set frequency)
1000 (8 HEX)	Rate Byte High	This **8-bit** value is used with the rate byte low (instruction 7) to build the 16-bit value, rate. refer to instruction 7 for details.
1001 (9 HEX)	Sweep Step Width	This **8-bit** value is used to set the sweep step width. The sweep step width is the # of cycles that the individual slot remains at a duty cycle before incrementing to the next duty cycle when in a sweep (or any auto increment) action.

IIII	**INSTRUCTION**	**DESCRIPTION**
1010 (A HEX)	Stack Control	Each slot has a local five-position stack. These stacks are used to generate custom waveforms, or to conduct tests requiring accurate timing. Each stack can hold one action (with the associated set parameters for that action).

1011 End Delay (B HEX)

This **8-bit** value is used to set the amount of time to pause at the stop value (and the start value) of one of the sweep continuously actions, prior to sweeping in the opposite direction. The actual length of time paused, in seconds, is determined using the equation below. This value actually represents the # of cycles to pause, and thus the duration is dependant on the frequency that the solenoid is currently operating at.

$$\text{Duration} = \frac{\text{EndDelay}}{\text{Frequency}}\text{sec}$$

IIII	**INSTR**	**DESCRIPTION**
1100 (C HEX)	Not Used	
1101 (D HEX)	Not Used	
1110 (E HEX)	Reserved	
1111 (F HEX)	Action	The least significant **4-bit**s of the data byte sent with this instruction are used to determine which action to take with the data that has been previously entered.

ACTION

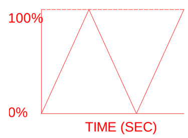
0000 Re-initialize:

0 This action will initialize the Individual slot to the same values it would have on power on.

SET PARAMETER VALUES ON POWER-UP:

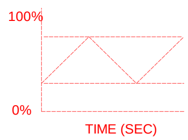
Set Duty Cycle = 0 (10-bit)
Start Duty Cycle = 0 (10-bit)
Stop Duty Cycle = 0 (10-bit)
Sweep Step = 1 (8-bit)
Sweep Width = 1 (8-bit)
End Delay = 85H (8-bit) (2sec @66Hz)
Down Count = 0000H (16-bit)
Mode = FIXED MODE

another action is executing, then the current duty cycle will be used. The frequency is set by instruction 4 (frequency divider).

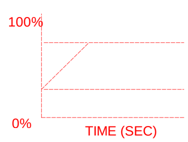


0010 Sweep Continuously:
2 In this mode the individual slot will sweep continuously from 0% to 99.9% to 0% duty cycle. The frequency of the duty cycle is determined by the power-up value (or re-initialize) or instruction 4 (frequency divider). Each cycle, the duty cycle is incremented or decrement by the power-up or re-initialize value (sweep step=1) or the value set with Instruction 3 (sweep step).

<u>ACTION</u>



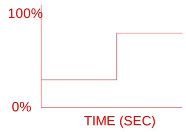
0011 Sweep Continuously Between Limits:
3 In this mode the individual slot sweeps continuously between the start duty cycle and the stop duty cycle. The action only tests to see if the duty cycle is equal to or has exceeded its bounds after the duty cycle has been incremented so it is possible to exceed the bounds by (step size-1). In addition, with sweep steps other than one, the limits depend on the duty cycle it is started from and therefore can vary in overall limits by twice (step size-1). If start duty cycle is equal to the stop duty cycle, then a fixed duty cycle (stop duty cycle +/- (step size-1)) will be output. Changing the start duty cycle or the stop duty cycle while in this mode will cause the sweep to immediately go to the new limits. It may change direction if the new value is already exceeded. The frequency of the duty cycle is determined by the power-up or re-initialize value, or instruction 4 (frequency divider). The duty cycle will be incremented or decrement by the value set with Instruction 3 (sweep step). Note that the power-up value of the step size is 1.



0100 Sweep Once from Start to Stop:
4 In this mode the individual slot immediately jumps to the start duty cycle and sweeps in the required direction to get to the stop duty cycle. The exact stop value may be exceeded by the step size-1. If the start duty cycle is equal to the stop duty cycle then a fixed duty cycle will be output. If the stop duty

power-up or re-initialize value (1), or the value set with instruction 3 (sweep step). At the completion of the sweep the driver box is put into fixed mode.

<u>ACTION</u>



0101 Go to Start, Down Count, Go to Stop (Step Mode):

5 In this mode the individual slot immediately jumps to the start duty cycle and jumps to the stop duty cycle after down count cycles. The value of down count cycles is set by the power-up or to the value determined by instructions 5 and 6 (down count low and high). If down count cycles is 0, then an immediate jump to the stop duty cycle is performed without going to the start duty cycle. The start duty cycle is determined by the power-up value or by instruction 1 (set duty cycle). The stop duty cycle is set by power-up value or instruction 2 (start duty cycle). Changing the down count high, down count low, start duty cycle or the stop duty cycle before the step is complete can cause unpredictable behavior. The frequency of the duty cycle is determined by the power-up value, or instruction 4 (frequency divider). At the completion of the step the driver box is put in fixed mode.

0110 RESERVED

6 This action is reserved

NOTE: The following actions are designed to be "stackable" so that complex waveforms/actions can be generated. Each slot can hold up to five(5) stackable actions. The interface board can hold an additional two(2) for each slot.

All stackable actions wait for the current action to complete prior to starting. Note that if the current action is not a stackable action, then the stackable action will take effect immediately.

Actions 7, 8, and 9 are intended to be chained to together in various combinations to construct custom waveforms to input to the solenoid. The following example will illustrate this.

The waveform to the left was created using Actions 7, 8, and stack, which in this case will have the mode fixed, and the solenoid will remain at that duty cycle.

ACTION

0111 Sweep From Current To Stop / Increment STACK:

7 In this mode, the individual slot first copies all set parameters to the next stack (with the mode set to fixed), and increments the Input stack pointer (to ensure that any new set parameters will be placed in the next stack). Then, it begins sweeping at the current duty cycle. This mode will sweep until it reaches the stop duty cycle value (again, previously set). Then, this mode will increment the output stack pointer to point to the next stack. If no new set parameters were entered, then the Individual slot will remain in fixed mode at the sop duty cycle value.

ACTION

1000 From Current, Down Count, Jump to Stop / Increment STACK:

8 in this mode, the individual slot first copies all current set parameters to the next stack (with the mode set to fixed), and increments the Input Stack pointer (to ensure that any new set parameters will be placed in the next stack). Then, it will start from the current duty cycle, stay there until it has counted the value stored in the 2-byte down count, then jump to the stop duty cycle. Then, this mode will increment the output stack pointer to point to the next stack. If no new set parameters were entered, then the individual slot will remain in fixed mode at the stop duty cycle.

1001 Sweep Once From Start To Stop / Increment STACK:

9 In this mode, the individual slot first copies all set parameters to the next stack (with the mode set to fixed), and increments the input stack pointer (to ensure that any new set parameters will be placed in the next stack). Then, it jumps to the start duty cycle, and begins sweeping in the required direction to get to the stop duty cycle (same as action 4, sweep once from start to stop). Then, the output stack pointer is incremented to

B

1100 Not Used

ACTION

1101 Not Used
D

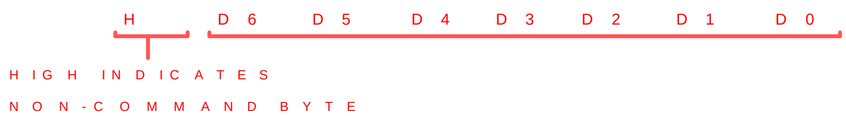
1110 Deactivate Preview Mode
E this action will undo the activate preview mode (#F below).

1111 Activate Preview Mode
F Setting a slot in this Mode will enable the user to verify a test prior to actually driving the solenoids. this mode will only send the output to the sweep out for that slot. NOTE: The harness must still be hooked up. To exit this mode, the slot must be reset, this mode will not drive the solenoids.

Section 11 - Data Byte

11.1 Purpose of the Data Byte

This byte is used in conjunction with the instruction byte. Refer to the discussion of the instruction byte for more information.



11.2 Bit Designation

The MSB is high to indicate a non-command byte.

The remaining seven bits, in conjunction with bits instruction.4, instruction.5, instruction.6, make up the 10 bits of data required with some of the instructions listed above.

Section 12 - Verify Byte

12.1 Purpose of the Verify Byte

This byte is used, in conjunction with the "verify slot" function (#72 hex), to verify a slot. This byte is sent by the host computer immediately after the three function bytes. This byte contains the slot # that is to be verified, and the solenoid ID that the host computer thinks is out there. The USD II will get the ID from the slot and place that value in the slot status byte (this is what is actually in the slot). Then, the USD II will send a message to the host computer indicating whether or not the slot was verified.

BIT	DESCRIPTION
7-4	Slot ID:

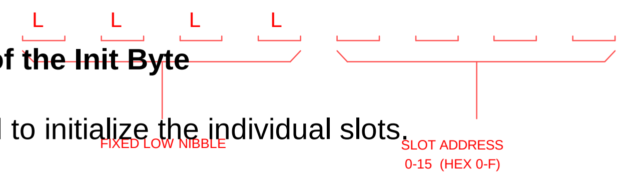
These bits represent the id of the wart board plugged onto the Individual slot located in that slot. The IDs currently defined appear in the table below. Note that when the unit is initially turned on (or re-initialized using command -- function #7F hex), the value of the slot id will be set to 0000 = power-on state.

0000 POWER-ON STATE
0001 PWMH
0010 ON/OFF
0011 VFS
0100 PWML
0101 INVALID
0110 INVALID
0111 Not Used
1000 "
1001 "
1010 "

Section 13 - Init Byte

13.1 Purpose of the Init Byte

This byte is used to initialize the individual slots.



<u>BIT</u>	<u>DESCRIPTION</u>
7-4	Fixed Low Nibble: This should always be sent low.
3-0	Slot Address: This nibble indicates which slot (0 through F Hex) you wish to re-initialize. Note that this will also reset the individual slot queue. Anything in the slot queue will be lost, and must be re-transmitted.

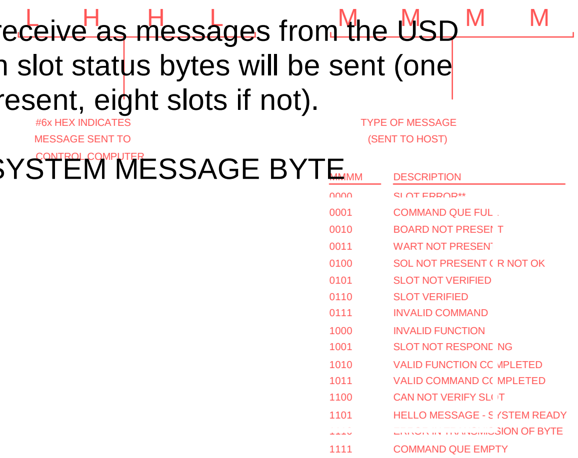
Section 14 - Categories of Messages

The five categories of messages that the USD II will send are shown on the following page. They are the echoed byte, the system information byte, the slot status bytes, slot message byte, and the transmission ID byte. The next sections provide further detail about each of the types of message bytes that the host computer can receive.

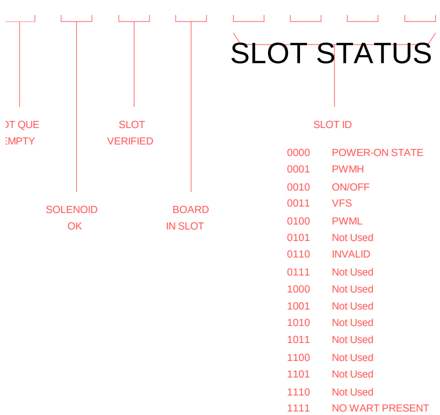
The five types of bytes that the host computer can receive as messages from the USD II are shown below. Note that either eight or sixteen slot status bytes will be sent (one byte per slot) (sixteen slots if the daughter rack is present, eight slots if not).

ECHO BYTE

SYSTEM MESSAGE BYTE



SLOT STATUS BYTE



SLOT MESSAGE BYTE TRANSMISSION ID BYTE

1) ECHOED BYTE

The USD II echoes every byte received from the host computer. It is the host computer's responsibility to verify that the information sent was received correctly by the USD II. This echoed byte is used for that purpose.

2) SYSTEM MESSAGE BYTE

This type of message is used to tell the host computer information about the current status of the system. This type of message will be the primary form of information for the host computer. These messages will be sent back whenever a condition calls for it (such as when the host tries to sixteen bytes long (depending on whether or not a daughter rack is installed). The message contains information about the individual slots in the rack-mount cage.

4) TRANSMISSION ID BYTE

This type of message is sent only when requested by the host computer. The host computer requests this message by sending the command -- send trans id (#74h, #74h, #74h).

5) SLOT MESSAGE BYTE

This type of message is sent only when the host computer requests information from a specific slot. Whenever a byte is sent from the USD II to the host, it is preceded by a 5x (where x represents the slot that is speaking).

Section 15 - Echoed Byte

THIS BYTE IS A DIRECT ECHO OF WHAT IS RECEIVED
BY THE USD II

The USD II will echo every byte that it receives. This is done to permit the host computer to verify that the correct bytes have been transmitted and have arrived at the USD II.

There are no echoed bytes of the form #6x hex or #5x hex (NOTE: There is no command that the host computer can execute with these forms). The USD II will not echo to the host any byte received of the form #6x hex or #5x hex. Instead, it will send a system information byte #6e hex (error in transmission of byte).

Section 16 - Slot Message Byte

PURPOSE:

The purpose of the slot message bytes is to return information that the host computer requests from an individual slot.

A slot message byte will be sent to the host computer only after some information is requested from a slot. The host computer detects a slot message byte by the form #5x hex. The byte immediately following the slot message byte will be the information that the host requested.

Using these messages, the host computer can determine whether or not a requested command has been executed or not, and the results of that command. the types of messages received are described below.

MMMM MMMM	DESCRIPTION
0110 xxxx	This will be sent to the host computer whenever the host requests information from an individual slot. Note that the upper nibble is a #5 Hex, and the lower nibble represents the slot number that is sending the byte.

Section 17 - System Message Byte

<u>PURPOSE</u>

The purpose of the system message bytes is to inform the host computer of the status of the USD II during operation.

A system message byte can be sent to the host computer at any time. The host computer detects a system message byte from other messages by the fact that they are of the form #6x hex. No other byte sent to the host computer will be of this form.

Using these messages, the host computer can determine whether or not a requested command has been executed or not, and the results of that command. The types of messages received are described below.

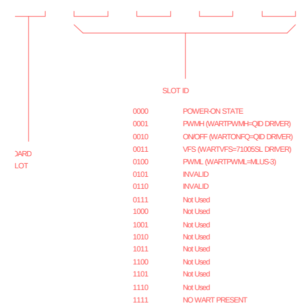
back in place of the echoed byte to indicate that the USD II did not receive that byte. The byte must be sent again.

0010 BOARD NOT PRESENT — This will be sent when a slot does not contain a smarts board. This will happen if you try to verify a slot that does not contain a board, or if the board loses power (blown fuse, etc.) and you try to control that board.

MMMM	MESSAGE	DESCRIPTION
0011	WART NOT PRESENT	This is only checked during a verify slot function (three #72 hex followed by the verify byte). If the USD II determines that there is not a wart board installed on the smarts board, this message will be sent.
0100	SOL NOT OK	This will be sent if the USD II detects a problem with the solenoid for a particular slot. This is checked every time a change to the slot is made (i.e. changing the duty cycle). NOTE: The USD II cannot determine whether the solenoid is bad, or if the fuse has blown, or if the driver is bad, or simply not installed. It can only detect that there is some problem with the solenoid.
0101	SLOT NOT VERIFIED	This will be sent if the host computer attempts to control a slot that has not yet been verified, or if the host computer tries to verify the slot with the wrong solenoid id. You must verify a slot (using the verify slot function, #72 hex, with the verify byte) prior to controlling it. Note, that this will clear bit SSB.5 of the slot status byte.
0110	SLOT VERIFIED	This will be sent to confirm that the slot has been verified. note that this will also set bit SSB.5 of the slot status byte.
0111	INVALID COMMAND	The USD II verifies that each of the three bytes is of the proper form. This message will be sent if any of the MSBs (most significant bits) are incorrect. Either the byte was received incorrectly, or it was sent incorrectly.

1001	SLOT NOT RESPONDING	This will be sent when the USD II is unable to complete communications with a slot. There is a built-in time-out of approximately 1 millisecond. If the slot does not respond within 1 millisecond, this message will be sent to the host computer.
1010	VALID FUNCTION COMPLETED	This message will be sent to indicate that the three function bytes sent did arrive, and that the requested function has been executed.
1011	VALID COMMAND COMPLETED	This message will be sent to indicate that a command has been performed. This can be used to track the status of the command queue. Once this message is received, the three bytes at the top of the command queue have been removed.
1100	CAN NOT VERIFY SLOT	This message will be sent to indicate that the function "verify slot" cannot be performed at this time. The will be sent if the host attempts to verify a slot when the individual slot queue was not empty.
1101	HELLO MESSAGE SYSTEM READY	This message will be sent to indicate that the USD II has completed its initialization routine and is ready to communicate with the Host.
1110	ERROR TRANSMITTING	This message will be sent to indicate that an error has occurred during transmission of a byte. If the USD II receives a byte of the form #6x hex, it knows that this must be an error (there are no commands of this form). This is the only time the USD II can detect an error in transmission. It is the host computer's responsibility to verify valid transmission of bytes by checking the verified bytes.
1111	COMMAND QUEUE EMPTY	This will be sent after the last byte is pulled from the command queue. This message will indicate the completion of the -- flush -- function.

Section 18 - Slot Status Byte



PURPOSE:

This byte is used to provide the host computer with information about each of the 8 slots (16 slots if a "Daughter" rack is installed). The USD II maintains one byte for each slot. These bytes will be sent only when requested by the host computer.

The host computer requests these bytes by using the administrative duties command -- send slot status bytes (#73h, #73h, #73h). The number of bytes returned will be 8 (16 if the "Daughter" rack is installed), regardless of how many individual slots are actually installed in the rack. The first byte sent will correspond to slot 0, followed by a byte corresponding to slot 1, and so on.

BIT DESCRIPTION

If this bit is high, the slot is verified. note that a slot must be verified prior to controlling the slot. when this bit is low an error has occurred.

4 Board Present in Slot:

If this bit is high, there is a board in the slot. If this bit is low, the board may not be completely plugged in, or there may not be a fuse for that slot, or the smarts or wart board may be bad.

3-0 Slot ID:

These bits represent the ID of the wart board plugged into the individual slot located in that slot. The IDs currently defined appear in the table below. Note that when the unit is initially turned on (or re-initialized using command -- function #7f hex), the value of the slot id will be set to 0000 = power-on state.

TABLE 2: Valid Slot IDs

Low Nibble of Slot Status Byte (bits 3-0)	Name of driver and WART Board	Type of Driver in Slot
0000 (0 Hex)	-	POWER-ON STATE
0001 (1 Hex)	QID wartpwmh	PWMH
0010 (2 Hex)	QID wartonofq	ON/OFF
0011 (3 Hex)	71005sl wartvfs	VFS
0100 (4 Hex)	MLUS III wartpwml	PWML
0101 (5 Hex)		Not Used
0110 (6 Hex)		INVALID!
0111 (7 Hex)		Not Used

1101 (D Hex)		"
1110 (E Hex)		Not Used
1111 (F Hex)		NO WART PRESENT

Section 19 - Transmission ID Byte

The transmission id byte is one type of message that the USD II returns to the host computer. This message will be sent back only when requested by the host computer.

The host computer requests this byte by using the administrative duty command -- send transmission id (#74h, #74h, #74h).

The two bytes shown below represent a mother and daughter USD II configuration. Note if a daughter rack is not connected only the mother byte id will be returned. ids of the form 5xx and 6xx should not be used (where xx is the lower nibble of the mother USD II).

Several IDs have been defined. Additional IDs can be added by connecting the correct program ID inputs (P7 through P0) located on the transmission harness. See Appendix for pin-out list.

Note: Connecting the pins Px- to Px+ will cause bit x to be low. Disconnecting pins will cause bit x to be high.

Addendum 1.0
ROC IC

SPI Input Command Data Format

The input command data format is shown in the table below. The command word will set the commanded current.

MSB															LSB
B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
x	x	x	x*	x*	x*	Dither on/off	9 Bit Data Word								

0 = Dither on,1 = Dither off, at reset dither is defaulted to "on."

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
Start Bit Logic 0	OL/SG	VSHT	x*	x*	x*	Previous Dither on/off	Previous 9 Bit Data Word								

Note: The command word could be set with bits B10, B11, and B12 at "0," and use them for fault checking with a software XOR of the fault data.

Addendum 1.0

ROC IC *(continued)*

Instruction Set

0 -	set duty cycle	with 10 bits of data
1 -	set start (low) count	with 8 bits of data
2 -	set stop (high) count	with 8 bits of data
3 -	set sweep count	with 8 bits of data
4 -	set frequency divider	with 8 bits of data
5 -	cycle count low byte	with 8 bits of data
6 -	cycle count high byte	with 8 bits of data
7 -	rate count low byte	with 8 bits of data
8 -	rate count high byte	with 8 bits of data
9 -	sweep width count byte	with 8 bits of data
A -	stack control utility	See list below
B -	sweep width end delay	with 8 bits of data
C -	send pulse at next interrupt	if in pulse mode
D -	get control (8 bit) byte for SPI	
E -	for debug only (sets bits in flag register RB0, R5) with bit 8 set data (bits 0-7) is	

Data - 5	enable short cycle flag, keep switching the output data stack until it finds a function
Data - 6	disable short cycle flag
Data - 7	output pulse at the start of each stack
Data - 8	output pulse at the start of each short cycle
Data - 9	enable pulse output
Data - A	disable pulse output and test mode (normal A/D output)
Data - B	send pulse output at the next interrupt, if pulse mode is enabled
Data - C	output pt. = input pt.
Data ->1F	upper nibble used for pointer, bit 0= - input, 1 - output 70H = reset pointer (20H)
Data ->FF	upper nibble used for pointer, 102 - 1FF set short cycle count (02 - ff)

Must set function 9 to enable pulse output. If enabled and set for pulse at each stack, this will pulse in or out of short cycle mode.

Addendum 1.0
ROC IC *(continued)*

Instruction F (Output Action Commands)

Data 0 -	re-init software
Data 1 -	fixed mode
Data 2 -	sweep continuously
Data 3 -	sweep continuously between start and stop
Data 4 -	jump to start and sweep to stop once
Data 5 -	step mode (go to start and change to stop after down count cycles)
Data 6 -	read data bus ID and output to latch
Data 7 -	sweep to stop once and switch stack
Data 8 -	step mode (after down count change to stop and switch stack)
Data 9 -	sweep start to stop once and switch stack
Data A-	send chip version
Data B-	wart ID error code
Data C-	error count

Appendix A
56 Pin Edac Solenoid Harness

J1

Pin Number	Description
A	Slot 4, solenoid 1
B	+12VDC to slot 5, solenoid 1 and 2
C	Slot 5 solenoid 1
D	+12VDC to slot 6, solenoid 1 and 2
E	+12VDC to slot 4, solenoid 1 and 2
F	+12VDC logic
H	+12VDC logic
J	Transmission ID 0
K	Slot 6, solenoid 1
L	
M	Transmission ID 7
N	

Y	Transmission ID 3
Z	
a	+12VDC to slot 3, solenoid 1 and 2
b	Transmission ID 6
c	
d	Slot 0, solenoid 1
e	Slot 2, solenoid 1
f	Transmission ID 5
h	
j	+12VDC to slot 0, solenoid 3 and 4
k	+12VDC to slot 2, solenoid 1 and 2
l	Transmission ID 4
m	+12VDC to slot 0, solenoid 3 and 4
n	Slot 0, solenoid 2
p	+12VDC Solenoid
r	+12VDC Solenoid
s	+12VDC Solenoid
t	+12VDC Solenoid
u	slot 1, solenoid 4
v	
w	
x	
y	Slot 0, solenoid 3
z	Ground Solenoid
AA	Ground Solenoid
BB	Ground Solenoid
CC	Ground Solenoid
DD	Slot 1, solenoid 3
EE	+12VDC to slot 1, solenoid 3 and 4
FF	
HH	+12VDC to slot 1, solenoid 1 and 2
JJ	Slot 0, solenoid 4
KK	Slot 1, solenoid 2
LL	+12VDC to slot 1, solenoid 3 and 4
MM	Slot 1, solenoid 1

Appendix B
20 Pin Edac Power Harness

J2

Pin Number	Description
A	+12VDC SOL
B	+12VDC SOL
C	+12VDC SOL
D	+12VDC SOL
E	
F	
H	
J	
K	+12VDC LOGIC

V	
W	GND SOLENOID
X	GND SOLENOID