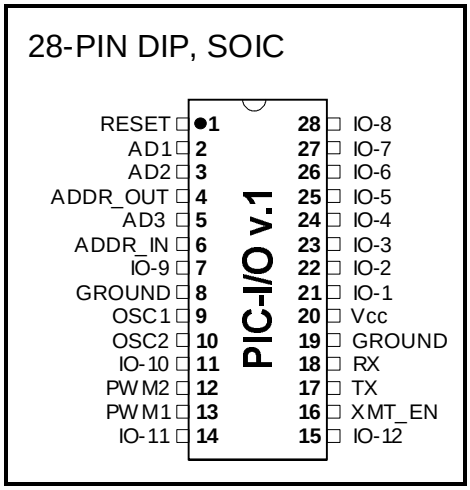
JR KERR
AUTOMATION
ENGINEERING

PIC-I/O

Multifunction
I/O Controller

The ***PIC-I/O*** multifunction I/O controller is compatible with the ***PIC-SERVO*** and ***PIC-STEP*** motor control modules and provides the following I/O capabilities:

- 12 digital I/O lines independently programmable as inputs or outputs
- Three 8-bit analog input channels (0 - 5v input)
- Two 20 KHz PWM output channels
- One 32 bit counter/timer
- RS485 serial interface allows up to 32 modules (***PIC-I/O, PIC-SERVO*** or ***PIC-STEP***) to be controlled from a single serial port
- Test and utility software available



1.0 Pin Description and Packaging

The ***PIC-I/O*** comes in a 28 pin, 0.3" DIP or an SOIC package. Generally, the device operates from a +5v supply and is compatible with TTL and CMOS logic. Please refer to the PIC16C73 data sheet from Microchip for complete electrical and physical specifications.

1.1 Pin Definitions

Pin	Symbol	Description
1	RESET	Reset pin, active low. Connects directly to Vcc for automatic reset on power-up.
2	AD1	Analog input 1 (0 - +5v)
3	AD2	Analog input 2 (0 - +5v)
4	ADDR_OUT	ADDR_OUT output. This signal is initialized HIGH, and drops LOW when a Set Address command is issued.
5	AD3	Analog input 3 (0 - +5v)
6	ADDR_IN	This pin must be pulled low to enable communications. Normally tied to the ADDR_OUT pin of the previous ***PIC-SERVO*** on the same RS485 network.
7	IO-9	General purpose I/O bit. Can be configured as an input or output.
8	GND	Ground connection
9	OSC1	Connects directly to a 20 MHz clock source or to one side of a 20 MHz crystal. See Microchip documentation for details of crystal connections.
10	OSC2	Connects to the other side of a 20 MHz crystal. N.C. if a clock source is used.
11	IO-10	General purpose I/O bit. Can be configured as an input or output.
12	PWM2	19.53 KHz PWM output signal
13	PWM1	19.53 KHz PWM output signal
14	IO-11	General purpose I/O bit. Can be configured as an input or output.
15	IO-12	General purpose I/O bit. Can be configured as an input or output.

16	XMT_EN	This output can be connected to the enable pin of an RS485 driver to enable output over a shared response line. Not used if a point-to-point serial connection (like RS232) is used.
17	TX	Serial transmit output. Connects to the transmit input of an RS485 or an RS232 driver chip.
18	RX	Serial receive input. Connects to the receive output of an RS485 or an RS232 driver chip.
19	GND	Ground connection
20	Vcc	Supply pin, connects to +5v D.C.
21-28	IO-1 - IO-8	General purpose I/O bits. Can be configured as an inputs or outputs. As inputs, these pins have internal 20 Kohm pull-up resistors.

1.2 Ordering Information

Part Number	*Description*
KAE-T2V1-DP	***PIC-I/O*** chip, version 1, 0.3" wide DIP package
KAE-T2V1-SO	***PIC-I/O*** chip, version 1, SOIC package

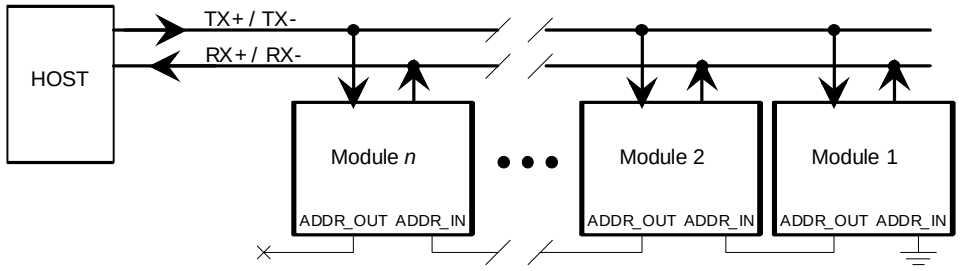
2.0 Theory of Operation

2.1 Communications & Initialization

NMC Communication Protocol

The ***PIC-I/O*** uses the same *full-duplex*, RS232 or RS485 based NMC (*Networked Modular Control*) communication protocol as used by the ***PIC-SERVO*** and the ***PIC-STEP*** controllers. It is a strict master/slave protocol, where command packets are sent to a controller module by the host computer, and a status packet is returned by the module. The default baud rate is 19,200, but it may be changed at any time to up to 115,200. The communication protocol uses 1 start bit, 1 stop bit and no parity.

Command packets are transmitted by the host over a dedicated command line. Status packets are received over a *separate* status line which is shared by all of the modules on the network. Because the host does not have to share the command line, the host communications port can be a standard RS232 port with a simple RS232 to RS485 signal level converter*. The slave ports, however, must be able to disable their transmitters to prevent data collisions over the shared status line. Therefore, all NMC compatible controllers provide an XMT_EN output used for enabling or disabling an RS485 transmitter. Please refer to the figure below.



* If only a single ***PIC-I/O*** controller is used, the ***PIC-I/O***'s communications port can be operated as an RS232 port, with TX and RX connected to an RS232 transceiver rather than to an RS485 transceiver. In this case, the XMT_EN output is not used and may be left open.

The command packets have the following structure:

Header byte (always 0xAA)
Module Address byte (0 - 255)
Command byte
Additional Data bytes (0 - 15 bytes)
Checksum byte (8-bit sum of the Module Address byte thru the last additional data byte)

The Header byte is used to signal the beginning of a command packet. When waiting for a new command, each module will ignore any incoming data until it sees a Header byte.

The Module Address byte is the address of the target module. The address can be an individual address, or the *group* address for the module. (See *Group Commands* below.)

The Command byte is broken up into an upper nibble (4 bits) and lower nibble (4 bits). The lower nibble contains the command value (0 - 15), and the upper nibble contains the number of additional data bytes required for that command (0 - 15). It is up to the host to insure that the upper nibble matches the number of additional data bytes actually sent.

The Additional Data bytes contain the specific data which may be required for a particular command. It is up to the host to make sure that the proper number of additional data bytes is sent for a particular command, and that the upper nibble of the command byte is equal to this number.

Once a module receives a complete packet, and the Address byte matches its address, it will verify the checksum and immediately (within 0.51 milliseconds) begin to process the command. If there is a checksum error in the command packet or any other sort of communications error (framing, overrun), the command will not be executed, but a status packet will still be returned. If there are no errors, the command will then be executed and a status packet returned.

The status packets have the following structure:

Status byte
Additional Status Data bytes (programmable)
Checksum byte (8-bit sum of all the bytes above)

The Status byte contains basic information about the state of the module, including whether or not the previous command had a checksum error. The specific bit definitions for the Status byte are in Section 3.2 below.

The number Additional Status Data bytes is programmable, and may contain information such as input bit values, A/D values, or the module type and version numbers. Exactly which data is included in these Additional Status bytes can be programmed using the Define Status or Read Status commands. On power-up or reset, each NMC module defaults to sending only the Status byte and Checksum byte, with no additional status data.

A command sent to a ***PIC-I/O*** controller is stored in an internal buffer until the end of the communications cycle (0.51 millisec. max.), when it is then executed and a status packet is returned. No new command should be sent until the status packet is received to prevent overwriting the command data buffer and to prevent collisions on the status line. If, however, the

host does send *any* data before a status packet is received, all slaves on the network will disable any status data transmission in progress and listen to the new command from the host. This insures that the host can always command the attention of all slaves on the network.

The Command Reference section below describes the data contained in the command packets and status packets.

Addressing

When multiple modules are connected to the same NMC network, they must be assigned unique addresses. This is done through the use of the ADDR_IN and ADDR_OUT signals on each NMC compatible controller. The ADDR_OUT signal from one controller is daisy-chained to the ADDR_IN signal of the adjacent controller on the network. Customarily, the ADDR_IN pin of the controller furthest from the host is tied to GND, and the ADDR_OUT signal of the controller closest to the host is left open. (See the figure above).

Unique addresses are assigned using the following procedure:

1. On power-up, all modules assume a default address of 0x00, and each will set its ADDR_OUT signal HIGH. Furthermore, a module's communications will be disabled completely until its ADDR_IN signal goes LOW. If the ADDR_OUT and ADDR_IN signals are daisy-chained as described above, all modules will be disabled except for the module furthest from the host.
2. The host starts by sending a Set Address command to module 0, changing its address to a value of 1. A side affect of the Set Address command is that the module will lower the its ADDR_OUT signal.
3. At this point, the next module in line is enabled with an address of 0. The host then sends a command to module 0 to change its address to a value of 2.
4. This process is continued until all modules have been assigned unique addresses.

Initialization of the addresses is performed by the host each time the NMC network is powered up or reset. The host can also use this mechanism to verify that the proper number of modules are present, and that their types match those expected for a particular application.

Once addresses are set, all other operations can be executed.

Group Commands

Each NMC controller module actually has two addresses: an individual address and a group address. On power-up or reset, the individual address defaults to 0x00 and the group address defaults to 0xFF. Both the individual address and the group address are set with the same Set Address Command. Individual addresses can have any value between 0 and 255, but group addresses are restricted to values between 128 and 255.

The purpose of the group address it to be able to send a single command (such as Start Move) to a several controllers at the same time. While the individual addresses of all controllers must be unique, a group of controllers can share a common group address. When a command packet is sent over the NMC network to a group address, all modules with a matching group address will execute the command.

The issue of which modules will send a status packet in response to a group command is resolved with the distinction between group *members* and group *leaders*. When the group address for a module is set, the Set Address command will also specify if the module is to be the leader or a member of that group. If a module is a member of its group and it receives a group command (one sent to its group address), it will execute the command but *not* send back a status packet. If a module is the leader of its group and it receives group command, it *will* send back a status packet in addition to executing the command. (The status packet is just the same as one sent in response to an individually addressed command.)

For any group of modules sharing the same group address, only one should be declared the group leader.

In certain instances (as when changing the Baud rate for all modules on the network), is necessary to send a command to a group without a group leader. In this case, no status will be coming back from any controllers, and the host should wait for at least 0.51 milliseconds before sending another command to keep from overwriting the previous command.

Network Initialization

The previous subsections have hinted at various operations required for network initialization. Here is a specific list of the actions which should be taken on power-up, or after a network-wide reset[†] :

1. Set the host baud communications port to 19,200 Baud, 1 start bit, 1 stop bit, no parity.
2. Send out a string of 16 null bytes (0x00) to fill up any partially filled command buffers. Wait for at least 1 millisecond, and then flush any incoming bytes from the host's receive buffer.
3. Use the Set Address command, as described above, to assign unique individual addresses to each module. At this point, set all group addresses to 0xFF, and do not declare any group leaders.
4. Verify that the number of modules found matches the number expected.
5. Different NMC controller modules will have different type numbers and different version numbers (***PIC-I/O*** = type 2). Use the Read Status command to read the type and version numbers for each module and verify that they match the types and versions expected.
6. Send a Set Baud command to the group address 0xFF to change the baud rate to the desired value. No status will be returned.
7. Change the host's Baud rate to match the rate just specified.
8. Poll each of the individual modules (using a No Op command) to verify that all modules are operating properly at the new Baud rate.
9. Use the Set Address command to assign any group addresses as needed.

At this point you are ready to send any module specific initialization commands to the individual modules and begin operation. Note that at any time, you may use the Set Address command to re-assign group addresses.

[†] For most basic applications which do not use group commands or faster baud rates, only steps 1 and 3 are really required.

2.2 I/O Capabilities

General Purpose I/O Bits

The ***PIC-I/O*** has 12 general purpose I/O bits (TTL/CMOS) which can be programmed as individually as either inputs or outputs. On power-up or reset, all bits default to inputs. As inputs, bits 1-8 have internal 20 Kohm pull-up resistors (bits 9 - 12 have no pull-up resistors). As outputs, all pins can source or sink up to 20 ma, although only 200 ma *total* may be sourced or sunk for the entire chip.

The Set Direction command is used to define which bits should be inputs and which should be outputs. The Set Outputs command can be used to change the state of the outputs immediately, or the Set Synch Output command can be used to load an output bit buffer with values to be set synchronously with other controllers.

The values of the I/O bits can be read using the Read Status command, or may be returned with every status packet using the Define Status command. Note that for bits defined as outputs, the values read should match the output values commanded.

Analog Inputs

The ***PIC-I/O*** has three 8-bit analog inputs. Inputs on the analog input pins should be between 0 and +5v. The analog input values can be read using the Read Status command, or may be returned with every status packet using the Define Status command.

PWM Outputs

The ***PIC-I/O*** has two 8-bit PWM outputs which put out 19.53 KHz square waves of programmable duty cycles. A PWM value of 255 correspond to a 100% duty cycle and a PWM value of 0 corresponds to a 0% duty cycle. On power-up or reset, the PWM outputs are set to 0% duty cycle.

The Set PWM command can be used to set both PWM values simultaneously, or the Set Synch Outputs command can be used to load PWM value buffers for synchronous output with other controllers.

Counter/Timer Channel

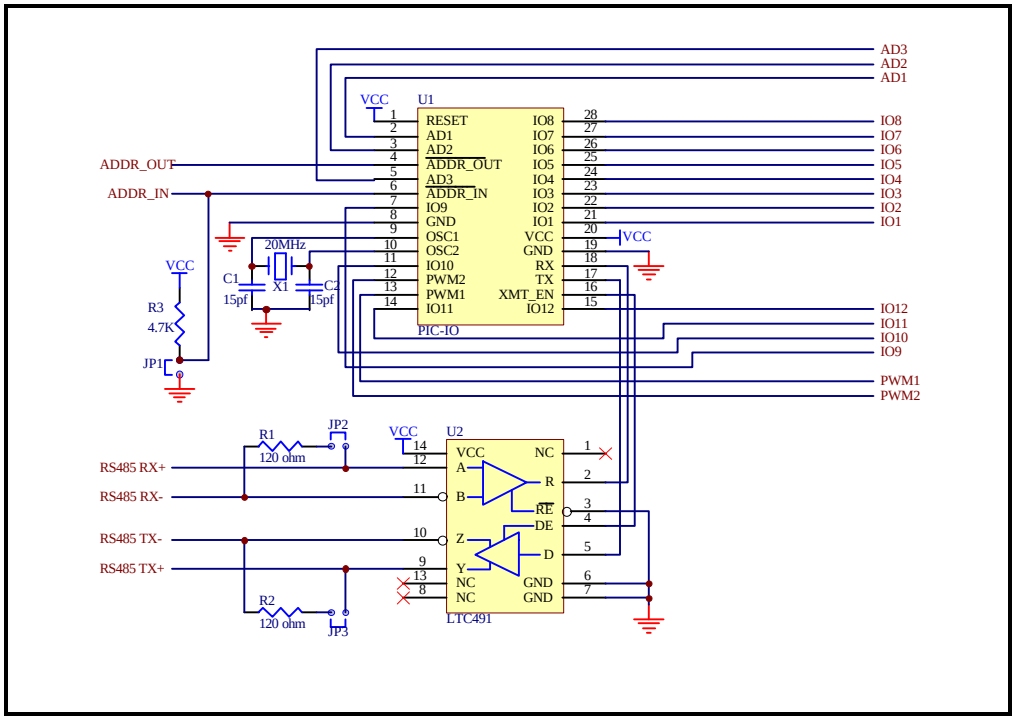
I/O bit 10 of the ***PIC-I/O***, when defined as an input, can be configured to be a 32 bit counter. A rising edge on this pin will increment the counter. The maximum counting frequency is 4 MHz, with a minimum HIGH and LOW time for the input signal of 250 nanoseconds.

If the counter is not used, a 32 bit timer is available. This timer has a maximum timing frequency of 5 MHz. A 2:1, 4:1 or 8:1 pre-scalar may be used with the timer to slow down timer frequency. (Note that this pre-scalar can also be applied in counter, counting every 2^{nd}, 4^{th} or 8^{th} rising edge.)

The counter/timer mode and pre-scalar are set using the Set Timer Mode command, and the Read Status or Define Status commands can be used to have the 32 bit counter or timer value returned in the status data packet.

2.3 Example Circuit

The schematic diagram below illustrates using the ***PIC-I/O*** with an RS485 interface. The I/O pins, A/D input pins and PWM output pins can all be connected to the user's specific control circuitry.



PIC-I/O With RS485 Interface.

3.0 *PIC-I/O* Command Reference

3.1 *PIC-I/O* Command Set

The following section describes in detail the commands available for the ***PIC-I/O***:

Command	CMD Code	# Data bytes	Description
Set Direction	0x0	2	Sets I/O bit directions as inputs or outputs.
Set Address*	0x1	2	Sets the individual and group addresses
Define Status*	0x2	1	Defines which data should be sent in every status packet
Read Status*	0x3	1	Causes particular status data to be returned just once
Set PWM	0x4	2	Immediately set the PWM output values
Synch Output	0x5	0	Outputs previously stored output bit and PWM output values
Set Output	0x6	2	Immediately set the states of the output bits
Set Synch Output	0x7	4	Store output bit values and PWM values to be synchronously set with the Synch Output command
Set Timer Mode	0x8	1	Sets mode of the counter/timer and its pre-scalar
---	0x9	0	Not used
Set Baud Rate*	0xA	1	Sets the baud rate (group command only)
---	0xB	0	Not used
Synch Input	0xC	0	Synchronously stores the input bit values and the counter/timer value
---	0xD	0	Not used
No Op*	0xE	0	Simply causes the defined status data to be returned
Hard Reset*	0xF	0	Resets the controller to its powerup state.

* These commands are common to all NMC compatible devices.

Set Direction

Command value: 0x0
Number of data bytes: 2
Command byte: **0x20**
Data bytes:

1. Direction bits for I/O bits 1-8:
   - Bit 0: Direction bit for I/O bit 1 (0 = output, 1 = input)
   - Bit 1: Direction bit for I/O bit 2
   - Bit 2: Direction bit for I/O bit 3
   - Bit 3: Direction bit for I/O bit 4
   - Bit 4: Direction bit for I/O bit 5
   - Bit 5: Direction bit for I/O bit 6
   - Bit 6: Direction bit for I/O bit 7
   - Bit 7: Direction bit for I/O bit 8
2. Direction bits for I/O bits 9-12:
   - Bit 0: Direction bit for I/O bit 9 (0 = output, 1 = input)
   - Bit 1: Direction bit for I/O bit 10
   - Bit 2: Direction bit for I/O bit 11
   - Bit 3: Direction bit for I/O bit 12
   - Bit 4-7: Don't care

Description:

Sets the directions of the individual I/O bits. On powerup, all bits are defined as inputs. Make sure that any I/O pins defined as outputs are not connected to the output of another device, or else the ***PIC-I/O*** or the other device may be damaged.

Set Address

Command value: 0x1
Number of data bytes: 2
Command byte: **0x21**
Data bytes:

1. Individual address: 0-0xFF (initial value 0x00)
2. Group Address: (initial value 0xFF)

Description:

Sets the individual address and group address. Group addresses are always interpreted as being between 0x80 and 0xFF. If a ***PIC-I/O*** is to be a group *leader*, clear bit 7 of the desired group address in the second data byte; the ***PIC-I/O*** will automatically set bit 7 internally after flagging itself as a group leader. (If bit 7 of the second data byte is set, the module will default to being a group *member*.) The first time this command is issued after power-up or reset, it will also enable communications for the next module in the network chain by lowering its ADDR_OUT signal.

Define Status

Command value: 0x2
Number of data bytes: 1
Command byte: **0x12**
Data bytes:

1. Status items: (default: 0x00)
   Bit 0: send input bit values (2 bytes)
   (the first byte will contain the input values for I/O bits 1-8, the second byte the input values for I/O bits 9 - 12 in the lower nibble)
   1: send A/D 1 value (1 byte)
   2: send A/D 2 value (1 byte)
   3: send A/D 3 value (1 byte)
   4: send counter/timer value (4 bytes, least significant first)
   5: send device type, version number (2 bytes)
   (***PIC-I/O*** device type = 2)
   6: send input bit values captured with the Synch Input command (2 bytes)
   7: send counter/timer value captured with the Synch Input command (4 bytes)

Description:

Defines what additional data will be sent in the status packet along with the status byte. Setting bits in the first data byte will cause the corresponding additional data to be included in the status packet. The status data will always be sent in the order listed. For example if bits 0 and 3 are set, all subsequent status packets will consist of the status byte followed by two bytes of input bit data, followed by the A/D 3 input byte, followed by the checksum. The status packet returned in response to *this* command will include the additional data bytes specified. On power-up or reset, the default status packet will include only the status byte and the checksum byte.

Read Status

Command value: 0x3
Number of data bytes: 1
Command byte: **0x13**
Data bytes:

1. Status items: (default: 0x00)
   Bit 0: send input bit values (2 bytes)
   (the first byte will contain the input values for I/O bits 1-8, the second byte the input values for I/O bits 9 - 12 in the lower nibble)
   1: send A/D 1 value (1 byte)
   2: send A/D 2 value (1 byte)
   3: send A/D 3 value (1 byte)
   4: send counter/timer value (4 bytes, least significant first)
   5: send device type, version number (2 bytes)
   (***PIC-I/O*** device type = 2)
   6: send input bit values captured with the Synch Input command (2 bytes)
   7: send counter/timer value captured with the Synch Input command (4 bytes)

Description:

This is a non-permanent version of the Define Status command. The status packet returned in response to *this* command will incorporate the data bytes specified, but subsequent status packets will include only the data bytes previously specified with the Define Status command.

Set PWM

Command value: 0x4
Number of data bytes: 2
Command byte: **0x24**
Data bytes:

1. PWM 1 output value (0 - 255)
2. PWM 2 output value (0 - 255)

Description:

Immediately set the two PWM output values. A value of 0 will turn off the PWM output driver, a value of 255 will turn it on with a 100% duty cycle.

Synch Output

Command value: 0x5
Number of data bytes: 0
Command byte: **0x05**
Data bytes:

None

Description:

Synchronously set the output bit values and PWM values previously stored with the Set Synch Output command. Note that the command byte value matches the ***PIC-SERVO***'s Start Move command. This means that if a ***PIC-I/O*** module shares the same group address with a ***PIC-SERVO*** module, the Start Move command of the ***PIC-SERVO*** can be synchronized with setting the outputs of the ***PIC-I/O*** module.

Set Output

Command value: 0x6
Number of data bytes: 2
Command byte: **0x26**
Data bytes:

1. Bit values for Output bits 1-8
2. Bit values for Output bits 9-12 (bits 4-7 of byte 2 are ignored)

Description:

Immediately sets the values for the I/O bits defined as outputs. If an I/O bit is defined as an input, the corresponding data byte bit value is ignored.

Set Synch Output

Command value: 0x7
Number of data bytes: 4
Command byte: **0x47**
Data bytes:

1. Bit values for Output bits 1-8
2. Bit values for Output bits 9-12 (bits 4-7 of byte 2 are ignored)
3. PWM1 output value (0-255)
4. PWM2 output value (0-255)

Description:

Stores output bit values and PWM values in internal registers to be set synchronously with the Synch Output command.

Set Timer Mode

Command value: 0x8
Number of data bytes: 1
Command byte: **0x18**
Data bytes:

1. Timer mode configuration byte
   - Bit 0: 0 = Counter/timer disabled, 1 = Counter/timer enabled
   - Bit 1: 0 = Select timer mode, 1 = Select counter mode
   - Bits 5,4: 00 = No prescaler (count every event)
     01 = 2:1 prescalar (every other event counted)
     10 = 4:1 prescalar (every 4^{th} event counted)
     11 = 8:1 prescalar (every 8^{th} event counted)
   - Bits 2,3,6,7: don't care

Description:

Sets the operating mode of the counter/timer. If in counter mode, each rising edge of I/O bit 10 (JP10, pin 5) will be counted. This bit should be set as an input to count external events. In timer mode, the counter counts the ***PIC-I/O***'s 5.0 MHz internal clock. The prescalar applies to both the counter and the timer modes.

Set Baud Rate

Command value: 0xA
Number of data bytes: 1
Command byte: **0x1A**
Data bytes:

1. Baud rate divisor, BRD

sample values:

9600	BRD = 129
19200	BRD = 63
57600	BRD = 20
115200	BRD = 10

Description:

Sets the communications baud rate. All control modules on the network must have their baud rates changed at the same time, therefore this command should only be

issued to a group including all of the modules on the network. A status packet returned from this command would be at the new baud rate, so typically, there should be no group leader when this command is issued. (Note that the host's baud rate must also be changed for subsequent communication.) The baud rate divisor is programmed directly into the PIC16C73's SPBRG register with the bit BRGH = 1. Please refer to the PIC16C7x data sheet for details in obtaining other baud rates.

Synch Input

Command value: 0xC
Number of data bytes: 0
Command byte: **0x0C**
Data bytes:
none
Description:

Causes the current Input bit values and the counter/timer value to be synchronously stored in the ***PIC-I/O***'s internal registers. These values can be read using the Read Status or the Define Status commands. Note that this command has the same value as the ***PIC-SERVO***'s Save Position as Home command. Therefore, input values can be stored synchronously with motor positions by issuing this command to a group containing both ***PIC-SERVO*** modules and ***PIC-I/O*** modules.

No Operation

Command value: 0xE
Number of data bytes: 0
Command byte: **0x0E**
Description:

Does nothing except cause a status packet with the currently defined status data to be returned.

Hard Reset

Command value: 0xF
Number of data bytes: 0
Command byte: **0x0F**
Description:

Resets modules to their power-up state. No status will be returned. Typically, this command is only issued to all the modules on the network using a group command. (All modules must have a common group address.)

3.2 Status Byte Bit Definitions

The first byte of every status packet is the status byte, whose bits contain basic operating information about the module. The ***PIC-I/O*** has only one of the eight bits defined: if Bit 1 of the status byte is set, the ***PIC-I/O*** detected a checksum error in the most recently sent command packet. Bits 0, 2, 3, 4, 5, 6 and 7 are all undefined and can be ignored.

4.0 Contact Information

Additional information may be found from these sources:

J R Kerr Automation Engineering **www.jrkerr.com**

Information about the ***PIC-SERVO*** motor controller and related products including ordering information, product documentation and test software. Technical support is provided via e-mail. Send your questions to **techsupport@jrkerr.com**.

Microchip **www.microchip.com**

Manufacturer of the PIC16C73 microcontroller used for the ***PIC-I/O***. Data sheets containing complete electical and timing specifications can be downloaded from this site.

HdB Electronics **www.hdbelectronics.com**

Distributor of ***PIC-SERVO*** products as well as of other electronic components, accessories and tools.