

IMC23/IMCE23 Command List

Version 1.00

TABLE OF CONTENTS

DT PROTOCOL SYNATAX.....	3
HOMING & POSITIONING	4
VELOCITY & ACCELERATION	4
SETTING CURRENT	5
LOOPING & BRANCHING	5
POSITION CORRECTION MODE.....	7
PROGRAM STORAGE & RECALL.....	7
PROGRAM EXECUTION	7
MICROSTEPPING	7
ON/OFF DRIVERS	7
QUERY COMMANDS.....	8
BAUD CONTROL.....	8
EXAMPLES	9
UNDERSTANDING THE RESPONSE.....	11
APPENDIX 1.....	12
APPENDIX 2.....	14
APPENDIX 3.....	14

IMC23/IMCE Command List

Product: IMC23/IMCE23
Version: 1.00
Date: 03/30/2006

Version History		
Version	Date	Description of Changes
1.00	03/30/2006	Initial Release

Commands to the IMC23 controller can be issued from a HyperTerminal connection. The syntax is intuitive and easy to follow. Setup instructions for HyperTerminal can be found in the IMC23 Manual.

DT PROTOCOL SYNATAX

The DT Protocol allows the unit to be commanded over a simple serial port.

Start Character	Address	Commands	Run	End of a string
/	1-9*	Command strings	R	<CR>

*To Access Drivers 10 – 16 use the following:

Driver #	Command	
A	:	(colon)
B	;	(semi colon)
C	<	(less than)
D	=	(equals)
E	>	(greater than)
F	?	(question mark)
0	@	(at sign)

Running two or more motors together:

Motors 1 and 2:	“A”	
Motors 3 and 4:	“C”	
Motors 5 and 6:	“E”	
Motors 7 and 8:	“G”	
Motors 9 and 10:	“I”	
Motors 11 and 12:	“K”	
Motors 13 and 14:	“M”	
Motors 15 and 16:	“O”	
Motors 1, 2, 3 and 4:	“Q”	
Motors 5, 6, 7 and 8:	“U”	
Motors 9, 10, 11 and 12:	“Y”	
Motors 13, 14, 15 and 16:	“]”	(close bracket)
For all motors:	“_”	(underscore)

Example: /CA5000R will move motors 3 and 4 to Absolute Position 5000.

LIST OF COMMANDS FOR THE IMC23

Command (Case Sensitive)	Operand	Example	Description
HOMING & POSITIONING			
Z	0-max*	/1Z1000R	Initialize the Motor. Motor will turn towards 0 until the home opto sensor or index channel is interrupted. If already interrupted it will back out of the opto or index and come back in until re-interrupted. Please see command 'N' for homing to opto or index channel. Current motor position is set to zero.
z	0-max*	/1z65536R	Sets current position without moving motor. New and old position must have same remainder when divided by 1024 else motor will jump up to 2 steps.
A	0-max*	/1A1000R	Move motor to Absolute position
f	0 or 1	/1f1R	Sets flag polarity of home sensor.
P	0-max*	/1P1000R	Move Motor relative number of steps in positive direction. A value of zero will cause an endless forward move at speed V. By doing so, it enters into Velocity Mode. Any other finite number will set the mode to be in Position Mode.
D	0-max*	/1D1000R	Move Motor relative number of steps in negative direction (Note: Motor will not run in the negative direction if the position is at 0 if final position is below zero.. You can use the 'z' command to set the 0 position to be a large positive number so that moves in neg dir are possible that are at a position that is further away in the negative direction. A value of zero will cause an endless backwards move at speed V. This will enter Velocity Mode. Any other finite number will set the mode to be in Position Mode.
B	0-max*	/1B1000R	Sets the distance for Pulse Jog Mode (/1n1R)
T		/1TR	Terminate current command
F	0, 1	/1F1R	Reverses the positive direction to be negative. Should only be done upon power up, since doing this command afterwards it will cause a loss in step.
VELOCITY & ACCELERATION			
V	100-2 ²⁴	/1V2000R	In Position Mode, this sets the Top Speed of the Motor in half steps per second.
V	100-2 ²⁴	/1V2000R	In Velocity Mode, changes the Top Speed "on the fly".
L	0-65000	/1L5000R	In Position Mode, this sets the Acceleration factor (acceleration = (L Value) x (400,000,000/65536) . Eg /1L1R takes 16.384 Seconds to get to a speed

			of V=100000
Command (Case Sensitive)	Operand	Example	Description
SETTING CURRENT			
m	0-100	/1m50R	Sets "Fast Move" Current on a scale of 0 to 100% of the max current, 3.0A. Default setting is m40.
h	0-50	/1h20R	Sets the Hold Current on a scale of 0 to 50% of the max current. Default setting is h10.
LOOPING & BRANCHING			
g		/1gP10G1R	Beginning of a repeat loop
G	0-30000	/1gP10G1R	End of a repeat loop. Loops can be nested up to 4 levels. A value of 0 causes the loop to be infinite.
M	0-30000	/1M2000R	Delay for "M" milliseconds
H	Blank 01 11 02 12 03 13 04 14	/1gH02P10 000G20R	Halt current command string and wait until condition specified. Wait for switch 2 closure Wait for low on input 1 (Pin 13) Wait for high on input 1 (Pin 13) Wait for low on input 2 (Pin 5) Wait for high on input 2 (Pin 5) Wait for low on input 3 (Pin 7) Wait for high on input 3 (Pin 7) Wait for low on input 4 (Pin 14) Wait for high on input 4 (Pin 14) Halted operation can also be resumed by typing /1R
S	01 11 02 12 03 13 04 14	/1gS02A100 00A0G20R	Skip next instruction if low on input 1 (Pin 13) Skip next instruction if hi on input 1 (Pin 13) Skip next instruction if low on input 2 (Pin 5) Skip next instruction if hi on input 2 (Pin 5) Skip next instruction if low on input 3 (Pin 7) Skip next instruction if hi on input 3 (Pin 7) Skip next instruction if low on input 4 (Pin 14) Skip next instruction if hi on input 4 (Pin 14)

Command (Case Sensitive)	Operand	Example	Description
N	1-2	/1N1R	<p>Special Modes</p> <p>1 = Encoder With No Index (Default). Homes to Opto.</p> <p>2 = Encoder With Index. Homes to Index.</p> <p>Available on the IMCE23</p>
n	0-4095	/1n2R	<p>Sets Modes – Interpret as combination of Binary Bits (LSB)</p> <p>Bit0: /1n1R Enable Pulse Jog Mode. Jog distance is given by “B” command. Velocity is given by “V” command . The Switch Inputs become the Jog Inputs.</p> <p>Bit1: /1n2R Enable Limits. (The two optos become limits switches). The polarity of the limits is set by the “f” command</p> <p>Bit2: /1n4R Enable Continuous Jog Mode. Continuous run of motor while switch is depressed. Velocity is given by the ”V” command. Note that the jog mode allows moves below zero, which will be interpreted by any subsequent ”A” command as a large positive number. If this is undesirable, please use the “z” command to define zero position to be some positive number so that underflow will not occur.</p> <p>Bit3: /1n8R Enables Position Correction Mode. (Appendix 1)</p> <p>Bit4: /1n16R Enabled Overload Report Mode. (Appendix 1)</p> <p>Bit5: /1n32R Enable Step And Direction Mode if (1) or enable Dual Encoder Mode if (0), (All SV23 and Later Models of SV17). Eg /1n96<CR> (96=32+64) Enables step and dir mode and slaves the motor to it. (See Appendix 3)</p> <p>Bit6: /1n64R Enable Motor slave to encoder/step-dir.</p> <p>Bit7: /1n128R Reserved</p> <p>Bit8: /1n256R Reserved</p> <p>Bit9: /1n512R Reserved</p> <p>Bit10: /1n1024R Reserved</p> <p>Bit11: /1n2048R Reserved.</p>

Command	Operand	Example	Description
POSITION CORRECTION MODE			
aC	1-65000	/1aC100R	When in position correction mode, set distance allowed to move before the motor corrects using encoder feedback. (See Appendix 1)
aE	1000-10 ⁶	/1aE12500R	Set Encoder ratio. This sets the ratio between the encoder ticks/rev and the microsteps/rev for the motor. (See Appendix 1)
Au	1-10 ⁶	/1au10000R	Set Overload Timeout. This sets the number of times the move is retried in case a move stalls. (See Appendix 1)
PROGRAM STORAGE & RECALL			
s	0-15	/1s1A10000A0R	Stores a program. Program 0 is executed on power up (Total of 25 commands max per string)
e	0-15	/1e1R	Executes the Stored Programs 0-15
PROGRAM EXECUTION			
R		/1R	Run the command string that is currently in the execution buffer – Always end commands with ‘R’.
X			Repeat the current command string
MICROSTEPPING			
Command	Operand	Example	Description
j	2, 4, 8, 16, 32, 64, 128, 256	/1j256R	Adjusts the resolution in micro-steps per step. For best performance operate in 256 mode.
o	0-1650		Allows user to correct any unevenness in microstep size. Adjusts audible noise and should be executed while motor is running. Default 1500, only need small changes /1o1520R
ON/OFF DRIVERS			
J	0-3		On/Off Driver. It's a two bit Binary value: 3=11=Both Drivers On, 2=10=Driver2 on, Driver1 off, etc.

Command	Operand	Example	Description
QUERY COMMANDS			
<i>The following commands are queries and cannot be cascaded in strings or stored. They can be executed while other commands are still running.</i>			
?	0	/1?0R	Returns the current motor position
?	1	/1?1R	Returns the current Start Velocity
?	2	/1?2R	Returns the current Slew Speed for Position mode
?	3	/1?3R	Returns the current Stop Speed
?	4	/1?4R	Returns the status of all four inputs, 0-15 representing a 4 bit binary pattern: Bit 0 = Input 1 (Pin 13) Bit 1 = Input 2 (Pin 5) Bit 2 = Input 3 (Pin 7) Bit 3 = Input 4 (Pin 14)
?	5	/1?5R	Returns the current Velocity mode speed
?	6	/1?6R	Returns the current step size
?	7	/1?7R	Returns the current 'o' value
?	8	/1?8R	Returns the Encoder Position (can be zeroed by "z" command)
?	9	/1?9R	Erases all stored commands in EEPROM
&			Returns the current Firmware revision and date
Q			Query current status of the controller: 0 = No Error 1 = Initialization error 2 = Bad Command 3 = Operand out of range
T		/1TR	Terminate current commands
\$		/1\$	Recalls current command executed. To see what is currently stored in a specific program, run the program and issue the /1\$ Example: /1e2R /1\$
max			*2^31
BAUD CONTROL			
b	9600 19200 38400	/1b19200R	Adjustable baud rate This command will usually be stored as program zero and execute on power up. Default baud rate is 9600.

RESPONSES FROM CONTROLLER IN HYPER TERMINAL

Syntax	Hex value	Description
/0'□	0x60	Command is terminated
/0@□	0x40	Good command, command received
/0C□	0x43	Command out of range
/0b□	0x62	Bad Command
/0O□	0x4F	Overflow

Disclaimer:

There are known issues involving the Halt command when stored in memory location zero. Upon power up, the remaining command string after the Halt command might be executed if the user types in a new command. If memory location zero is not being used, the user is advised to always clear everything in memory by typing */I?9R*. Otherwise, the user may terminate the remaining command string in the buffer by issuing a */ITR*.

EXAMPLES

Example: */lgP1000D1000G10R* will move motor 1000 steps counterclockwise, then 1000 steps clockwise, in a loop for 10 times.

/	Always begin a program with the forward slash
1	Address of controller (Check with the Red Dial on the unit)
g	Beginning of loop (All commands within 'g' and 'G' will repeat)
P1000	Move counterclockwise 1000 steps
D1000	Move clockwise 1000 steps
G10	End loop, repeat 10 times (G0 will repeat infinitely, type <i>/ITR</i> to terminate)
R	Run this command

Example: */ls0gH01A100H01A0G0R* will store a program to memory, and run upon power up. This program will move 100 steps (90° for a 1.8° step motor) when you press a push button. And it will return to its original position when pressing the button a second time. This will repeat infinitely.

/	Always begin programs with a forward slash
1	Address of Controller. Check on the Red Dial of the unit
s0	Store to program 0 – defined as running upon power up
g	Beginning of loop. Anything between 'g' and 'G' will repeat
H01	Halt commands until a low '0' is seen on input 1. Push button is pressed.
A100	Then move 100 steps (absolute position)
H01	Halt again until a low '0' is seen on input 1. Push button is pressed.
A0	Move back to Position 0.
G0	End loop. Repeat infinitely, type <i>/ITR</i> to terminate
R	Run commands

To execute program, type */1e0R*. Or, power down and power up. Only program 0 will start upon power up. To terminate out of this infinite loop, type */ITR*.

Example:

Enable Pulse Jog Mode

/1n1R

Enable Opto Limit Mode

/1n2R

Enable Pulse Jog Mode and Opto Limit Mode

/1n3R

Enable Continuous Jog Mode

/1n4R

Enable Opto Limit Mode and Continuous Job Mode

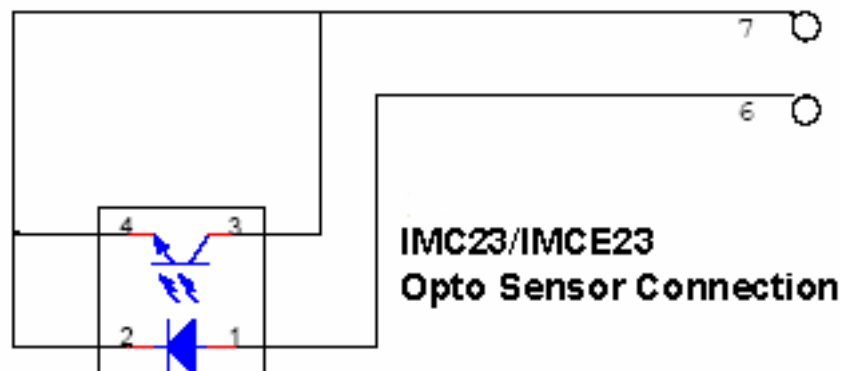
/1n6R

Homing Sensor

The “Z” command is used to initialize the motor to a generally known amount of steps (a maximum of 10000 steps + 400 default steps). When issued, i.e. /1Z5000R, the motor will turn towards zero at a maximum step of 5400 until the home opto sensor is interrupted. If issued a /1Z0R, motor will only move 400 steps to find opto sensor.

If the sensor is already interrupted, and /1Z5000R was issued, the motor will move in the opposite direction until the sensor is un-cut again. At this time, the motor moves towards home in the same way described above. When sensor is cut, motor stops motion and current position is reset to zero. Speed is set by lower case v, i.e. /1v4000Z5000R.

The Z command is used in conjunction with Pins 6 and 7. An appropriate optical sensor must be attached to Pins 6 and 7 in order for the homing command to work properly. The Z command allows the motor to rotate until Pin 7, Input 3, goes from low to high.



UNDERSTANDING THE RESPONSE

The IMC23 controller board sends commands to a Master Device, i.e. a PC, and is dedicated as Address 0. The Master Device looks for the response from the controller in a certain syntax that is partially transparent in HyperTerminal. /0 is the beginning of the syntax which the Master Device looks for.

Following the /0 characters are the status characters which is a compilation of 8 bits:

Bit 7	Reserved
Bit 6	Always set
Bit 5	Ready Bit – Set when controller is ready to accept commands
Bit 4	Reserved
Bit 3-0	Status code: 0 = No Error 1 = Initialization Error 2 = Bad Command 3 = Operand out of Range 7 = Overload error

Example Responses to command /1?4

FFh	RS485 line turn around character
2Fh	ASCII '/' Start character. The DT Protocol uses '/' as the start
30h	ASCII 0. Master Device's address
60h	Status character as explained above
31h	Two bytes are the actual answer in ASCII. This is an eleven (11) which represents the status of the four inputs: Bit 0 = Switch 1 Bit 1 = Switch 2 Bit 2 = Opto 1 Bit 3 = Opto 2 (eleven is 1011 in binary)
03h	This is the ETX or end of text character. It's the end of the answer string
0Dh	Carriage Return
0Ah	Line Feed

APPENDIX 1

Position Correction Mode and Overload Report Mode

Position Correction Mode:

Position correction mode, when enabled will move until the encoder reads the correct number. Once enabled Positions are given in Quadrature Encoder Counts of the encoder (Not in microsteps). If the motor stalls during a move then this mode will reattempt the move until the encoder reads the correct number. (Available in software version 3.7 and above)

First Set the Encoder ratio:

Encoder Ratio = (MicroSteps/Rev Divided by Quadrature Encoder Ticks/rev) x 1000
This must be a whole number after the multiply by 1000.

The Motor must be left in 256 Microstep mode for correct operation of feedback mode. For Example A: 1.8 Degree Stepper (running in 1/256th step mode), which has 200x256 Microsteps/Rev is Hooked up to a 400 line encoder which has 1600 quadrature encoder counts.

$$aE = ((200 \times 256) / (400 \times 4)) \times 1000 = 32000$$

Issue the command /1aE32000R to set this Encoder ratio.

If the encoder ratio is unknown do the following:

Leave the Encoder ratio at its power up default of 1000.

Ensure that the encoder increases its count when the motor moves in the positive direction. If not, switch either the AB lines on the encoder which will reverse the count direction OR switch the wires to one of the windings on the stepper motor, which will reverse the direction of rotation.

- Issue /1n0R to clear any special modes.
- Issue /1z0R which will zero both the encoder and step position.
- Issue /1A100000R and ensure that the move completes at a velocity that does not stall.
- Issue /1?0 which reads back the current position – This should be 100000.
- Issue /1?8 which reads back the encoder position.
- Issue /1aE0R which auto divides these two numbers.
- Issue /1?aE which reads back the encoder ratio computed.
This value read back is only a rough guide and will be out by a few counts due to inaccuracies in the motor position and run-out in the encoder.
The value read back MUST be overwritten by the EXACT value that represents ratio.
- Now Issue /1aE32000R or whatever exact number represents the encoder ratio.

Second: (optional) set the Error in Quadrature Encoder Ticks allowed before a Correction is issued:

Eg /1aC50R (default is 50)

Third: (optional) set the Overload Timeout Value:

This is the number of retries allowed under a stall condition. Eg /1au10000R (default is 10)

Appendix 1 : Position Correction Mode Continued.....

Fourth: Enable the Feedback mode:

Zero positions just prior to enabling feedback mode the by issuing /1z0R.

(Or issue /1z10000R etc if you need to at this time)

Enable Position Correction Mode by issuing /1n8R.

Notes:

- (1) When any command is received by the drive it will always respond with its status. The drive will only accept a command when it is not busy. This status byte received must be checked, to ensure that the unit was not busy and that the command was accepted. This is especially important when position correction mode is enabled, because the drive may be attempting to correct position all by its self, and will reject an externally (via RS232) received command if it is busy in the middle of a correction move.
- (2) When position correction mode is enabled, /1n8R, then the drive will keep retrying any stalled moves, and will NOT halt any strings or loops upon detection of a stall.
- (3) During position correction mode /1T will halt any move, but there is a possibility that the drive may instantly reissue itself a position correction command, especially if it is fighting a constant disturbance. It may be necessary to issue a /1n0R to positively halt a move in progress.
- (4) Position correction mode is inhibited if the encoder underflows and goes negative (but will automatically resume if a move is made into the positive range). If position correction is required at the zero point, please redefine zero to be a slightly positive number with the “z” command. Eg /1z10000R
- (5) If the encoder ratio is changed from its default of 1000, the allowed max position will be decreased from $+2^{31}$ by the same ratio.

Overload Report Mode:

Overload report mode when enabled, will compare the encoder value to the commanded position at the end of a move and report an error if the two values do not match within the range given by “aC”. When this error occurs the drive will exit from any loops or strings it may be executing.

Overload report mode is enabled by /1n16R, and requires the encoder ratio to be entered correctly via the “aE” command. Issue a /1zR to zero both the encoder and position counter just prior to issuing /1n16R.

APPENDIX 2
ANALOG INPUTS AND ANALOG FEEDBACK
Available on Software version 3.90 and later

ANALOG INPUTS

The 4 Inputs on the IMC23 are all ADC Inputs.

- 1) These ADC values can be read via RS485. E.g. `/1?aa<CR>`. These values are on a scale of 0-16368 as the input varies from 0-3.3V. The inputs as shipped are good to about 7 bits, but can be made to be better than 10 bits with the removal of the input over-voltage protection circuitry, (call factory for details).
- 2) The threshold upon which a Digital “One” or “Zero” is called can be varied with the “at” command and hence affect the Halt H command or Skip S command.
E.g. `/1at309999R<CR>` – This sets the threshold on input 3 to be 09999. Note that it is necessary to insert leading zeros after the Input Number (3) since the threshold value must always be entered as 5 digits (00000-16368).
- 3) The thresholds for all 4 inputs can be read back with the `/1?at<CR>` command. The units have a default threshold value of 6144 = (1.24V).
- 4) It is possible for example to regulate pressure, by turning a pump on or off depending on an analog value read back, by setting the threshold of the One/Zero call to be the regulation Point. E.g. `/1at308000gS03P1000G0R`
- 5) A potentiometer can be placed as shown in the wiring diagram and its position read back via the `/1?aa<CR>` command. Note that the supply provided (which normally drives an LED) has 200 ohm in series to 5V, so the use of a 500 ohm potentiometer will give almost a 0-3.3V range on the inputs.

APPENDIX 3
STEP AND DRIVE MODE

- 1) Issue the command string to enable Step and driver mode `/1n96R`
- 2) Connect +ve Signal of the Step to Pin 4
- 3) Connect –ve Signal of the Step to Pin 3
- 4) Connect Direction input signal into Pin 11