Command Set Overview

Reference Key:

- is the IO Bit Number

m - is the mask value of which bits are affected

W - defines it as a word (16 bits)

expression - an expression must contain no more than a total maximum of 32 operators, values, and parenthesis.

value - a number, variable or math expression with one operand

constant - means a fixed integer

gen# Trajectory generator number: 1 or 2

i - Interrupt number , valid values are from 0 to 7

Communication Commands:

ADDR=*expression*	Set motor's serial communications address. Applies for both RS232 and RS485
BAUD(x)=*y*	This allows for COM0 or COM1 to be changed, x is the channel (0 or 1) and y is baud rate
CADDR=*expression*	Set CAN address, can be different from serial address, default is 63
CBAUD=*expression*	Set CAN baud rate, default is 125000
CCHN(*RS2,0***)**	Close communication channel command
ECHO	Must be used to insure all data received in one motor will be echoed to next motor
ECHO_OFF	Default, turn communication's echo off
GETCHR	Get the next character from channel 0
GETCHR1	Get the next character from channel 1
LEN	Number of characters in channel 0 buffer
LEN1	Number of characters in channel 1 buffer
RCADDR	Reports CAN address
RCBAUD	Reports CAN baud rate
RCHN(0)	Report channel 0 error bits
RCHN(1)	Report channel 1 error bits
SILENT	Ignore print commands to channel 0 from user program
SILENT1	Ignore print commands to channel 1 from user program
SLEEP	Ignore commands for channel 0 except the WAKE command
SLEEP1	Ignore commands for channel 1 except the WAKE command
STDOUT=0	Sets internal report commands to RS232 (default)
STDOUT=1	Sets internal report commands to RS485
TALK	Enable prints for channel 0 from user program
TALK1	Enable prints for channel 1 from user program
WAKE	Wake for channel 0
WAKE1	Wake for channel 1

Program Flow Commands:

CASE *expression*	Switch case statement
C *constant*	Subroutine label, e.g. C10 for subroutine 10, must have a RETURN for each C label
DEFAULT	Default action for switch case statement
DITR(*i***)**	Individual interrupt disable
EITR(*i***)**	Individual interrupt enable
ELSEIF *expression*	Used for IF statements to test another condition, if expression is true, then execute code
END	End program execution
ENDIF	End statement for IF code structures
ENDS	Command for end of switch case statement
GOSUB(*value***)**	Call a subroutine, value up to 999
GOTO(*value***)**	Jump program execution to a label, value up to 999
IF *expression*	Conditional Test, expression can be multiple math operations
ITR(*i, status_wrd#, bit#, s ,label#***)**	Interrupt setup
ITRD	Global interrupt scanner disable
ITRE	Global interrupt scanner enable
LOOP	Loop command for while loops
PAUSE	Pause program execution, used for interrupts
RESUME	Resume program execution
RETURN	Return from subroutine
RETURNI	Return from interrupt
RUN	Start program execution
RUN?	Wait at this point for RUN command before program starts to execute
STACK	Resets all GOSUB stack returns and Interrupts
SWITCH *expression*	Switch case statement
TWAIT	Wait for trajectory to complete, only used in program
TWAIT(*gen#***)**	Wait for trajectory generator (gen#) to complete it's move
TSWAIT	Wait for synchronized trajectory to complete, down loaded program only * COMBITRONIC™
WAIT=*expression*	Set wait time in milliseconds
WHILE *expression*	
LOOP	While loop format

I/O Commands:

EIGN(#)	Assign a single I/O point as general use input
EIGN(W,0)	Assign all local I/O as general use inputs
EIGN(W,0,12)	Assign inputs 2 and 3 as general use inputs at once (disabling over-travel limits)
EIGN(W,*0,m***)**	Assign a masked word-sized set of local I/O as general use inputs at once
EILN	Set port C (I/O-2) as negative over travel limit
EILP	Set port D (I/O-3) as positive over travel limit
EIRE	Set I/O 6 to capture external encoder's current value
EIRI	Set I/O 6 to capture internal encoder's current value
EISM(6)	Issue (G) when local input 6 goes low
EOBK(#)	Configure a given output to control an external brake
IN(#)	x=IN( ), assign the state of a specific I/O to a variable (x in this case)
IN(W,0)	x=IN(W,0), assign the state of the first word of local I/O to the variable x
INA(A,#)	x=INA(A,#), raw analog reading: 10 bit resolution spanned over signed 16 bit range

* COMBITRONIC™ **These commands require Combitronic with -C or -DN product configuration option to execute.**

Command Set Overview

INA(V,#)	x=INA(V,#), input voltage in millivolts of analog input value for a given I/O defined by #
INA(V1,#)	x=INA(V1,#), scaled 0-5 VDC reading in millivolts directly, 3456 would be 3.456 VDC
OC(#)	x=OC(#), individual output status, bit 1 if output is being driven
OC(W,#)	x=OC(W,#), block output status, bit 1 if output is being driven
OF(#)	x=OF(#), returns present fault state for I/O defined by #
OF(L,#)	x=OF(W,#), returns bit mask fault latched for I/O points
OF(W,#)	x=OF(W,#), returns bit mask of present faulted I/O points
OR(*value*)	Reset output (turn off)
OS(*value*)	Set output (turn on)
OUT(#)=*expression*	if expression LSB = 1, then it's true(1), otherwise it's false (0)

Math Commands:

-	Subtract
!|	Bitwise exclusive OR
!=	Not equal to
%	Modulo (remainder) division
&	Bitwise AND
*	Multiply
/	Divide
^	Power limited to 4th power and below, integers only
|	Bitwise inclusive OR
+	Add
<	Less than
<=	Less than or equal to
==	Equal to
>	Greater than
>=	Greater than or equal to
ABS(*value*)	Absolute Value
ACOS(*value*)	Arc Cosine
ASIN(*value*)	Arc Sine
ATAN(*value*)	Arc Tangent
COS(*value*)	Cosine
FABS(*value*)	Floating point absolute value
FSQRT(*value*)	Floating point square root
RANDOM=*expression*	Set the random seed value 0 to 2^31 - 1
RRANDOM	Report the next available random number in the range 0 to 2^31 -1
SIN(*value*)	Sine
SQRT(*value*)	Square Root
TAN(*value*)	Tangent
TMR(*x,t*)	Sets timer x for t milliseconds

Motion Commands:

ADT=*expression*	Set the accel/decel at once for a move
ADTS=*expression*	Set sync accel/decel at once for a move * COMBITRONIC™
Ai(0)	Arm index rising edge of internal encoder
Ai(1)	Arm index rising edge of external encoder
Aij(0)	Arm index rising edge then falling edge internal encoder
Aij(1)	Arm index rising edge then falling edge external encoder
Aj(0)	Arm index falling edge of internal encoder
Aj(1)	Arm index falling edge of external encoder
Aji(0)	Arm index falling edge then rising edge internal encoder
Aji(1)	Arm index falling edge then rising edge external encoder
AMPS=*expression*	Current limit value. 0-1023
AT=*expression*	Set the acceleration target for a move
ATS=*expression*	Set sync acceleration target for a move * COMBITRONIC™
BREAK	Break out of while loop
BRKENG	Manually Engage the brake
BRKRLS	Manually Release the brake
BRKSRV	Brake Servo, engage the brake when the drive is not active (default)
BRKTRJ	Brake Trajectory
CTR(0)	Present value of internal encoder
CTR(1)	Present value of external encoder
DEL=*expression*	Set maximum allowable derivative error limit
DT=*expression*	Set the deceleration target for a move
DTS=*expression*	Set sync deceleration for a move * COMBITRONIC™
EL=*expression*	Set maximum allowable following error limit
ENC1	Enable external encoder for servo
ENC0	Enable internal encoder for servo
F	Set tuning values
G	Go, initiates all buffered modes of operation
G(*gen#*)	Go, initiate motion in trajectory generator (gen#)
GS	Go synchronized, initiates linear interpolated moves * COMBITRONIC™
KA=*expression*	Feed forward gain
KD=*expression*	Derivative gain coefficient
KG=*expression*	Gravity offset
KI=*expression*	PID integral gain
KL=*expression*	PID integral limit
KP=*expression*	PID proportional gain
KS=*expression*	Differential sample rate
KV=*expression*	Velocity feed forward gain
MC	Initiate electronic camming
MC(2)	Set Trajectory Generator 2 to run in electronic camming
MDB	Enable TOB when in one of the 2 trapezoidal modes
MDE	Set motor to enhanced trapezoidal mode commutation by using encoder
MDS	Set motor to sine mode commutation

* COMBITRONIC™ **These commands require Combitronic with -C or -DN product configuration option to execute.**

Command Set Overview

MDT	Set motor to trapezoidal mode communtation using hall sensors (default mode)
MFA(*value*)	Accel over *value* master distance. Default is zero (off)
MFD(*value*)	Decel over *value* master distance. Default is zero (off)
MFDIV=*expression*	Assign Incoming counts Divisor
MFMUL=*expression*	Assign Incoming counts Multiplier
MF0	Initiate and zero counter, but do not follow
MFR	Select follow mode using quadrature encoder input.
MFR(2)	Set Trajectory Generator 2 to run in Mode Follow Ratio (electronic Gearing)
MFSLEW(*value*)	Stay at slew for *value* distance, then decel
MINV(0)	Default motor commutation direction
MINV(1)	Invert commutation, shaft rotates opposite direction
MP	Initiate Position Mode
MP(1)	Set Trajectory Generator 1 to run in Position Mode
MS0	Initiate and zero counter, but do not follow
MSR	Calculate Mode Step Ratio and prepare to follow
MT	Initiate Torque Mode (Open Loop)
MTB	Enable mode torque brake
MV	Initiate Velocity Mode
MV(1)	Set Trajectory Generator 1 to run in Velocity Mode
O=*expression*	Set origin, set present position to some value
O(*gen#*)=*expression*	Set origin for move gen# to some value
OFF	Turn the amplifier off
OSH=*expression*	Origin shift of position counter on the fly
OSH(*gen#*)=*expression*	Shift origin for move gen# by some value
PID1	Set default PID update rate
PID2	Set default PID/2 update rate
PID4	Set default PID/4 update rate
PID8	Set default PID/8 update rate
PML=*expression*	Sets the position modulo limit wrap value
PMT=*expression*	Set the position modulo target
PRT=*expression*	Set the relative target position
PRTS=*(dist1;axis1,dist2;axis2,dist3;axis3)*	Set synchronized relative target position * COMBITRONIC™
PRTSS=*(dis1;axis)*	Set supplemental synchronized relative target position * COMBITRONIC™
PT=*expression*	Set the absolute target position
PTS=*(dist1;axis1,dist2;axis2,dist3;axis3)*	Set synchronized absolute target position * COMBITRONIC™
PTSS=*(dis1;axis)*	Set supplemental synchronized absolute target position * COMBITRONIC™
S	Instantly stop motor
S(*gen#*)	Instantly stop trajectory generator (gen#)
SLD	Disable software travel limits
SLE	Enable software travel limits
SLM (0)	Make a soft limit only trigger the flag, but not cause a fault
SLM (1)	Make s soft limit trigger the flag and cause a fault (default mode)
SLN=*expression*	Set the negative software travel limit
SLP=*expression*	Set the positive software travel limit
T=*expression*	Set the commanded torque while in MT mode
TH=*expression*	Set maximum allowable thermal limit (degrees C)
VT=*expression*	Set the velocity target for a move
VTS=*expression*	Set synchronized velocity target for a move * COMBITRONIC™
X	Decelerate to a stop at present deceleration rate
X(*gen#*)	Decelerate to a stop , trajectory generate (gen#)

Status Commands:

Ba	*Over current bit, status word 0, bit 4* status word 1, bit 3
Be	*Excessive position error, status word 0, bit 6*
Bh	Excessive temperature occurred, status word 0, bit 5
Bi(0)	Rising Edge Capture on Encoder 0 (internal), status word 1, bit 2
Bi(1)	Rising Edge Capture on Encoder 1 (external), status word 1, bit 6
Bj(0)	Falling Edge Capture on Encoder 0 (internal),
Bj(1)	Falling Edge Capture on Encoder 1 (external), status word 1, bit 7
Bk	Main program checksum error, program is corrupt and cannot run, status word 2, bit 15
Bl	Left (-) over travel limit, status word 0, bit 13
Bls	Left (-) over travel software limit occurred, status word 1, bit 13
Bm	Left (-) over travel limit active, status word 0, bit 15
Bms	Left (-) over travel software limit active, status word 1, bit 15
Bo	Motor is off, status word 0, bit 1
Bp	Right (+) over travel limit active, status word 0, bit 14
Bps	Right (+) over travel software limit active, status word 1, bit 14
Br	Right (+) over travel limit, status word 0, bit 12
Brs	Right (+) over travel software limit occurred, status word 1, bit 12
Bs	Command Syntax error note, status word 2, bit 14
Bt	Trajectory in progress, status word 0, bit 2
Bv	Velocity limit, status word 0, bit 7
Bw	Wrap around occurred, position wrapped through +/- 2^31, status word 3, bit 3
Bx(0)	Hardware index input probe state for internal encoder, status word 1, bit 8
Bx(1)	Hardware index input probe state for external encoder, status word 1, bit 9
CLK=*expression*	System Clock value in milliseconds
ERRC	Get most recent command error code
ERRW	Where/Who commanded most recent error
FSA ( )	FSA(0,0) is default, sets all types of faults to result in MTB
RAC	Report commanded acceleration
RAT	Report target acceleration

* COMBITRONIC™ **These commands require Combitronic with -C or -DN product configuration option to execute.**

Moog Inc., Animatics • All integrated motor products made by Moog Animatics are covered by patent number 5,912,541
tel: 408.748.8721 • fax: 408.748.8725 • www.animatics.com

Command Set Overview

Ra	Report value of variable 'a'
Rab[0]	Report value of ab[0]
Raf[0]	Report floating point value of af[0]
Ral[0]	Report value of al[0]
Raw[0]	Report value of aw[0]
REPTR	Reports EEPROM pointer value
RCKS	Report Checksum
RB(*sw,b*)	Report status bit, b, from status word, sw
RCLK	Report system clock in milliseconds
RCTR(0)	Report present value of internal encoder
RCTR(1)	Report present value of external encoder
RDEA	Report actual derivative error
RDEL	Report commanded derivative error limit
RDT	Report target deceleration
REA	Report actual following error
REL	Report commanded following error limit
RI (0)	Report where the rising edge of the internal index was detected
RI (1)	Report where the rising edge of the external index was detected
RIN(#)	Report the state of a I/O
RIN(W,0)	Report the first word of local I/O
RINA(A,#)	Reports analog input value for a given I/O defined by #
RINA(V,#)	Reports voltage level (scaled from supply) of analog input value for a given I/O defined by #
RINA(V1,#)	Reports voltage level (scaled 0-5 VDC) of analog input value for a given I/O defined by #
RJ(0)	Report where the falling edge of the internal index was detected
RJ(1)	Report where the falling edge of the external index was detected
RMFDIV	Report Divisor
RMFMUL	Report Multiplier
RMODE	Report mode of operation
RPA	Report present actual position
RPC	Report present commanded position
RPC(*gen#*)	Report commanded position for trajectory generator (gen#)
RPMA	Report the current modulo counter
RPML	Report position modulo limit
RPMT	Report the most recent setting of PMT (position modulo target)
RPRA	Report actual relative position
RPRC	Report commanded relative position
RPRT	Report present relative target position
RPT	Report present target position
RRES	Report encoder resolution of motor
RSLN	Report value of negative software limit
RSLP	Report value of positive software limit
RSP	Report sampling rate and firmware version
RSP1	Report firmware revision date
RTH	Report maximum allowable thermal limit
RTMR(*x*)	Report timer x (present time left in milliseconds)
RT	Report commanded torque
RVC	Report commanded velocity
RVT	Report target velocity
RUIA	Reports current (Amps=UIA/100)
RUJA	Reports bus voltage (Volts=UJA/10)
RVA	Report actual velocity
RW(*value*)	Report status word
Z(*sw,b*)	Clears/zeros status word bits
Za	Reset over current bit
Ze	Reset position error bit
Zh	Reset over temperature bit
Zl	Reset left(-) historical limit bit
Zls	Reset left(-) software historical limit bit
Zr	Reset right(+) historical limit bit
Zrs	Reset right(+) software historical limit bit
Zs	Reset syntax error bit
ZS	Clear all errors, reset system latches to power up state
Zw	Reset wraparound bit

Variable Commands:

a=*expression*	Variable, 32 bit signed integers, a-z, aa-zz, aaa-zzz, 78 total variables
ab[*x*]=*expression*	Array variables, 8 bit byte arrays, x can be 0-203
af[*x*]=*expression*	Floating point array variables, x can be 0-7
al[*x*]=*expression*	Array variables, 32 bit long arrays, x can be 0-50
aw[*x*]=*expression*	Array variables, 16 bit word arrays, x can be 0-101
EPTR=*expression*	EEPROM pointer, non-volatile memory, use before VLD and VST commands
VLD(*variable,quantity*)	Load values from EEPROM to variables starting at EPTR location
VST(*variable,quantity*)	Store values to EEPROM from variables starting at EPTR location

Other Commands:

LOCKP	Disable program (EEPROM) upload
UPLOAD	Upload the program

OCHN(*RS2,0,*N,*9600,*1,8,*C,1000*) Default: (RS232,chan=0, no parity, 9600 baud,1 stopbit, 8 databits, command,1000 ms timeout)

PRINT("Hello World",#13) Print command to say "Hello World", see print section for more detailed examples

PRINT1("Hello World",#13) Print command to say "Hello World" on channel 1, see print section for more detailed example

Note: See users guide for complete list of commands and full syntax.

Many commands such as Cam mode and dual trajectory mode commands are not fully explained here.