

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	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 #	ADTS=expression	Set sync accel/decel at once for a move * COMBITRONIC™
INA(V1,#)	x=INA(V1,#), scaled 0-5 VDC reading in millivolts directly, 3456 would be 3.456 VDC	Ai(0)	Arm index rising edge of internal encoder
OC(#)	x=OC(#), individual output status, bit 1 if output is being driven	Ai(1)	Arm index rising edge of external encoder
OC(W,#)	x=OC(W,#), block output status, bit 1 if output is being driven	Aij(0)	Arm index rising edge then falling edge internal encoder
OF(#)	x=OF(#), returns present fault state for I/O defined by #	Aij(1)	Arm index rising edge then falling edge external encoder
OF(L,#)	x=OF(W,#), returns bit mask fault latched for I/O points	Aj(0)	Arm index falling edge of internal encoder
OF(W,#)	x=OF(W,#), returns bit mask of present faulted I/O points	Aj(1)	Arm index falling edge of external encoder
OR(value)	Reset output (turn off)	Aji(0)	Arm index falling edge then rising edge internal encoder
OS(value)	Set output (turn on)	Aji(1)	Arm index falling edge then rising edge external encoder
OUT(#)=expression	if expression LSB = 1, then it's true(1), otherwise it's false (0)	AMPS=expression	Current limit value. 0-1023
<hr/>		AT=expression	Set the acceleration target for a move
Math Commands:		ATS=expression	Set sync acceleration target for a move * COMBITRONIC™
-	Subtract	BREAK	Break out of while loop
!	Bitwise exclusive OR	BRKENG	Manually Engage the brake
!=	Not equal to	BRKRLS	Manually Release the brake
%	Modulo (remainder) division	BRKSRV	Brake Servo, engage the brake when the drive is not active (default)
&	Bitwise AND	BRKTRJ	Brake Trajectory
*	Multiply	CTR(0)	Present value of internal encoder
/	Divide	CTR(1)	Present value of external encoder
^	Power limited to 4th power and below, integers only	DEL=expression	Set maximum allowable derivative error limit
	Bitwise inclusive OR	DT=expression	Set the deceleration target for a move
+	Add	DTS=expression	Set sync deceleration for a move * COMBITRONIC™
<	Less than	EL=expression	Set maximum allowable following error limit
<=	Less than or equal to	ENC1	Enable external encoder for servo
==	Equal to	ENC0	Enable internal encoder for servo
>	Greater than	F	Set tuning values
>=	Greater than or equal to	G	Go, initiates all buffered modes of operation
ABS(value)	Absolute Value	G(gen#)	Go, initiate motion in trajectory generator (gen#)
ACOS(value)	Arc Cosine	GS	Go synchronized, initiates linear interpolated moves * COMBITRONIC™
ASIN(value)	Arc Sine	KA=expression	Feed forward gain
ATAN(value)	Arc Tangent	KD=expression	Derivative gain coefficient
COS(value)	Cosine	KG=expression	Gravity offset
FABS(value)	Floating point absolute value	KI=expression	PID integral gain
FSQRT(value)	Floating point square root	KL=expression	PID integral limit
RANDOM=expression	Set the random seed value 0 to 2^31 - 1	KP=expression	PID proportional gain
RRANDOM	Report the next available random number in the range 0 to 2^31 - 1	KS=expression	Differential sample rate
SIN(value)	Sine	KV=expression	Velocity feed forward gain
SQRT(value)	Square Root	MC	Initiate electronic camming
TAN(value)	Tangent	MC(2)	Set Trajectory Generator 2 to run in electronic camming
TMR(x,t)	Sets timer x for t milliseconds	MDB	Enable TOB when in one of the 2 trapezoidal modes
<hr/>		MDE	Set motor to enhanced trapezoidal mode commutation by using encoder
Motion Commands:		MDS	Set motor to sine mode commutation
ADT=expression	Set the accel/decel at once for a move		

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

Command Set Overview

MDT	Set motor to trapezoidal mode commutation using hall sensors (default mode)	SLN=<i>expression</i>	Set the negative software travel limit
MFA(<i>value</i>)	Accel over <i>value</i> master distance. Default is zero (off)	SLP=<i>expression</i>	Set the positive software travel limit
MFD(<i>value</i>)	Decel over <i>value</i> master distance. Default is zero (off)	T=<i>expression</i>	Set the commanded torque while in MT mode
MFDIV=<i>expression</i>	Assign Incoming counts Divisor	TH=<i>expression</i>	Set maximum allowable thermal limit (degrees C)
MFMUL=<i>expression</i>	Assign Incoming counts Multiplier	VT=<i>expression</i>	Set the velocity target for a move
MFO	Initiate and zero counter, but do not follow	VTS=<i>expression</i>	Set synchronized velocity target for a move <i>* COMBITRONIC™</i>
MFR	Select follow mode using quadrature encoder input.	X	Decelerate to a stop at present deceleration rate
MFR(2)	Set Trajectory Generator 2 to run in Mode Follow Ratio (electronic Gearing)	X(<i>gen#</i>)	Decelerate to a stop, trajectory generate (<i>gen#</i>)
MFSLEW(<i>value</i>)	Stay at slew for <i>value</i> 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=<i>expression</i>	Set origin, set present position to some value		
O(<i>gen#</i>)=<i>expression</i>	Set origin for move <i>gen#</i> to some value		
OFF	Turn the amplifier off		
OSH=<i>expression</i>	Origin shift of position counter on the fly		
OSH(<i>gen#</i>)=<i>expression</i>	Shift origin for move <i>gen#</i> 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=<i>expression</i>	Sets the position modulo limit wrap value		
PMT=<i>expression</i>	Set the position modulo target		
PRT=<i>expression</i>	Set the relative target position		
PRTS=(<i>dist1;axis1,dist2;axis2,dist3;axis3</i>)	Set synchronized relative target position <i>* COMBITRONIC™</i>		
PRTSS=(<i>dis1;axis</i>)	Set supplemental synchronized relative target position <i>* COMBITRONIC™</i>		
PT=<i>expression</i>	Set the absolute target position		
PTS=(<i>dist1;axis1,dist2;axis2,dist3;axis3</i>)	Set synchronized absolute target position <i>* COMBITRONIC™</i>		
PTSS=(<i>dis1;axis</i>)	Set supplemental synchronized absolute target position <i>* COMBITRONIC™</i>		
S	Instantly stop motor		
S(<i>gen#</i>)	Instantly stop trajectory generator (<i>gen#</i>)		
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 a soft limit trigger the flag and cause a fault (default mode)		
		CLK=<i>expression</i>	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

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), status word 1, bit 7
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

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



Command Set Overview

RAT	Report target acceleration
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(<i>sw,b</i>)	Report status bit, <i>b</i> , from status word, <i>sw</i>
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(<i>gen#</i>)	Report commanded position for trajectory generator (<i>gen#</i>)
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(<i>x</i>)	Report timer <i>x</i> (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(<i>value</i>)	Report status word
Z(<i>sw,b</i>)	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=<i>expression</i>	Variable, 32 bit signed integers, a-z, aa-zz, aaa-zzz, 78 total variables
ab[<i>x</i>]=<i>expression</i>	Array variables, 8 bit byte arrays, x can be 0-203
af[<i>x</i>]=<i>expression</i>	Floating point array variables, x can be 0-7
al[<i>x</i>]=<i>expression</i>	Array variables, 32 bit long arrays, x can be 0-50
aw[<i>x</i>]=<i>expression</i>	Array variables, 16 bit word arrays, x can be 0-101
EPTR=<i>expression</i>	EEPROM pointer, non-volatile memory, use before VLD and VST commands
VLD(<i>variable,quantity</i>)	Load values from EEPROM to variables starting at EPTR location
VST(<i>variable,quantity</i>)	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.