KOLLMORGEN™

P7000 Series Stepper Drives

MODBUS Documentation for User Parameters

Revision C
June 05, 2013

Warning: Non-Volatile Memory

The P7000 products automatically update non-volatile memory when a variable is changed. Designing a system that changes one or more variables on a repetitive basis could exceed the storage device's life expectancy of 1,000,000 writes. **Exceeding the specification will cause a drive failure - requiring repair**. This can (and must be) prevented by disabling the automatic writes to non-volatile memory by setting command index 96 to 65535. This command should be executed as the first command sent to the P7000 after power up or reset, as the value of command index 96 itself is volatile. For more information see Index 96, below.

1.1 MODBUS RTU

The base unit uses the MODBUS RTU Class 0 serial protocol for communication via the RS232 RJ12 (J5) port or RS485 option card (J2-J3, P7000 AC). RTU is the binary implementation of the MODBUS protocol and Class 0 means that only the Read Holding Registers (3) and Preset Multiple Registers (16) functions of the protocol are supported. Detailed information regarding the MODBUS standard can be found at www.modbus.org.

The serial port settings are as follows:

BAUD:	19200
Start:	1 bit
Data:	8 bits
Parity:	Even
Stop:	1 bit

Reading Variables:

To read a variable use the Modbus Function "Read Holding Registers" (Function Code 3).

Read Request:

UA=xx (1 byte), FC=03 (1 byte), Index Number (16bit), Number of 16bit Registers (16bit), CRC (16bit)

Where:
UA is the unit address of the drive.
FC is the Read Multiple Holding Register Function Code (03).
Index Number is the variable register address.
Number of registers to read should be set to 1, 2, or 3 depending on the variable.

Read Response:

UA=xx (1 byte), FC=03 (1 byte), Data Length, Data, CRC (16bit)

Where:
UA is the unit address.
FC is the function code (03).
Data length will be 2, 4, or 6 (number of bytes)
Data will be the value of the variable read.
CRC is to be added to the telegram.

Example: read DRes (Index 2)
```
000-Tx: 01 03 00 02 00 01 25 CA
001-Rx: 01 03 02 01 F4 B8 53
```
Response is 2 bytes read from Index 2 and data value is 0x01F4 (500 dec), which is 25,000 / 50 (step resolution / tooth count).

Example: read Jog High (Index 41 – 42)
```
000-Tx: 01 03 00 29 00 02 15 C3
001-Rx: 01 03 04 00 01 41 FA 1B E0
```
Response is 4 bytes read from Index 41-42 and data value is 0x41FA 0001 (the data is in Little Endian) Converting from hex to floating point decimal gives 31.250002. Solving for velocity in RPS = 31.250002 / 25000 / 250E-6 = 5.00000032 revs/s.

Writing Variables:

To write variables use the Modbus Function “Write Multiple Registers” (Function Code 16).

<u>Write Request:</u>

UA=xx, FC=16, Index Number(16bit), Number of 16bit Registers(16bit), Data Length(1byte), Data, CRC

Where:
UA is the unit address of the drive (1 byte).
FC is the Write Multiple Registers function code (16 decimal = 0x10) (1 byte).
Number of registers to write should be set to 1, 2, or 3 depending on the variable.
Data Length is the number of bytes of data being written.
Data will be the value being written to the variable.
CRC is to be added to the telegram.

<u>Write Response:</u>

UA=xx, FC=16, Index Number(16bit), Number of 16bit Registers(16bit), CRC

Where:
UA is the unit address.
FC is the function code echoed back (16d / 0x10).
Address is the index number to which the data was written.
Number of Registers is the number of 16bit words of data written.

Example: Make the Step Resolution = 20,000 steps/rev
DRes (Index 2) = 20,000 / 50 = 400
Write 400 to DRes.
```
000-Tx: 01 10 00 02 00 01 02 01 90 A6 4E
001-Rx: 01 10 00 02 00 01 A0 09
```
Response is 1 word (16bits) successfully written to Index 2.

1.2 Parameter Index Definitions

Syntax for the MODBUS protocol parameter index table is defined below:

INT[x] Take the integer part of the expression x

FRAC[x] Take the fractional part of the expression x

R **Read Only** – Parameters with this attribute can only be read, never written to. For example, index 112, the drive’s heat sink temperature, is read-only.

RAM **Read** / **Write RAM** – These parameters are volatile and are lost (initialized to default values) on power-up or reset. An example is index 15, software disable.

EE **Read / Write EEPROM** – Most of the index parameters are of type EE. The parameter’s value is stored in non-volatile memory. Whenever the parameter is changed, the drive automatically stores the new value.

P **Protected** – A small group of parameters are stored in a protected segment of non-volitile (eeprom) memory. They are loaded during the manufacturing process and will remain intact, even after a firmware upgrade or parameter reset. The serial number, index 117-120 is an example of a protected index parameter.

Index	Name	Description	Type
2	DRes	Command step resolution electrical cycle modulo count. Steps (or microsteps) per rotor tooth. DRes is used in the drive to calculate the step resolution (steps per rev).	EE
		Formula	
		$DRes = \frac{Step\ Resolution}{Motor\ Tooth\ Count}$ Motor Tooth Count = 50 for all motors with 200 full-steps per rev.	
		Default Value	
		500	

Index	Name	Description	Type
7	OutputCfg	Programmable output configuration (Rev II Only). Selects the function of the digital output.	EE
		Formula	
		OutputCfg = 0 (No Function) *1 (EOT Latched)* *2 (Motor Moving)* *3 (Motion Node Active)* *4 (Stalled)*	
		Default Value	
		0x0001	

Index	Name	Description	Type
8	OutputPol	Output active state polarities (Rev II Only).	EE
		Formula	
		Bit 0: Output #1 (the only digital output) *OutputPol Bit 0 = 0 (dec. 0) Output #1 is Active Open* *OutputPol Bit 0 = 1 (dec. 1) Output #1 is Active Closed*	
		Default Value	
		0x0001	

Index	Name	Description	Type
9	H4thPercent	Amplitude value for 4th harmonic compensation (Set by L1 slider on X-Smoothness screen in GUI)	EE
		Formula	
		$H4thPercent = 0 \le INT\left[\frac{\%\ Amplitude}{100}(24575)\right] \le 2457$	
		Default Value	
		0 (Disabled)	

Index	Name	Description	Type
10	H4thPhase	Phase value for 4th harmonic compensation (Set by L2 slider on X-Smoothness screen in GUI)	EE
		Formula	
		H4thPhase =±100	
		Default Value	

0	(Disabled)

Index	Name	Description	Type
11	IdleTrig	Timer value for Idle Current Reduction, which reduces the motor current to a specified percentage of the peak commanded current. The Idle mode timer begins counting in 250us increments after the absence of step deltas for 6.4ms.	EE
		Formula	
		IdleTrig = 0 (Disabled) IdleTrig is in increments of ¼ of a ms. Time in milliseconds is (IdleTrig) * 0.25	
		Default Value	
		0 (Disabled)	

Index	Name	Description	Type
12	IdlePercent	Percentage reduction of the peak commanded current vector magnitude when timer trigger is satisfied	EE
		Formula	
		$IdlePercent = 0 \le INT\left[\frac{\%}{100}(32767)\right] \le 32767$	
		Default Value	
		32767	

Index	Name	Description	Type
15	SoftShtDwn	Software override of the amplifier enable input (Shutdown).	RAM
		Formula	
		SoftShtDwn = 0 (Enable input is in control of amplifier) *SoftShtDwn = Non-zero value. Useable range is -65535 to 65535 (Amplifier is disabled regardless of enable input state)*	
		Default Value	
		0	

Index	Name	Description	Type
16-17	StepCnt_L	32 bit command step position. This is the currently commanded position value (position value at which the drive thinks it is located). (Index 16 is the low 16 bits; Index 17 is the high 16 bits.)	RAM
	StepCnt_H	**Formula**	
		N/A	
		Default Value	
		0	

Index	Name	Description	Type
18	DirPol	System direction polarity for direction input and internal move engine. This is the Rotation Polarity setting on the GUI Command screen.	EE
		Formula	

DirPol is set with bit 15 (sign bit) of the 16 bit index #18.

DirPol = 0 or positive value (Normal) Useable range is 0 to 32767.
DirPol = negative value (Inverted) A value of -1 also works, since it is looking for the sign bit (bit 15) of a 16 bit variable to be high. Useable range is 32768 to 65535.

Default Value
0

Index	Name	Description	Type
19	SDPol	Sets the active polarity of Shutdown input. This is the Enable Polarity setting on the GUI Command screen.	EE

Formula

SDPol = 0 (Active Low or Active Open)
SDPol = Non-zero value in 16 bit variable (Active High or Active Closed). Useable range is 1 to 65535 (any bits high).

Default Value
0

Index	Name	Description	Type
22-23	SerJog	Serial jog velocity - A non-zero value will start a continuous move at the specified speed.	RAM

Formula

$$Velocity = (Velocity\ RPS)(Step\ Resolution)(250E-6)$$

32 bit, single precision floating point value

31	30	23	0
s	e	f	

s = 1 bit sign
e = 8 bit biased exponent
f = 23 bit fraction

Index 22 is the low 16 bits; Index 23 is the high 16 bits.

To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal.
To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (22 and 23).

Default Value
0.0

Index	Name	Description	Type
27	FeatureSel	Feature enable / disable selection bits.	EE

Formula

Bit 15-06:	*Unused*
Bit 05:	*Enable Time / Distance Move Profile Definition*
Bit 04:	*Enable Multi-Stepping*
Bit 03:	*Enable Dynamic Smoothing*

Bit 02: Enable Current Reduction *Bit 01: Enable Anti-Resonance* *Bit 00: Enable Encoderless Stall Detect* *Set each bit high to enable and low to disable each feature.* *Setting this through Modbus will not necessarily overwrite the checkboxes in the GUI.*
Default Value
0

Index	Name	Description	Type
28	H2ndPercent	Amplitude value for 2nd harmonic compensation (Set by M1 slider on X-Smoothness screen in GUI)	EE
		Formula	
		H2ndPercent = 0 to 32767	
		Default Value	
		0 (Disabled)	

Index	Name	Description	Type
29	H2ndPhase	Phase value for 2nd harmonic compensation (Set by M2 slider on X-Smoothness screen in GUI)	EE
		Formula	
		H2ndPhase = 0 to 512	
		Default Value	
		0 (Disabled)	

ndex	Name	Description	Type
34	UnitAddress	Unit address for MODBUS RTU communications.	EE
		Formula	
		UnitAddress $= 1 \le x \le 255$	
		Default Value	
		1	

ndex	Name	Description	Type
35	ExConfig	Read-Only - Previous power-up state of the external setup switches. Users can read this parameter, but should not write to it.	EE
		Formula	
		16 Bit state value 15 12 3 0 Unused | DIPS (Bits) | Rotary (Hex)	
		Default Value	
		0	

Index	Name	Description	Type
36	UnitsLabel	Selectable name of the user units (defines rotary or linear motion)	EE
		Formula	
		UnitsLabel = 0 = Steps	

1 = Revs *2 = Millimeters* *3 = Inches*
Default Value
1

Index	Name	Description	Type
37-38	GRatio	User units gear ratio	EE
		Formula	
		$GRatio = \frac{Motor\ Revs}{User\ Unit}$ *32 bit, single precision floating point value* 31 (s), 30–23 (e), 22–0 (f) *s = 1 bit sign* *e = 8 bit biased exponent* *f = 23 bit fraction* Index 37 is the low 16 bits; Index 38 is the high 16 bits. To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal. To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (37 and 38).	
		Default Value	
		1.0	

Index	Name	Description	Type
39-40	Jog Rate	Jog acceleration and deceleration rates in user units used by jog inputs and serial velocity commands. This value is scaled by the user units gear ratio GRatio into Revs / s^2	EE
		Formula	
		$JogRate = \left(Rate\ RPS^2\right)\left(Step\ Resolution\right)\left(6.25E-8\right)$ *32 bit, single precision floating point value* 31 (s), 30–23 (e), 22–0 (f) *s = 1 bit sign* *e = 8 bit biased exponent* *f = 23 bit fraction* Index 39 is the low 16 bits; Index 40 is the high 16 bits. To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal. To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (39 and 40).	

Default Value
0.015625 (10 RPS² at 25000 steps / rev)

Index	Name	Description	Type
41-42	Jog High	Jog high speed velocity used by the jog inputs. This value is scaled by the user units gear ratio GRatio into Revs / s	EE
		Formula	
		$Velocity = (Velocity\ RPS)(Step\ Resolution)(250E-6)$ *32 bit, single precision floating point value* 31 30 23 0 | s | e | f | *s = 1 bit sign* *e = 8 bit biased exponent* *f = 23 bit fraction* Index 41 is the low 16 bits; Index 42 is the high 16 bits. To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal. To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (41 and 42).	
		Default Value	
		12.5 (2 RPS at 25000 steps / rev)	

Index	Name	Description	Type
43-44	Jog Low	Jog low speed velocity used by the jog inputs. This value is scaled by the user units gear ratio GRatio into Revs / s	EE
		Formula	
		$Velocity = (Velocity\ RPS)(Step\ Resolution)(250E-6)$ *32 bit, single precision floating point value* 31 30 23 0 | s | e | f | *s = 1 bit sign* *e = 8 bit biased exponent* *f = 23 bit fraction* Index 43 is the low 16 bits; Index 44 is the high 16 bits. To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal. To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (43 and 44).	
		Default Value	
		3.125 (0.5 RPS at 25000 steps / rev)	

Index	Name	Description	Type
45-46	Stop Rate	Stop decel rate used by stop inputs and the serial break command. This value is scaled by the user units gear ratio GRatio into Revs / s²	EE

Formula

$$StopRate = \left(Rate\ RPS^{2}\right)\left(Step\ Resolution\right)\left(6.25E\ \text{-}\ 8\right)$$

32 bit, single precision floating point value

31	30 … 23	… 0
s	e	f

s = 1 bit sign
e = 8 bit biased exponent
f = 23 bit fraction

Index 45 is the low 16 bits; Index 46 is the high 16 bits.

To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal.
To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (45 and 46).

Default Value

0.15625 (100 RPS² at 25000 steps /rev)

Index	Name	Description	Type
47-48	MtrInertia	Rotor inertia value of the motor in kg-cm²	EE

Formula

32 bit, single precision floating point value

31	30 … 23	… 0
s	e	f

s = 1 bit sign
e = 8 bit biased exponent
f = 23 bit fraction

Index 47 is the low 16 bits; Index 48 is the high 16 bits.

To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal.
To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (47 and 48).

Default Value

0.0

Index	Name	Description	Type
49-51	InputCfg	Programmable input configurations for inputs #1-#9	EE

Formula

Every 4 bits defines an input for a total of 36 bits over 3 words.
Input #1 is the least significant 4 bits:

15-13: Undefined *12: Stop Move on Edge* *11: Stop Move* *10: Start / Stop Pulsed* *09: Start Move* *08: Soft Reset* *07: Move Select* *06: Jog Speed Select* *05: Jog-* *04: Jog+* *03: Home* *02: EOT-* *01: EOT+* *00: No Function* Example: 1: MoveSelect_1 (mode 7) 2: MoveSelect_2 (mode 7) 3: Start Move (mode 9) 4: EOT+ (mode 1) 5: EOT- (mode 2) 6: Jog Speed (mode 6) 7: Jog+ (mode 4) 8: Jog- (mode 5) 9: Home (mode 3) Index 49 = 6519 (0x1977) Index 50 = 21602 (0x5462) Index 51 = 3 (0x0003) Total Value = 0x354621977 or `0011 0101 0100 0110 0010 0001 1001 0111 0111 Bin` `#9=3 #8=5 #7=4 #6=6 #5=2 #4=1 #3=9 #2=7 #1=7`
Default Value
3,2,1,7,7,7,7,7,7

Index	Name	Description	Type
52	InputPol	Input active state polarities (Active Open or Active Closed) configuration.	EE
		Formula	
		Bits 8 - 0: Inputs #9-#1 respectively (Input#1 is bit 0) *InputPol Bit = 0 for Active Open* *InputPol Bit = 1 for Active Closed*	
		Default Value	
		511 (0x01FF) (All inputs set to Active Closed)	

Index	Name	Description	Type
53	InputDB	Input debounce time in 250us increments	EE
		Formula	
		$InputDB = INT\left[\frac{Delay\ ms}{0.25}\right]$	

Default Value
0.0ms

Index	Name	Description	Type
54-55	TestSpeed1	Test speed for tuning 4th harmonic in RPS. Set by GUI.	EE

Formula

32 bit, single precision floating point value

31	30	23	0
s	e	f	

s = 1 bit sign
e = 8 bit biased exponent
f = 23 bit fraction

Index 54 is the low 16 bits; Index 55 is the high 16 bits.

To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal.
To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (54 and 55).

Default Value
0.0

Index	Name	Description	Type
56-57	TestSpeed2	Test speed for tuning 2nd harmonic in RPS. Set by GUI.	EE

Formula

32 bit, single precision floating point value

31	30	23	0
s	e	f	

s = 1 bit sign
e = 8 bit biased exponent
f = 23 bit fraction

Index 56 is the low 16 bits; Index 57 is the high 16 bits.

To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal.
To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (56 and 57).

Default Value
0.0

Index	Name	Description	Type
58-59	TestSpeed3	Test speed for tuning DC offset fundamental in RPS. Set by GUI.	EE

Formula

32 bit, single precision floating point value

31	30	23	0
s	e	f	

s = 1 bit sign *e = 8 bit biased exponent* *f = 23 bit fraction* Index 58 is the low 16 bits; Index 59 is the high 16 bits. To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal. To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (58 and 59).
Default Value
0.0

Index	Name	Description	Type
60-65	CfgName	Configuration Name	EE
		Formula	
		10 byte string	
		Default Value	
		“Untitled1”	

Index	Name	Description	Type
66-71	MtrName	Motor Name	EE
		Formula	
		10 byte string	
		Default Value	
		“None”	

Index	Name	Description	Type
72-73	RateLimit	Accel and decel limit in RPS² for move profiles (used by GUI)	EE
		Formula	
		32 bit, single precision floating point value 31 30 23 0 | s | e | f | *s = 1 bit sign* *e = 8 bit biased exponent* *f = 23 bit fraction* Index 72 is the low 16 bits; Index 73 is the high 16 bits. To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal. To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (72 and 73).	
		Default Value	
		100000.0	

Index	Name	Description	Type
74-75	VelLimit	Velocity limit in RPS for move profiles (used by GUI)	EE

Formula
32 bit, single precision floating point value
31 30 23 0
s | e | f
s = 1 bit sign *e = 8 bit biased exponent* *f = 23 bit fraction*
Index 74 is the low 16 bits; Index 75 is the high 16 bits.
To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal. To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (74 and 75).
Default Value
50.0

Index	Name	Description	Type
76-77	LdInertia	Load inertia in kg-cm² (used by GUI to calculate recommended values)	EE

Formula
32 bit, single precision floating point value
31 30 23 0
s | e | f
s = 1 bit sign *e = 8 bit biased exponent* *f = 23 bit fraction*
Index 76 is the low 16 bits; Index 77 is the high 16 bits.
To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal. To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (76 and 77).
Default Value
0.0

Index	Name	Description	Type
78-79	PeakTorque	Peak motor torque value in N-m (used by GUI to calculate recommended values)	EE

Formula
32 bit, single precision floating point value
31 30 23 0
s | e | f
s = 1 bit sign *e = 8 bit biased exponent* *f = 23 bit fraction*

Index 78 is the low 16 bits; Index 79 is the high 16 bits. To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal. To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (78 and 79).
Default Value
0.0

Index	Name	Description	Type
80	CmdVel	Command velocity	R
		Formula	
		$Velocity\ RPS = \frac{(CmdVel)(1000)}{25600}$	
		Default Value	
		0	

Index	Name	Description	Type
96	EEDisable	EEPROM disable flag. Disables the EEPROM so all writes are only stored to RAM.	RAM
		Formula	
		EEDisable = 0 (EEPROM enabled - writes values to EEPROM) *EEDisable = 65535 (EEPROM disabled - writes values to RAM only)*	
		Default Value	
		0	

Index	Name	Description	Type
98	BaseStat1	Primary base unit status word	R
		Formula	
		Bit 15: P70560 DC Unit Identification (Set for DC unit) *Bit 14: Motion Node Flag (Set if Motion Node Enabled)* *Bit 13-10: Unused* *Bit 09: Motion Node Active (Set if sequence is active)* *Bit 08: Open Motor Phase Detected* *Bit 07: EOT- Latched* *Bit 06: EOT+ Latched* *Bit 05: EEPROM Busy Flag (Set if 5ms write cycle is active)* *Bit 04: Amp Enabled Flag (Set is base unit is enabled)* *Bit 03-00: 4 Bit Base Unit Fault Code* *Fh-8h [Reserved]* *7h EEPROM Checksum* *6h Undervoltage* *5h General Fault* *4h Drive Overtemperature* *3h Overvoltage* *2h Overcurrent* *1h Stalled* *0h No Faults*	
		Default Value	

0

Index	Name	Description	Type
99	BaseStat2	Secondary base unit status word	R
		Formula	
		Bits 15-10: Unused *Bit 09: Output State #1* *Bit 08: Input State #9* *Bit 07: Input State #8* *Bit 06: Input State #7* *Bit 05: Input State #6* *Bit 04: Input State #5* *Bit 03: Input State #4* *Bit 02: Input State #3* *Bit 01: Input State #2* *Bit 00: Input State #1*	
		Default Value	
		0	

Index	Name	Description	Type
102	OpReqFlags	Operation Request Flags	RAM
		Formula	
		Bit 15-05: Reserved *Bit 04: Auto Null Request* *Bit 03: Load FLASH Memory Request* *Bit 02: System Reset Request* *Bit 01: Default Configuration Request* *Bit 00: Motor Probe Request*	
		Default Value	
		0	

Index	Name	Description	Type
103	ActiveMove	Holds the currently active move or the last move active prior to a fault	R
		Formula	
		Default Value	
		0	

Index	Name	Description	Type
104-105	StoredFlt1	Last 8 drive faults	P
	StoredFlt2	**Formula**	
		32 bit value containing 8, 4 bit fault codes	

31	28	3	0
Oldest fault		Latest Fault	

Bit 03-00: 4 Bit Base Unit Fault Code

Fh-8h	*[Reserved]*
7h	*EEPROM Checksum*
6h	*Undervoltage*
5h	*General Fault*

4h Drive Overtemperature *3h Overvoltage* *2h Overcurrent* *1h Stalled* *0h No Faults*
Default Value
0

Index	Name	Description	Type
106	MvSelect	Selects one of the 63 stored moves for motion and immediately starts the stored move	RAM
		Formula	
		$MvSelect = 1 \le x \le 63$	
		Default Value	
		0	

Index	Name	Description	Type
107	Break	Halts all motion and stored move execution	RAM
		Formula	
		Break = non-zero value halts all motion, including stored moves	
		Default Value	
		0	

Index	Name	Description	Type
112	DriveTemp	Temperature of drive heat sink	R
		Formula	
		$C^\circ = \frac{DriveTemp}{208.523636} - 50.0$	
		Default Value	
		0	

Index	Name	Description	Type
113	DC_BusV	Actual voltage level of the DC bus	R
		Formula	
		AC Version: $Volts = \frac{DC\_BusV}{59.459263}$ *DC Version:* $Volts = \frac{DC\_BusV}{327.68}$	
		Default Value	
		0	

Index	Name	Description	Type
114	OSVer	Operating System Version	R

Formula
Version = OSVer / 100 *Example: 210 for version 2.10*
Default Value
1.00

Index	Name	Description	Type
115-116	ExVerInfo	Extended Operating System Version Information	R
		Formula	
		4 Byte String Index 115 is the high word; Index 116 is the low word. Each word is in Little Endian format. Example: OS Verson in GUI reads v2.10 B_A1 Index 115 = 24386 (0x5F42) Index 116 = 12609 (0x3141) Hex value (4 bytes) = 5F 42 31 41 ASCII codes: 5F = _ (underscore) 42 = B 31 = 1 41 = A So, 42 5F 41 31 (hex) = B_A1	
		Default Value	
		Str="i115Lsb + i115Msb + i*116Lsb + i116Msb"*	

Index	Name	Description	Type
117-120	SerNum_LDW	Unit serial number ASCII string Index 117 is the high word; Index 120 is the low word. Each word is in Little Endian format. Example: Serial Number in GUI reads B06A1234 Index 117 = 12354 (0x3042) Index 118 = 16694 (0x4136) Index 119 = 12849 (0x3231) Index 120 = 13363 (0x3433) Hex value (8 bytes) = 30 42 41 36 32 31 34 33 ASCII codes: 30 = 0 42 = B 41 = A 36 = 6 32 = 2 31 = 1 34 = 4 33 = 3 So, 42 30 36 41 31 32 33 34 (hex) = B06A1234	P
		Formula	
		Str="i117Lsb + i117Msb + i*118Lsb + i118Msb + i119Lsb + i119Msb + i120Lsb + i120Msb"*	
		Default Value	

Index	Name	Description	Type
128+i	Move Type & Jump #	Defines the stored move type and move jump number (following move number). Index = 128 + (stored move number – 1) * 10 Example: Index for move #2 would be 128 + (2-1) * 10 = 138 Example: Index for move #28 would be 128 + (28-1) * 10 = 298	EE

Formula
Bits 15-08: Jump Number *Jump Number = 0 (No end of move jump)* *Jump Number* $= 1 \le x \le 63$ *Bits 07-00: Move Type* *Move Type = 0 (Incremental Move)* *= 1 (Absolute Move)* *= 2 (Home Move)*

Default Value
0

Index	Name	Description	Type
129-130 +i	Move Accel	Defines the stored move acceleration in user units. Index for high 16 bits = 130 + (stored move number – 1) * 10 Index for low 16 bits = 129 + (stored move number – 1) * 10 Example: Index for move #2 would be 130 + (2-1) * 10 = 140 and 129 + (2-1) * 10 = 139 Example: Index for move #28 would be 130 + (28-1) * 10 = 400 and 129 + (28-1) * 10 = 399	EE

Formula

$$MoveAccel = (Rate\ RPS^2)(Step\ Resolution)(6.25E-8)$$

32 bit, single precision floating point value

31	30 — 23	23 — 0
s	e	f

s = 1 bit sign
e = 8 bit biased exponent
f = 23 bit fraction

Index 129+i is the low 16 bits; Index 130+i is the high 16 bits.

To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal.
To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex

values to the corresponding index numbers (129+i and 130+i).
Default Value
0.0

Index	Name	Description	Type
131-132 +i	Move Vel	Defines the stored move velocity in user units. Index for high 16 bits = 132 + (stored move number – 1) * 10 Index for low 16 bits = 131 + (stored move number – 1) * 10 Example: Index for move #2 would be 132 + (2-1) * 10 = 142 and 131 + (2-1) * 10 = 141 Example: Index for move #28 would be 132 + (28-1) * 10 = 402 and 131 + (28-1) * 10 = 401	EE

Formula

$$MoveVel = (Velocity\ RPS)(Step\ Resolution)(250E\text{-}6)$$

32 bit, single precision floating point value

31	30	23	0
s	e	f	

s = 1 bit sign
e = 8 bit biased exponent
f = 23 bit fraction

Index 131+i is the low 16 bits; Index 132+i is the high 16 bits.

To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal.
To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (131+i and 132+i).

Default Value

0.0

Index	Name	Description	Type
133-134 +i	Move Decel	Defines the stored move deceleration in user units. Index for high 16 bits = 134 + (stored move number – 1) * 10 Index for low 16 bits = 133 + (stored move number – 1) * 10 Example: Index for move #2 would be 134 + (2-1) * 10 = 144 and 133 + (2-1) * 10 = 143 Example: Index for move #28 would be 134 + (28-1) * 10 = 404 and 133 + (28-1) * 10 = 403	EE

Formula

$$MoveDecel = (Rate\ RPS^2)(Step\ Resolution)(6.25E\text{-}8)$$

32 bit, single precision floating point value

31	30	23	0
s	e		f

s = 1 bit sign
e = 8 bit biased exponent
f = 23 bit fraction

Index 133+i is the low 16 bits; Index 134+i is the high 16 bits.

To read through Modbus, read as hexadecimal and then convert from Hex to Floating-Point Decimal.
To assign the value through Modbus, convert the desired value (Floating-Point Decimal) to hexadecimal. Then write the hex values to the corresponding index numbers (133+i and 134+i).

Default Value
0.0

Index	Name	Description	Type
135-136 +i	Move Distance	Defines the stored move distance in user units. Index for high 16 bits = 136 + (stored move number – 1) * 10 Index for low 16 bits = 135 + (stored move number – 1) * 10 Example: Index for move #2 would be 136 + (2-1) * 10 = 146 and 135 + (2-1) * 10 = 145 Example: Index for move #28 would be 136 + (28-1) * 10 = 406 and 135 + (28-1) * 10 = 405	EE
		Formula	
		*MoveDistance = (Distance in Revs) * (Step Resolution)*	
		Default Value	
		0	

Index	Name	Description	Type
137+i	Move Time Delay	Defines the stored move end of move time delay in increments of 250 microseconds. For example, setting the move time to 4000 will produce a 1 sec delay. (1 ms = 4 increments). Index = 137 + (stored move number – 1) * 10 Example: Index for move #2 would be 137 + (2-1) * 10 = 147 Example: Index for move #28 would be 137 + (28-1) * 10 = 407	EE
		Formula	

$\mathit{Move\,Time\,Delay} = 0 \leq (x\ ms)\,(4) \leq 32767$
Default Value
0

Entering move data via index registers

Typically, move profile data is created and loaded into a P7000 using the Motion screen in P7000Tools. Move data can also be loaded into a drive via the ModBus interface. The data for all 63 move profiles are stored in a table of 630 registers, where each move occupies ten registers. When populating the table with move data via P7000 Tools, the software takes care of certain lower level details. When using the ModBus interface, these details must now be done explicitly by the user. Following are some examples to aid in understanding how to do this.
Data storage and representation

All index registers in the P7000 are 16 bit. Move profile data may be integer, floating point, or bit level in its representation. A given move's parameters my occupy one or two registers. To write a move speed expressed in meaningful units like 2.5 RPS (revolutions per second), the desired value must first be scaled to a format the P7000 uses internally, converted to a 32 bit floating point format, and written to the drive as two individual 16 bit values.

Example 1
Given a desired move speed of 2.5 RPS, what data should be written to the appropriate registers for move #8 ?
First, scale user units to internal drive units:
MoveVelocity = (Velocity in RPS) * (Step per Revolution) * (250 E –6)
= (2.5) * (25,000) * (250E-8)
=15.625
Next, convert the move velocity from decimal to 32 bit single precision floating point representation:
15.625 = 0x417A 0000 (see http://babbage.cs.qc.edu/IEEE-754/ for a converter)
In the P7000, the MSW (most significant word) goes at the higher index and the LSW (least significant word) goes at the lower index,(little endian) so:
LSW = 0x0000 (write to base index)
MSW = 0x417A (write to base index + 1)
Finally, determine the index registers to write the data to:
Method A) Use the formula for index 131-132 from above: the base index for the move velocity for stored move #8 = 131+(move#-1) * 10)
= 131 + (8-1)*10
= 201
Method B) using the table below, note that the LSW for the move velocity for stored move #8 = 201 and the MSW = 202
Setting index 201=0 and index 202=0x417A causes the move velocity parameter for stored move #8 to be 2.5 RPS. This can be confirmed using P7000 Tools.
P7000 Series MODBUS Documentation 24 of 25

Example 2
A P7000 user would like to configure move profile #8 using the ModBus interface. What data needs to be sent to the drive to achieve the following move parameters?
Acceleration: 2.5 rev/sec^2
Deceleration: 5.6 rev/sec^2
Distance: 11.9 revolutions, in negative direction
Velocity: 619 rpm
Delay time: 1.0 sec
Incremental move, link to move #5
Assume steps per rev = 25,000

a) Move Acceleration and Deceleration
See the descriptions for index 129/130 and 133/134

Register acc/dec= (acc/dec in RPS^2) * (Step per Revolution) * (6.25E -8)
Register acc = (2.5) * (25000) * (6.25E -8) = 0.00390625
Register dec = (5.6) * (25000) * (6.25E-8) = 0.00875000

b) Move Velocity
See the description for index 131/132
Register velocity = (Velocity in RPS) * (Step per Revolution) * (250 E –6)
619 rpm = 619/60 = 10.31666667 RPS
Register velocity = (10.31666667) * (25000) * (250E-6) = 64.47916667

c) Move Distance
See the description for index 135/136
Register distance = (distance in revolutions) * (Step per Revolution)
Register distance = (11.9) * (25000) = 297500

d) Delay Time
See description for index 137
Register delay value = (delay time in seconds) / (250E-6)
Register delay value = 1.000) / (250E-6) = 4000
P7000 Series MODBUS Documentation 25 of 25

e) Move Type and Jump Number
See description for index 128
Move type = incremental = 0 (LSB)
Jump number = 5 (MSB)
Register value = 0500 hex = 1280 decimal

198 Move #8 type and jump number
199 Move #8Accel, LSW of 32 bit float
200 Move #8Accel, MSW of 32 bit float
201 Move #8 Velocity, LSW of 32 bit float
202 Move #8 Velocity, MSW of 32 bit float
203 Move #8 Decel, LSW of 32 bit float
204 Move #8 Decel, MSW of 32 bit float
205 Move #8 Distance, LSW of 32 bit INT
206 Move #8 Distance, MSW of 32 bit INT
207 Move #8 delay time