



Animatics

SmartMotor Series, RTC Series

Overview

Maple Systems' **Silver Series/HMI500 Series** Family of Human-Machine Interfaces (Maple HMIs) communicate with Animatics SmartMotor and RTC controllers using the **SmartMotor ASCII protocol**. The Maple HMI is the master in a single-master, single-slave or multiple-slave format.

The driver file is: **AnimaticsSMC.pds** and will show up as “**Animatics SMC v1.00**” in the PLC Type list.

Compatible Controllers	
Family	Model
SmartMotor	SM17xx, SM23xx, SM34xx, SM42xx, SM56xx
RTC	RTC3000

Communications Cable

Connect the Maple HMI via 3-wire RS-232 (M-M, Null Modem) through the Animatics communications Y-cable (CBL SMA1-10) to the main port on the SmartMotor controller. To connect multiple controllers, use the Animatics “ADD-A-MOTOR” cable.

A list of communications cables offered by Maple Systems as well as cable assembly instructions to assist you in assembling your own communications cable are available on our website at www.maple-systems.com/cables.htm.

WARNING: If your HMI communications cable is not wired exactly as shown in our cable assembly instructions, damage to the Maple HMI or loss of communications can result.

Controller Settings

The SmartMotor Controller must be set to “ECHO” mode.

Accessible Controller Commands and Memory

The following tables list the controller's commands and memory ranges that Maple's HMIs are able to access. (Please note that your controller's memory range may be *smaller* or *larger* than that supported by Maple's HMIs.)

Register Memory:

Device	Address Range	Details
VarAZ ^{1,2}	0 - 25	Read/Write values in single letter variables 'a to z'. This is a 32-bit register type ²
ArrayAB ⁴	0 – 199	Read/Write values in ab[]. This is an 8-bit signed byte.
ArrayAW ⁴	0 – 99	Read/Write values in aw[]. This is a 16-bit signed register.
ArrayAL ^{2, 4}	0 – 49	Read/Write values in al[]. This is a 32-bit signed register type ²
Clock	0 – 0	Read/Write the system clock. For values > 32767, set the Number of Words to 2.
Counter	0 – 0	Read the secondary encoder. Set the Number of Words to 2.
ModeFollow ³	0 – 0	Set the controller's mode to Follow. Use a Set Word object; Style = Set Constant; Value = 0, 1, 2, or 4.
ModePosition ³	0 – 0	Set the controller's mode to Position.
ModeStepDirection ³	0 – 0	Set the controller's mode to Step & Direction.
ModeTorque ³	0 – 0	Set the controller's mode to Torque.
ModeVelocity ³	0 – 0	Set the controller's mode to Velocity.
MotionGo	0 – 0	Start motion with the current parameters.
MotionStopAbrupt	0 – 0	Stop motion with no deceleration.
MotionStopDecel	0 – 0	Stop motion with with current deceleration settings.
MotionAccel	0 – 0	Read/Write the Acceleration setting.
MotionDistance	0 – 0	Read the current distance, write the target distance.
MotionOrigin	0 – 0	Set the Origin position.
MotionPosition	0 – 0	Read the current position, write the target position.
MotionPosError	0 – 0	Read the current position error.
MotionPosErrLim	0 – 0	Read/Write the Position Error Limit.
MotionTorque	0 – 0	Read/Write the Torque (Torque mode only)
MotionVelocity	0 – 0	Read/Write the maximum allowed velocity.

Status	0 – 0	Read the 16 status bits as a word.
ProgramRun	0 – 0	Run the current program in the SMC.
ProgramEnd	0 – 0	Terminate the current program in the SMC.
ProgramGosub	0 – 0	Run the specified label in the SMC program as a Subroutine. Use a Set Word object; Style = Set Constant; Value = 0 to 999, to specify the label.
ProgramGoto	0 – 0	Jump to the specified label in the SMC program. Use a Set Word object; Style = Set Constant; Value = 0 to 999, to specify the label.

Discreet Memory:

Device	Address Range	Details
VarAZBit ¹	0.00 - 25.15	Read/Write bits in the single letter variable registers. Bit must be specified to 2 digits. Bit access is for the lowest 16-bit order. The upper 16-bit access is not supported at this time.
ArrayAWBit	0.00 – 99.15	Read/Write bits in aw[]. Bit must be specified to 2 digits.
StatusBit	0.00 – 0.15	Read bits in Status Word. Bit must be specified to 2 digits.
ResetZFlags	0 – 0	Reset all Z flags.

Notes:

- Specify the letter variable a - z with the number 0 - 25 as the address. (I.e. Var a = 0, b = 1, z = 25)
- When using this register type, set the number of words to '2'
- A Read operation (Numeric Data or Word Lamp) using these Devices will return a numeric value representing the current mode:
 - 0 – Unknown or Undefined
 - 1 – Absolute Position
 - 2 – Relative Position
 - 3 – Velocity
 - 4 - Torque
 - 5 - Follow
 - 6 – Step & Direction
 - 7 – Cam Table
 - 8 – Drive
 - 9 – Follow w/ Multiplier
 - 10 – Position Error
 - 11 – Motor Off
 - 12 – Contouring
- Array AB, array AW and array AL are all the same memory area. See the User Assigned Variable Memory Map chart that follows for details. As this chart shows the array AL[0], which is a long word that uses the first two 16-bit words (array AW[0] and array AW[1]); it also uses the first four 8-bit bytes (array AB[0], array AB[1], array AB[2], and array AB[3]).

User Assigned Variables - Memory Map:

Single variables a through z are addressed as: VarAZ[0] through VarAZ[25], respectively.

Double & triple variables are accessed by using ArrayAL, ArrayAW, or ArrayAB register types as shown in the table below:

Var.	Long	Word	Byte		Var.	Long	Word	Byte
aa	ArrayAL[0]	ArrayAW[0] ArrayAW[1]	ArrayAB[0] ArrayAB[1] ArrayAB[2] ArrayAB[3]		ii	ArrayAL[8]	ArrayAW[16] ArrayAW[17]	ArrayAB[32] ArrayAB[33] ArrayAB[34] ArrayAB[35]
bb	ArrayAL[1]	ArrayAW[2] ArrayAW[3]	ArrayAB[4] ArrayAB[5] ArrayAB[6] ArrayAB[7]		jj	ArrayAL[9]	ArrayAW[18] ArrayAW[19]	ArrayAB[36] ArrayAB[37] ArrayAB[38] ArrayAB[39]
cc	ArrayAL[2]	ArrayAW[4] ArrayAW[5]	ArrayAB[8] ArrayAB[9] ArrayAB[10] ArrayAB[11]		kk	ArrayAL[10]	ArrayAW[20] ArrayAW[21]	ArrayAB[40] ArrayAB[41] ArrayAB[42] ArrayAB[43]
dd	ArrayAL[3]	ArrayAW[6] ArrayAW[7]	ArrayAB[12] ArrayAB[13] ArrayAB[14] ArrayAB[15]		ll	ArrayAL[11]	ArrayAW[22] ArrayAW[23]	ArrayAB[44] ArrayAB[45] ArrayAB[46] ArrayAB[47]
ee	ArrayAL[4]	ArrayAW[8] ArrayAW[9]	ArrayAB[16] ArrayAB[17] ArrayAB[18] ArrayAB[19]		mm	ArrayAL[12]	ArrayAW[24] ArrayAW[25]	ArrayAB[48] ArrayAB[49] ArrayAB[50] ArrayAB[51]
ff	ArrayAL[5]	ArrayAW[10] ArrayAW[11]	ArrayAB[20] ArrayAB[21] ArrayAB[22] ArrayAB[23]		nn	ArrayAL[13]	ArrayAW[26] ArrayAW[27]	ArrayAB[52] ArrayAB[53] ArrayAB[54] ArrayAB[55]
gg	ArrayAL[6]	ArrayAW[12] ArrayAW[13]	ArrayAB[24] ArrayAB[25] ArrayAB[26] ArrayAB[27]		oo	ArrayAL[14]	ArrayAW[28] ArrayAW[29]	ArrayAB[56] ArrayAB[57] ArrayAB[58] ArrayAB[59]
hh	ArrayAL[7]	ArrayAW[14] ArrayAW[15]	ArrayAB[28] ArrayAB[29] ArrayAB[30] ArrayAB[31]		pp	ArrayAL[15]	ArrayAW[30] ArrayAW[31]	ArrayAB[60] ArrayAB[61] ArrayAB[62] ArrayAB[63]

Var.	Long	Word	Byte		Var.	Long	Word	Byte
qq	ArrayAL[16]	ArrayAW[32] ArrayAW[33]	ArrayAB[64] ArrayAB[65] ArrayAB[66] ArrayAB[67]		vv	ArrayAL[21]	ArrayAW[42] ArrayAW[43]	ArrayAB[84] ArrayAB[85] ArrayAB[86] ArrayAB[87]
rr	ArrayAL[17]	ArrayAW[34] ArrayAW[35]	ArrayAB[68] ArrayAB[69] ArrayAB[70] ArrayAB[71]		ww	ArrayAL[22]	ArrayAW[44] ArrayAW[45]	ArrayAB[88] ArrayAB[89] ArrayAB[90] ArrayAB[91]
ss	ArrayAL[18]	ArrayAW[36] ArrayAW[37]	ArrayAB[72] ArrayAB[73] ArrayAB[74] ArrayAB[75]		xx	ArrayAL[23]	ArrayAW[46] ArrayAW[47]	ArrayAB[92] ArrayAB[93] ArrayAB[94] ArrayAB[95]
tt	ArrayAL[19]	ArrayAW[38] ArrayAW[39]	ArrayAB[76] ArrayAB[77] ArrayAB[78] ArrayAB[79]		yy	ArrayAL[24]	ArrayAW[48] ArrayAW[49]	ArrayAB[96] ArrayAB[97] ArrayAB[98] ArrayAB[99]
uu	ArrayAL[20]	ArrayAW[40] ArrayAW[41]	ArrayAB[80] ArrayAB[81] ArrayAB[82] ArrayAB[83]		zz	ArrayAL[25]	ArrayAW[50] ArrayAW[51]	ArrayAB[100] ArrayAB[101] ArrayAB[102] ArrayAB[103]

Var.	Long	Word	Byte		Var.	Long	Word	Byte
aaa	ArrayAL[26]	ArrayAW[52] ArrayAW[53]	ArrayAB[104] ArrayAB[105] ArrayAB[106] ArrayAB[107]		eee	ArrayAL[30]	ArrayAW[60] ArrayAW[61]	ArrayAB[120] ArrayAB[121] ArrayAB[122] ArrayAB[123]
bbb	ArrayAL[27]	ArrayAW[54] ArrayAW[55]	ArrayAB[108] ArrayAB[109] ArrayAB[110] ArrayAB[111]		fff	ArrayAL[31]	ArrayAW[62] ArrayAW[63]	ArrayAB[124] ArrayAB[125] ArrayAB[126] ArrayAB[127]
ccc	ArrayAL[28]	ArrayAW[56] ArrayAW[57]	ArrayAB[112] ArrayAB[113] ArrayAB[114] ArrayAB[115]		ggg	ArrayAL[32]	ArrayAW[64] ArrayAW[65]	ArrayAB[128] ArrayAB[129] ArrayAB[130] ArrayAB[131]
ddd	ArrayAL[29]	ArrayAW[58] ArrayAW[59]	ArrayAB[116] ArrayAB[117] ArrayAB[118] ArrayAB[119]		hhh	ArrayAL[33]	ArrayAW[66] ArrayAW[67]	ArrayAB[132] ArrayAB[133] ArrayAB[134] ArrayAB[135]

Var.	Long	Word	Byte		Var.	Long	Word	Byte
iii	ArrayAL[34]	ArrayAW[68] ArrayAW[69]	ArrayAB[136] ArrayAB[137] ArrayAB[138] ArrayAB[139]		rrr	ArrayAL[43]	ArrayAW[86] ArrayAW[87]	ArrayAB[172] ArrayAB[173] ArrayAB[174] ArrayAB[175]
jjj	ArrayAL[35]	ArrayAW[70] ArrayAW[71]	ArrayAB[140] ArrayAB[141] ArrayAB[142] ArrayAB[143]		sss	ArrayAL[43]	ArrayAW[88] ArrayAW[89]	ArrayAB[176] ArrayAB[177] ArrayAB[178] ArrayAB[179]
kkk	ArrayAL[36]	ArrayAW[72] ArrayAW[73]	ArrayAB[144] ArrayAB[145] ArrayAB[146] ArrayAB[147]		ttt	ArrayAL[45]	ArrayAW[90] ArrayAW[91]	ArrayAB[180] ArrayAB[181] ArrayAB[182] ArrayAB[183]
lll	ArrayAL[37]	ArrayAW[74] ArrayAW[75]	ArrayAB[148] ArrayAB[149] ArrayAB[150] ArrayAB[151]		uuu	ArrayAL[46]	ArrayAW[92] ArrayAW[93]	ArrayAB[184] ArrayAB[185] ArrayAB[186] ArrayAB[187]
mmm	ArrayAL[38]	ArrayAW[76] ArrayAW[77]	ArrayAB[152] ArrayAB[153] ArrayAB[154] ArrayAB[155]		vvv	ArrayAL[47]	ArrayAW[94] ArrayAW[95]	ArrayAB[188] ArrayAB[189] ArrayAB[190] ArrayAB[191]
nnn	ArrayAL[39]	ArrayAW[78] ArrayAW[79]	ArrayAB[156] ArrayAB[157] ArrayAB[158] ArrayAB[159]		www	ArrayAL[48]	ArrayAW[96] ArrayAW[97]	ArrayAB[192] ArrayAB[193] ArrayAB[194] ArrayAB[195]
ooo	ArrayAL[40]	ArrayAW[80] ArrayAW[81]	ArrayAB[160] ArrayAB[161] ArrayAB[162] ArrayAB[163]		xxx	ArrayAL[49]	ArrayAW[98] ArrayAW[99]	ArrayAB[196] ArrayAB[197] ArrayAB[198] ArrayAB[199]
ppp	ArrayAL[41]	ArrayAW[82] ArrayAW[83]	ArrayAB[164] ArrayAB[165] ArrayAB[166] ArrayAB[167]		yyy	ArrayAL[50]	ArrayAW[100] ArrayAW[101]	ArrayAB[200] ArrayAB[201] ArrayAB[202] ArrayAB[203]
qqq	ArrayAL[42]	ArrayAW[84] ArrayAW[85]	ArrayAB[168] ArrayAB[169] ArrayAB[170] ArrayAB[171]					

Connecting to Multiple Controllers:

The Silver Series HMI supports communicating with multiple controller by using the controller's node-address. On the System Parameters dialog (Edit menu -> System Parameters), under the PLC tab, set the "PLC Station No." to the address of the motor to be communicated with. If this setting is 0, the HMI will send commands to all smart motors (broadcast address).

To access multiple controllers within the same HMI application, the HMI's address mode must be changed to *Extended*. On the System Parameters dialog, select the Editor tab. Change the *Address Mode* to **Extended**. When configuring an object, use the following format for the object's Address box:

[motor number]#[address]

For example, to read aw[7] from motor number 3, the "Device Address" box would contain:

3#7

If the *[motor number]#* is not in the address box, the HMI will use the motor number specified on the PLC tab of the System Parameters dialog.

To make the physical connection between the Silver Series HMI and multiple Smart Motor Controllers, use the Animatics Add-A-Motor cable(s). Connect the HMI to the first motor, the second motor to the first motor, and so on.

TIPS:

To control multiple motors at the same time from one push button, create multiple objects of the same kind and layer them one on top of the other. Give one a button shape and then clear the shape on the others so they appear invisible. For example, if the HOME command for the X-motor (node:1) and the Y-motor (node:2) is set to go home when single-letter variable 'd' is set to 1, then you would create two set-bit objects (one with a shape & a label, the other with no shape or label). Set their attributes to "momentary". One button, set to Device Type: VarAZBit, Device address: 1#3.00 (node#1, d variable, bit 0) and set the other button to VarAZBit, Device address: 2#3.00 (node#2, d variable, bit 0).

If you notice erratic (fluttering) numbers showing from your numeric data or numeric input displays: This is possibly caused by a timing problem. The Animatics doesn't reply as fast as the HMI can send requests. This can cause the HMI to get stuck in a retry loop, and can display bad data. Try adding some turn-around delay into Parameter 2 or the "Edit->System Parameters" menu. Entering a number of 5 in parameter 2, will add 50mS to the turn around delay, a value of 10 will add 100mS. Also try increasing the baud rate.

EZware-500 Settings

The following table lists the communications settings that must be configured in EZware-500. These settings can be found in the Edit-Set System Parameters menu under the PLC tab. Please note:

- the **Recommended Settings** column provides the recommended setting based upon the default settings most commonly used in the Animatics SmartMotor Controller.
- the **Options** column lists EZware-500's options; your PLC may not support every option

Name	Recommended Settings	Options	Important Notes
PLC type:	Animatics SMC Vn.xy		
Serial port I/F:	RS232	RS232, RS485	
Data Bits:	8	7 or 8	Must match the SMC port setting.
Stop Bits:	1	1 or 2	Must match the SMC port setting.
Baud Rate:	9600	9600,19200, 38400,57600, 115200	Must match the SMC port setting. Use the fastest baud rate supported by the controller.
Parity:	None	Even, Odd, None	Must match the SMC port setting.
HMI station No.:	0	N/A	Not used by this protocol
PLC station No.:	1	0-120	Must match the motor number to the SMC; or set to 0 for unaddressed motor(s)
Multiple HMI:	Disable	Disable, Master, Slave	use for multiple OITs
HMI-HMI link speed:	38400	38400, 115200	use for multiple OITs
PLC time out constant (sec)	3.0	1.5 to 5.0	adjust if longer timeout is required
PLC block pack:	0	0-10	see <i>Silver Series Installation and Operation Manual</i>
Parameter 2	0	0-100	Turn-around delay (in 10mS incr.), i.e. 5 = 50mS turn around delay