AKD2G PushCorp Standard Interface

Hardwired Signals

X21/A3 (DIN 1) – Fault Reset

X21/A4 (DIN 2) – Analog Run[2]

X21/A5 (DIN 3) – Hardware Enable[1]

X21/B7 (DOUT 1) – Motor Overload Warning[2]

X21/B5-B6 (DOUT 9) – BTB/RTO

X21/A1-A2 (AIN ±10VDC) – Command Velocity[2]

X21/B1-B2 (AOUT 0-10VDC) – Actual Velocity[2]

X21/A11-B11 – STO Inputs[1]

X21/B3 – DIO 24V Supply*

X21/B4 – DIO 0V/COM[3]

[1] Required for Operation

[2] Analog/Discrete Control Only

[3] Should be bonded to same common as STO

* AKD2G will show a warning if not supplied with 24V

EthernetIP

Scanlist Data

VendCode	452
VendName	Kollmorgen
ProdType	43
ProdTypeStr	Generic
ProdCode	20
MajRev	1
MinRev	3
ProdName	AKD2G-SPI

Assembly Data

	Instance	Bytes	Words
Producing	104	14	7
Consuming	103	6	3

IP Addresses

EthernetIP	192.168.1.13
Service Port	192.168.1.14

IO Map

Control Inputs from AKD2G			
Word(s)	**Byte(s)**	**Bit(s)**	**Description**
0	0	0	Fault
		1	User Configurable
		2	At Tool Change*
		3	User Configurable
		4	User Configurable
		5	User Configurable
		6	User Configurable
		7	User Configurable
	1	8	User Configurable
		9	User Configurable
		10	User Configurable
		11	User Configurable
		12	User Configurable
		13	User Configurable
		14	User Configurable
		15	User Configurable
1-2	2-5	16-47	Actual Velocity
3-4	6-9	48-79	Actual Amperage
5-6	10-13	80-111	Motor Temperature

Control Outputs to AKD2G			
Word(s)	**Byte(s)**	**Bit(s)**	**Description**
0	0	0	Clear Fault
		1	Velocity Enable
		2	Go Tool Change*
		3	User Configurable
		4	User Configurable
		5	User Configurable
		6	User Configurable
		7	User Configurable
	1	8	User Configurable
		9	User Configurable
		10	User Configurable
		11	User Configurable
		12	User Configurable
		13	User Configurable
		14	User Configurable
		15	User Configurable
1-2	2-5	16-47	Velocity Command

*For STC1015 and STC1515 Only

Scaling

Command and Actuals are 32-bit signed integers.

Velocity scaling is whole digit RPM
Example: Reading 6,000(base 10) on the integer is 6,000 RPM

Amperage scaling is 0.001mA.
Example: Reading 2,356(base 10) on the integer is 2.356 amps

Temperature scaling is whole digit Ohms.
Example: Reading 1,018 on the integer is 1,018 Ohm

Equations for converting Ohms to degrees are in the respective spindle manuals.

16-bit Tips and Tricks

The response words from the AKD2G are 32-bit numbers; some systems are limited to reading 16-bit numbers.

These are pseudo-code examples of resolving the 32-bit numbers using its comprising 16-bit numbers (upper and lower words) into a coherent value.

```
IF Word2 = 0 THEN
        ActVel = Word2*2^16+Word1
ELSE
        ActVel = Word2*2^16+Word1 – 2^32
ENDIF
```

```
IF Word4 = 0 THEN
        ActAmp = (Word4*2^16+Word3)/1000
ELSE
        ActAmp = (Word4*2^16+Word3 – 2^32)/1000
ENDIF
```

```
IF Word6 = 0 THEN
        MotTemp = (Word6*2^16+Word5)/1000
ELSE
        MotTemp = (Word6*2^16+Word5 – 2^32)/1000
ENDIF
```

Negative Command Values

Operating the spindle in the reverse direction via EthernetIP requires a negative value at the Group/BTD/BITS output word for the Command Velocity.

Some systems are not capable to directly resolving a negative value when putting it to the output word, so an intermediate step is necessary to convert the negative number.

From a scalar point of view, this is managed with the 2's compliment of the number across the 32-bit word – some robots and most PLCs will automatically resolve the 2's compliment; simpler systems require manipulation of the value.

For example, the target is to reverse 5,000 RPM, and we are working with the 32 bit word for the AKD2G EIP Velocity Command.

5,000(b10) is 00000000000000000001001110001000(b2)
Invert all the bits -> 11111111111111111110110001110111(b2)
Add one -> 11111111111111111110110001111000(b2)
11111111111111111110110001111000(b2) is 4,294,962,296(b10)

So, for the 32 bit word, -5,000 has the scalar value of 4,294,962,296

A simple algorithm can accomplish this with basic mathematic operators (pseudo code):

IF [desired velocity] < 0 THEN
  [command velocity] = 2^{32} +[desired velocity]
ELSE
  [command velocity] = [desired velocity]
END

2^{32} = 4,294,967,296

4,294,967,296 – 5000 = 4,294,962,296

Assigning Output Words

Further manipulation if the output side needs to be split into two groups of 16 bits:

Word1 = [command velocity] MOD 2^{16}

Word2 = ([command velocity] – Word1)/2^{16}