FrameWork Firmware 1.3
Command List

February 11, 1998

FrameWork 1.3 Firmware Commands

Sample Terminal Command Session

Framework commands and responses can contain strings of ASCII characters and numerical information. Numbers can be in one of three formats - Decimal, Hexadecimal, or Binary. All commands and responses are prefixed with a character identifying the command format. These prefix characters are :

Format	Command	Reply	End Reply
Decimal	#	$	%
Hexadecimal	*	(	)
Binary	0xAA	0xA8	0xA9

In Decimal mode, all numerical parameters are separated by a space character. All commands are terminated with a CR. Each reply sequence is completed with an EndReply with a 0 or 1 representing success or error, respectively, and a then followed by a ? or ! for success or error.

To bring up a FrameWork terminal, run FrameWork from within Windows and connect to a DVT Series 800 running FrameWork 1.2. Press **Control Alt T** to open a terminal window. Then use the commands shown below to configure your specific sensors. Commands entered by the user are in **bold** and responses are in *Italics*.

Example: System has two products defined. Product 'Clip' (id=1) is selected as the inspection product and the power on inspection product.

Command/*Response*	Description
#PQ	Query system for products
$ -1 1 1	No selected Product, Product 1 set as power-on and
$ 0 Test Product	inspection product. Product 0 is named 'Test Product'.
$ 1 Clip	Product 1 is named 'Clip'.
% 0	Command completed without error.
?	Command completed without error.
#Pq 2	Query product 2 for pass/fail stats.
% 1	Error performing command.
!	Error performing command.
#Pq 1	Query product 1 for pass/fail stats.
$ 100 2 3 -1	Product 1 has performed 100 Pass, 2 Warn, 3 Fail
% 0	inspections, and the last inspection was Pass.
?	Command completed without error.
#PS 1	Select Product 1
% 0	Command completed without error.
?	Command completed without error.
#SQ	Query selected product ('Clip') for defined sensors.
$ 2 1 Sensor 2	'Sensor 2' has an ID of 2 and is a type 1 sensor.
$ 5 2 Test Path	'Test Path' has an ID of 5 and is a type 2 sensor.

% 0	Command completed without error.
?	Command completed without error.
#SPO 2 48	Set ‘Sensor 2’ (ID=2) to activate user outputs OUT5 and OUT6 on failure. (16(OUT5)+32(OUT6) = 48)
% 0	Command completed without error.
?	Command completed without error.
#SPO 5 16	Set ‘Test Path’ (ID=5) to activate user output OUT5 on failure.
% 0	Command completed without error.
?	Command completed without error.
#Sq 0	Query Sensor 0 of selected product for pass/fail stats.
$ 0 1 Clip Fold	Sensor 0 id is 0, is active, and is named ‘Clip Fold’.
$ 101 2 2 0	Sensor 0 has performed 101 Pass, 2 Warm, and 3 Fail
%0	inspections, and the last inspection result was Pass
?	Command completed without error.

Background Transfer Commands

The background transfer thread is used to transfer information from the IVU to the PC without constant polling from the PC. The background thread uses a separate ('background') communication channel to automatically send requested information from the IVU to the PC, while the 'background' commands are issued from the PC over the primary channel. This means that currently these commands will not work when using the standard ASCII protocol.

B+

Start background transfers. If inspections are not enabled, then user mode pictures will be acquired based on sync mode.

B-

Stop background transfers.

BAG <*Product ID* : INT16> <*Sensor ID* : INT16> <*Graph Type* : INT8>

Add background graph transfers to list of items to transfer. *Sensor ID* = -1 for all sensors.

BAH <*Product ID* : INT16> <*Sensor ID* : INT16> <*History Type* : INT8>

Add background performance/history transfers to list of items to transfer. *Sensor ID* = -1 for all sensors.

BAI <*Win.P1.X* : INT16> <*Win.P1.Y* : INT16> <*Win.P2.X* : INT16>
<*Win.P2.Y* : INT16> <*Compression Format* : UINT8>

Add background image transfers to list of items to transfer. *Compression Format* ignored. Huffman 1D and 2bit residual interlaced image transfers performed.

BDA

Remove all background transfers from list of items to transfer.

BDG <*Product ID* : INT16> <*Sensor ID* : INT16> <*History Type* : INT8>

Remove background graphs from list of items to transfer.

BDH <*Product ID* : INT16> <*Sensor ID* : INT16> <*History Type* : INT8>

Remove background performance/history transfers from list of items to transfer.

BDI

Remove background image transfers from list of items to transfer.

BG <*Image ID* : INT32> <*Product ID* : INT16> <*Sensor ID* : INT16> <*Graph Type* : INT8>

Send a graph on the background channel. Transfers must be stopped for this command to work.

Bg <*Product ID* : INT16> <*Sensor ID* : INT16> <*Graph Type* : INT8>

Send graph on the background channel. Transfers must be stopped for this command to work.

BH

Send performance info for products and sensors.

BI <*Image ID*: INT32> <*Win.P1.X* : INT16> <*Win.P1.Y* : INT16> <*Win.P2.X* : INT16> <*Win.P2.Y* : INT16>

Send image on the background channel. An *Image ID* == -1 means transfer learn image.

Bi <*Win.P1.X* : INT16> <*Win.P1.Y* : INT16> <*Win.P2.X* : INT16> <Win.P2.Y : INT16>

Send the newest image on the background channel.

BK

Kill the background thread.

BM <*autodisable* : INT8> <*mode* : INT8> <<*priority low* : INT8>
<*priority high* : INT8> <*Image ID* : INT32>>

Set background transfer mode. If *Autodisable* == TRUE, thread will auto-disable after each transfer. *image ID* is ignored.

BP <*Product ID* : INT16>

Select *Product ID* as the selected background product for image marking/evaluation and Image/CCD Head selection.

BS <*Destination Channel* : INT8>

Start a thread for background transfers. Background Channel is associated with this terminal, and is shutdown when this terminal/channel is shutdown.
Response:
<*New Channel Number* : INT8>

BT <*Image ID* : INT32>

Send image and sensor graphs on the background channel. An *Image ID* == -1 means transfer **learn** image.

Bt

Send the newest image and sensor graphs on the background channel.

Background Packet Format and Sequence

Header Packet:
<*(packet type) 0* : UINT8> <*Image ID* : INT32> <*Time Acq* : INT32>
<*Product ID* : INT16>

Graph Header Packet:
<*(packet type)* 32 : UINT8> <*Image ID* : INT32> <*Product ID* : INT16> <*Sensor ID* : INT16> <*Graph Type* : INT8> <*Data Format* : INT8> <*Data Length* : INT32> <*Graph Specific Data*>

Graph Data Packet:
<*(packet type)* 34 : UINT8> <*Graph Data*>

Graph End Packet:
<*(packet type)* 33 : UINT8> <*Image ID* : INT32> <*Product ID* : INT16> <*Sensor ID* : INT16> <*Graph Type* : INT8>
...

End Packet:

```
<(packet type) 1 : UINT8> <Image ID : INT32>
```

Header Packet:

```
<(packet type) 2 : UINT8>
```

Perf/Hist. Packet:

```
<(packet type) 48 : UINT8> <Perf/Hist Type : UINT8>
   <Perf/Hist Data>
...
```

End Packet:

```
<(packet type) 3 : UINT8>
```

Graph Type Info:

Graph Type	Description	Graph Specific Data
0	Pixel Intensity Graph	<Threshold : INT32> <Contrast : INT32> <Min Intensity : INT32> <Max Intensity : INT32>
1	Intensity Histogram Graph	<Threshold : INT32>
2	Pixel Gradient graph	
3	Blob Info graph	<Num Blobs : INT32> <Num Blobs Selected : INT32> <Valid Features : UINT16> <Valid Checks : UINT16>

Data Format:

Data Format	Description
3	256 level greyscale data

Performance Packet Info:

Entry Type	Entry Data
0	Product Pass/Warn/Fail counts
1	Sensor PassWarn/Fail counts
2	same as Sensor Specific ‘Output’ section of ‘SQ’ command
4	same as ‘Ii’ command
5	same as ‘YQ’ command
6	Product Pass/Fail status
7	Sensor Pass/Fail status
8	I/O history data
9	<input polarity: UINT16> <inputs: UINT16> <raw inputs: UINT16> <output polarity: UINT16> <outputs: UINT16>

Communications Commands

Cb
CB <*New Baud Rate* : INT32>
change baud rate
Response:
<\r \n?: string>

Ce
CE <+|- : char>
enable or disable echo

Cq
CQ
query current communication configuration
Response:
<*Current Baud Rate* : INT32> <*Flow Control* : INT8>
Flow Control:
0 = NONE
1 = XON/XOFF
2 = RTS/CTS

Cr
CR <*PortNumber* : INT8>
Reset port statistics.

Cs
CS <*PortNumber* : INT8>
Query port statistics.
Response:

Cx
CX <*Data*>
Echo data back (for checking communication errors)

Digital I/O Commands

(output bits 15 .. 0 : Busy, U12(Warn), n/a, Pass, Fail, U11(ResCon), U10(DigSelPass), U9(DigSelFail), U8, U7, U6, U5, U4(MUX4), U3(MUX3), U2(MUX2), U1(MUX1))
(input bits 7 .. 0 : I6, I5, I4, I3, I2, I1, Sync, Select)

DA<*I/O Isolation* : INT8>
Set the Input/Output Isolation type used for the Series 800 unit
I/O Isolation:
-1 : Unknown isolation,
0 : DC I/O,
1: AC I/O (DC Sync),
2: AC I/O (AC Sync),
3 : TTL I/O

DD <*Input bit* : UINT8> <*Dactive* : UINT16> <*Dinactive* : UINT16>
Set Digital Input Debounce time (in ms)

DI <*NormPolarity Bits* : UINT32> <*Inv Polarity Bits* : UINT32>
Sets input *NormPolarity Bits* to normal polarity and *InvPolarity Bits* to inverted polarity.

DM <*Virtual Input bit* : UINT8> <*Mask*: UINT32>
Sets the virtual input mask for the specified bit.

DN <*Virtual Output bit* : UINT8> <*Mask High*: UINT32> <*Mask Low*: UINT32>
Sets the virtual output mask for the specified bit.

DO <*NormPolarity Bits* : UINT32> <*Inv Polarity Bits* : UINT32>
Sets output *NormPolarity Bits* to normal polarity and *InvPolarity Bits* to inverted polarity.

DQ
Query general digital I/O info
Response:
<*Number physical inputs* : UINT8> <*Number physical outputs* : UINT8> <*I/O Isolation* : INT8> <*physical input polarity* : UINT32> <*physical input state* : UINT32> <*physical raw input* : UINT32> <*virtual input state* : UINT32> <*physical output polarity* : UINT32> <*physical output state* : UINT32> <*virtual output state high* : UINT32> <*virtual output state low* : UINT32>
<*Bit 0 Active debounce time* : UINT16> <*Bit 0 Inactive debounce time* : UINT16> ... <*Bit N Active debounce time* : UINT16> <*Bit N Inactive debounce time* : UINT16>
<*virtual input mask 0* : UINT32> … <*virtual input mask 15* : UINT32>

<*virtual output mask high 0* : UINT32> <*virtual output mask low 0* : UINT32> … <*virtual output mask high 7* : UINT32> <*virtual output mask low 7* : UINT32>
<*virtual output mask high 8* : UINT32> <*virtual output mask low 8* : UINT32> … <*virtual output mask high 15* : UINT32> <*virtual output mask low 15* : UINT32>

Dq

Query digital I/O states

Response:

<*Number physical inputs* : UINT8> <*Number physical outputs* : UINT8> <*I/O Isolation* : INT8> <*physical input polarity* : UINT32> <*physical input state* : UINT32> <*physical raw input* : UINT32> <*virtual input state* : UINT32> <*physical output polarity* : UINT32> <*physical output state* : UINT32> <*virtual output state high* : UINT32> <*virtual output state low* : UINT32>

DS <*Active Bits* : UINT32> <*Inactive Bits* : UINT32>

Set physical digital output bits *Active Bits* to the active state and *Inactive Bits* to the inactive state.

Image Management Commands

I

Query version description
Response:
<*version* : string>

IC

Query CCD info
Response:
<*select ID* : INT16> <*num installed* : UINT8> <*num acquisition/pipeline buffers* : UINT8> <*num buffers*: UINT8>
<ID_1 : INT16> <*#XPixels* : INT16> <*#YPixels* : INT16> <*PixelWidth* : INT16> <*PixelHeight* : INT16> <*Name* : char 50>
...
<ID_n : INT16> <*#XPixels* : INT16> <*#YPixels* : INT16> <*PixelWidth* : INT16> <*PixelHeight* : INT16> <*Name* : char 50>

IC <*ID* : INT16>

Change CCD (0=810, 1=820). Enter an ID of -1 for autodetection of CCD type. Response with a selected ID of -1 indicates that no CCD was detected.
Response:
<*select ID* : INT16> <*num installed* : UINT8><*num system buffers* : UINT8><*num buffers*: INT8>

Ic

Calibrate CCD
Response:
<*status* : INT32>
Calibration status. (0=success, -1=did not converge, -2=gain maximized)

ID <*DeleteStrategy* : UINT8>

Set delete strategy
0 : delete newest,
1 : delete oldest
2 : delete newest by priority
3 : delete oldest by priority

Ii

Query Inspection Statistics
Response :
<*Total Inspections* : UINT32> <*Missed Inspections* : UINT32> <*Acquisition Conflicts* : UINT32> <*Timeout Failures* : UINT32> <*Min Time in Capture Queue* : UINT32> <*Min Inspection Time* : UINT32> <*Mean Inspection Time* :

UINT32> <*Max Inspection Time* : UINT32> <*Min Time between Acquisitions* : UINT32> <*Mean Time between Acquisitions* : UINT32>

Ii <*Inspection Stat* : Char>

Query Specific Inspection Statistics

Response: (Inspection Stat)

a: <*Min Time Between Acquisitions* : UINT32>
b: <*Mean Time Between Acquisitions* : UINT32>
c: <*Acquisition Conflicts* : UINT32>
f: <*Timeout Failures* : UINT32>
i: <*Min Inspection Time* : UINT32>
m: <*Missed Inspections* : UINT32>
n: <*Mean Inspection Time* : UINT32>
q: <*Min Time in Capture Queue* : UINT32>
t: <*Total Inspections* : UINT32>
x: <*Max Inspection Time* : UINT32>

Iiz

Reset all Inspection Statistics

Iiz <*Inspection Stat* : Char>

Reset Specific Inspection Statistic

Response: (Inspection Stat)

a: Min Time Between Acquisitions (reset to 0x7FFFFFFF)
b: Mean Time Between Acquisitins
c: Acquisition Conflicts
f: Timeout Failures
i: Min Inspection Time (reset to 0x7FFFFFFF)
m: Missed Inspections
n: Mean Inspection Time
q: Min Time in Capture Que (reset to 0x7FFFFFFF)
t: Total Inspections
x: Max Inspection Time

Il <*PacketType* : UINT8> <*Specific Packet Data*>

Response: (Header Only)

<*Assigned ImageID* : INT32 >

Specific Packet Data

Header Packet:

<Packet Type = 0x10> <image ID : INT32> <x1 : INT16> <y1 : INT16> <x2 : INT16> <y2 : INT16> <compression format : UINT8> <RowSkip : UINT8> <ColSkip : UINT8> <RowOffset : UINT8> <ColOffset : UINT8>

Uncompressed Data Packet

<Packet Type = 0x12> <Compression Format = 0> <image ID : INT32> <image data : UINT8> ..... <image data : UINT8>

End Of Image Sending Packet
<Packet Type = 0x11>

IL

Lock the Saved Image Queue. Save Image Queue is automatically unlocked if terminal terminates.

IP <# system buffers : UINT8>

Set system image buffers

IQ

Query image information

Response:

<delete strategy : UINT8>
<ID_1 : INT32> <result : INT8> <product : INT16> <image handle : INT16> <First Failing/Warning Sensor ID : INT16> <First Failing /Warning Sensor Result Code: INT8>
...
<ID_n : INT32> <result : INT8> <product : INT16> <image handle : INT16> <First Failing/Warning Sensor ID : INT16> <First Failing /Warning Sensor Result Code: INT8>

IQ<image ID : INT32>

Query specific image information

Response:

<Result : INT8> <Capture Queue Time Stamp : UINT32> <Acquisition Time Stamp : UINT32> <Inspection Time Stamp : UINT32> <Saved Time Stamp : UINT32><ProductID : INT16> <Lock Count : INT16>
<Exposure Time : UINT32> <Antiblooming Enabled : UINT8> <IlluminationEnabled : UINT8> <X1 : INT16> <Y1 : INT16> <X2 : INT16> <Y2 : INT16> <AfterSyncDelay : UINT32> <AntiBloomingLevel : INT16> <Gain : INT16>

IR<image ID : INT32>

Release the selected image from the Saved Queue.

IS <image ID : INT32> <x1 : INT16> <y1 : INT16> <x2: INT16> <y2: INT16> <compression format : UINT8> <rowskip : UINT8> <colskip : UINT8> <rowoffset : INT8> <coloffset : UINT8>

send image (*last image sent remains locked)

Response: see separate sheet given earlier

compression format:

0 = Uncompressed
1 = Huffman 2D
2 = TwoBit Residual

Is (<low : INT16> <high : INT16>)

send newest image within *low* .. *high* result code priority. (*last image successfully sent remains locked as current learn image)

IU

Unlock the Saved Image Queue

Product Management Commands

Note: If a *Product ID* = -2 the current default product is used

PA

PA <*name* : char 50>

Add new product and auto name or set name to *name*. Product is created active.
Response:

<*ID* : INT16> <*name* : char 50>

PD <*Product ID* : INT16>

Delete product and its sensors from RAM and Flash

PFD<*Product ID* : INT16>

Delete product from flash

PFS

Save all products to flash

PFS< *Product ID* : INT16>

Save product to flash

PI < *Product ID*: INT16>

Select inspection product

PL < *Sensor ID* : INT16>

Learn all sensors (or optional selected sensor) for the selected product on the currently locked learn-image.

PM < *Product ID*: INT16>

Query general Data Link information.
Response:

< *Product ID*: INT16> <*Transmit Rate* : INT8> <*Fail Character* : CHAR> < *Placeholder Type* : INT8>

Placeholder Type –	0	Spaces
	1	Zeros

PM <*Product ID*: INT16> <*Data Link ID* : INT8>

Query information on a specific Data Link string
Response:

<*Product ID*: INT16> <*Data Link ID* : INT8> <*Function* : INT8> <*Priority Flag* : INT8> <*Data Link String* : CHAR[200]>

Function –	1	Never
	2	Always
	3	AND
	4	OR

PN < *Product ID*: INT16> <*name* : char50>

Rename product

PP < *Product ID*: INT16>

Set product parameters (-1 sets default parameters):

A <flag : UINT4>	active product flag
B <flag : UINT4>	antiblooming enable flag
b <ABL Gain : INT16>	antiblooming level
D <time : UINT32>	delay after synch signal (microseconds)
E <time : INT32>	exposure time (microseconds, 0 .. 60 sec.)
G <WL Gain : INT16>	gain (0-245)
H <Selected Head : INT8>	selected head (0 .. 15)
I <flag : UINT4>	illumination enable flag
M <Trans Rate: INT8> <Fail Char : CHAR> <Terminate Type : INT8>	General Data Link parameters
m <Data Link ID : INT8> <Function : INT8> <Priority : INT8> <Data Link String : CHAR[200]>	Parameters for specific Data Link string (see note above for Function values)
O <FailTime : INT16> <OutputPulseWidth : INT16> <PostOutputDelay : INT16> <ResConflictPulseWidth : INT16>	
P <all pass bits : UINT16>	set bits that should go active if all sensors (with appropriate output bits set) pass
S <digital signature : INT16>	digital signature code
W<x1 : INT16> <y1 : INT16> <x2 : INT 16> <y2 : INT16>	window for image (will be clipped to actuall ccd dimensions during acquisition).
Response: <x1 : INT16> <y1 : INT16> <x2 : INT 16> <y2 : INT16>	actual window after clipping

Pp < *Product ID* : INT16>

Select power on product

PQ

Query general product information

Response:

<*selected ID* : INT16> <*inspection ID* : INT16>
<*Power On Product ID* : INT16>
<ID_1 : INT16> <*name* : char 50>
...
<Id_n : INT16> <*name* : char 50>

PQ <*-1* : INT16>

Query default product IAParams information

Response:

<*exposure time* : INT32> <*antiblooming* : UINT4>
<*illumination* : UINT4> <*delay after sync* : INT32>
<*antiblooming gain* : INT16> <*gain* : INT16>
<*window x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16> <*Selected Head* : INT8>

PQ < *Product ID* : INT16>

Query a product for information

Response:

<*ID* : INT16> <*name* : char 50>
<*lock count* : INT8> <*active* : UINT4> <*flash* : UINT4> <*dirty* : UINT4> <*signature* : INT16> <*fail time* : INT16>
<*OutputPulseWidth* : INT16> <*PostOutputDelay* : INT16>
<*ResourceConflictPulseWidth* : INT16>
<*AllPassBits* : UINT16>
<*exposure* time : INT32> <*antiblooming* : UINT4>
<*illumination* : UINT4> <*delay after sync* : INT32>
<*antiblooming gain* : INT16> <*gain* : INT16>
<*window x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16> <*Selected Head* : INT8>
<*pass count* : UINT32> <*alarm count* : UINT32> <*fail count* : UINT32>
<*description*>

Pq < *Product ID* : INT16>

Query a product for specific information

Response:

<*pass count* : UINT32> <*alarm count* : UINT32> <*fail count* : UINT32> <*Latest Result*: INT8>

PR < *Product ID* : INT16>

Reset product *ID* results.

PS < *Product ID* : INT16>

Select default product (for sensors)

PSI < *Digital ID* : INT16>

Select default product by digital ID (for sensors)

PSN <*Name* : char [50]>

Select default product by name (for sensors)

PT < *Product ID* : INT16> <*description* : char 200>

Set descriptive text of product

Sensor Management Commands

SA <*type* : INT16>
SA <*type* : INT16> <*name* : char 50>
Add new sensor of *type* and auto name or set name to *name*.
Response:
<*ID* : INT16><*name* : char 50>

SD <*ID* : INT16>
Delete sensor

SI <*ID* : INT16> <new *ID* : INT16>
Change sensor ID

SM
List installed sensor modules
Response:
<*number installed modules* : INT16>
<ID_1 : INT16> <*version* : INT16> <*name* : char 50>
...
<ID_n : INT16> <*version* : INT16> <*name* : char 50>

SN <*ID* : INT16> <*name* : char 50>
Rename sensor

SO <*ID* : INT16>
Query sensor output results.

SP <*ID* : INT16> <... *sensor specific* ...>
Set sensor parameters (See later sections for the format specific to each type of sensor)

SPA <*ID* : INT16> <*ActiveFlag* : UINT4>
Set sensor active flag.

SPO <*ID* : INT16> <*Output Pass Bits* : UINT16> <*Output Warn Bits* : UINT16> <*Output Fail Bits* : UINT16> <*Data Link Pass Bits* : UINT16> <*Data Link Warn Bits* : UINT16> <*Data Link Fail Bits* : UINT16>
Set output and Data Link bit pattern for that sensor.

SQ
Query general sensor information
Response:
<*number of sensors* : UINT16>
<ID_1 : INT16> <*type* : INT16> <*name* : char 50>
...
<ID_n : INT16> <*type* : INT16> <*name* : char 50>

SQ <*ID* : INT16>

Query a sensor for information

Response:

<*ID* : INT16> <*ActiveFlag* : UINT4> <*name* : char 50>
<*pass bits* : UINT16> < *warn bits* : UINT16> <*fail bits* :
UINT16> <*Data Link pass bits* : UINT16> <*Data Link warn
bits* : UINT16> <*Data Link fail bits* : UINT16>
< ... *sensor specific information (see list at end)* ... >
<*description* : char 100>

Sq <*ID* : INT16>

Query a sensor for information

Response:

<*ID* : INT16> <*ActiveFlag* : UINT4> <*name* : char 50>
<*LatestResult* : INT8> <*PassCount* : UINT32> <*WarnCount* :
UINT32> <*FailCount* : UINT32> <*SpecificLastResult* : INT8>
(see Sensor Input/Output Scheme for "Specific LastResult" codes)

SR <*ID* : INT16>

Reset sensor *ID* results.

ST <*ID* : INT16> <*description* : char 100>

Set descriptive text of sensor

Terminal Management Commands

T+

Change to selected terminal driver

T-

Change to terminal driver 0 (1st installed)

Tb

Query default terminal drivers and port information

Response:

```
<Port Number : UINT8> < Terminal ID : INT16> < Baud Rate :
   INT32> <Data Bits : INT8> <Stop Bits : INT8> <Parity :
   INT8> <Flow Control : INT8> <Echo : INT8>
                ...
<Port Number : UINT8> < Terminal ID : INT16> < Baud Rate :
   INT32> <Data Bits : INT8> <Stop Bits : INT8> <Parity :
   INT8> <Flow Control : INT8> <Echo : INT8>
```

Note:

Stop Bits –	0	1 stop bit
	1	1.5 stop bits
	2	2 stop bits
Parity –	0	No Parity
	1	Odd Parity
	2	Even Parity
Flow Control –	0	No flow control
	1	XON/XOFF flow control
	2	RTSCTS flow control

Tb <*Port Number* : UINT8> < *Terminal ID* : INT16> < *Baud Rate* : INT32> <*Data Bits* : INT8> <*Stop Bits* : INT8> <*Parity* : INT8> <*Flow Control* : INT8> <*Echo* : INT8>

Change default terminal drivers and port information

TB

Query port default terminal and baud rate.

Response:

```
<Port Number : UINT8><Terminal ID : INT16>
   <Baud Rate : INT32>
   ...
<Port Number : UINT8> <Terminal ID : INT16>
   <Baud Rate : INT32>
```

TB<*Port Number* : UINT8> <*Terminal ID* : INT16> <*Baud Rate* : INT32>

Change default Terminal and Baud rate for *Port Number.*

Td<*DriverID* : INT8>
TD<*DriverID* : INT8>

Change serial driver(0:TERM, 1:DVTSP)

Tn<*Destination Channel* : INT8>
TN<*Destination Channel* : INT8>

Create new channel using DVTSP and terminal driver 0 (1st installed).

Response:

<*New Channel Number* : INT8>

Tq

Return the current terminal driver

Response:

<*Selected Terminal ID* : INT16> <*Terminal Name* : string>

TQ

Display available terminal drivers

Response:

<$TerminalID_1$: INT16> <$Terminal\ Name_1$: string>

…

<$TerminalID_n$: INT16> <$Terminal\ Name_n$: string>

Tr
TR

Reset to default serial driver (TERM) and selected terminal driver

Ts<*DriverID* : INT16>
TS<*DriverID* : INT16>

Select terminal driver

Tt
TT

Terminate terminal

System Management Commands

Y<+|- : char>
(+)Enable / (-)disable image acquisition/inspection process.

YD
Query Digital Selection Mode

YD<*dsdelay* : INT32char>
If *dsdelay* < 0, Digital Selection is disabled. If *dsdelay* >= 0, Digital Selection is enabled, and the digital ID will be read *dsdelay* milliseconds after the Digital Select Line is activated.

YF
Query if Firmware is saved in flash.
Response:
<*InFlash* : UINT8>

YI
Trigger an inspection if the system is in external trigger mode.

YM
Query Marking Mode

YM<*Marking Mode* : INT16>
Set image marking mode
Marking Mode >= 0 : mark specified sensor.
Marking Mode == -1 : no marking
Marking Mode == -2 : mark active sensors
Marking Mode == -3 : mark inactive sensors
Marking Mode == -4 : mark all sensors

YN
Get IVU Name

YN<*IVUName* : char50>
Set IVU Name

YP<*-1|0|value* : INT32>
Set image acquisition sync source.
-1 : acquire image after old image is processed.
0 : acquire new image immediately after old image.
value : acquire image every *value* milliseconds.

Yp<+|- : char>
Set Sync source to (+)Internal or (-)External.

YQ

Query system status

Response:

<+|- : char> <+|- : char> <+|- : char> <*period* : INT32> <*cpu load* : UINT8> <*Unused Flash* : INT32> <*System Clock* : UINT32> < *CCD ID* : INT16 > <*InspectionProduct ID* : INT16>

(+)Running / (-)not running inspections.
(+)Internal / (-)external sync.
Digital Selection (+)enabled / (-)disabled.
Sync *period* and *CPU load* (0 - 100)% over the previous 1024 millisecond time interval.

Yq

Query system type

Response:

<*CPU Type* : UINT8> <*CPU Clock* : UINT16>
<*FPU Present* : UINT8> <*Installed RAM* : UINT32>

YR3742

Reset IVU and 'reboot' to Diagnostics. '3742' is an ASCII string.

YS1bdc

Save application to flash. Flash is reformatted and all Flash Parameters and Flash Heap memory is erased. '1bdc' is an ascii string. The FLASH_FORMAT parameter is set to 0x11081995. (8/11/95)

YV

Query firmware version.

Response:

<*Flags* : UINT8> <*Major* : UINT8> <*Minor* : UINT8>
<*Refresh* : UINT8> <*Build* : UINT16>
<*build date* : UINT32>

Flags : 0x01 Internal Beta Release. Build date is ddmmyyyy in hex format.

Debug/Test Commands

Yd

Set current channel as the debug output channel.

Yr<*RecordMode* : UINT8>

If (*RecordMode* & 0x01) then send newer image (not newest). If (*RecordMode* & 0x02) then mark image with its inspection product, not the selected product.

Sensor Input/Output Scheme

Type	High Nibble	Valid Values	Examples
Position	0	x,y location and rotation vector	x,y location & rotation, equation of line.
Pixel Value	1	0-255	intensities contrast
Quantity	2	positive number	# edges # features # pixels
Percentage	3	0-100	% below threshold
Boolean	4	0-1	go/no-go, presence
Statistical Numbers	5		
SubPixel Position	6		equation of line or circle
Real Number	7		angle, roundness
String	8		Bar code string
Vectors	9		Blob feature vector
Integer Lists	10		Blob areas
SubPixel Position Lists	11		Blob positions
Real Number Lists	12		Blob angles

Specific Input/Output Numbers

Output	*High Nibble*	*Low Nibble*	*Decimal Number*	*Structure Values*	
				ID	*Param*
Position	0	0	0	0	X
				1	Y
				2	Theta X
				3	Theta Y
Minimum Intensity	1	0	16		
Maximum Intensity	1	1	17		
Mean Intensity	1	2	18		
Median Intensity	1	3	19		
Contrast	1	4	20		
Threshold	1	5	21		
# of Edges	2	0	32		
# of Features	2	1	33		
# of Pixels	2	2	34		
# of Blobs	2	3	35		
% below threshold	3	0	48		
SubPixel Point	6	0	96	0	X
				1	Y

SubPixel Line	6	1	97	0	A
				1	B
				2	Xo
				3	Yo
SubPixel Circle	6	2	98	0	X
				1	Y
				2	Radius
Distance/Radius	7	0	112		
Angle	7	1	113		
Straightness	7	2	114		
Roundness	7	3	115		
Scale Factor	7	4	116		
Bar Code String	8	0	128		
Blob Feature Vector	9	0	144		
Blob 1 Area	10	0	256		
…	…	…	…	…	
Blob 8 Area	10	7	263		
Blob 1 Subpix Pos	11	0	272		
…	…	…	…	…	
Blob 8 Subpix Pos	11	7	279		
Blob 1 Angle	12	0	288		
…	…	…	…	…	
Blob 8 Angle	12	7	295		

Specific SoftSensor Result codes

- Negative numbers are failure codes.
- Positive numbers are warning codes.
- Zero indicates the sensor passed the inspection.

Codes for pass/warn violations are from abs(1 to 63). General failure codes are from abs(64 and up)

Failure Code	All Sensors
-64	Unable to allocate memory
-65	SoftSensor out of bounds
-66	Bad position reference
-67	Bad low intensity reference
-68	Bad High intensity reference
-69	Bad Feature vector reference

Failure Code	Warning Code	EdgeCount SoftSensors
-1	1	min contrast not met ((failing) value < min # edges)
-2	2	value < minimum number of edges
-3	3	value > maximum number of edges
-4	4	value < maximum passing percent (scanning sensor)

Failure Code	Warning Code	FeatureCount SoftSensors
-1	1	min contrast not met ((failing) value < min # features)
-2	2	value < minimum number of features
-3	3	value > maximum number of features
-4	4	value < maximum passing percent (scanning sensor)

Failure Code	Warning Code	Intensity SoftSensors
-1	1	value < minimum contrast
-2	2	value > maximum contrast
-3	3	value < minimum percent of pixels below threshold
-4	4	value > maximum percent of pixels below threshold

Failure Code	Warning Code	Positioning (Translational/Rotational) SoftSensors
-1	1	value < minimum contrast
-2	2	value < minimum percent offset1
-3	3	value > maximum percent offset1
-4	4	value < minimum percent offset2
-5	5	value > maximum percent offset2
-6	6	bad line fit (area edge)
-7		no intersection (area edge)
-8		no edge found (lines, arcs)

Failure Code	Warning Code	Measurement SoftSensors
-1	1	value < minimum contrast
-2	2	value < minimum distance
-3	3	value > maximum distance
-4	4	value < minimum radius/angle
-5	5	value > maximum radius/angle
-6	6	not enough edges found
-7		no intersection (area edge)
-8		straightness/roundness

Failure Code	Warning Code	Math SoftSensors
-1	1	minimum distance/angle
-2	2	maximum distance/angle

Failure Code	Warning Code	DataMatrix SoftSensors
-1	1	String does not match pass criteria

Failure Code	Warning Code	Blob Generation SoftSensors
-1	1	Minimum contrast
-2	2	Minimum blobs
-3	3	Maximum blobs

Failure Code	Warning Code	Blob Selection SoftSensors
-1	1	Minimum blobs
-2	2	Maximum blobs

Failure Code	Warning Code	Scripting SoftSensors
-1	1	Result set to FAIL or WARN
-2	2	Processing error
<-2	>2	Result set to unknown fail or warn value

Edge Counting Sensors (Module ID: 1)

Command Format For Setting EdgeCounting Sensor Parameters

SP<*sensor ID* : INT16> <*parameter ID* : UINT8> <*data*>

<*parameter ID*>:

0 Set sensor type and coordinates
(set bit 7 of *type* for feature counting sensor)

<*data*> ⇒ <*type* : UINT8> <*coordinate data*>

<*type*>:

1 PolyLine

<*coordinate data*> ⇒ <*num points* : INT16> <*x1* : INT16> <*y1* : INT16> ... <x_n : INT16> <y_n : INT16>

2 Elliptical arc

<*coordinate data*> ⇒ <*xc* : INT16> <*yc* : INT16> <*xr1* : INT16> <*yr1* : INT16> <*xr2* : INT16> <*yr2* : INT16> <<*xa1* : INT16> <*ya1* : INT16> <*xa2* : INT16> <*ya2* : INT16>>

<<...>> are optional - for open arc.

1 Set sensor threshold

<*data*> ⇒ <*type* : UINT4> <*value* : INT16> <*data*>

<*type*>:

0 Fixed greyscale value

1 Percentage of contrast along path

2 Percentage of reference inputs

<<*data*> ⇒ <*LowRef SensorID* : INT16> <*LowRef OutputID* : INT16> <*HighRef SensorID* : INT16> <*HighRef OutputID* : INT16>>

<<...>> - optional.

2 Set feature parameters

<*data*> ⇒ <*edge type* : UINT4><<*min light size* : UINT16> <*max light size* : UINT16> <*min dark size* : UINT16> <*max dark size* : UINT16> <*Require Alternating Features* : UINT8> <*Enable End Features* : UINT8>>

<<...>> are for feature counting.

3 Set scanning parameter

<*data*> ⇒ <*x vector component* : INT16> <*y vector component*: INT16> <*main vector number of steps*: INT16> <*main vector scan density* : INT16> <*perpendicular vector number of steps* : INT16> <*perpendicular scan density* : INT16> <*output type* : INT 16>

Comments:

- vector components are for the main direction vector and are entered as a unit vector scaled by 16384 (2^{14})
- the number of steps are in the positive x direction for the main vector and in the y direction for the perpendicular vector. Then the coordinates are rotated to the vector direction entered above. However, the number of steps are taken in both the positive and negative directions.

- The scan density is similar to the scan density for area edge scanning and is entered as a number from 0 to 10000.
- Output Type:
  0 = check passing percent
  1 = check minimum number of edges versus the constraints
  2 = check maximum number of edges versus the constraints

`8` Set sensor warning levels

`<data> ⇒ <min contrast : INT16> <min edges : INT16> <max edges : INT16> << pass conditions - same as first 3 params>> < min percent passing : INT16>`

`9` Set sensor passing levels

`<data> ⇒ < same <data> as #8> << warn conditions - same <data> as #8>>`

<< ... >> - optional

`10` Set input reference

`<data> ⇒ <input ID : INT16> <reference sensor ID : INT16> <reference output ID : INT16>`

`11` Set learned position

<*data*> ⇒ <*x* : INT32> <*y* : INT32> <*theta* : FIXED32, FIXED32>

EdgeCounting Sensor Query Response

<*type/shape* : UINT8> (bit 7 set if feature counting)

- 0 No shape defined
- 1 PolyLine
  - <*num points*>
  - <x_1><y_1> ... <x_{10}><y_{10}>
  - ...
  - <x_{n-9}><y_{n-9}> ... <x_n><y_n>
- 2 Ellipse
  - *<cx> <cy> <x_1> <y_1> <x2> <y2> <<xa1><ya1><xa2><ya2>>*

<*threshold type* : UINT4> <*value* : UINT8>

- 0 Fixed greyscale value
- 1 Percent of contrast along path
- 2 Percent of reference input contrast

<*edge type* : UINT4> <<*min light size* : UINT16> <*max light size* : UINT16> <*min dark size* : UINT16> <*max dark size* : UINT16> <*Require Alternating Features* : UINT8> <*Force End Transitions* : UINT8>>

edge type:

- 0 Either
- 1 Positive edge / bright fiducial
- 2 Negative edge / dark fiducial

scanning parameter: <*x vector component* : INT16> <*y vector component*: INT16> <*main vector number of steps*: INT16> <*main vector scan density* : INT16> <*perpendicular vector number of steps* : INT16> <*perpendicular scan density* : INT16> <*output type* : INT16>

output type:

- 0 Percent passing
- 1 Minimum number of edges
- 2 Maximum number of edges

warning levels: <*min contrast* : INT16> <*min edges* : INT16> <*max edges* : INT16><*min percent passing while scanning* : INT16>

pass levels: <*min contrast* : INT16> <*min edges* : INT16> <*max edges* : INT16><*min percent passing while scanning* : INT16>

Input References: <*position - sensor ID*:INT16> <*position - output ID*:INT16> <*low intensity - sensor ID*:INT16> <*output ID*:INT16> < *high intensity - sensor ID*: INT16> <*output ID* : INT16>

Outputs: <*min intensity* : INT32> <*max intensity* : INT32> <*contrast* : INT32> <*threshold* : INT32> <*num edges* : INT32> < *Percent Passing* : INT32>

For scanning, *num edges* corresponds to the min or maximum number of edges if the output type is min or max edges respectively. *num edges* is -1 if output type is percent passing.
For non-scanning, *Percent Passing* is -1.

Position during Learning: <*x* : INT32><*y*: INT32><*theta* : FIXED32,FIXED32>

Edge/Feature Count Sensor Referencing IDs	
Input (i.e. – Will reference other sensors by):	
Position	0
Low Intensity Reference	16
High Intensity Reference	17
Output (i.e. – Can be referenced by)	
Min Intensity	16
Max Intensity	17
Contrast	20
Threshold	21
Number Edges	32

Failure Code	**Warning Code**	**EdgeCount SoftSensors**
-1	1	min contrast not met ((failing) value < min # edges)
-2	2	value < minimum number of edges
-3	3	value > maximum number of edges
-4	4	value < maximum passing percent (scanning sensor)

Failure Code	**Warning Code**	**FeatureCount SoftSensors**
-1	1	min contrast not met ((failing) value < min # features)
-2	2	value < minimum number of features
-3	3	value > maximum number of features
-4	4	value < maximum passing percent (scanning sensor)

Intensity Sensors (Module ID: 2)

Command Format For Setting Intensity Reference Sensor Parameters

{...} are for WebCheck Sensors

SP<*sensor ID* : INT16><*parameter ID* : UINT8><*data*>

<*parameter ID*>:

0 Set sensor type and coordinates (set bit 7 for WebCheck sensor)

<data> ⇒ <type : UINT8> <coord data>

<type>:

1 Poly-line

<*coord data*> ⇒ <*num points* : INT16> <*x1* : INT16> <*y1* : INT16> ... <x_n : INT16> <y_n : INT16>

2 Elliptical arc

<*coord data*> ⇒ <*xc* : INT16> <*yc* : INT16> <*xr1* : INT16> <*yr1* : INT16> <*xr2* : INT16> <*yr2* : INT16> <<*xa1* : INT16> <*ya1* : INT16> <*xa2* : INT16> <*ya2* : INT16>>

<<...>> are optional - for open arc.

3 Rectangle (area)

<*coord data*> ⇒ <*x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16>

5 Ellipse (area)

<*coord data*> ⇒ <*xc* : INT16> <*yc* : INT16> <*xr1* : INT16> <*yr1* : INT16> <*xr2* : INT16> <*yr2* : INT16>

9 PolyLine

<*coord data*> ⇒ <*num points* : INT16> <*x1* : INT16> <*y1* : INT16> ... <x_n : INT16> <y_n : INT16>

1 Set sensor threshold

<*data*> ⇒ <*type* : UINT4><*value* : INT16> <*data*>

<*type*>:

0 Fixed greyscale value

1 Percentage of contrast along path

2 Percentage of reference inputs

<<*data*> ⇒ <*LowRef SensorID* : INT16> <*LowRef OutputID* : INT16> <*HighRef SensorID* : INT16> <*HighRef OutputID* : INT16>>

<<...>> - optional.

2 Set sampling parameters

<*data*> ⇒ <*x sampling density* : INT16 (0-10000)> <*y sampling density* : INT16 (0-10000)>

3 Set WebCheck parameters

<*data*> ⇒ <*num of bins*(1-8 = 2, 4, 8, 16, 32, 64, 128, 256): UINT4> <*histogram scaling flag* (0-1): UINT4>

8 Set sensor warning levels

<*data*> ⇒ <*min contrast* : INT16> <*max contrast* : INT16> <*min % pixel below threshold* : INT16> <*max % pixel below threshold* : INT16> {<*max Chi Square Result* : INT32 (value scaled by 256) > <*min Chi % Result* : INT16 (% scale by 10000)> }

9 Set sensor passing levels

<*data*> ⇒ < *same <data> as #8*> << *warn conditions - same <data> as #8*>>

<< ... >> - optional

10 Set input reference

<*data*> ⇒ <*input ID* : INT16> <*reference sensor ID* : INT16> <*reference output ID* : INT16>

11 Set learned position

<*data*> ⇒ <*x* : INT16> <*y* : INT16> <*theta* : INT16, INT16>

12 Set learned histogram

<*data*> ⇒ <*Number Data Points* : INT16> <*Starting Histogram Location* : INT16> <*data* : UINT32> <...>

14 Set/clear WebCheck bit for sensor

<*data*> ⇒ <*set/clear web check sensor* : UINT4>

15 Outputs 16 learned histogram data points <input: starting histogram location>

<*data*> ⇒ <*data* : UINT32> <...>

Intensity Reference Sensor Query Response

<*type/shape* : UINT8> (bit 7 high for web check sensor)

0 No shape defined

1 PolyLine

<*number points*>

<x_1><y_1> ... <x_{10}><y_{10}>

...

<x_{n-9}><y_{n-9}> ... <x_n><y_n>

2 Elliptical arc

<*xc*><*yc*><*xr1*> <*yr1*><*xr2*><*yr2*><<*xa1*><*ya1*><*xa2*><*ya2*>>

3 Rectangle

<*x1*> <*y1*> <*x2*> <*y2*>

5 ellipse

<xc> < yc> <xr1> <yr1> <xr2> <yr2>

<*threshold type* : UINT4> <*value* : UINT8>

0 fixed greyscale value

1 percent of contrast along path

2 percent of reference input contrast

<*x_sampling density* : INT16 (0-10000)> <*y sampling density* : INT16 (0-10000)>

{(webcheck parameters): <*num of bins* : UINT4> <*histogram scaling* : UINT4>}

warning levels: <*min contrast* : INT16> <*max contrast* : INT16> <*min %pixels below threshold* : INT16> <*max %pixel below threshold* : INT16>{<*max ChiSquared Value* : INT32> <*min Chi % Value* : INT16>}

pass levels: <*min contrast* : INT16> <*max contrast* : INT16> <*min %pixel below threshold* : INT16> <*max %pixel below threshold* : INT16> {<*max ChiSquared Value* : INT32> <*min Chi % Value* : INT16>}

Input References: <*position - sensor ID*:INT16> <*position - output ID*:INT16> <*LowIntensity Reference - sensor ID*:INT16>.<*LowIntensity Reference - output ID*:INT16>.<*HighIntensity Reference - sensor output ID*:INT16> <*HighIntensity Reference - output* ID:INT16>

Outputs: <*min intensity* : INT32><*max intensity* : INT 32> <*MeanIntensity* : INT 32> <*MedianIntensity* : INT 32> <*contrast* : INT 32> <*threshold* : INT 32> <*pixel count* : INT 32>{<*Chi Squared Result* : INT32> (value scaled by 256) <*Chi % Result* : INT32> (% scaled by 10000) }

Position during Learning: <*x* : INT32><*y* : INT32><*theta* : FIXED32,FIXED32>

Intensity Sensor Referencing IDs	
Input (i.e. – Will reference other sensors by):	
Position	0
Low Intensity Reference	16
High Intensity Reference	17
Output (i.e. – Can be referenced by)	
Min Intensity	16
Max Intensity	17
Contrast	20
Threshold	21
% Below Threshold	48
Chi Squared	60

Failure Code	Warning Code	Intensity SoftSensors
-1	1	value < minimum contrast
-2	2	value > maximum contrast
-3	3	value < minimum percent of pixels below threshold
-4	4	value > maximum percent of pixels below threshold

Positioning Sensors (Module ID: 3)

Command Format For Setting Positioning Sensor Parameters

{...} are for area based positioning sensors. [...] are for area-edge positioning sensor.

SP <*sensor ID* : INT16><*parameter ID* : UINT8><data>
<*parameter ID*>

0 Set sensor type and coordinates
<data> ⇒ <type : UINT8><coord data>
<type>

1 straight line
<coord data> ⇒ <*x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16>

2 elliptical arc
<coord data> ⇒ <*xc* : INT16> <*yc* : INT16> <*xr1* : INT16> <*yr1* : INT16> <*xr2* : INT16> <*yr2* : INT16> <<*xa1* : INT16> <*ya1* : INT16> <*xa2* : INT16> <*ya2* : INT16>>
<<...>> are optional - for open arc.

3 rectangle (fiducial)
<*coord data*> ⇒ <*x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16>

4 area-edge sensor
<*coord data*> ⇒ <*x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16> <*x3* : INT16> <*y3* : INT16>

5 elliptical area
<*coord data*> ⇒ <*xc* : INT16> <*yc* : INT16> <*xr1* : INT16> <*yr1* : INT16> <*xr2* : INT16> <*yr2* : INT16>

1 set sensor threshold
<data> ⇒ <type : UINT4> <value : INT16> <data>
<type>

0 fixed greyscale value

1 percentage of contrast along path

2 percentage of reference inputs
<<data> ⇒ <*LowRef SensorID* : INT16> <*LowRef OutputID* : INT16> <*HighRef SensorID* : INT16> <*HighRef OutputID* : INT16>>
<<...>> - optional.

2 set feature parameters
<data> ⇒ <*edge/feature type* : UINT4> <*feature parameters*>

edge type:
0 either
1 positive edge / bright fiducial
2 negative edge / dark fiducial

scan sides [bit 3, 2, 1, 0] ⇒ [right, bottom, left, top]

<*feature parameters*> ⇒

straight line

<*edge index* : UINT16> <*Minimum Light Feature Size* : UINT16> <*Maximum Light Feature Size* : UINT16> <*Minimum Dark Feature Size* : UINT16> <*Maximum Dark Feature Size* : UINT16> <*Require Alternating Feature* : UINT4> <*Enable End Feature* : UINT4>

elliptical arc

<*edge index* : UINT16> <*Minimum Light Feature Size* : UINT16> <*Maximum Light Feature Size* : UINT16> <*Minimum Dark Feature Size* : UINT16> <*Maximum Dark Feature Size* : UINT16> <*Require Alternating Feature* : UINT4> <*Enable End Feature* : UINT4>

rectangle (fiducial)

<*scan sides* : UINT4>

area-edge sensor

<*edge index* : UINT16> <*Minimum Light Feature Size* : UINT16> <*Maximum Light Feature Size* : UINT16> <*Minimum Dark Feature Size* : UINT16> <*Maximum Dark Feature Size* : UINT16> <*Require Alternating Feature* : UINT4> <*Enable End Feature* : UINT4> <*scan density* : INT16>

elliptical area

<*scan sides* : UINT4>

8 set sensor warning levels

<data> ⇒ <*min contrast* : INT16> <*min offset* : INT16> <*max offset* : INT16> {<*min y offset 2* : INT16> <*max y offset* : INT16>} [<*min theta* : INT16> <*max theta* : INT16>]

9 set sensor passing levels

<data> ⇒ < same <data> as #8> << warn conditions - same <data> as #8>>

<< ... >> - optional

10 set input reference

<data> ⇒ <*input ID* : INT16><*reference sensor ID* : INT16> <*reference output ID* : INT16>

11 set learned position

<data> ⇒ <*x* : INT16> <*y* : INT16> <*theta* : INT16, INT16>

Positioning Sensor Query Response

<*type/shape* : UINT8>

0 no shape defined

1 straight line
<*x1*> <*y1*> <*x2*> <*y2*>

2 elliptical arc
<*cx*> <*cy*> <*x1*> <*y1*> <*x2*> <*y2*> <<*xa1*><*ya1*><*xa2*><*ya2*>>

3 rectangular area
<*x1*> <*y1*> <*x2*> <*y2*>

4 area-edge locator
<*x1*> <*y1*> <*x2*> <*y2*> <*x3*> <*y3*> <*x4*> <*y4*>

5 elliptical area
<*cx*> <*cy*> <*x1*> <*y1*> <*x2*> <*y2*>

<*threshold type* : UINT4> <*value* : UINT8>

0 fixed grayscale value

1 percent of contrast along path

2 percent of reference input contrast

<*edge type* : UINT4>…

straight line
<*edge index* : UINT16> <*Minimum Light Feature Size* : UINT16> <*Maximum Light Feature Size* : UINT16> <*Minimum Dark Feature Size* : UINT16> <*Maximum Dark Feature Size* : UINT16> <*Require Alternating Feature* : UINT4> <*Enable End Feature* : UINT4>

elliptical arc
<*edge index* : UINT16> <*Minimum Light Feature Size* : UINT16> <*Maximum Light Feature Size* : UINT16> <*Minimum Dark Feature Size* : UINT16> <*Maximum Dark Feature Size* : UINT16> <*Require Alternating Feature* : UINT4> <*Enable End Feature* : UINT4>

rectangle (fiducial)
<*scan sides* : UINT4>

area-edge sensor
<*edge index* : UINT16> <*Minimum Light Feature Size* : UINT16> <*Maximum Light Feature Size* : UINT16> <*Minimum Dark Feature Size* : UINT16> <*Maximum Dark Feature Size* : UINT16> <*Require Alternating Feature* : UINT4> <*Enable End Feature* : UINT4>
<*scan density* : INT16>

elliptical area
<*scan sides* : UINT4>

edge type:

0 either

1 positive edge / bright fiducial

2 negative edge / dark fiducial

scan sides [bit 3, 2, 1, 0] ⇒ [right, bottom, left, top]

warning levels:

```
<min contrast : INT16> <min offset : INT16> <max offset :
   INT16> {<min y offset : INT16> <max y offset : INT16>}
   [<min theta : INT16> <max theta : INT16>]
```

failure levels:

```
<min contrast : INT16> <min offset : INT16> <max offset :
   INT16> {<min y offset : INT16> <max y offset : INT16>}
   [<min theta : INT16> <max theta : INT16>]
```

Input References:

```
<position - sensor ID:INT16><position - output
   ID:INT16><low intensity - sensor ID:INT16> <low
   intensity - output ID:INT16>< high intensity - sensor
   ID: INT16> <high intensity - output ID : INT16>
```

Outputs:

```
<x : INT32> <y : INT32> <theta : INT32,INT32>
   <offset1 (%) : INT32> <offset2 (%) : INT32>
   <Min Intensity : INT32> <Max Intensity : INT32>
   <contrast : INT32> <Threshold : INT32>
```

Position during Learning:

```
<x : INT32> <y : INT32> <theta : INT32,INT32>
```

Positioning Sensor Referencing IDs	
Input (i.e. – Will reference other sensors by):	
Position	0
Low Intensity Reference	16
High Intensity Reference	17
Output (i.e. – Can be referenced by)	
Position	0
Min Intensity	16
Max Intensity	17
Contrast	20
Threshold	21

Failure Code	**Warning Code**	**Positioning (Translational/Rotational) SoftSensors**
-1	1	value < minimum contrast
-2	2	value < minimum percent offset1
-3	3	value > maximum percent offset1
-4	4	value < minimum percent offset2
-5	5	value > maximum percent offset2
-6	6	bad line fit (area edge)
-7		no intersection (area edge)
-8		no edge found (lines, arcs)

PrecisionMeasure Sensors (Module ID: 4)

Command Format For Setting PrecisionMeasure Sensor Parameters

SP <*sensor ID* : INT16><*parameter ID* : UINT8><*data*>

<parameter ID>

0 Set sensor type and coordinates
<*data*> ⇒ <*type* : UINT8><*coord data*>

<*type*>

1 straight line
<*coord data*> ⇒ <*x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16>

2 circular arc (toroid)
<*coord data*> ⇒ <*xc* : INT16> <*yc* : INT16> <*xr1* : INT16> <*xr2* : INT16>

4 area-edge sensor (line)
<*coord data*> ⇒ <*x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16> <*x3* : INT16> <*y3* : INT16>

12 area-edge sensor (circle) (defunct)
<*coord data*> ⇒ <*x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16> <*x3* : INT16> <*y3* : INT16>

17 straight line gauge
<*coord data*> ⇒ <*x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16>

20 area-edge gauge
<*coord data*> ⇒ <*x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16> <*x3* : INT16> <*y3* : INT16>

1 Set sensor threshold
<*data*> ⇒ <*type* : UINT4> <*value* : INT16> <<*data*>>

<*type*>

0 fixed gradient value
1 percentage of contrast along path
2 percentage of reference inputs

<*value*>

100 * gradient, or
100 * percent of reference/contrast

2 set edge type
<*data*> ⇒ <edge *type* : UINT4>

<edge *type*>

0 Positive or Negative gradient
1 Positive gradient
2 Negative gradient

3 Set scan/algorithm parameters

<*data*> ⇒ <*algorithm* : UINT4> <*scan density* : INT32> <*num scan regions* : UINT4> <*start angle 1 * 16384* : INT32> <*stop angle 1 * 16384* : INT32> … <*start angle n * 16384*: INT32> <*stop angle n * 16384* : INT32>

<*algorithm*>

for area-edge (line) and area-edge gauge sensors

0 Line Fit
1 Minimum Distance
2 Maximum Distance
3 Mean Distance
4 Median Distance

for circular arc and area-edge (circle) sensors

0 Circle Fit Radius
1 Minimum Radius
2 Maximum Radius

4 set scale factor parameters

<*data*> ⇒ <*type* : UINT4> <<*data*>>

<*type*>

0 no scaling.
1 reference scale.
<<*data*>> ⇒ <*sensorID* : INT16>

8 set sensor warning levels

<data> ⇒ <*min contrast* : INT16> <*min distance/radius* : FLOAT64> <*max distance/radius* : FLOAT64> <<*min theta* : FLOAT64> <*max theta* : FLOAT64> <*max straightness* : FLOAT64>> <<*max roundness* : FLOAT64>>

theta = (angle in degrees(-90..+90)) + 90

9 set sensor passing levels

<*data*> ⇒ < same <*data*> as #8>

10 set input reference

<*data*> ⇒ <*input ID* : INT16><*reference sensor ID* : INT16> <*reference output ID* : INT16>

11 set learned position

<*data*> ⇒ <*x* : INT16> <*y* : INT16> <*theta* : INT16, INT16>

theta = x,y orientation vector with mag(theta)=16384

PrecisionMeasure Sensor Query Response

<*type/shape* : UINT8><<*data*>>

0	no shape defined
1	straight line <*data*> = <x1> <y1> <x2> <y2>
2	circular toroid <*data*> = <cx> <cy> <xr1> <xr2>
4	area-edge locator (line) <*data*> = <x1> <y1> <x2> <y2> <x3> <y3> <x4> <y4>
12	area-edge locator (circle) <*data*> = <x1> <y1> <x2> <y2> <x3> <y3> <x4> <y4>
17	straight line gauge <*data*> = <x1> <y1> <x2> <y2>
20	area-edge gauge <*data*> = <x1> <y1> <x2> <y2> <x3> <y3> <x4> <y4>

<*threshold type* : UINT4> <*value* : INT16>

0	fixed gradient value
1	percent of contrast along path
2	percent of reference input contrast

<*edge type* : UINT4>

0	either
1	positive edge
2	negative edge

<scaling type : UINT4> <<sensorID : INT16>>

warning levels:

<*min contrast* : INT16> <*min distance/radius* : FLOAT64>
<*max distance/radius* : FLOAT64> <<*min theta* : FLOAT64>
<*max theta* : FLOAT64>> <<*max straightness* : FLOAT64>>
<<*max roundness* : FLOAT64>>

failure levels:

same format as warning levels

Input References:

<*position - sensor ID*:INT16><*position - output ID*:INT16><*low intensity - sensor ID*:INT16> <*low intensity - output ID*:INT16>< *high intensity - sensor ID*: INT16> <*high intensity - output ID* : INT16>

Outputs:

```
<x : INT32> <y : INT32> <theta : INT32,INT32>
   <<distance : FLOAT64>> <<radius : FLOAT64>>
   <<angle : FLOAT64>> <Min Intensity : INT32>
   <Max Intensity : INT32> <contrast : INT32>
   <Gradient Threshold : INT32> <<Straightness : FLOAT64>>
   <<Roundness : FLOAT64>> <scale factor : FLOAT64>
```

Position during Learning:

```
<x : INT32> <y : INT32> <theta : INT32,INT32>
```

Measurement Sensor Referencing IDs	
Input (i.e. – Will reference other sensors by):	
Position	0
Output (i.e. – Can be referenced by)	
Position	0
Min Intensity	16
Max Intensity	17
Contrast	20
Gradient Threshold	21
Subpixel Position	96
Subpixel Line	97
Subpixel Circle	98
Distance	112
Angle	113
Straightness	114
Roundness	115
Scale	116

Failure Code	**Warning Code**	**Measurement SoftSensors**
-1	1	value < minimum contrast
-2	2	value < minimum distance
-3	3	value > maximum distance
-4	4	value < minimum radius/angle
-5	5	value > maximum radius/angle
-6	6	not enough edges found
-7		no intersection (area edge)
-8		straightness/roundness

Math Sensors (Module ID: 5)

Command Format For Setting Math Sensor Parameters

SP<*sensor ID* : INT16><*parameter ID* : UINT8><data>

<*parameter ID*>

0 Set sensor type and coordinates
<*data*> ⇒ <*type* : UINT8><*reference data*>

<*type*>

1 distance calculator
<*reference data*> ⇒ <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

2 angle calculator
<*reference data*> ⇒ <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

3 intersection calculator
<*reference data*> ⇒ <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

4 midpoint calculator
<*reference data*> ⇒ <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

5 midline calculator
<*reference data*> ⇒ <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

6 line through two points calculator
<*reference data*> ⇒ <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

7 perpendicular line calculator
<*reference data*> ⇒ <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

8 scale factor calculator

9 coordinate transform
<*reference data*> ⇒ <*Image Pt0 X* : FLOAT64> < *Image Pt0*
Y: FLOAT64> < *Image Pt1 X* : FLOAT64> < *Image Pt1*
Y : FLOAT64> <*World Pt0 X* : FLOAT64> < *World Pt0*
Y: FLOAT64> < *World Pt1 X* : FLOAT64> < *World Pt1*
Y : FLOAT64> < *Negative Z* : UINT8> < *Angle*
Scale : FLOAT64> < *Angle Offset* : FLOAT64>
<*sensor ID1* : INT16> <*param ID1* : INT16>

3 set angle range (for angle math sensor)
<*data*> ⇒ <*type* : UINT4>

<type>

0 -180 .. +180 degrees
1 0 .. 360 degrees
2 -90 .. +90 degrees
3 0 .. 180 degrees

4 set scale factor params for distance SoftSensor
<*data*> ⇒ <*type* : UINT4> <<*data*>>

<type>

0 no scaling.

1 reference scale.
<<*data*>> ⇒ <*sensorID* : INT16>

5 set scale factor params for Scale SoftSensor.
<*data*> ⇒ <*type* : UINT4> <<*data*>> <*precision* : INT8>

<*type*>

0 static scale.
<<*data*>> ⇒ <*scale* : FLOAT64> <*sensorID* : INT16>
<*desired value* : FLOAT64>

1 dynamic scale.
<<*data*>> ⇒ <*sensorID* : INT16> <*desired value* : FLOAT64>

8 set sensor warning levels
<*data*> ⇒ <*min distance/angle* : FLOAT64> <*max distance/angle* : FLOAT64>

9 set sensor passing levels
<*data*> ⇒ < same <*data*> as #8>

Math Sensor Query Response

<*type/shape* : UINT8><<*data*>>

0 no shape defined

1 distance calculator
<*data*> = <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

2 angle calculator
<*data*> = <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

3 intersection calculator
<*data*> = <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

4 midpoint calculator
<*data*> = <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

5 midline calculator
<*data*> = <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

6 line through two points calculator
<*data*> = <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

7 perpendicular line calculator
<*data*> = <*sensor ID1* : INT16> <*param ID1* : INT16>
<*sensor ID2* : INT16> <*param ID2* : INT16>

8 scale factor calculator

9 coordinate transformation
<*data*> = <*Image Pt0 X* : FLOAT64> < *Image Pt0 Y*: FLOAT64> <
Image Pt1 X : FLOAT64> < *Image Pt1 Y* : FLOAT64> <*World*
Pt0 X : FLOAT64> < *World Pt0 Y*: FLOAT64> < *World Pt1*
X : FLOAT64> < *World Pt1 Y* : FLOAT64> < *Negative*
Z : UINT8> < *Angle Scale* : FLOAT64> < *Angle*
Offset : FLOAT64> <*sensor ID1* : INT16>
<*param ID1* : INT16>

angle range:
<*angle range* : UINT4>

Scaling Params (for distance SoftSensor):
<*type* : UINT4> <<*sensorID* : INT16>>

Scaling Params (for scale SoftSensor):
<*type* : UINT4> <<*scale factor* : FLOAT64> <*sensorID* : INT16>
<*desired value* : FLOAT64>> <<*sensorID* : INT16>
<*desired value* : FLOAT64>> <*precision* : INT8>

warning levels:
<<*min distance/angle* : FLOAT64> <*max distance/angle* :
FLOAT64>>

failure levels:

```
<<min distance/angle : FLOAT64> <max distance/angle :
   FLOAT64>>
```

Outputs:

```
<<distance : FLOAT64>> <<angle : FLOAT64>> <<x : FLOAT64>
   <y : FLOAT64>> <<x0 : FLOAT64> <y0 : FLOAT64>
   <a : FLOAT64> <b : FLOAT64>> <<scale factor : FLOAT64>>
```

Math Sensor Referencing IDs	
Input (i.e. – Will reference other sensors by):	
Distance	112
Scale	116
Output (i.e. – Can be referenced by)	
Position	0
Subpixel Position	96
Subpixel Line Position	97
Distance	112
Angle	113
Scale	116

Failure Code	**Warning Code**	**Math SoftSensors**
-1	1	minimum distance/angle
-2	2	maximum distance/angle

DataMatrix SoftSensor (Module ID: 6)

Command Format For Setting DataMatrix Sensor Parameters

SP<*sensor ID* : INT16><*parameter ID* : UINT8><*data*>

<*parameter ID*>

0 Set sensor shape/type

<*data*> ⇒ <*type* : UINT8><*coord data*>

<*type*>

3 rectangle

<*coord data*> ⇒ <*x0* : INT16> <*y0* : INT16> <*x1* : INT16> <*y1* : INT16>

4 parallelogram sensor

<*coord data*> ⇒ <*x0* : INT16> <*y0* : INT16> <*x1* : INT16> <*y1* : INT16> <*x2* : INT16> <*y2* : INT16>

2 set code parameters

<*data*> ⇒ <*edge type* : UINT4> <*threshold* : INT16> <*symbol size* : UINT8> <<*max rotation* : UINT8>> <*max skew* : UINT8> <*min cell size* : UINT16> <*max cell size* : UINT16> <*fill density* : UINT4> <*max scans* : UINT8> <*max edges* : UINT8> <*debug1* : INT32>

edge type:

0 Dark on Light Code

1 Light on Dark Code

threshold:

threshold*100

symbol size:

0	SIZE_10_10	1	SIZE_12_12	2	SIZE_14_14	3	SIZE_16_16
4	SIZE_18_18	5	SIZE_20_20	6	SIZE_22_22	7	SIZE_24_24
8	SIZE_26_26	9	SIZE_32_32	10	SIZE_36_36	11	SIZE_40_40
12	SIZE_44_44	13	SIZE_48_48	14	SIZE_52_52	15	SIZE_64_64
16	SIZE_72_72	17	SIZE_80_80	18	SIZE_88_88	19	SIZE_96_96
20	SIZE_104_104	21	SIZE_120_120	22	SIZE_132_132	23	SIZE_144_144
24	SIZE_8_18	25	SIZE_8_32	26	SIZE_12_26	27	SIZE_12_36
28	SIZE_16_36	29	SIZE_16_48				

max rotation and skew:

angle in degrees

aspect ratio:

(cell height/cell width)*100

min/max cell size (width):

(size in pixels)*100

fill density:

0 > 30%

1 ≤ 30%

9 Set Pass Criteria

<*data*> ⇒ <*start location* : INT16> <*string* : char 200>

if *start location* = -1, then match entire string, otherwise match substring starting at *start location*.

10 set input reference

<*data*> ⇒ <*input ID* : INT16><*reference sensor ID* : INT16> <*reference output ID* : INT16>

11 set learned position

<*data*> ⇒ <*x* : INT16> <*y* : INT16> <*theta* : INT16, INT16>

theta = x,y orientation vector with mag(*theta*)=16384

DataMatrix Sensor Query Response

```
<type/shape : UINT8><<data>>
```

0 no shape defined

3 rectangle

```
   <data> = <x0> <y0> <x1> <y1>
```

4 parallelogram shape

```
   <data> = <x0> <y0> <x1> <y1> <x2> <y2>
```

```
<edge type : UINT4> <threshold : INT16> <symbol size : UINT8>
   <<max rotation : UINT8>> <max skew : UINT8>
   <min cell size : UINT16> <max cell size : UINT16>
   <fill density : UINT4> <max scans : UINT8> <max edges : UINT8>
   <debug1 : INT32>
```

Pass Criteria:

```
<start location : INT16> <string : char 200>
```

Input References:

```
<position - sensor ID:INT16><position - output ID:INT16>
```

Outputs:

```
<string : char 200>
```

Position during Learning:

```
<x : INT32> <y : INT32> <theta : INT32,INT32>
```

DataMatrix Sensor Referencing IDs	
Input (i.e. – Will reference other sensors by):	
Position	0
Output (i.e. – Can be referenced by)	
String	128

Format of encoded string:
A '~' is an escape character. Any non-printing character is printed as a '~*xx*', where *xx* is the ASCII code, in hex. A '~7e' is used to represent a '~' in the data string.

Failure Code	Warning Code	DataMatrix SoftSensors
-1	1	String does not match pass criteria

Blob Generation SoftSensor (Module ID: 7)

Command Format For Setting Blob Generation Sensor Parameters

SP<*sensor ID* : INT16><*parameter ID* : UINT8><*data*>

<*parameter ID*>

0 Set sensor shape/type

<*data*> ⇒ <*type* : UINT8><*coord data*>

<*type*>

3 rectangle

<*coord data*> ⇒ <*x0* : INT16> <*y0* : INT16> <*x1* : INT16> <*y1* : INT16>

1 Set sensor threshold

<*data*> ⇒ <*type* : UINT4> <*value* : INT16> <<*data*>>

<*type*>

0 fixed gradient value
1 percentage of contrast along path
2 percentage of reference inputs

<*value*>

100 * gradient, or
100 * percent of reference/contrast

2 Set general information

<*data*> ⇒ <*Connectivity Type* : UINT4> <*Use Boundary Blobs* : UINT4> <*Min Blob Size* : INT32> <*Max Blob Size* : INT32>

Connectivity type:

0 4-Connectedness
1 8-Connectedness

3 Set preprocessing information

<*data*> ⇒ <*Binarization Type* : UINT4> <*Preproc. 1 Type* : INT8> <*Preproc. 1 Size* : INT32> … <*Preproc. 6 Type* : INT8> <*Preproc. 6 Size* : INT32>

Binarization type:

0 Dark Blobs
1 Light Blobs
2 Edges: Roberts Operator
3 Edges: Prewitt Operator
4 Edges: Sobel Operator

Preprocessing type:

0 None
1 Dilation
2 Erosion
3 Open
4 Close

8,9 Set Warn/Pass Criteria

<*data*> ⇒ <*Min Contrast* : INT16> <*Min Number Blobs* : INT16>
<*Max Number Blobs* : INT16>

10 set input reference

<*data*> ⇒ <*input ID* : INT16><*reference sensor ID* : INT16>
<*reference output ID* : INT16>

11 set learned position

<*data*> ⇒ <*x* : INT16> <*y* : INT16> <*theta* : INT16, INT16>
theta = x,y orientation vector with mag(*theta*)=16384

Blob Generation Sensor Query Response

<*type/shape* : UINT8><<*data*>>

0 no shape defined

3 rectangle

<*data*> = <x0> <y0> <x1> <y1>

<*threshold type* : UINT4> <*value* : UINT8>

0 Fixed greyscale value

1 Percent of contrast along path

2 Percent of reference input contrast

<*Connectivity Type* : UINT4> <*Use Boundary Blobs* : UINT4> <*Min Blob Size* : INT32> <*Max Blob Size* : INT32>

<*Binarization Type* : UINT4> <*Preproc. 1 Type* : INT8> <*Preproc. 1 Size* : INT32> … <*Preproc. 6 Type* : INT8> <*Preproc. 6 Size* : INT32>

Warn Criteria:

<*Min Contrast* : INT16> <*Min Num Blobs* : INT16> <*Max Num Blobs* : INT16>

Pass Criteria:

<*Min Contrast* : INT16> <*Min Num Blobs* : INT16> <*Max Num Blobs* : INT16>

Input References:

<*position - sensor ID*:INT16><*position - output ID*:INT16>
<*Min Intensity - sensor ID*:INT16>< *Min Intensity - output ID*:INT16>
< *Max Intensity - sensor ID*:INT16>< *Max Intensity - output ID*:INT16>

Outputs:

< *Min Intensity* :INT32>< *Max Intensity* :INT32> < *Contrast* :INT32> < *Threshold* :INT32> < *Num Blobs* :INT32>

Position during Learning:

<*x* : INT32> <*y* : INT32> <*theta* : INT32,INT32>

Blob Generation Sensor Referencing IDs	
Input (i.e. – Will reference other sensors by):	
Position	0
Low Intensity Reference	16
High Intensity Reference	17
Output (i.e. – Can be referenced by)	
Min Intensity	16
Max Intensity	17
Contrast	20
Threshold	21
Number of Blobs	35
Blob Feature Vector	144

Failure Code	Warning Code	Blob Generation SoftSensors
-1	1	Minimum contrast
-2	2	Minimum blobs
-3	3	Maximum blobs

Blob Selection SoftSensor (Module ID: 8)

Command Format For Setting Blob Selection Sensor Parameters

SP<*sensor ID* : INT16><*parameter ID* : UINT8><*data*>

<*parameter ID*>

0 Set sensor shape/type
<*data*> ⇒ <*type* : UINT8><*coord data*>

<*type*>

0 Group selection

2 General Blob Selector parameters
<*data*> ⇒ <*angle type* : INT16> <*blob reference* : INT16>
angle type:
0 Principle Axis (± 180^{o})
1 Principle Axis (0-360^{o})
2 Maximum Point from Centroid

3 Group algorithm parameters
<*data*> ⇒ <*Min Area* : INT32> <*Max Area* : INT32> <*Min Perimeter* : FLOAT64> <*Max Perimeter* : FLOAT64> <*Min Position - X* : FLOAT64> <*Min Position - Y* : FLOAT64> <*Max Position - X* : FLOAT64> <*Max Position - Y* : FLOAT64> <*Min Angle* : FLOAT64> <*Max Angle* : FLOAT64> <*Min Intensity* : INT32> <*Max Intensity* : INT32> <*Min Eccentricity* : FLOAT64> <*Max Eccentricity* : FLOAT64> <*Min Compactness* : FLOAT64> <*Max Compactness* : FLOAT64> <*Min Bounding Box Width* : INT32> < *Max Bounding Box Width* : INT32> < *Min Bounding Box Height* : INT32> < *Max Bounding Box Height* : INT32> <*Min Radius* : FLOAT64> <*Max Radius* : FLOAT64>

8,9 Set Warn/Pass Criteria
<*data*> ⇒ <*Minimum Number Blobs* : INT32> <*Maximum Number Blobs* : INT32>

10 Set input reference
<data> ⇒ <Blob Generation Sensor ID : INT16>

Blob Selection Sensor Query Response

<*type/shape* : UINT8><<*data*>>

0 Group selection algorithm

<*angle type* : INT16> <*blob reference* : INT16>

<*Min Area* : INT32> <*Max Area* : INT32> <*Min Perimeter* : FLOAT64> <*Max Perimeter* : FLOAT64> <*Min Position - X* : FLOAT64> <*Min Position - Y* : FLOAT64> <*Max Position - X* : FLOAT64> <*Max Position - Y* : FLOAT64> <*Min Angle* : FLOAT64> <*Max Angle* : FLOAT64> <*Min Intensity* : INT32> <*Max Intensity* : INT32> <*Min Eccentricity* : FLOAT64> <*Max Eccentricity* : FLOAT64> <*Min Compactness* : FLOAT64> <*Max Compactness* : FLOAT64> <*Min Bounding Box Width* : INT32> < *Max Bounding Box Width* : INT32> < *Min Bounding Box Height* : INT32> < *Max Bounding Box Height* : INT32> <*Min Radius* : FLOAT64> <*Max Radius* : FLOAT64>

Warn/Pass Criteria:
<*Minimum Number Blobs* : INT32> <*Maximum Number Blobs* : INT32>

Input References:
<Blob Generation Sensor ID : INT16>

Outputs:
<*x* : INT32> <*y* : INT32> <*theta* : INT32,INT32> <*Number of Blobs* : INT32>

Blob Selection Sensor Referencing IDs	
Input (i.e. – Will reference other sensors by):	
Blob Feature Vector	144
Output (i.e. – Can be referenced by)	
Position	0
Number of Blobs	35
Subpixel Position	96
Blob Area (Blobs 1-8)	256-263
Blob Subpixel Position (Blobs 1-8)	272-279
Blob Angle (Blobs 1-8)	288-295

Failure Code	**Warning Code**	**Blob Selection SoftSensors**
-1	1	Minimum blobs
-2	2	Maximum blobs

Scripting SoftSensor (Module ID: 9)

Command Format For Setting Scripting Sensor Parameters

SP<*sensor ID* : INT16><*parameter ID* : UINT8><*data*>

<*parameter ID*>

0 Set general sensor parameters

<*data*> ⇒ <*Program Length* : UINT16> <*Num Tokens* : UINT16> <*Stack Depth* : UINT16> <*Output Bits* : UINT16>

1 Add a program line

<*data*> ⇒ <*Number of Line Characters* : UINT8> <*Line Char 1* : UINT8> … <*Line Char n* : UINT8>

NOTE: Line Characters are transmitted as their ASCII values.

2 Add program tokens

<*data*> ⇒ <*Number of Tokens sent* : INT16> <*Token 1* : INT16><<*token data*>> … <*Token n* : INT16><<*token data*>>

token	<token data>
TOKEN_PUSHI(1)	<*i* : INT32>
TOKEN_PUSHF(2)	<*d* : FLOAT64>
TOKEN_PUSHSTR (12)	*not implimented*
TOKEN_JUMP_FALSE (3)	<*jump addr* : INT32>
TOKEN_JUMP (4)	
TOKEN_LOAD (5)	<*id* : INT32>
TOKEN_STORE (6)	
TOKEN_FUNCTION (8)	
TOKEN_LOAD_SREF (9)	<*Sid* : INT16> <*Oid* : INT16> <*Mid* : INT16>
TOKEN_CHECK_SREF (11)	
TOKEN_STORE_SREF (10)	<*Oid* : INT16> <*Mid* : INT16>

3 Set Warn string

<*data*> ⇒ <*Warn String* : string 200>

4 Set Fail string

<*data*> ⇒ <*Fail String* : string 200>

Scripting Sensor Query Response

<*type* : UINT8> <*Program Length* : UINT16> <*Number of Tokens* : UINT16>
<*Stack Depth* : UINT16> <*Output Bits* : UINT16>

<*Number Program Line Characters* : UINT8> <*Line Char 1* : UINT8> … <*Line Char n* : UINT8>

.
.
.

<*Number of Tokens Sent* : UINT8> <*Token 1* : UINT8> <<*Token Data* >> …
<*Token n* : UINT8> <<*Token Data* >>

<<*Token Data* >> See above

.
.
.

<*Warn String* : STRING200>

<*Fail String* : STRING200>

Outputs:

<*x* : INT32> <*y* : INT32> <*theta* : INT32,INT32> <*Intensity* : INT32>
<*Distance* : FLOAT64> <*Angle* : FLOAT64> <*Subpixel X* : FLOAT64>
<*Subpixel Y* : FLOAT64> <*Line Xo*: FLOAT64> <*Line Yo* : FLOAT64> <*Line a* : FLOAT64> <*Line b* : FLOAT64>

Scripting Sensor Referencing IDs	
Input (i.e. – Will reference other sensors by):	
Position	0
Minimum Intensity	16
Maximum Intensity	17
Mean Intensity	18
Median Intensity	19
Contrast	20
Threshold	21
# of Edges	32
# of Features	33
# of Pixels	34
# of Blobs	35
% below threshold	48
Subpixel Position	96
Subpixel Line	97
Subpixel Circle	98
Distance/Radius	112
Angle	113
Straightness	114
Roundness	115
Bar Code String	128
Blob Area (Blobs 1-8)	256-263
Blob Subpixel Position (Blobs 1-8)	272-279
Blob Angle (Blobs 1-8)	288-295
Output (i.e. – Can be referenced by)	
Position	0
Intensity	16
Subpixel Position	96
Subpixel Line	97
Distance	112
Angle	113
String	128

Failure Code	Warning Code	Scripting SoftSensors
-1	1	Result set to FAIL or WARN
-2	2	Processing error
<-2	>2	Result set to unknown fail or warn value