Process Eye
SECS Host
ProcSECS (A)
Manual

SP104011 Rev 1.00
September 1999

As part of our continuous product improvement policy, we are always

pleased to receive your comments and suggestions about how we should develop our product range. We believe that the manual is an important part of the product and would welcome your feedback particularly relating to any omissions or inaccuracies you may discover.

You can send your comments to:-

Spectra SensorTech Ltd.
Cowley Way
Crewe
Cheshire
CW1 6AG
U.K.

+44 1270 250150 Tel. International
+44 1270 251939 Fax. International

01270 250150 Tel. U.K.
01270 251939 Fax. U.K.

In North America you can send your comments to:-

Spectra Instruments
380 Woodview Ave.
Morgan Hill
CA 95037

(408) 778 6060 Office
(408) 776 8575 Fax

Contents

This page is deliberately left blank.

Errata and addenda

This page is deliberately left blank.

This page is deliberately left blank.

Section 1.
Introduction

1.1. Overview

There are two versions of ProcSECS. One is designed for use with tools having a generic SECS II or GEM/SECS II interface and is known as ProcSECS (G). The other is designed for use with Endura and Centura Process tools from Applied Materials via the RGA SECS port and is known as ProcSECS (A). This manual covers ProcSECS (A).

The ProcSECS application gives Process Eye the ability to communicate with Endura and Centura Process tools from Applied Materials via the RGA SECS port. At its simplest it may just be used to ensure the correct RGA recipe is run at the appropriate time. A more ambitious link could run appropriate recipes, log tool data as part of the RGA data and perform statistical monitoring of the data and even stop wafer processing. The computer running Process Eye, i.e. the RGA(s), and ProcSECS is the Host, the Endura or Centura Process tool is the Equipment.

Data is transmitted using the SECS I protocol.

There are four methods that can be used to utilise RGA(s) to Process Tool communications: Pre-Programmed, Grid Display, ProcSECS Macros and Process Eye Macros.

The ProcSECS Wizard is provided to assist in configuring the pre-programmed actions and Grid Display. It also adds the required information to RGA configuration files.

1.1.1. Pre-Programmed Actions

Certain messages sent over the RGA SECS interface can be configured to cause one or more actions at the RGA computer.

1.1.2. Grid display

Status variables may be polled from the tool at a user defined interval and displayed in a spreadsheet like grid. Row and column headings may be

defined as well as the placement of the data.

1.1.3. ProcSECS Macros

Messages received by ProcSECS are pre-processed and then made available to the VBA compatible macro language. Additionally any or all messages may be configured to be intercepted by the Macro. Within the macro arbitrarily complex actions may be taken.

1.1.4. Process Eye Macros

Using the Macro facility of Process Eye virtually anything that can be done over the RGA SECS interface may be integrated with RGA operation. Using the Macro method requires a detailed understanding of the RGA SECS interface, VBA programming and Process Eye's implementation of the link. (Note that in Multi-head situations it may not be possible to direct the appropriate message to the appropriate head and ProcSECS macros should be used if the pre-programmed capabilities are not sufficient.)

Section 2.
Installation

Close any programs that are running, insert the ProcSECS install disk in floppy drive A: (or B:) and either click Start|Settings|Control Panel|Add/Remove Programs or double click the Setup.Exe file from explorer. Follow the on-screen instructions. It is suggested that you leave all settings at their default values. The only item that should be changed is the directory if the default is not the same as the directory in which Process Eye is installed.

2.1. Configuration RGA

For the SECS II link to function the ProcSECS software must be correctly configured. The ProcSECS .Cfg files must be correctly set using a text editor such as NotePad. In addition Win.Ini must be modified to show the location of the ProcSECS software but this is handled automatically by the install program. The ProcSECS.Iin file will be created or modified using the ProcSECS Wizard.

2.1.1. ProcSECS.Cfg – SECS I (RS232)

Below is a typical configuration for a SECS I (RS232) link.

```
PORT 0:
  LINE COM1
  PROTOCOL SECS
  BAUD 9600
  T1 0.5
  T2 3.0
  T3 30.0
  T4 10.0
  RETRY 3
  HOST
```

```
DEVICE 0:
  PORT 0
  DEVICEID 0x0000
```

Port 0: parameters

LINE Defines the RS232 port the SECS link is using. Must be COM1 or COM2

PROTOCOL Must be SECS

BAUD defines the Baud rate. May be 110, 150, 300, 600, 1200, 2400, 4800, 9600 or 19200

T1 The SECS T1 (Inter Character) timeout. Must be between 0.1 and 25.5 seconds

T2 The SECS T2 (Block Protocol) timeout. Must be between 0.1 and 25.5 seconds

T3 The SECS T3 (Reply) timeout. Must be between 1.0 and 120 seconds

T4 The SECS T4 (Inter Block) timeout. Must be between 1.0 and 120 seconds

RETRY The SECS Retry Limit (RTY). Must be between 0 and 31

HOST Specifies that ProcSECS is going to act as the SECS host. Generally should be left alone but if the other end cannot be specified as Equipment it may be set to EQUIPMENT.

DEVICE 0: parameters

PORT must match the Port number defined above. Should be left at 0

DEVICEID The SECS Device ID. A number between 0 and 32767

2.1.2. Configure Quick Start

RS232
Set LINE to COM1 or COM2
Set BAUD (typically 9600)
Set DEVICEID (typically 0)

TCP/IP
Set PASSIVE ENTRY IPADDRESS
Set PASSIVE ENTITY TCPPORT (typically 5000)
Set DEVICEID (typically 0)

2.2. Configuration Endura/Centura

The Endura/Centura must be configured as the SECS Master (Equipment)
Enable Collection Events as required
Enable Ping
Either Set Device ID to 0 or change RGA Device ID setting
Enable link

2.3. ProcSECS Wizard

ProcSECS is configured using the ProcSECS Wizard. Normally when run the ProcSECS Wizard will appear as shown in Figure 1. If the Configuration window appears then select Endura/Centura and the appropriate Process Eye version. Running ProcSECS Wizard from the icon created during installation will bypass the configuration window.

Generally the following order of data entry should be used.
1 Enter all Heads
2 Enter S1F3 Polls (if required)
3 Configure the Grid (if required)
4 Define Events (if required)
5 Recipe lookup (if required)

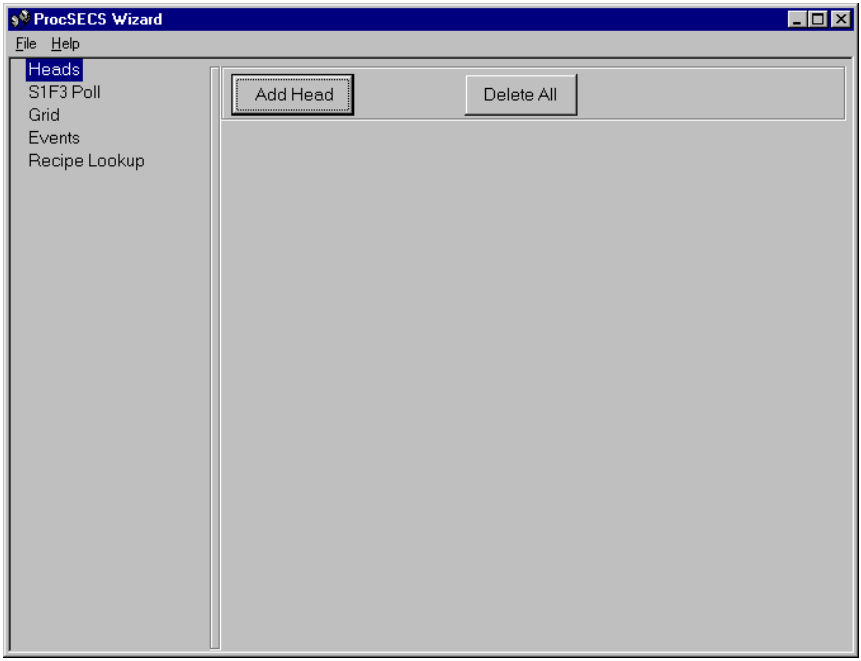

Figure 1

2.3.1. RGA (Head) settings

Each head that communicates with ProcSECS must be defined in the Heads section. Click on the Heads branch of the tree view and then click on the Add Head button and the Head configuration form will appear.

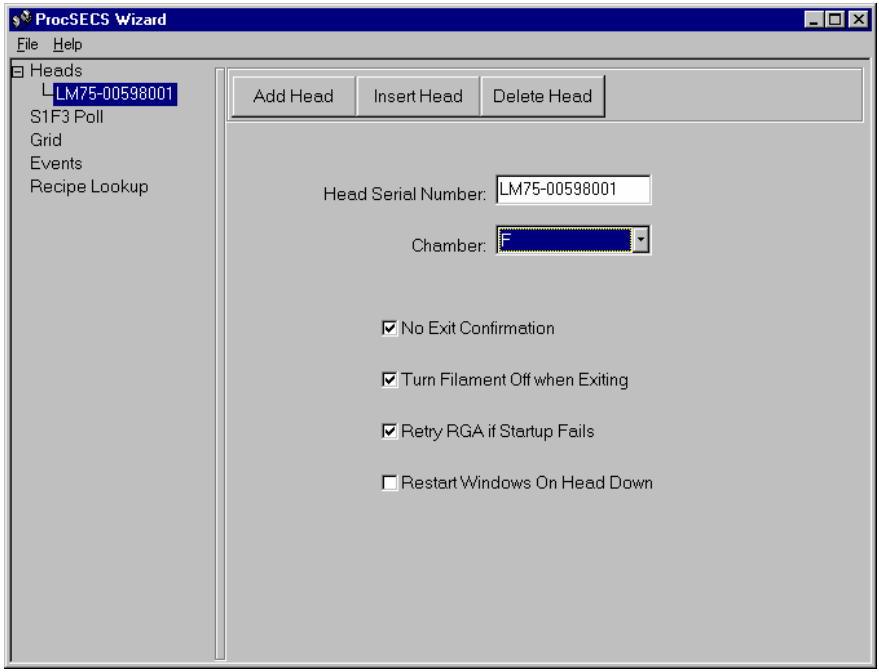

Figure 2 Head Configuration Form

Enter the Serial Number in the Head Serial Number box (V1.6x or later) or the Head Number (V1.4x). You must associate the Head with the correct Chamber on the Endura by selection from the list provided in the Chamber List box.

The following items may be set to facilitate automatic start and stop of Process Eye for that head.

No Exit Confirmation:
Skip the exit confirmation dialog box.

Turn Filament Off when Exiting:
Force the RGA's filament to be switched off if it is on when the head closes down.

Retry RGA if Startup Fails:
If the RGA fails to communicate at startup retry (V1.6x only).

Restart Windows On Head Down:
If the RGA head stops communicating restart Windows (V1.6x only).

2.3.2. Status Variable Polling (S1F3 Poll)

All tools maintain a number of Status Variables giving details of the tools current status that may be polled using the standard SECS II S1F3 message with the data coming back in an S1F4 reply. The data may be displayed in the Grid Display (see Grid section for details).

The S1F3 Poll section allows the Status Variables, that will be polled, to be defined.

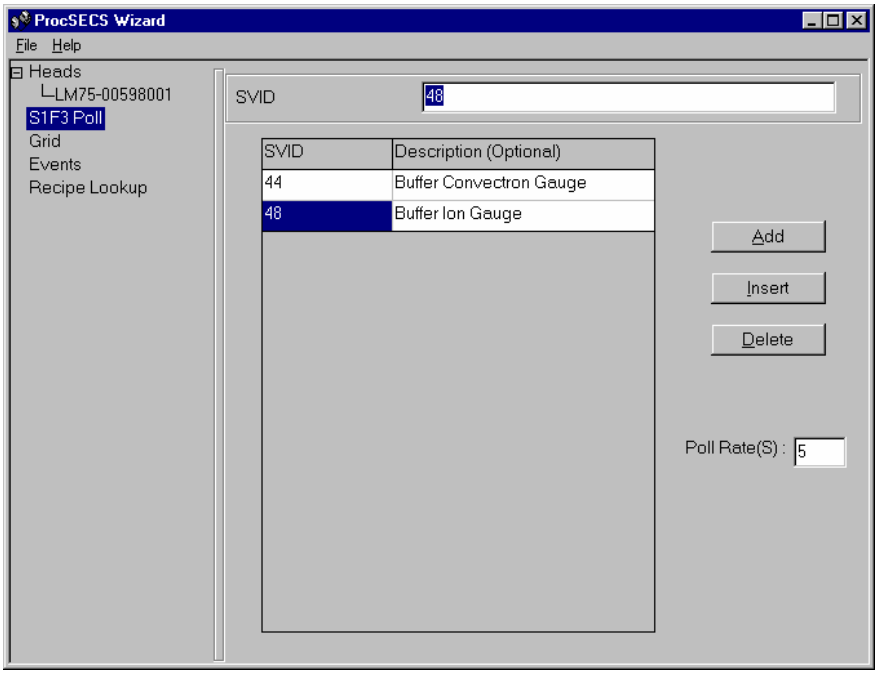

Figure 3 S1F3 Poll Form

Select the Poll Rate required (1 to 60 seconds) in the Poll Rate box.

To add a Status Variable to the Poll list click the Add Button. Enter the Status Variable ID (SVID number taken from the tools SECS manual) and a

useful description if required.

2.3.3. Grid Display

The Grid section is used to configure the optional grid display available in ProcSECS. It has two functions:-

1 To display data either directly from an S1F3 poll, or from the ProcSECS Macro.
2 To enable the selection of automatic or manual control of individual RGA heads.

Click on the Grid section in the Tree View to display the Grid Form.

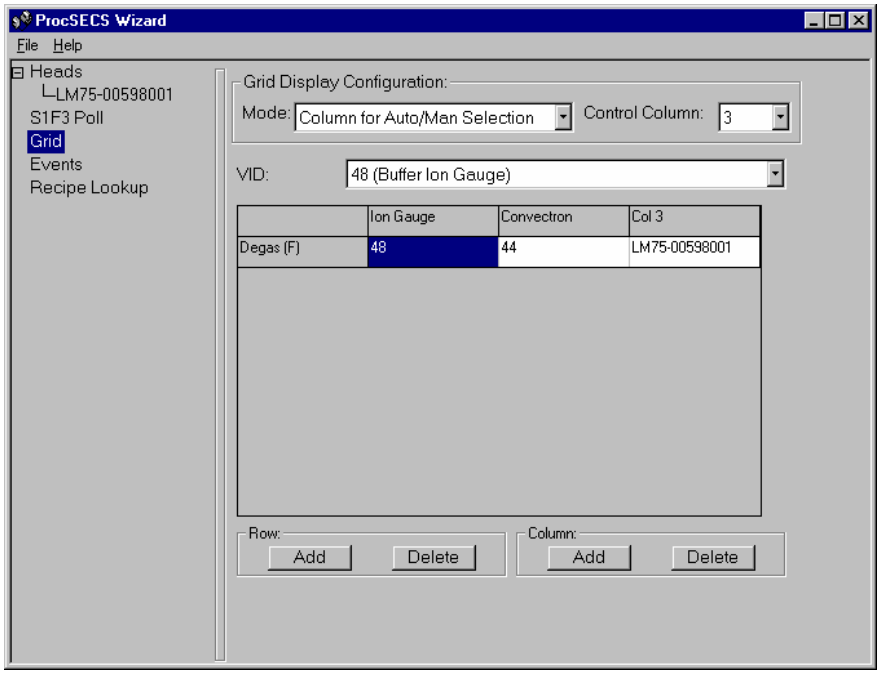

Figure 4 Grid Form

Select the required mode of operation for the grid from the Mode list box.
Disabled:
Will prevent the grid from being displayed.

Column for Auto/Man Selection:
Will reserve one column and a row for each head defined for Auto/Manual selection at run time. In the ProcSECS Wizard the reserved cells will display the Head serial number or Head Number. The actual column reserved can be specified by the Control Column list box. The reserved rows will always be first and the heads will appear in the order they were defined in the Heads Section.
Unused Cells in the grid are available to display Poll Status Variables or data from the Macro.

Normal:
Cells in the grid are used to display Poll Status Variables or data from the Macro.

When the desired mode has been selected additional rows and columns may be added using the Row and Column Add buttons.

Each cell can then be selected in turn and a Status Variable chosen from the VID list box. The “Unused” entry can be selected to remove an unwanted VID.

You may also define meaningful row and column titles by clicking on the appropriate cell and entering the row or column title.

Warning: Editing the Heads section or changing the Grid mode after defining a grid will invalidate all grid settings.

Section 3.
Operation

3.1. Pre-Programmed Operation

In Pre-Programmed operation you define what action should be taken when a specific even occurs. Not all Actions are applicable to all events (see the table below for a summary)

	Program Start	Program Stop	SECS Link Fail	S1F4 Poll	S6F67	S6F69	S6F71
1 – Run Recipe: Fixed Name	X	-	X	-	X	X	X
2 – Run Recipe: Event Defined	-	-	-	-	-	X	-
3 – Run Recipe: Event Defined with Lookup	-	-	-	-	-	X	-
4 – Reserved	-	-	-	-	-	-	-
5 – Set User Disk Header	-	-	-	-	-	X	-
6 – Stop Recipe	X	-	X	-	X	X	X
7 – Run External Program	X	-	X	-	X	X	X
8 – Add to RGA Recipe Comment	-	-	-	-	X	X	X
9 – Set Global Variable	-	-	-	X	X	X	X
10 – Switch On Filament	-	-	-	-	X	X	X
11 – Switch Off Filament	-	-	X	-	X	X	X
12 – Write Comment to disk	X	X	X	-	X	X	X
13 – Stop Process Eye program	-	X	X	-	-	-	-
14 – Restart Windows	-	X	X	-	-	-	-

“X” = available for that event
“-“ = unavailable for that event

Note that the same action, e.g. Run an External Program, may be executed many times for a single event with different parameters for each instance of the action as well as multiple different actions.

All Actions are defined in Events Section. Some events have actions directly available e.g. Start, Stop etc. while others are associated with specific Chamber events e.g. S6F69 S6F71 etc.

The events you wish to handle are selected in the Events Form using the Add button. All selected events will then appear as a branch under the Events Section of the Tree View. Selecting these will either allow the direct actions or the specific chamber to be chosen.

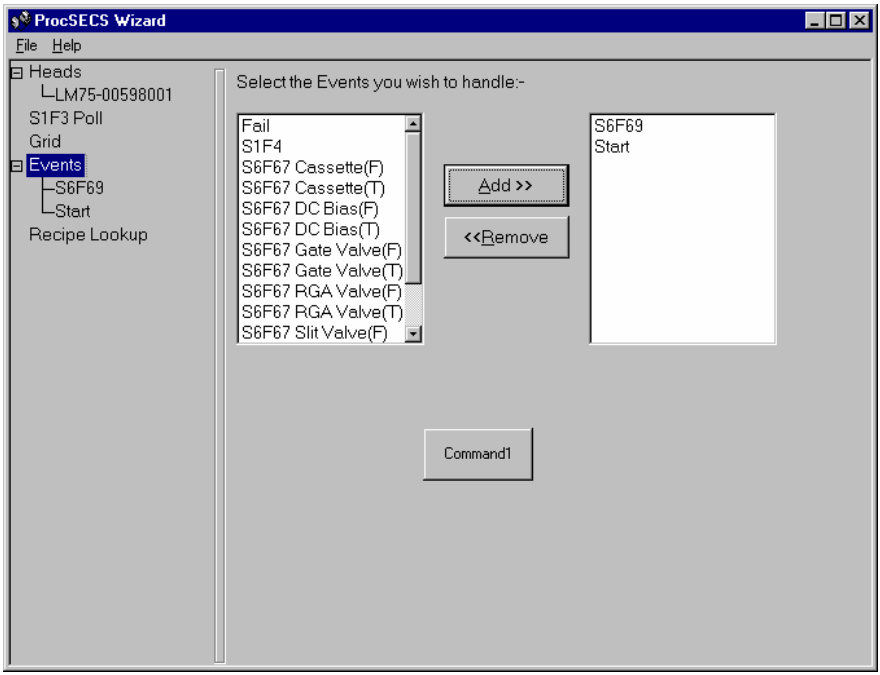

Figure 5 Events Form

For Endura Events (S6F67, S6F69 and S6F71) the Chamber Select Form allows you to specify the specific Chamber that actions are associated with. Chambers are selected by highlighting the chamber on the left and clicking the Add Button. Events may be generated by Chambers (and have actions associated with them) that do not have an RGA installed.

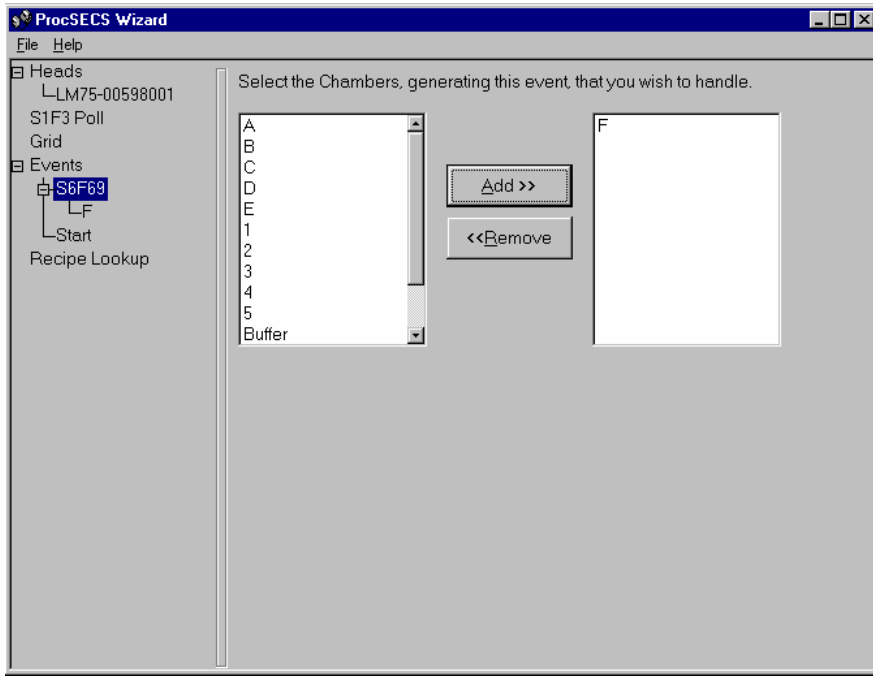

Figure 6 Chamber Select Form

3.2. Program Start

When the ProcSECS program runs there are four possible actions. The main use is to run the RGA(s) so that starting a single program runs both the ProcSECS application and the Process Eye application(s). If one or more RGAs are already running, they may have recipes started or stopped. Finally the log file SECSlog.Txt can have a line written to it.

Action: “Run Recipe: Fixed Name”
The RGA on which to run the Recipe is defined in the Target either by Chamber or by Head.
The Recipe name to run is defined in the “Recipe Name” box. The Recipe name must match the text on the recipe button within Process Eye.
If DDE Filename has been selected as the file naming method in the recipe the DDE filename is defined using the “DDE Filename” box. Note that Process Eye is a 16 bit application and the DDE filename must be in DOS 8.3 format. Also using an existing filename will overwrite the previous file with no warnings.

Action: “Stop Recipe”

The RGA on which to stop the Recipe is defined in the Target either by Chamber or by Head.

Action: “Run External Program”

Starts an external program. The name (including path if required) is defined in the “Program to Start” box. Note that Process Eye is a 16 bit application and any path must be in DOS 8.3 format. Any arguments may be added to the “Program to Start” box with a space to separate the path/filename and arguments.
The window style that the program is to start with is defined in the “Window Style” list box.

Action: “Log message to SECSlog.Txt”

Appends “Date Time Process SECS host started” to the SECSlog.Txt file

3.3. Program Stop

When the ProcSECS program runs there are only three possible actions. The main use is to stop the RGA(s) so that when ProcSECS is restarted the heads will be restarted and the link between ProcSECS and Process Eye(s) established. The other actions available are to write to the SECSlog.Txt file and/or to restart Windows.

Action: "Log message to SECSlog.Txt"

Appends "Date Time Process SECS host stopped" to the SECSlog.Txt file

Action: "Stop Process Eye Program"

The RGA on which to stop is defined in the Target either by Chamber or by Head.

Action: "Restart Windows"

Windows will be terminated and then restarted.

3.4. SECS Link Fail

When the ProcSECS link to the Endura fails there are seven possible actions. The main use is to restart windows if the and cause the computer to re-initialize and re-make the SECS link. If one or more RGAs are already running, they may have recipes started or stopped, the filament(s) turned off. An external program may be run and the log file SECSlog.Txt can have a line written to it. Finally Windows may be terminated and restarted.

Action: "Run Recipe: Fixed Name"

The RGA on which to run the Recipe is defined in the Target either by Chamber or by Head.
The Recipe name to run is defined in the "Recipe Name" box. The Recipe name must match the text on the recipe button within Process Eye.
If DDE Filename has been selected as the file naming method in the recipe the DDE filename is defined using the "DDE Filename" box. Note that Process Eye is a 16 bit application and the DDE filename must be in DOS 8.3 format. Also using an existing filename will overwrite the previous file

with no warnings.

Action: "Stop Recipe"

- The RGA on which to stop the Recipe is defined in the Target either by Chamber or by Head.

Action: "Run External Program"

- Starts an external program. The name (including path if required) is defined in the "Program to Start" box. Note that Process Eye is a 16 bit application and any path must be in DOS 8.3 format. Any arguments may be added to the "Program to Start" box with a space to separate the path/filename and arguments.
- The window style that the program is to start with is defined in the "Window Style" list box.

Action " Switch Off Filament"

- The RGA on which to switch off the filament is defined in the Target either by Chamber or by Head.

Action: "Log message to SECSlog.Txt"

- Appends {Date} {Time} "SECS link failed" to the SECSlog.Txt file

Action: "Stop Recipe"

- The RGA on which to stop the Recipe is defined in the Target either by Chamber or by Head.

Action: "Restart Windows"

- Windows will be terminated and then restarted.

3.5. Status variable poll (S1F4)

When data from a Status Variable poll is received by ProcSECS only a single action is configurable – send the data to a global variable within an RGA head.

Action: "Set Global Variable"

- The RGA on which to set a Global Variable is defined in the Target either by Chamber or by Head.
- The Global Variable to set is defined in the "Global Variable" box. Typically global variables are numbered 1 through 8 (defined in Process Eye configuration) but this may be extended.
- The data to store in the global variable is defined in the "SVID" list box.

3.6. Events

Endura and Centura Process tools from Applied Materials offer a number of useful messages over its RGA SECS interface. There are three events that may have actions associated with them: S6F67, S6F69 and S6F71. As described above each of these events are specific to an Endura chamber.

3.6.1. S6F67

The S6F67 Event is broken down into five groups: Cassette Present, RGA Valve, Gate Valve, DC Bias and Slit Valve. Each group is further divided into two Events – a true (T) Event and a false(F) Event. Each Event is treated individually.

The following applies equally to all Collection Event messages (Cassette Present, RGA Valve, Gate Valve, DC Bias and Slit Valve) however you should be aware that certain combinations cannot occur, for example there will be no Chamber 1 events for the Cassette Present Collection Event.

Action: "Run Recipe: Fixed Name"

- The RGA on which to run the Recipe is defined in the Target either by Chamber or by Head.
- The Recipe name to run is defined in the "Recipe Name" box. The Recipe name must match the text on the recipe button within Process Eye.
- If DDE Filename has been selected as the file naming method in the recipe the DDE filename is defined using the "DDE Filename" box. Note that Process Eye is a 16 bit application and the DDE filename

must be in DOS 8.3 format. Also using an existing filename will overwrite the previous file with no warnings.

Action: "Stop Recipe"

- The RGA on which to stop the Recipe is defined in the Target either by Chamber or by Head.

Action: "Run External Program"

- Starts an external program. The name (including path if required) is defined in the "Program to Start" box. Note that Process Eye is a 16 bit application and any path must be in DOS 8.3 format. Any arguments may be added to the "Program to Start" box with a space to separate the path/filename and arguments.
- The window style that the program is to start with is defined in the "Window Style" list box.

Action: "Add to Comment"

- The RGA on which to add the comment is defined in the Target either by Chamber or by Head.
- The comment to add is defined in the form: "{Date} {Time} Collection Event {Collection Event Name} on chamber {Chamber Name} became {Value}". The selected RGA must be running a Bar Chart or Peak Jump recipe.

Action: "Set Global Variable"

- The RGA on which to set a Global Variable is defined in the Target either by Chamber or by Head.
- The Global Variable to set is defined in the "Global Variable" box. Typically global variables are numbered 1 through 8 (defined in Process Eye configuration) but this may be extended.
- The data to store in the global variable is defined in the "Data" list box. Either Collection Event Number or Collection Event Value.

Action Type: "Switch On Filament"

- The RGA to have its filament switched on is defined in the Target either by Chamber or by Head.

Action " Switch Off Filament"

- The RGA on which to switch off the filament is defined in the Target either by Chamber or by Head.

Action: "Log message to SECSlog.Txt"

- Appends "{Date} {Time} Collection Event {Collection Event Name} on chamber {Chamber Name} became {Value} to the SECSlog.Txt file

3.6.2. S6F69

This message allows recipes on the RGA to be synchronized with recipes on the tool. It is arguably the most useful SECS message the Endura/Centura provides.

If DDE Filename is being used to define that name of the data file for a Process Eye recipe it can have embedded in it any of the following data items by inserting the text <n> in the DDE Filename where:
<1> is replaced by {Lot ID}
<2> is replaced by {Load Lock (A or B)}
<3> is replaced by {Wafer Number}
<4> is replaced by {Recipe Name}
<5> is replaced by {Chamber Number}

It is important that the DDE filename keep to DOS 8.3 filename conventions. Note that the "\" character is permitted and this allows the path for data storage to be dynamically defined at run time.

Action: "Run Recipe: Fixed Name"

- The RGA on which to run the Recipe is defined in the Target either by Chamber or by Head.
- The Recipe name to run is defined in the "Recipe Name" box. The Recipe name must match the text on the recipe button within Process Eye.
- If DDE Filename has been selected as the file naming method in the recipe the DDE filename is defined using the "DDE Filename" box. Note that Process Eye is a 16 bit application and the DDE filename

must be in DOS 8.3 format. Also using an existing filename will overwrite the previous file with no warnings.

Action Type: "Run Recipe: Event Defined"

- The RGA on which to run the Recipe is defined in the Target either by Chamber or by Head.
- The Recipe name to run is defined by the Process Chamber Recipe Name. The Recipe name must match the text on the recipe button within Process Eye.
- If DDE Filename has been selected as the file naming method in the recipe the DDE filename is defined using the "DDE Filename" box. Note that Process Eye is a 16 bit application and the DDE filename must be in DOS 8.3 format. Also using an existing filename will overwrite the previous file with no warnings.

Action Type: "Run Recipe: Event Defined with Lookup"

- The RGA on which to run the Recipe is defined in the Target either by Chamber or by Head.
- The Recipe name to run is defined by the Process Chamber Recipe Name and using that to lookup a corresponding RGA recipe name. The lookup is defined in the [RECIPE LOOKUP] section. Each Tool Recipe name becomes an item. The looked up recipe name must match the text on the recipe button within Process Eye.
- If DDE Filename has been selected as the file naming method in the recipe the DDE filename is defined using the "DDE Filename" box. Note that Process Eye is a 16 bit application and the DDE filename must be in DOS 8.3 format. Also using an existing filename will overwrite the previous file with no warnings.

Action Type: "Set User Disk Header"

- The RGA on which to run the Recipe is defined in the Target either by Chamber or by Head.
- Uses the 5 data items (Lot Name, Load Lock, Wafer Number, Process Chamber Recipe Name and Chamber Number) as the values for Process Eye's User Disk Header. It is important that the user disk header definition be set for this format with 5 values. Setting the user disk header will cause the next recipe that has User Disk Header enabled to use the information sent with this command

without the usual screen dialog. This is true even if the recipe is run locally.

Action: "Stop Recipe"

- The RGA on which to stop the Recipe is defined in the Target either by Chamber or by Head.

Action: "Run External Program"

- Starts an external program. The name (including path if required) is defined in the "Program to Start" box. Note that Process Eye is a 16 bit application and any path must be in DOS 8.3 format. Any arguments may be added to the "Program to Start" box with a space to separate the path/filename and arguments.
- The window style that the program is to start with is defined in the "Window Style" list box.

Action: "Add to Comment"

- The RGA on which to add the comment is defined in the Target either by Chamber or by Head.
- The comment to add is defined in the form: "{Date} {Time} Recipe Start for Chamber: {Chamber Name} Lot: {MID} Load Lock: {LLx} Wafer: {Wafer Number}". The selected RGA must be running a Bar Chart or Peak Jump recipe.

Action: "Set Global Variable"

- The RGA on which to set a Global Variable is defined in the Target either by Chamber or by Head.
- The Global Variable to set is defined in the "Global Variable" box. Typically global variables are numbered 1 through 8 (defined in Process Eye configuration) but this may be extended.
- The data to store in the global variable is defined in the "Data" list box. It can be: Lot Name, Load Lock, Wafer Number, Process Chamber Recipe Name or Chamber Number.

Action Type: "Switch On Filament"

- The RGA to have its filament switched on is defined in the Target either by Chamber or by Head.

Action " Switch Off Filament"

- The RGA on which to switch off the filament is defined in the Target either by Chamber or by Head.

Action: "Log message to SECSlog.Txt"

- Appends "{Date} {Time} Recipe Start for Chamber: {Chamber Name} Lot: {MID} Load Lock: {LLx} Wafer: {Wafer Number}" to the SECSlog.Txt file

3.6.3. S6F71

This message allows end of recipe on a tool to cause of actions on an RGA.

Action: "Run Recipe: Fixed Name"

- The RGA on which to run the Recipe is defined in the Target either by Chamber or by Head.
- The Recipe name to run is defined in the "Recipe Name" box. The Recipe name must match the text on the recipe button within Process Eye.
- If DDE Filename has been selected as the file naming method in the recipe the DDE filename is defined using the "DDE Filename" box. Note that Process Eye is a 16 bit application and the DDE filename must be in DOS 8.3 format. Also using an existing filename will overwrite the previous file with no warnings.

Action: "Stop Recipe"

- The RGA on which to stop the Recipe is defined in the Target either by Chamber or by Head.

Action: "Run External Program"

- Starts an external program. The name (including path if required) is defined in the "Program to Start" box. Note that Process Eye is a 16 bit application and any path must be in DOS 8.3 format. Any arguments may be added to the "Program to Start" box with a space to separate the path/filename and arguments.
- The window style that the program is to start with is defined in the "Window Style" list box.

Action: "Add to Comment"

- The RGA on which to add the comment is defined in the Target either by Chamber or by Head.
- The comment to add is defined in the form: "{Date} {Time} Recipe Stop for Chamber: {Chamber Name}". The selected RGA must be running a Bar Chart or Peak Jump recipe.

Action: "Set Global Variable"

- The RGA on which to set a Global Variable is defined in the Target either by Chamber or by Head.
- The Global Variable to set is defined in the "Global Variable" box. Typically global variables are numbered 1 through 8 (defined in Process Eye configuration) but this may be extended.
- The data stored is always the Chamber Number in which the recipe stopped.

Action Type: "Switch On Filament"

- The RGA to have its filament switched on is defined in the Target either by Chamber or by Head.

Action " Switch Off Filament"

- The RGA on which to switch off the filament is defined in the Target either by Chamber or by Head.

Action: "Log message to SECSlog.Txt"

- Appends: "{Date} {Time} Recipe Stop for Chamber: {Chamber Name}" to the SECSlog.Txt file

3.7. Recipe Lookup

If the "Run Recipe: Event Defined with Lookup" action has been defined the lookup table must be set up.

Click on the "Recipe Lookup" item in the tree and then the "Add" button. Click in the grid under "Tool Recipe" and type in the Tool Recipe name (case insensitive). Then click on the corresponding "RGA Recipe" and type the RGA recipe (again case insensitive) that is to be associated with the Tool Recipe. Add as many lookup pairs as required.

Section 4.
Macros

4.1. Overview

The ProcSECS macro capability was added for two general reasons. Firstly the basic functionality of the pre-programmed operation may be augmented where it is not powerful enough and secondly some or all of the handling of SECS messages from the tool can be performed using the macro. In addition other actions such as linking with external programs, displaying data in the ProcSECS grid etc. can be performed.

The macro language is Visual Basic for Applications ™ (VBA) compatible with extensions that allow manipulation of SECS messages in addition to controlling RGA(s).

Each time one of a number of events occurs the macro will be called with a variable set that indicates which event has happened. The blank form of the macro that should be used is in MTSECS.BAS. It should be edited either using VB as a editor or a text editor such as Notepad.Exe and then renamed SECS.BAS and placed in the same directory as the ProcSECS.EXE file. The form of a ProcSECS macro is:

```
Const PRIMARY_MSG  = -3
Const STATUS_MSG = -2
Const REPLY_MSG = -1
Const APP_START = 1
Const APP_STOP = 2
Const S1F4_POLL = 3
Const S6F67 = 4
Const S6F69 = 5
Const S6F71 = 6

Sub Main ()

Do
        Select Case EventID
                Case PRIMARY_MSG ‘ Macro is handling this primary
message
```

```
            Case STATUS_MSG          ‘ RGA status message
            Case REPLY_MSG           ‘ Reply to message sent in macro
            Case APP_START   ‘ ProcSECS started
            Case APP_STOP    ‘ ProcSECS stopped
            Case S1F4_POLL   ‘ S1F4 Polled data message
            Case S6F67       ‘Tool generated a Collection Event
            Case S6F69       ‘ Tool generated an Recipe Start Event
            Case S6F71       ‘ Tool generated a Recipe Stop Event
        End Select
        Stop
    Loop
End Sub
```

The appropriate code going in each Case section as required.
Most of the events occur with no additional configuration but to allow “raw” SECS II messages to be passed through the ProcSECS application to the macro running in ProcSECS the following items in the ProcSECS.Ini file must be set:

ProcSECS Default Handler
Alternate Message Handler
Alternate Handler Count
Alt Stream N
Alt Function N
Auto Drop Ticket

See the Advanced Features section for more details.

To facilitate Macro to tool and Macro to RGA communication the follow language extensions have been made available.

4.2. Read Only Variables

4.2.1. EventID

An Integer giving the event that caused the macro to be called. Value will be from –3 to 6 with 0 excluded.

-3 = Primary SECS message that the macro is handling
-2 = Status message from an RGA

-1 = Reply message to a primary message sent from within the macro
1 = ProcSECS program start
2 = ProcSECS program stop
3 = SVID Poll data available
4 = S6F67 Collection Event
5 = S6F69 Recipe Start Event
6 = S6F71 Recipe Stop Event

4.2.2. ReplyFromTool

If a SECS message is sent to the tool with the W bit set to 1 a reply will be generated and this reply is saved in the ReplyFromTool variable. See the example in below.

Important Note. Reading the variable sets it to nothing (“”)

It is a comma delimited string in the following format:

Stream – the SECS Stream – 0 to 255
Function – the SECS Function – 0 to 255
Item Count – the number of SECS data items
Data Type Item 1 – the type of data for the first item (See Appendix 1 for details)
Data Item 1 – the first data item
.
.
Data Type Item n – the type of data for the last item
Data Item n – the last data item

4.2.3. S6F71Data

An Integer giving the chamber number as documented in the Applied Materials RGA SECS manual of the chamber that has stopped running a recipe.

4.3.Read Only Arrays

4.3.1. S1F4Data(Index)

A variant array that holds the data from the most recent S1F3/4 poll. Index must be an Integer in the range 1 to SVID Count. If Index is outside of this range 0 will be read for that array element.

4.3.2. StatusData(Index)

A variant array of two elements that holds the data from the most recent RGA Status Event. Index must be an Integer in the range 1 to 2. If Index is outside of this range 0 will be read for that array element.
Index = 1 gives the Status Data
Index = 2 gives the time (in Now form) that the last status was received, this may be used to determine

4.3.3. S6F67Data(Index)

A variant array of three elements that holds the data from the most recent S6F67 Event. Index must be an Integer in the range 1 to 3. If Index is outside of this range 0 will be read for that array element.
Index = 1 gives the Collection Event ID
Index = 2 gives the data (1 or 0) for the collection event
Index = 3 gives a pointer into the EnduraEventData/EnduraEventTime arrays.

4.3.4. S6F69Data(Index)

A variant array of three elements that holds the data from the most recent S6F69 Recipe Start Event. Index must be an Integer in the range 1 to 5. If Index is outside of this range 0 will be read for that array element.
Index = 1 gives the Lot ID (a.k.a. MID)
Index = 2 gives the Load Lock the wafer came from (LLA or LLB)
Index = 3 gives the Wafer Number (0 if unknown by the tool or 1 – 30)
Index = 4 gives the Endura Recipe Name
Index = 5 gives the Endura chamber number the recipe is running on

4.3.5. EnduraEventData(Index)

A variant array of 75 elements that holds the data from Collection Events.

Index must be an Integer in the range 1 to 75. If Index is outside of this range 0 will be read for that array element.
Index = 1 – 15 = Cassette (only 14 & 15 actually used)
Index = 16 - 30 = RGA Valve (only 16 - 28 actually used)
Index = 31 - 45 = Gate Valve (only 31 - 43 actually used)
Index = 46 - 60 = Plasma (DC Bias) (only 46 - 58 actually used)
Index = 61 - 75 = Slit Valve (all used)

4.3.6. EnduraEventTime(Index)

A variant array of 75 elements that holds the time each item in EnduraEventData was last updated. Index must be an Integer in the range 1 to 75. If Index is outside of this range 0 will be read for that array element. The ordering of the elements is the same as for EnduraEventData above.

Note the EnduraEventData and EnduraEventTime arrays should be initialised with an S1F65 message to the tool using the MessageToTool subroutine with the MacroHandlesReply parameter set false so ProcSECS will handle the S1F66 reply.

4.4. Subroutines

4.4.1. MessageToTool MessageString, MacroHandlesReply

MessageToTool sends the message in MessageString to the tool.
Important Note: When MessageToTool is used with the W bit set the reply will be placed in ReplyFromTool regardless of where the reply message would usually be handled. As there is only space for one reply in ReplyFromTool you must wait for the reply before sending the next message.
MessageString is a string consisting of a comma delimited series of parameters:

Stream the SECS Stream – 0 to 255
Function the SECS Function – 0 to 255
W bit the SECS W bit – 0 or 1
Item Count the number of SECS data items
Data Type Item 1 the type of data for the first item (See Appendix 1 for details)
Data Item 1 the first data item
.

.
Data Type Item n the type of data for the last item
Data Item n the last data item

MacroHandlesReply is an optional Boolean parameter – if omitted it will be taken as true which means if the W bit is 1 the macro must handle the reply message. If false then the reply will be handled by ProcSECS.

Example to request Equipment Constant 5 by sending an S2F13 and having the macro handle the reply (i.e. Stream = 2, Function = 13, W bit = 1, Item Count = 2, Data Item 1 is a list of 1 so Data Type Item 1 = List = 2, Data Item 1 = 1, Data Item 2 is an 2 byte signed Integer of value 5 so Data Type Item 2 = I2 = 29, Data Item 2 = 15):

```
Dim ReplyMessage as String
        MessageToTool "2,13,1,2,4,1,29,5"
        Do
                DoEvents
                ReplyMessage = ReplyFromTool
        Loop Until ReplyFromTool <> ""
        ' Then Process ReplyMessage
```

4.4.2. MessageToRGA MessageString, RGA

Sends the MessageString to the RGA in the RGA string. For V1.6x RGA is the serial number, for V1.4 it is the head number (as a string)
Valid commands are:
END RECIPE – halts currently running recipe
LOCKOUT,{Lock} – disables/enables the head. Lock = 1 locks it, Lock = 0 unlocks it
RUN,{Recipe name},{DDE Filename} –
runs a recipe. DDE filename is optional
SET UDH,{UDH} – defines the user disk header, UDH is a comma delimited string of the individual headers
SECS,{Message} – a SECS message that may be processed by the Process Eye macro
ADD COMMENT,{Comment} –
adds a text comment to the comment file for a running recipe
FIL,{Setting} – turns the currently selected filament on (1) or off (0)
SET GLOBAL,{Global Variable Number},{Data} –

stores the data in the selected global variable for use by a recipe macro.
TERMINATE – closes down the instance of Process Eye

4.4.3. WriteToGrid Row, Col, Data, FormatString

Write the Variant Data into Row, Col of the Grid. By default it will be formatted in scientific notation but the optional variant FormatString may be used and the data will be formatted using Format$(Data,FormatString)

4.5. Functions

4.5.1. IsRGARegistered (RGA)

Returns a Boolean to indicate if the RGA (serial number or head) in the String RGA has run up and registered itself with ProcSECS and is therefore ready to take commands.

4.5.2. IsRGAInAutomatic (RGA)

Returns a Boolean to indicate if the RGA (serial number or head) in the String RGA is currently in Automatic mode (returns True) or Manual mode (returns false).

4.6. Process Eye Programming

Process Eye Macro & Stand Alone Programming .
Messages not generated or handled within ProcSECS can either be generated and handled by macro code within Process Eye or by a stand alone program.

There are two types of message in to your application:
Responses to Primary messages with the W bit set sent by your application
Primary messages from the tool

To allow “raw” SECS II messages to be passed through the ProcSECS application to either a macro running in Process Eye or a stand alone application the following items in the ProcSECS.Ini file must be set:

ProcSECS Default Handler
Alternate Message Handler

Alternate Handler Count
Alt Stream N
Alt Function N
Auto Drop Ticket

See the Advanced Features section for more details.

4.7. Macro Language Extensions

Process Eye hides some of the details of a SECS link with the following Functions, Subroutines and Variables. These are in addition to the existing macro extensions detailed in the Process Eye manual

4.7.1. Read Only Variables

SECSMessageCount – an integer giving the number of messages currently waiting in the SECS message input buffer.

4.7.2. Functions

SendSECSMessage(Message As String) – Transmits a primary SECS message (for secondary messages use SendSECSReply) and returns an Integer to indicate the message was successfully transmitted to ProcSECS. This does not guarantee the message made it to the tool. The Message is a string of comma delimited values:
Stream the SECS Stream – 0 to 255
Function the SECS Function – 0 to 255
W bit the SECS W bit – 0 or 1
SECS ID a string used to allow your application to identify which of the incoming reply messages belongs with this primary.
Item Count the number of SECS data items
Data Type Item 1 the type of data for the first item (See Appendix 1 for details)
Data Item 1 the first data item
Data Type Item n the type of data for the last item
Data Item n the last data item

SendSECSReply(Message As String) – Transmits a primary SECS message (for secondary messages use SendSECSReply) and returns an Integer to indicate the message was successfully transmitted to ProcSECS. This does not guarantee the message made it to the tool. The Message is a string of comma delimited values:

Stream the SECS Stream – 0 to 255 – the same as the incoming message
Function the SECS Function – 0 to 255
Ticket the SDR ticket received with the incoming message
Item Count the number of SECS data items
Data Type Item 1 the type of data for the first item (See Appendix 1 for details)
Data Item 1 the first data item
.
.
Data Type Item n the type of data for the last item
Data Item n the last data item

PollSECSData(SECS_ID As String) – searches the messages currently waiting in the SECS message input buffer for the reply to your primary message with the specified SECS ID, returns a Boolean, true if the message is waiting, false if it is not.

GetSECSData(SECS_ID As String) – searches the messages currently waiting in the SECS message input buffer for the reply to your primary message with the specified SECS ID. If the message is found it is returned as a string, if it is not found a null string is returned. The message string is a comma delimited string in the following format:

SECS ID the string sent by your application when transmitting the primary message.
Ticket the SDR transaction ticket. It can usually be ignored except it is used in the SendSECSReply message or if, despite all our warnings, you disable Auto Drop Ticket feature.
Stream the SECS Stream – 0 to 255 – the same as the incoming message
Function the SECS Function – 0 to 255
W Bit the SECS W bit – 0 or 1
Item Count the number of SECS data items
Data Type Item 1 the type of data for the first item (See Appendix 1 for details)
Data Item 1 the first data item

Data Type Item n the type of data for the last item
Data Item n the last data item

PopSECSData() – returns a String that is the last message put into the SECS message input buffer. If no messages are in the buffer it returns a null. The message is in exactly the same format as the GetSECSData described above.

SendSECSDrop(Ticket As String) – if you have taken responsibility for dealing with SDR tickets this function is used to drop tickets when they are no longer required. If tickets are not properly dropped the SDR application will eventually (after about 255 tickets are used) stop functioning. It is strongly recommended that the Auto Drop Ticket capability of ProcSECS is used.

RemoveSECSHeader(SECSMsg As String) – returns the Item Count (integer) in a message and the message in SECSMsg is returned with the SECS_ID, Ticket, Stream, Function, W Bit and Item Count data removed.

NthCommaValue(CommaDelimitedString As String, N As Integer) returns a variant which is the Nth position in a comma delimited string. Note each item consists of two values, the data type and the data. Multiply the item number by 2 and subtract 1 to point at the data type and multiply the item number by 2 to point at the data.

Example: To poll Status Variable ID 4 once per scan and store it in Action Channel 1.
(Note there are a number of different methods of implementing this particular feature and using the ProcSECS Macro capability should also be considered)

```
Global Result as Integer        ‘ general variable to store SECS command status in
Global SECS_ID as Long          ‘ incrementing variable that tracks message

Sub Main ()
        PreTrigger
        PauseMacro

        PostTrigger
        PauseMacro

        PreRun
        PauseMacro

        Do While Scanning
              EndOfScan
              PauseMacro
        Loop

        PostRun
End Sub

Sub PreTrigger ()
‘ Request value of SVID #4 for the first time using S1F3 with W set
        Result = SendSECSMessage(“1,3,1,” & Cstr(SECS_ID) & “2,4,1,29,4")
End Sub

Sub PostTrigger ()
End Sub

Sub PreRun ()
End Sub

Sub EndOfScan ()

Dim S1F4Message as String       ‘ Variable to hold the S1F4 reply to our S1F3 message
Dim S1F4ItemCount as Integer    ‘ Variable to hold number of data items in S1F4 reply
Dim SVID4 as Variant            ‘ Variable to hold SVID 4 value

        S1F4Message= GetSECSData(Cstr(SECS_ID))
        S1F4ItemCount = RemoveSECSHeader(S1F4Message)       ‘ Remove header,
        get ItemCount
        SVID4 = NthCommaValue(S1F4ItemCount * 2, S1F3Message)       ‘ *2 as 2
        values per Item
        WriteHugeArrayData (1,ScanNumber,ActionChannel(1),CSng(SVID4))‘ Save
        SECS_ID = SECS_ID + 1                               ‘ Use a different SECS ID
        Result = SendSECSMessage(“1,3,1,” & Cstr(SECS_ID) & “2,4,1,29,4") ‘ Request
        SVID 4 again
```

```
End Sub

Sub PostRun ()
End Sub
```

4.8. Miscellaneous Notes

It is essential that any instances of Process Eye that are to communicate with ProcSECS are run after ProcSECS has initialised.

Appendix 1
Date type names

Names to use when defining data types

Name	Macro Value	SECS II Name
S2_L	4	List
S2_B	8	1 Byte Binary
S2_BOOLEAN	12	Boolean
S2_A	16	ASCII
S2_U1	24	1 Byte Unsigned Integer
S2_U2	25	2 Byte Unsigned Integer
S2_U4	26	4 Byte Unsigned Integer
S2_U8	27	8 Byte Unsigned Integer
S2_I1	28	1 Byte Signed Integer
S2_I2	29	2 Byte Signed Integer
S2_I4	30	4 Byte Signed Integer
S2_I8	31	8 Byte Signed Integer
S2_F4	34	4 Byte Floating Point Number
S2_F8	35	8 Byte Floating Point Number

Recipe Lookup syntax

[RECIPE LOOKUP]
ToolRecipeName1 = RGARecipeName1
ToolRecipeName2 = RGARecipeName2
ToolRecipeName3 = RGARecipeName3

This page is deliberately left blank.

Appendix 2
Chamber names

Endura/Centura Chamber names

Chamber A	Pre clean
Chamber B	Cooldown
Chamber C	Normally Unused
Chamber D	Same as C
Chamber E	O/D or unused
Chamber F	O/D or unused
Chamber 1	Process Chamber 1
Chamber 2	Process Chamber 2
Chamber 3	Process Chamber 3
Chamber 4	Process Chamber 4
Chamber 5	Wafer Park or Wafer Temperature Control (rare)
Bfr	Buffer Chamber (front wafer Transfer no Process)
Xfr	Transfer Chamber
LLA	Load Lock A
LLB	Load Lock B

Endura/Centura chamber to RGA lookup

[CHAMBER TO RGA]
Chamber A = RGA Serial Number
Chamber B = RGA Serial Number
Chamber C = RGA Serial Number
Chamber D = RGA Serial Number
Chamber E = RGA Serial Number
Chamber F = RGA Serial Number
Chamber 1 = RGA Serial Number
Chamber 2 = RGA Serial Number
Chamber 3 = RGA Serial Number
Chamber 4 = RGA Serial Number
Chamber 5 = RGA Serial Number
Bfr = RGA Serial Number

Xfr = RGA Serial Number
LLA = RGA Serial Number
LLB = RGA Serial Number

Note for Process Eye V1.4x RGA Serial Number must be replaced by the head number (1 through 9)

Appendix 3
Advanced features

The ProcSECS.Ini file
In the [SYSTEM] section there are a number of possible configuration settings. In general they should be left at the default settings (i.e. nothing in the file). If editing the ProcSECS.Ini file it is essential that a text editor that saves text in plain format (such as Notepad) be used.

App Is Active
When set to 0 will cause ProcSECS to wait for the other end to connect using S1F1. When set to 1 will cause the ProcSECS application to send an S1F1 connect message. By default it will be 0,

Log Errors
All error messages can be written to a file called ProcSECS.Err. By default all errors are logged. (Log Errors = 1). Set to 0 to disable error logging.

The following configurations only apply if the Pre-Programmed method of handling SECS messages is to be replaced or augmented using the ProcSECS Macro, Process Eye Macros or another stand alone application.

ProcSECS Default Handler
When set to 1 the ProcSECS main application handles all messages that are not specified (see below). When 0 the Alternative Message Handler handles all messages by default.

Alternate Message Handler
Specifies the place messages not handled by ProcSECS are sent to in terms of the Serial number for Microvision Plus and HPQ 2 RGAs, Head number for Microvision RGAs and an arbitrary string for Stand Alone applications (See registering stand alone applications) or the ProcSECS macro code (set Alternative Message Handler = Macro)

Alternate Handler Count
Defines how many messages are handled by the Alternative Message

Handler. Default is zero.

Alt Stream N
Alt Function N
For each message to handled by the Alternate Message Handler the message Stream and function are define. N = 1, 2, 3 …. Alternate Handler Count

Auto Drop Ticket
ProcSECS uses the SDR toolkit licensed from GW Associates to perform SECS I or HSMS-SS communications. The SDR uses the Ticket method to deal with multiple messages. When set to 1 ProcSECS drops all tickets except Primary messages with the W bit set (it is the Process Eye Macro or Stand Alone applications responsibility to deal with these tickets). Should be left at 1 unless you really know why you want to change it.