PowerLogic® Ethernet Communications Card (ECC) and PowerLogic® Ethernet Gateway (EGX400) PowerLogic® Series 800 Power Meter Ethernet Communications Card (PM8ECC)

Reference Guide

Retain for future use.

INTRODUCTION

This reference manual covers web page development for the PowerLogic® Ethernet Communications Card (ECC), PowerLogic® Ethernet Gateway EGX400, and the PowerLogic® Series 800 Power Meter Ethernet Communications Card (PM8ECC).

For additional information, refer to the following documentation:

- ECC installation manual 63230-304-201
- ECC user's guide 63230-304-204
- EGX installation manual 63230-314-200
- EGX user's guide 63230-314-208
- PM8ECC installation guide 63230-506-200
- PM8ECC user's guide 63230-506-204

To download the documenation from www.PowerLogic.com, select your country > Literature > Communication Devices > Ethernet ECC or Ethernet EGX > Instructional, then select the manual you want to download. Follow the instructions on the website for obtaining a password to access online literature.

APPLICATIONS FOR THE ECC/EGX400/ PM8ECC

The ECC/EGX400/PM8ECC functions primarily as an Ethernet gateway to allow Ethernet access to Modbus/Jbus and PowerLogic (SY/MAX) serial devices. Additionally, the ECC/EGX400/PM8ECC functions as a web and file server.

Supported Ethernet Protocols

The ECC/EGX400/PM8ECC supports the following Ethernet protocols:

- **Modbus TCP/IP:** Used to serve compatible Modbus TCP/IP masters such as the System Manager™ Software (SMS) system via TCP port 502.
- **Hypertext Transfer Protocol (HTTP)**: Provides web server functionality via TCP port 80. Remote configuration and the viewing of real-time and historical data is possible using a web browser. Nonvolatile memory in the EGX400/PM8ECC is used to store the web pages, graphics, documentation, controls, applets, and other files.
- **File Transfer Protocol (FTP)**: Provides the ability to transfer the following file types to and from the EGX400/PM8ECC via TCP port 21:

  NOTE: The ECC only supports HTM/HTML files. The file types with an asterisk (*) are parsed by the EGX400/PM8ECC as described in the following sections.

  — GIF and JPEG graphics files
  — PDFs
  — Java applets
  — ActiveX controls
  — HTM/HTML*

  - — XML*
  - — XSL*
  - — XSD*
  - — DTD*
  - — Txt files*
- **Simple Mail Transfer Protocol (SMTP)**: Provides the ability to send e-mail messages using TCP port 25.
- **Simple Network Management Protocol (SNMP):** Based on MIB2 format, SNMP provides the ability to store and send identifying and diagnostic information used for network management purposes via UDP port 161 .
- **Simple Network Time Protocol (SNTP)**: SNTP is a protocol used to synchronize the clocks of networked devices using a SNTP server via UDP port 123.

CREATING CUSTOM WEB PAGES FOR THE ECC/EGX400/PM8ECC

To create custom web pages for the ECC/EGX400/PM8ECC, you should have the following:

- A general understanding of the PowerLogic Power Monitoring and Control System
- A general understanding of the Internet and the World Wide Web (WWW)
- Basic skills with text editor software
- Working knowledge of HyperText Markup Language (HTML) and JavaScripting.

Hardware, Software, and Logistics Requirements

Before proceeding, ensure that you have the following:

- The ECC/EGX400/PM8ECC installed and assigned an IP address
- Access to the ECC/EGX400/PM8ECC via a LAN connection
- Web page editor or text editor software

File Storage Considerations

Custom HTML page examples are included on a CD-ROM shipped with the ECC/PM8ECC. The pages are configured to read data from the host device.

NOTE: Each HTML page for the ECC can have a maximum file size of 50 KB. A maximum of ten pages totalling 500 KB can be stored in the host circuit monitor. If you need to delete a custom page from the circuit monitor to make room for one you've created, you can restore the deleted page in the future from the CD-ROM.

Components of Custom HTML Pages

In general, each ECC/EGX400/PM8ECC custom page has two components and each one is developed using different tools.

Static Components: These components include the page layout, static text, color schemes, lines, and tables. This part of the custom page is usually created with a web page editor and customized by adding or modifying HTML tags.

Because the static portion of the web page is dependant on the user, it is left to the web designer to decide how to write the HTML code. Therefore, it is not addressed in this manual.

Dynamic Components: These components include special delimiters known as server side includes (SSI) that tell the ECC/EGX400/PM8ECC to dynamically get register information from attached devices and display it in the HTML page.

Accessing Devices

In general, the custom pages will be written in HTML with a special tag that tells the ECC/EGX400/PM8ECC to dynamically get register information from a device. The delimiter at the beginning of a tag is (PL_ _) and the delimiter at the end is (_ _PL). This tells the ECC/EGX400/PM8ECC to parse this string and dynamically fill it with register data. Table 1 lists the supported PowerLogic tags.

Table 1: PowerLogic Tags and Usage

Function Name	Function Code	PowerLogic TAG
SyMax Block Read - Registers	SyMax Function Code 0	<DeviceID>^<StartingRegisterAddress>[<NumberOfRegisters>] example tag = PL__1^1003[5]__PL example of data returned = 85,86,84,25,56
SyMax Scattered Read – Registers	SyMax Function Code 4	<DeviceID>^<RegisterAddress1>,<RegisterAddress2>,etc example tag = PL__1^1003,1004,1005,1006,1007__PL example of data returned = 85,86,84,25,56
Modbus Block Read- Coil Status	Modbus Function Code 1	<DeviceID>^C<StartingCoilAddress>[<NumberOfCoils>] example tag = PL__1^C1003[5]__PL example of data returned = 1,0,0,1,1
Modbus Block Read – Input Status	Modbus Function Code 2	<DeviceID>^D<StartingInputAddress>[<NumberOfInputs>] example tag = PL__1^D1003[5]__PL example of data returned = 1,0,0,1,1
Modbus Block Read – Holding Registers	Modbus Function Code 3	<DeviceID>^H<StartingRegisterAddress>[<NumberOfRegisters>] example tag = PL__1^H1003[5]__PL example of data returned = 85,86,84,25,56
Modbus Block Read – Input Registers	Modbus Function Code 4	<DeviceID>^I<StartingRegisterAddress>[<NumberOfRegisters>] example tag = PL__1^I1003[5]__PL example of data returned = 85,86,84,25,56
Modbus Scattered Read – Holding Registers	Modbus Function Code 100	<DeviceID>^S<RegisterAddress1>,<RegisterAddress2>,etc example tag = PL__1^S1003,1004,1005,1006,1007__PL example of data returned = 85,86,84,25,56
Modbus Block Read – General Reference	Modbus Function Code 20	<DeviceID>^F<StartingRegisterAddress>,[<NumberOfRegisters>]<File> example tag = PL__1^F1003[5]2__PL example of data returned = 85,86,84,25,56

NOTE: When creating custom HTML pages for the ECC/PM8ECC, an asterisk (*) can be used for the device ID. The asterisk acts as a wildcard so when it is parsed by the ECC/PM8ECC, the address of the host circuit monitor is automatically used for the device ID.

WEBPAGE GENERATOR UTILITY

The WebPageGenerator utility is a wizard-based tool developed by Schneider Electric that creates web pages. When you complete five simple steps in the utility, WebPageGenerator creates web pages and sends them to the web server to provide real-time and historical data views. This utility supports CM3000 and CM4000 circuit monitors with an ECC or stand-alone EGX400.

The WebPageGenerator creates five summary web pages for real-time data viewing from multiple devices, as well as detailed single-device web pages for data views from standard devices listed below. In addition, the WebPageGenerator can be used as an FTP interface to send the pages to the ECC or EGX web server.

You can download the WebPageGenerator utility from the Downloads section on www.PowerLogic.com. The utility is also provided on the EGX Technical Library CD that ships with the EGX and on the CM3000/CM4000 Technical Library CD that ships with the CM3000, CM4000, and ECC.

Minimum System Requirements

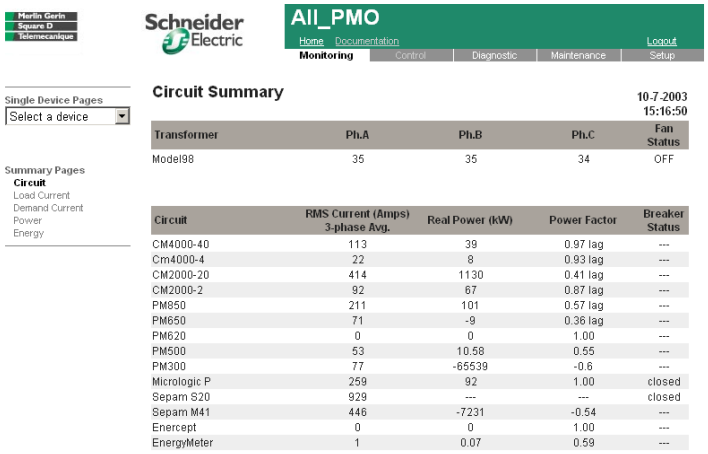
- Windows 98 SE, Windows NT 4.0, Windows 2000 Service Pack 2, Windows XP
- Internet Explorer 6.0 or later
- 30MB of available hard drive space

Supported Devices

The following devices are supported by PowerLogic WebPageGenerator:

- CM4000, CM3000, CM2000, CM200, and CM100 series circuit monitors
- Enercept and Energy submeters
- Micrologic low voltage trip units
- PM800, PM700, PM600, PM500, PM300, and PM9C series power meters
- Sepam protective relays series 80, 40, 20, and 2000
- Model98 transformer temperature controllers
- ATV and ATS drives and soft starters
- Motorlogic Plus motor starters
- Motorpact (Medium Voltage Soft Starter [MVSS])

Figure 1: Sample Circuit Summary Page



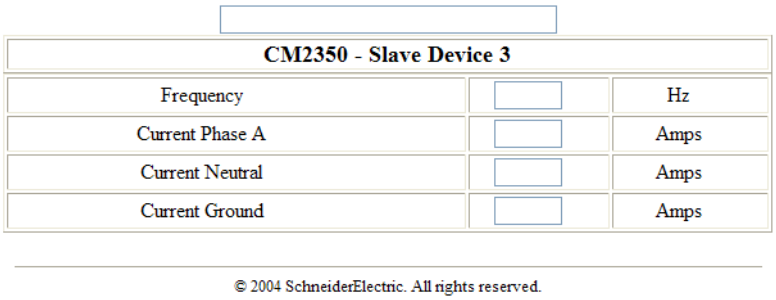
Transformer	Ph.A	Ph.B	Ph.C	Fan Status
Model98	35	35	34	OFF

Circuit	RMS Current (Amps) 3-phase Avg.	Real Power (kW)	Power Factor	Breaker Status
CM4000-40	113	39	0.97 lag	---
Cm4000-4	22	8	0.93 lag	---
CM2000-20	414	1130	0.41 lag	---
CM2000-2	92	67	0.87 lag	---
PM850	211	101	0.57 lag	---
PM650	71	-9	0.36 lag	---
PM620	0	0	1.00	---
PM500	53	10.58	0.55	---
PM300	77	-65539	-0.6	---
Micrologic P	259	92	1.00	closed
Sepam S20	929	---	---	closed
Sepam M41	446	-7231	-0.54	---
Enercept	0	0	1.00	---
EnergyMeter	1	0.07	0.59	---

EXAMPLE 1—CREATING A NEW HTML CUSTOM PAGE

Example 1 is an exercise in creating the following new HTML page to be transferred to the ECC/EGX400/PM8ECC.

Figure 2: Example 1 HTML Page



CM2350 - Slave Device 3		
Frequency		Hz
Current Phase A		Amps
Current Neutral		Amps
Current Ground		Amps

HTML Source Code for Example 1

The following is the HTML source code for the Example 1 web page. See Table 2 on page 9 for descriptions of source code.

NOTE: Bold type indicates the HTML and JavaScript that are key points to consider when making a custom page. Also note that the line numbers are for reference only; they are not part of the HTML.

Line No. HTML Syntax

```
1.  <html>
2.  <head>
3.  <META HTTP-EQUIV="refresh" CONTENT="5">
4.  <title>CM2350 - Slave Device 3</title>
5.  </head>
6.  <body>
7.  <form name="view_form">
8.  <p align="center">
9.  <input type = "text" name = "time_spot" size = "40">
10. <table border="1" width="600">
11. <tr>
12. <td width="600"><p align="center"><font size="4"><b>
13. CM2350 - Slave Device 3</b></font></p>
14. </td>
15. </tr>
16. </table>
17. <table border="1" width="600">
18. <tr>
19. <td width="300">
20. <p align="center">Frequency</p>
21. </td>
22. <td align="center" width="90"><p align="center">
23. <input type="text" size="5" name="frequency"></p>
24. <td width="100">
25. <p align="center">Hz</p>
26. </td>
27. </tr>
28. <tr>
29. <td width="300">
30. <p align="center">Current Phase A</p>
31. </td>
32. <td align="center" width="90"><p align="center">
33. <input type="text" size="5" name="currentphasea"></
    p>
34. <td width="100">
35. <p align="center">Amps</p>
```

```
36. </td>
37. </tr>
38. <tr>
39. <td width="300">
40. <p align="center">Current Neutral</p>
41. </td>
42. <td align="center" width="90"><p align="center">
43. <input type="text" size="5" name="currentneutral"></
    p>
44. <td width="100">
45. <p align="center">Amps</p>
46. </td>
47. </tr>
48. <tr>
49. <td width="300">
50. <p align="center">Current Ground</p>
51. </td>
52. <td align="center" width="90"><p align="center">
53. <input type="text" size="5" name="currentground"></
    p>
54. <td width="100">
55. <p align="center">Amps</p>
56. </td>
57. </tr>
58. </table>
59. <br><HR SIZE="1" width="66%"><CENTER><font
    face="Times Roman" size="2">© 2004
    SchneiderElectric. All rights reserved.</font></
    CENTER>
60. </form>
61. <script language="JavaScript">
62. function ShowFreq()
63. {
64. Registers =
    [PL__3^2020,2021,2022,2025,1001,1003,1006,1007__PL]
    ;
65. ScaleFactorA = Registers[0];
66. ScaleFactorB = Registers[1];
67. ScaleFactorC = Registers[2];
68. ScaleFactorF = Registers[3];
69. Frequency = Registers[4];
70. CurrentPhaseA = Registers[5];
71. CurrentNeutral = Registers[6];
72. CurrentGround = Registers[7];
73. ScaleFactorAMultiplier = 0;
```

```
74. ScaleFactorBMultiplier = 0;
75. ScaleFactorCMultiplier = 0;
76. ScaleFactorFMultiplier = 0;
77. TheTime = new Date();
78. switch (ScaleFactorA)
79. {
80. case -2:
81. ScaleFactorAMultiplier = 0.01;
82. break;
83. case -1:
84. ScaleFactorAMultiplier = 0.1;
85. break;
86. case 1:
87. ScaleFactorAMultiplier = 10;
88. break;
89. default:
90. ScaleFactorAMultiplier = 1;
91. break;
92. }
93. switch (ScaleFactorB)
94. {
95. case -2:
96. ScaleFactorBMultiplier = 0.01;
97. break;
98. case -1:
99. ScaleFactorBMultiplier = 0.1;
100.break;
101.case 1:
102.ScaleFactorBMultiplier = 10;
103.break;
104.default:
105.ScaleFactorBMultiplier = 1;
106.break;
107.}
108.switch (ScaleFactorC)
109.{
110.case -2:
111.ScaleFactorCMultiplier = 0.01;
112.break;
113.case -1:
114.ScaleFactorCMultiplier = 0.1;
115.break;
116.case 1:
```

```
117. ScaleFactorCMultiplier = 10;
118. break;
119. default:
120. ScaleFactorCMultiplier = 1;
121. break;
122. }
123. switch (ScaleFactorF)
124. {
125. case -1:
126. ScaleFactorFMultiplier = 0.1;
127. break;
128. default:
129. ScaleFactorFMultiplier = 0.01;
130. break;
131. }
132. Frequency *= ScaleFactorFMultiplier;
133. CurrentPhaseA *= ScaleFactorAMultiplier;
134. if (CurrentNeutral == -32768)
135. CurrentNeutral = "N/A";
136. else
137. CurrentNeutral *= ScaleFactorBMultiplier;
138. if (CurrentGround == -32768)
139. CurrentGround = "N/A";
140. else
141. CurrentGround *= ScaleFactorCMultiplier;
142. document.view_form.frequency.value = Frequency;
143. document.view_form.currentphasea.value =
     CurrentPhaseA;
144. document.view_form.currentneutral.value =
     CurrentNeutral;
145. document.view_form.currentground.value =
     CurrentGround;
146. document.view_form.time_spot.value = TheTime;
147. }
148. ShowFreq();
149. </script>
150. </body>
151. </html>
```

Table 2: Description of HTML Source Code for Example 1

HTML Code Line No.	Description
HTML Source for the Static Elements	
3	HTML tag to set up page refresh cycle in seconds.
4	HTML tag to define page title label. This title appears on the browser title bar and is used in the main links page of the ECC/EGX400/PM8ECC.
13	HTML syntax to write the title of the table "CM2350 - Slave Device 3".
20	HTML syntax to write "Frequency" table cell text label.
23	HTML syntax for the input control to be filled by dynamic data.
25	HTML syntax to write "Hz".
30, 33, 35	HTML syntax for displaying Current Phase A.
40, 43, 45	HTML syntax for displaying Current Neutral.
50, 53, 55	HTML syntax for displaying Current Ground.
JavaScripting Code for the Dynamic elements	
64	This line contains the following: "PL" delimiters at the beginning and end to signify to the ECC/EGX400/PM8ECC to parse this string and dynamically fill it with register data. 3^ to signify the serial slave device address on the daisy chain. 2020,2021, …,1007 a list of register numbers, which contain necessary CM2350 data.
65	Register #2020 of the CM2350 has the Scale Factor A value.
66	Register #2021 of the CM2350 has the Scale Factor B value.
67	Register #2022 of the CM2350 has the Scale Factor C value.
68	Register #2025 of the CM2350 has the Scale Factor F value.
69	Register #1001 of the CM2350 has the Frequency value.
70	Register #1003 of the CM2350 has the Current Phase A value.
71	Register #1006 of the CM2350 has the CurrentNeutral value.
72	Register #1007 of the CM2350 has the Current Ground value.
142-145	JavaScript statements to print variable values into their field.

Once you have created the HTML page, you must transfer this page to the ECC/EGX400/PM8ECC. For details, see the ECC user's guide (63230-304-204), the EGX user's guide (63230-314-208), or the PM8ECC user's guide (63230-506-204).

EXAMPLE 2—MODIFYING AN EXISTING HTML CUSTOM PAGE

Example 2 is an exercise in modifying the Example 1 custom page to pull data from a CM4000 rather than a CM2000. For Example 2, the CM4000 has a device address of 1 on the ECC/EGX400/PM8ECC daisy chain.

Solution

In this exercise, we will keep the web page layout and colors exactly the same. However, the real-time data tags must be changed to get information from the CM4000 rather than the CM2000. To do that, open the HTML source file using any web page or text editor and make the following changes.

HTML Modification

Replace line 4:

```
<title>CM2350 - Slave Device 3</title>
```

with:

```
<title>CM4000 - Slave Device 1</title>
```

Replace line 13:

```
CM2350 - Slave Device 3</b></font></p>
```

with:

```
CM4000 - Slave Device 1</b></font></p>
```

JavaScripting Source Code Modification

To receive the dynamic data from the CM4000 rather than the CM2000, change the CM2000 device address and register numbers to the CM4000 device address and its corresponding register numbers. Make the following changes:

Replace line 64:

```
Registers =
[PL__3^2020,2021,2022,2025,1001,1003,1006,1007__PL];
```

with:

```
Registers =
[PL__1^3209,3210,3211,1180,1100,1103,1104__PL];
```

NOTE: The CM4000 and CM2000 register lists are different. The frequency parameter in the CM2000 is stored in register number 1001 and must be formatted by a scaling factor that is saved in register 2025. While the CM4000 frequency reading located in register 1180 also requires formatting, a fixed scaling factor of 0.01 is always applied and does not require a register. Therefore, make the following modification as well:

Replace line 76:

```
ScaleFactorFMultiplier = 0;
```

with:

```
ScaleFactorFMultiplier = 0.01;
```

Also, you need to delete lines 123 through 131.

Once complete, save the file. Then, transfer this page into the ECC/EGX400/PM8ECC.

CREATING A LINK TO THE HOME PAGE

If you want to have a dynamic HOME link to give users a way to get back to the main links of the ECC/EGX, you can copy this section of script and insert it at the bottom of the HTM page.

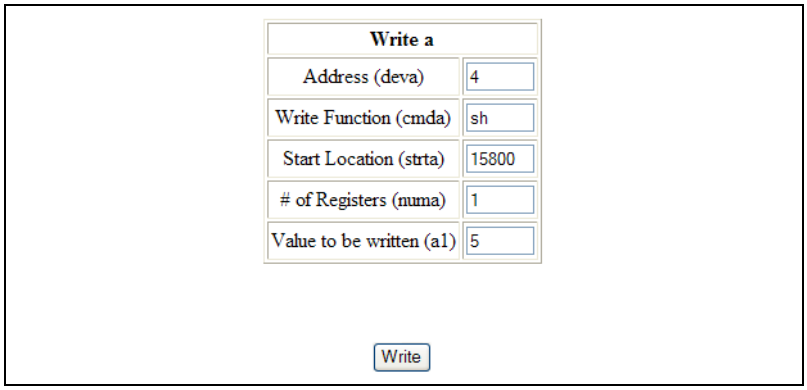
```
<script>
UrlString = document.URL;
Length = UrlString.length;
Code = UrlString.substring(Length - 4, Length);
if (Code != ".htm"){
Code = UrlString.substring(Length - 5, Length);
document.write("<center><a href='/Home.htm" + Code +
"'><fontsize='2'><b>Home</b></a></center>");
}
</script>
```

CREATING A CUSTOM PAGE TO WRITE TO SLAVE DEVICES

The simplest way to create a custom HTML page that allows you to write to slave devices is to copy and modify the sample provided in “Sample Page for Slave Device Writes” on page 12.

We recommend that you set pages capable of doing writes to “Admin-level” password access. If you are using the write capability in a remote control application, be especially careful about assigning passwords and using this feature as writing to registers changes the configuration of a device and can energize and de-energize devices. See the ECC/EGX400/PM8ECC user’s guide for more about password administration.

The sample results in the following basic page for the ECC/EGX400/ PM8ECC:



Required Format for Posting

Register writes are performed using the HTML “post” function. The form post name must equal “Post*PageName*xxxxx” where:

- *PageName* is the name of the custom page from which the form post is initiated
- xxxxx is the security token assigned during the login. The security token xxxxx will have to be dynamically generated with DHTML.

  NOTE: This only applies to the ECC/EGX400

A single custom page can contain multiple form posts as long as each form has a unique name attribute. Within each form post, there can be from one to five write functions with the following maximum number of register or coil data to be written.

Number of writes in individual form post	Number of max locations per write
1	95
2	45
3	25
4	20
5	15

Each write function must be represented by the following input variables:

	Write #1	Write #2	Write #3	Write #4	Write #5
dev*x*	“deva”	“devb”	“devc”	“devd”	“deve”
md*x*	“cmda”	“cmdb”	“cmdc”	“cmdd”	“cmde”
strt*x*	“strta”	“strtb”	“strtc”	“strtd”	“strte”
num*x*	“numa”	“numb”	“numc”	“numd”	“nume”
x#	“a1”	“b1”	“c1”	“d1”	“e1”
x#	“a2”	“b2”	“c2”	“d2”	“e2”

x#	“a3”	“b3”	“c3”	“d3”	“e3”
x#	…	…	…	…	…

Where:

- “dev*x*” is equal to the slave address of the device to be written.
- “cmd*x*” can equal any of the strings in the table below.
- “strt*x*” is equal to the start location (register or coil) to be written.
- “num*x*” is equal to the number of registers or coils to be written.
- “x#” is equal to the value or data to be written.

String	Description	Modbus Write Function
“sh”	Write Single Holding Register	0x06 Preset Single Register
“mh”	Write Multiple Holding Register(s)	0x10 Preset Multiple Registers
“sc”	Write Single Coil	0x05 Force Single Coil
“mc”	Write Multiple Coil(s)	0x0f Force Multiple Coils

Sample Page for Slave Device Writes

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<HTML><HEAD><TITLE>Example Write Register Page</TITLE>

<META http-equiv=Content-Type content="text/html; charset=windows-1252">

<META content="MSHTML 6.00.2800.1170" name=GENERATOR></HEAD>

<BODY><!--

This script creates a <FORM> tag using the URL string Copy and paste this script to the
beginning of your custom page.

-->

<SCRIPT>

/* get string from url box */

UrlString = document.URL;

/* -- create a string that contains the page name and the token (ECC/EGX400 only) -- */

/* get the length of the string */

Length = UrlString.length;

/* find the location where the page name begins */

index_string = UrlString.lastIndexOf("/", Length - 2);

/* remember the page name (including the token if it exists) */

end_string = UrlString.substring(index_string + 1,Length);

/* if this page is the result of a post the "Post" will be prepended to the beginning of the
string.  We do not want this, so delete it */

end_string = end_string.replace("Post","");

/* create the post string for the action attribute */

new_string = "Post" + end_string;

/* If there are multiple forms on a single page and the variable for the posts is accessed
outside of the forms,each form must have a unique name.  If the variables are not accessed
outside the form or if there is only oneform, this is not necessary. It is done here to show
how to use a variable to create a form name. For this pageit is hard-coded but it could be
generated at run time. */
```

```
form_name = "my_form";
/* create the string to create the form and send it */
post_string = "<FORM name='"+form_name+"' action='" + new_string + "' method='POST'
align='center'>";
document.write(post_string);
</SCRIPT>

<TABLE cellPadding=2 align=center border=0>
  <TBODY>
  <TR>
    <TD vAlign=top>
      <TABLE cellPadding=2 align=center border=1>
        <TBODY>
        <TR>
          <TD align=middle colSpan=2><B>Write a</B></TD></TR>
        <TR>
          <TD align=middle><FONT size=3>Address (deva)</FONT></TD>
          <TD align=middle><FONT size=3><INPUT size=5 value=4 name=deva>
            </FONT></TD></TR>
        <TR>
          <TD align=middle><FONT size=3>Write Function (cmda)</FONT></TD>
          <TD align=middle><FONT size=3><INPUT size=5 value=sh
            name=cmda><BR></FONT></TD></TR>
        <TR>
          <TD align=middle><FONT size=3>Start Location (strta)</FONT></TD>
          <TD align=middle><FONT size=3><INPUT size=5 value=15800 name=strta>
            </FONT></TD></TR>
        <TR>
          <TD align=middle><FONT size=3># of Registers (numa)</FONT></TD>
          <TD align=middle><FONT size=3><INPUT size=5 value=1 name=numa>
            </FONT></TD></TR>
        <TR>
          <TD align=middle><FONT size=3>Value to be written (a1)</FONT></TD>
          <TD align=middle><FONT size=3><INPUT size=5 value=5 name=a1>
          </FONT></TD></TR><!-- For four registers, change numa value to 4, and remove the
comment tags below --><!--
        <TR>
             <TD align=middle><FONT size=3>Value to be written (a2)</FONT></TD>
             <TD align=middle><FONT size=3>
             <input type=text name=a2 value=10 size=5>
```

```
            </FONT></TD>
        </TR>
        <TR>
            <TD align=middle><FONT size=3>Value to be written (a3)</FONT></TD>
            <TD align=middle><FONT size=3>
            <input type=text name=a3 value=100 size=5>
            </FONT></TD>
        </TR>
        <TR>
            <TD align=middle><FONT size=3>Value to be written (a4)</FONT></TD>
            <TD align=middle><FONT size=3>
            <input type=text name=a4 value=0 size=5>
            </FONT></TD>
        </TR>
--></TBODY></TABLE></TD></TR></TBODY></TABLE><BR><BR><BR>
<CENTER><INPUT type=submit value=Write></CENTER></FORM>
<!-- Home link for current session --><BR>
<SCRIPT>
   UrlString = document.URL;
   Length = UrlString.length;
   Code = UrlString.substring(Length - 4, Length);
   if (Code != ".htm")
   {
        Code = UrlString.substring(Length - 5, Length);
        Home = "<center><a href='/Home.htm" + Code + "'><font size='2'><b>Home</b></a></
center>";
        document.write(Home);
   }
</SCRIPT>
</BODY></HTM
```