User Manual

A1363 Samples Programmer

By Robert Bate,
Allegro MicroSystems

Setup

SETTING UP THE HARDWARE

1. Unpack the ASEK05. This should include the following:
   A. ASEK05
   B. Ethernet crossover cable
   C. Ribbon cable
   D. Power supply
   E. Daughterboard

Figure 1: Components of the ASEK05

2. Plug one end of the ribbon cable into the daughterboard and plug the other end of the ribbon cable into the ASEK05. The ribbon cable is keyed so that it can only be inserted one way.

Figure 2: Ribbon Cable Attached to the Daughter Board and the ASEK05

3. Connect the round end of the power supply cord to the 5 V Power port on the ASEK05, and connect the wall plug to a 120 V wall socket.
4. Plug one end of the Ethernet cable into the Ethernet port on the ASEK05 and plug the other end of the Ethernet cable into the computer that will be used.

Table of Contents

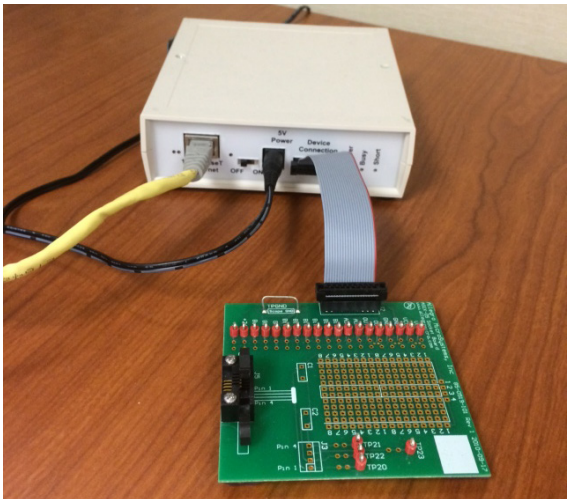

Figure 3: Ethernet Cable and Power Supply Attached to the ASEK05

5. Turn on the ASEK05.
6. After the orange "Busy" light turns off, insert the device to be programmed into the socket on the daughter board, ensuring that pin 1 aligns with the "Pin 1" marking on the board.

CAUTION!
Turning on the ASEK05 power switch while a device is inserted in the socket on the ASEK05 Daughter Board can cause damage to the device and/or the ASEK05.

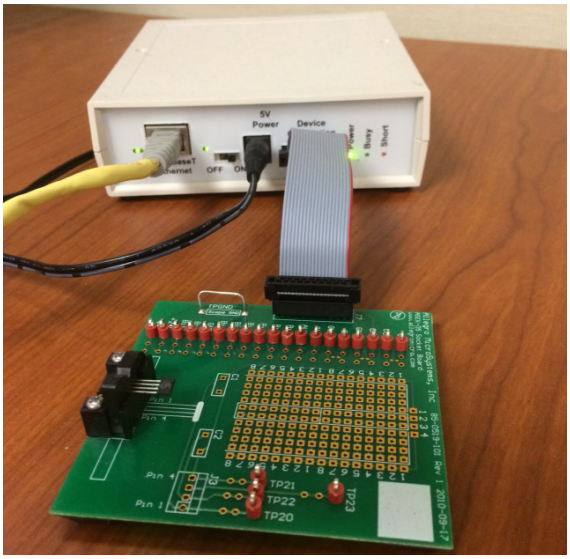

Figure 4: Powered On

ESTABLISHING COMMUNICATION WITH THE PC

A new TCP/IP network must be established on a PC to provide communication between the PC and the ASEK05 daughter board. The TCP/IP connection can be established using the crossover cable or a hub and two straight cables. The ASEK05 IP address is 192.1.2.3. The TPC IP address must be configured so that both devices are located inside of the same network.

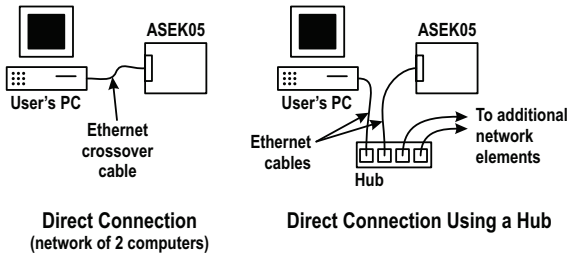

Figure 5: Options for Connecting the ASEK05 to a PC

Configure the TCP/IP address as follows:

1. On the Windows desktop, click **Start > Control Panel > View Network Status and Tasks**. The LAN or High-Speed Internet selection window will appear.
2. Right click on **Local Area Connection** selection to highlight it.
3. Click **Properties**.
4. When the Local Area Connection Properties dialog box appears, click **Internet Protocol (TCP/IP)**.
5. Click **Properties** again.

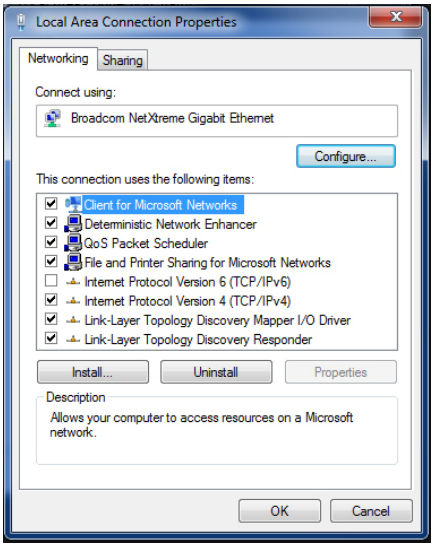

Figure 6: Finding TCP/IP Properties

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

6. When the Internet Protocols (TCP/IP) Properties dialog box appears, click the **Use the following IP address:** radio button.
7. Type **192.1.2.6** into the text entry box next to **IP address**. The DNS server address does not need to be entered.

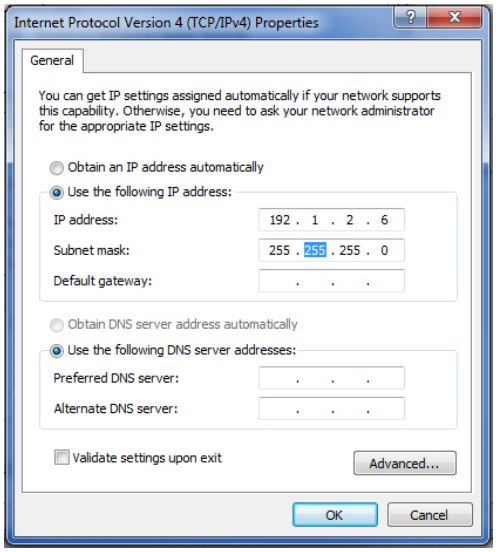

Figure 7: Entering Custom IP Addresses

8. Click **OK** to save the settings and close the Internet Protocols (TCP/IP) Properties dialog box.
9. Click **OK** to save the settings and close the Local Area Connection Properties dialog box.
10. To ensure that the connection has been properly established, ping the ASEK05 IP address as follows:
    A. Click **Start > Run** on the Windows desktop.
    B. When the Run dialog box appears, type **cmd** in the text entry box, then click **OK** to open a command window.
    C. At the DOS prompt, type **ping 192.1.2.3**, then press **Enter**.
    D. If the connection has been properly established, the time duration for the sent packages between the two devices will be displayed. If the connection has not been established, the system will display the message, "Request timed out."

SETTING UP THE SOFTWARE

Installing the Software

1. Save the zip file to the directory where you want the software to be run.
2. Unpack the zip file. A folder called Allego A1363 Samples Programmer should have been created.
3. Double-click **Allego A1363 Samples Programmer** to open it. All the files need to run should be in the folder. The application file is named Allegro A1363 Samples Programmer.exe.

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

Starting the Application

1. Double-click **Allegro A1363 Samples Programmer.exe**. The Allegro A1363 Samples Programmer window will appear (see Figure 8).

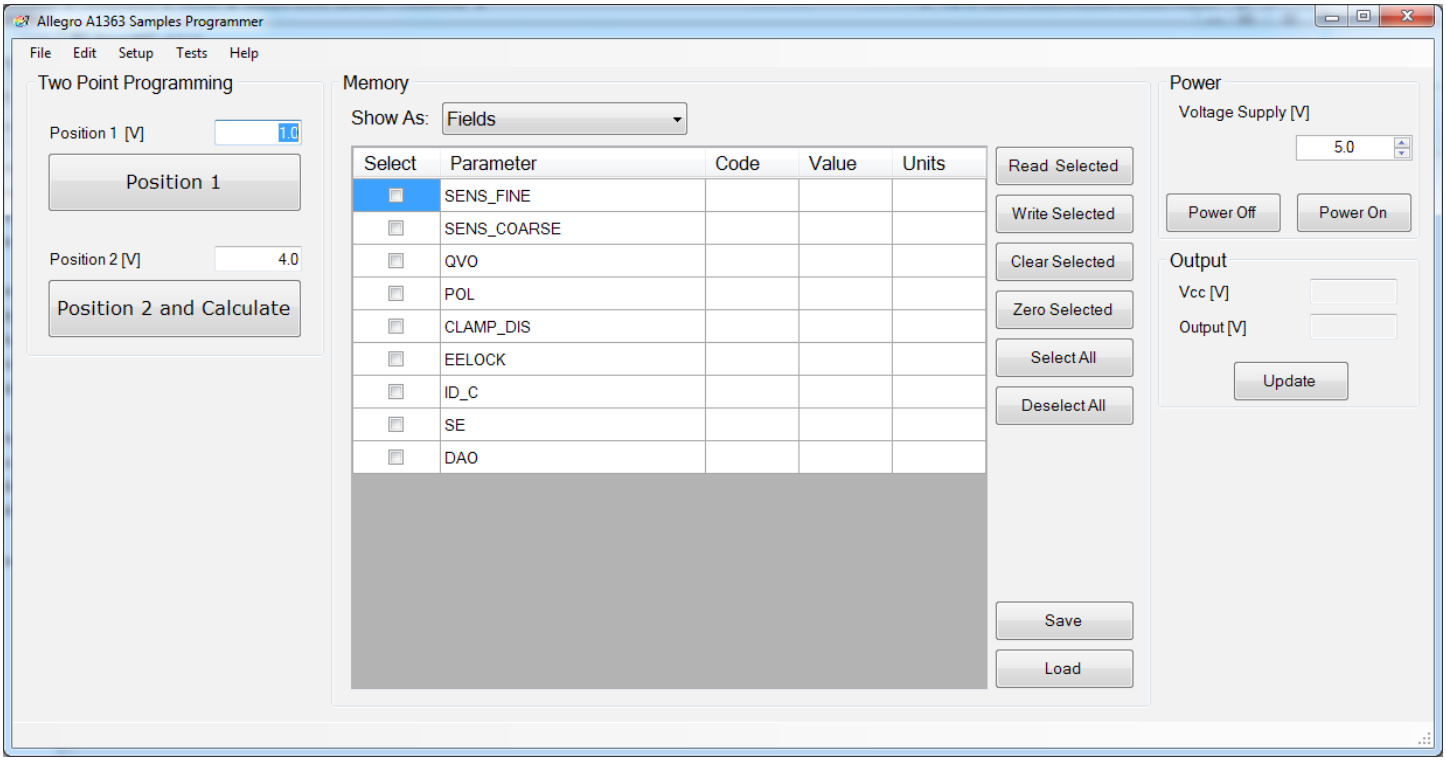

Figure 8: Allegro 1363 Samples Programmer Window

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

2. On the main window of the Allegro A1363 Samples Programmer, click **Power On**. The Vcc [V] and Output [V] fields should be populated, as shown in Figure 9.

3. If an error dialog box is received, check the connections, make sure the power is on at the ASEK05, and make sure a device is in the socket. After trying again and still receiving an error message, have IT department check the network connections.

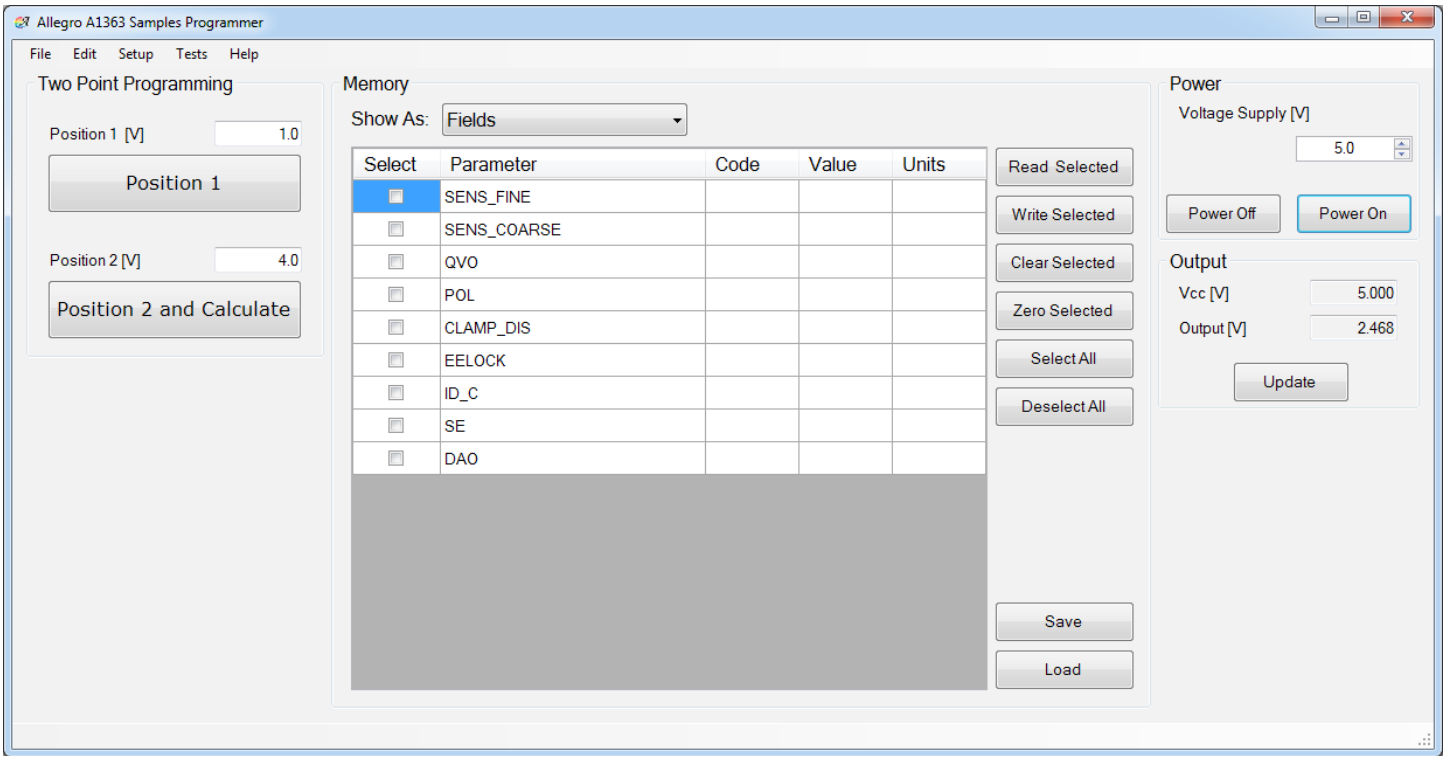

Figure 9: Power-On State

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

Reading from a Device

READING A FIELD

1. In the **Memory** field of the Allegro A1363 Samples Programmer window, select **Fields** from the **Show As:** pull-down menu.
2. Click the check box next to **ID_C**.
3. Click **Read Selected**. The window should appear as it does in Figure 10.

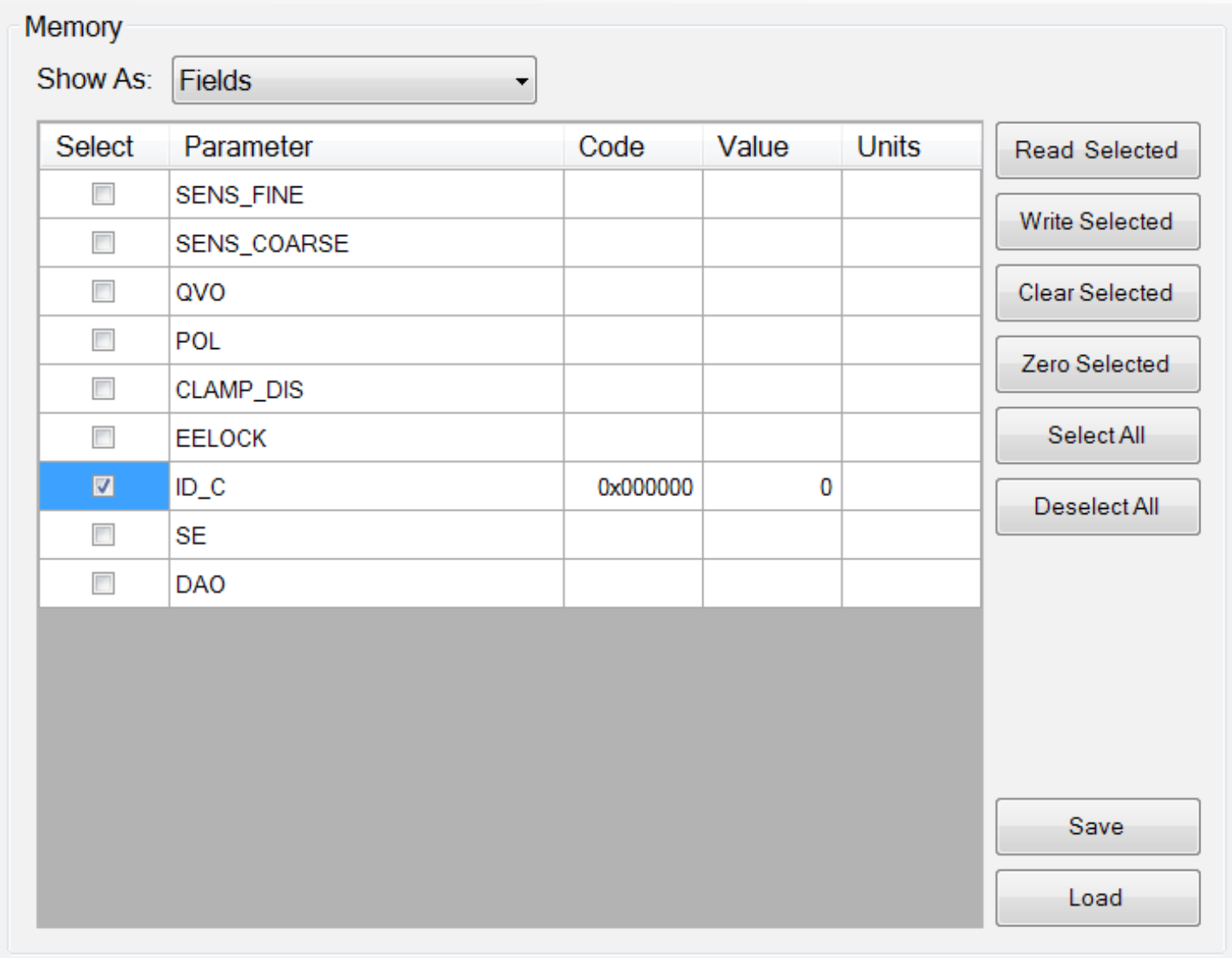

Figure 10: Reading a Field

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

READING MEMORY LOCATION

1. On the **Show As:** pull-down menu, select **Memory Location**. Click the check box next to 0x01.
2. Click **Read Selected**. The ID_C and 0x01 fields should show the same values in the Code column, as shown in Figure 11.

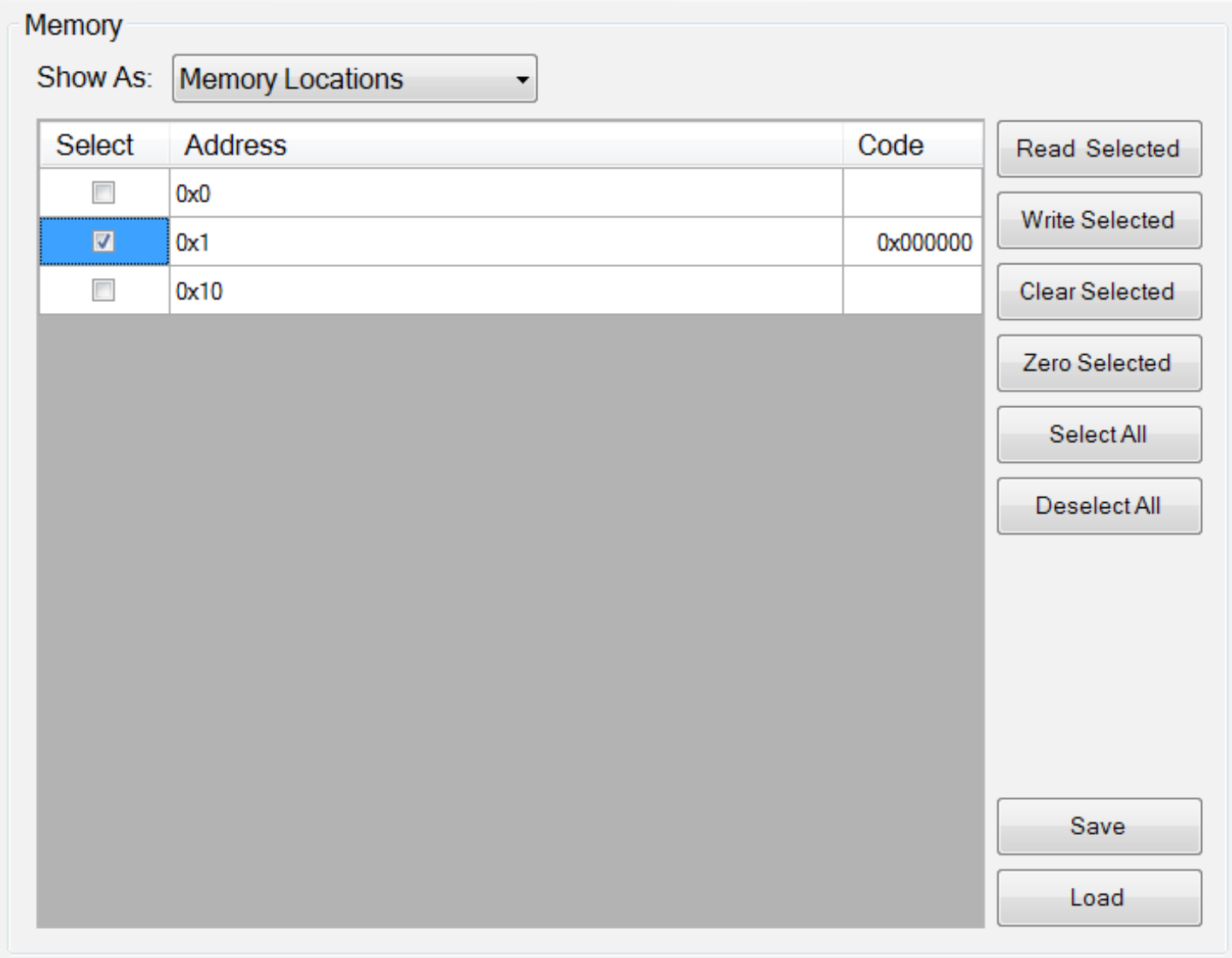

Figure 11: Reading a Memory Location

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

Writing to a Device

WRITING A FIELD

1. In the **Memory** field of the Allegro A1363 Samples Programmer window, select **Fields** from the **Show As:** pull-down menu.
2. Double-click the cell in the **Code** column of the **ID_C** row.
3. Type **0x123** in the field, then press **Enter**.
4. Click **Write Selected**. The window should appear as it does in Figure 12.
5. To make sure that the ID_C field was written to the device, click **Clear Selected**. The values in the Code and Value cells should disappear.
6. Click **Read Selected**. If the values were written properly, they should reappear in the Code and Value cells.

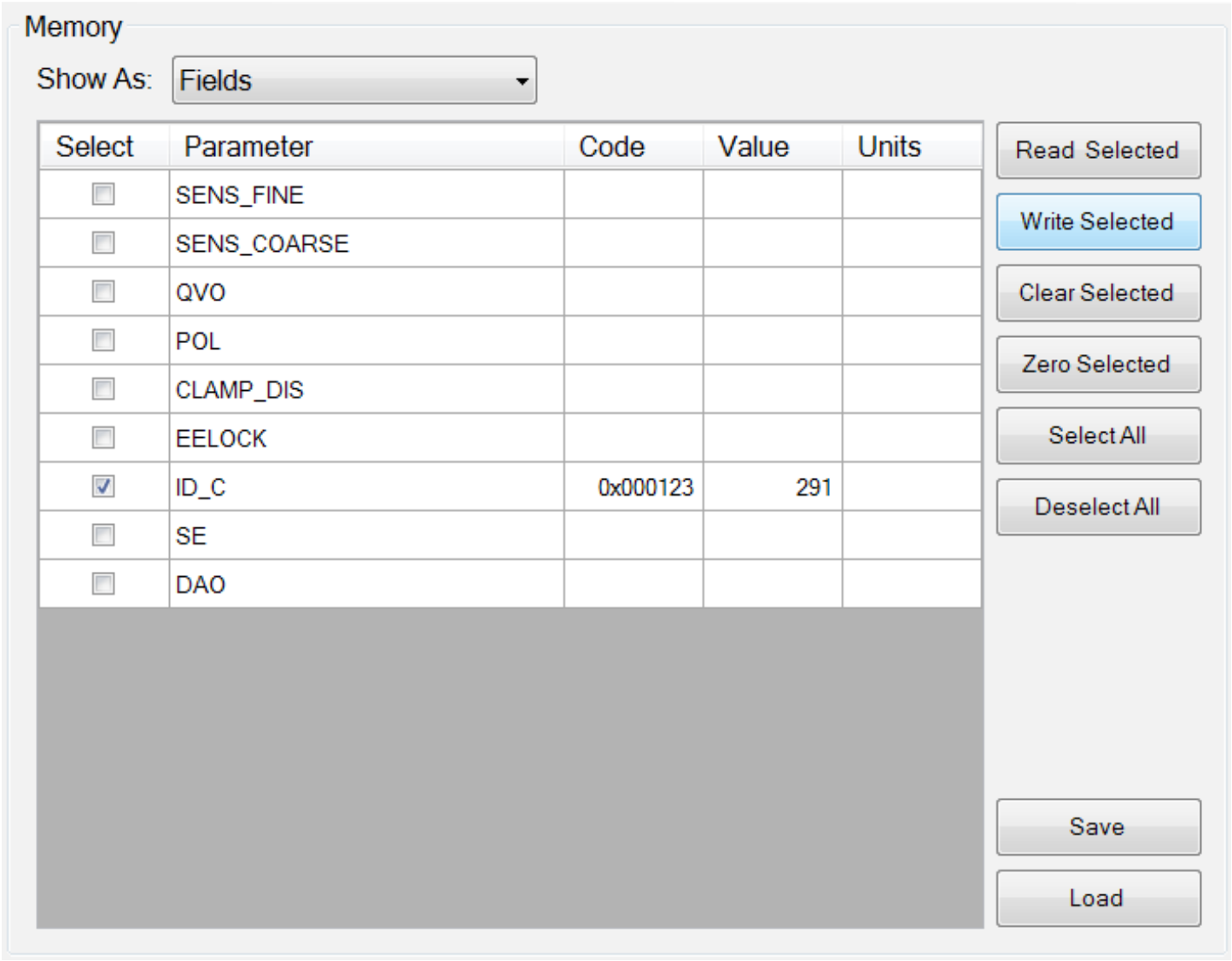

Figure 12: Writing a Field

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

WRITING A MEMORY LOCATION

1. From the **Show As:** pull-down menu, select **Memory Locations**.
2. Click **Deselect All** to clear any memory locations that may already be selected.
3. In the **Select** column, click the check box next to **0x01**
4. Click **Read Selected**.
5. In the **Code** column, double-click the field next to **0x01**.
6. Type **0x000321**, and press **Enter**.
7. Click **Write Selected**. The memory location is now changed to 0x000321.
8. To ensure that the memory was properly written, select **Fields** from the **Show As:** pulldown menu.
9. In the **Select** column, click the check box next to **ID_C**.
10. Click **Read Selected**. If the memory was written properly, it should appear in the ID_C row.

TWO-POINT PROGRAMMING

Two-point programming uses the data from two readings to calculate the desired fine offset and sensitivity multiplier values.

1. Apply a magnetic field that generates the **lowest output value**, then click **Position 1**.
2. Next, apply a magnetic field that generates the **highest output value**, then click **Position 2 and Calculate**. The application will take another series of readings and calculate the desired fine offset and the sensitivity multipliers. It will then write the values to the device.

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

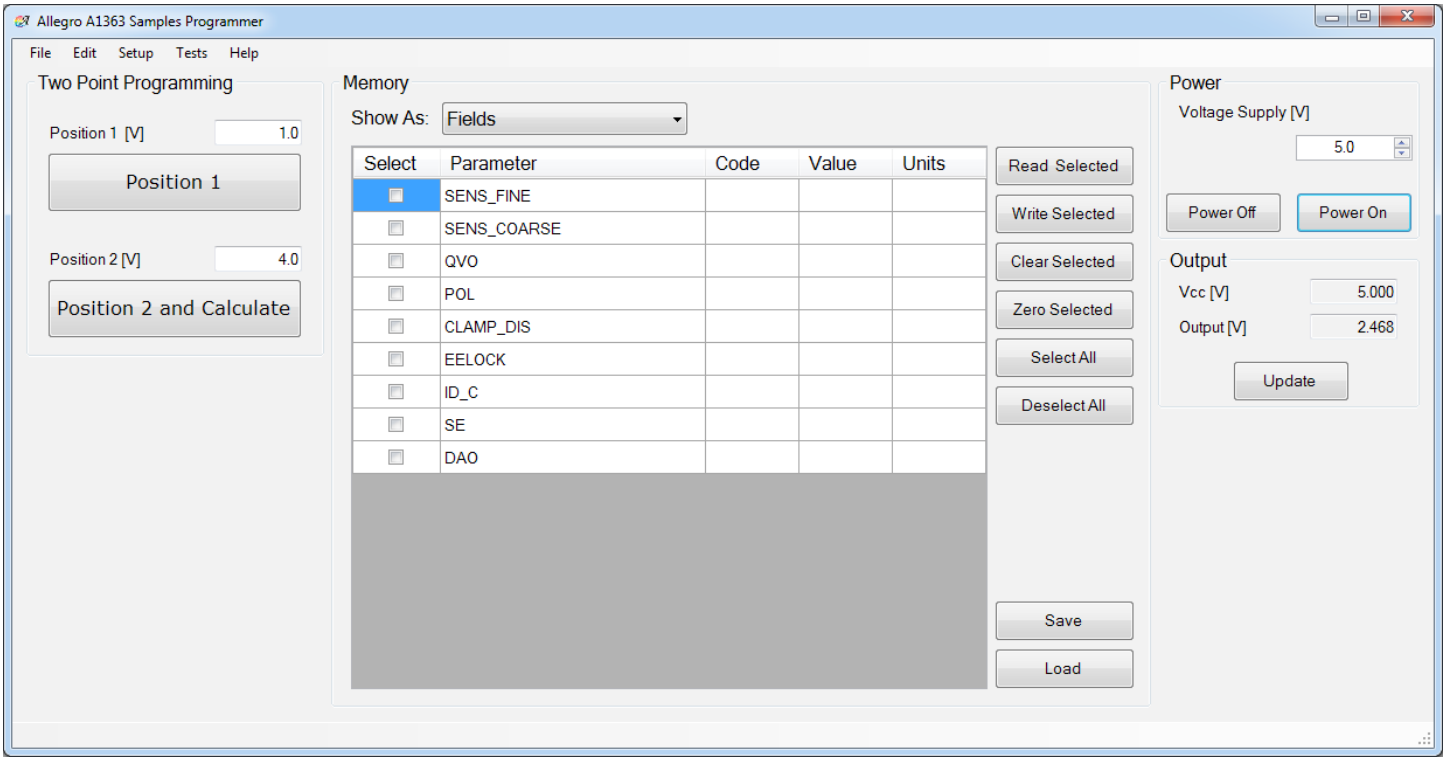

Figure 13: Main Window

MENU BAR

FILE MENU

Load Memory File

After the user selects a file from the standard file browser, this command loads the parameters into the memory grid and selects them. The files do not have to contain all of the parameters.

Save File Memory

After the user names the file to be saved using the standard file browser, this command saves the selected parameters into the file. The files do not have to contain all of the parameters.

Exit

The command exits the application.

EDIT MENU

Undo

This command undoes the last text box action. It is only enabled if the cursor is in the text box.

Cut

This command deletes selected text and copies it onto the clipboard. It is only enabled if the cursor is in the text box.

Copy

This command copies selected text onto the clipboard. It is only enabled if the cursor is in the text box.

Paste

This command pastes text from the clipboard into a highlighted text box. It is only enabled if the cursor is in a text box.

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

SETUP MENU

Communication...

This command displays the Communication Setup window. For more information, see the Communication Setup Window section.

TESTS MENU

Run Script...

This command allows the user to run scripts written in the Allegro Script Language. C#, or Visual Basic.

HELP MENU

View Help

This command opens the Help window and displays the help file for the Allegro A1363 Samples Programmer.

A1363 Spec Sheet

This command opens the web browser and downloads the A1363 datasheet from the Allegro website.

Allegro Website

This command opens the web browser and navigates to the Allegro website.

About Allegro A1363 Samples Programmer

This command displays information about the application, including the version number (refer to Figure 14).

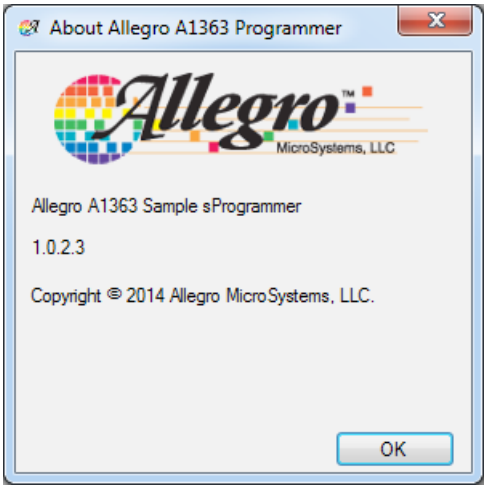

Figure 14: About Window

STATUS BAR

STATUS FIELD

This field displays the current operations that are ongoing as well as any error messages.

POWER FIELD

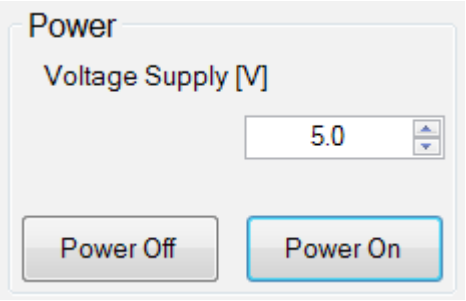

Figure 15: Power Field

Voltage Supply Text Entry Box

The Voltage Supply text entry box allows the user to set the voltage that will be supplied to the part. Any changes in value will not be used until the Power On button is pressed.

Power On Button

The Power On button is used to set the supply voltage for the device to the value set in the Voltage Supply text entry box and turn the power on. If the power is already on, it will turn the power off and then back on again.

Power Off Button

The Power Off button is used to turn off the power to the device.

Output

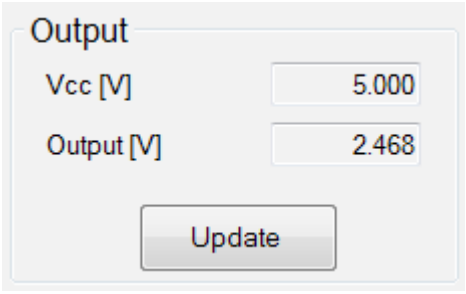

Figure 16: Output Field

Update Button

The Update button is used to read the current values for Vcc and Output from the device and display them in their respective text boxes.

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

Vcc Text Box

The Vcc text box displays the current value of the supply voltage for the device.

Output Text Box

The Output text box displays the current value of the output voltage for the device.

MEMORY FIELD

Show As Pull-Sown Menu

The Show As pulldown menu is used to select which fields and parameters will be displayed in the table.

TABLE AREA

The table area displays the fields and memory locations in a spreadsheet format.

Select Column

The Select column is used to select which rows the various function buttons will operate on. Clicking an empty check box selects the row. Clicking a checked box deselects the row.

Parameter/Address Column

The Parameter/Address column displays the name of the parameter or the memory location, depending on what is selected in the Show As: pull-down menu. Holding the cursor over a parameter displays a longer version of the name in a pop-up window.

Code Column

The code column displays the hexadecimal version of the data in the field or memory location. If the field or register is writable, double clicking on the cell will allow the user to change the value. The new value is not written to the device until the Write Selected button is clicked. Holding the cursor over the cell displays the range of values in a pop-up window.

Value Column

The value column displays a version of the data that has been interpreted in the units that are defined for the field or register. For example, the EELOCK field displays locked or unlocked. If the field or register is writable, double-clicking on the cell will allow the user to change the value. The new value is not written to the device until the Write Selected button is clicked. Holding the cursor over the cell displays the range of values in a pop-up window. This column is only displayed for fields.

Units Column

If the value column has units, then the units are displayed in this column. This column is only displayed for fields.

READ SELECTED BUTTON

Clicking this button reads all of the data in the selected rows from the device.

WRITE SELECTED BUTTON

Clicking this button writes all of the data in the selected rows to the device.

ZERO SELECTED BUTTON

Clicking this button places a zero in the Code column all of the selected rows.

CLEAR SELECTED BUTTON

Clicking this button clears all of the Code and Value columns in the selected row.

SELECT ALL BUTTON

Clicking this button selects all of the rows.

DESELECT ALL BUTTON

Clicking this button deselects all of the rows.

LOAD BUTTON

Clicking this button loads the field or memory values from a file.

SAVE BUTTON

Clicking this button saves the selected field or memory location values to a file.

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

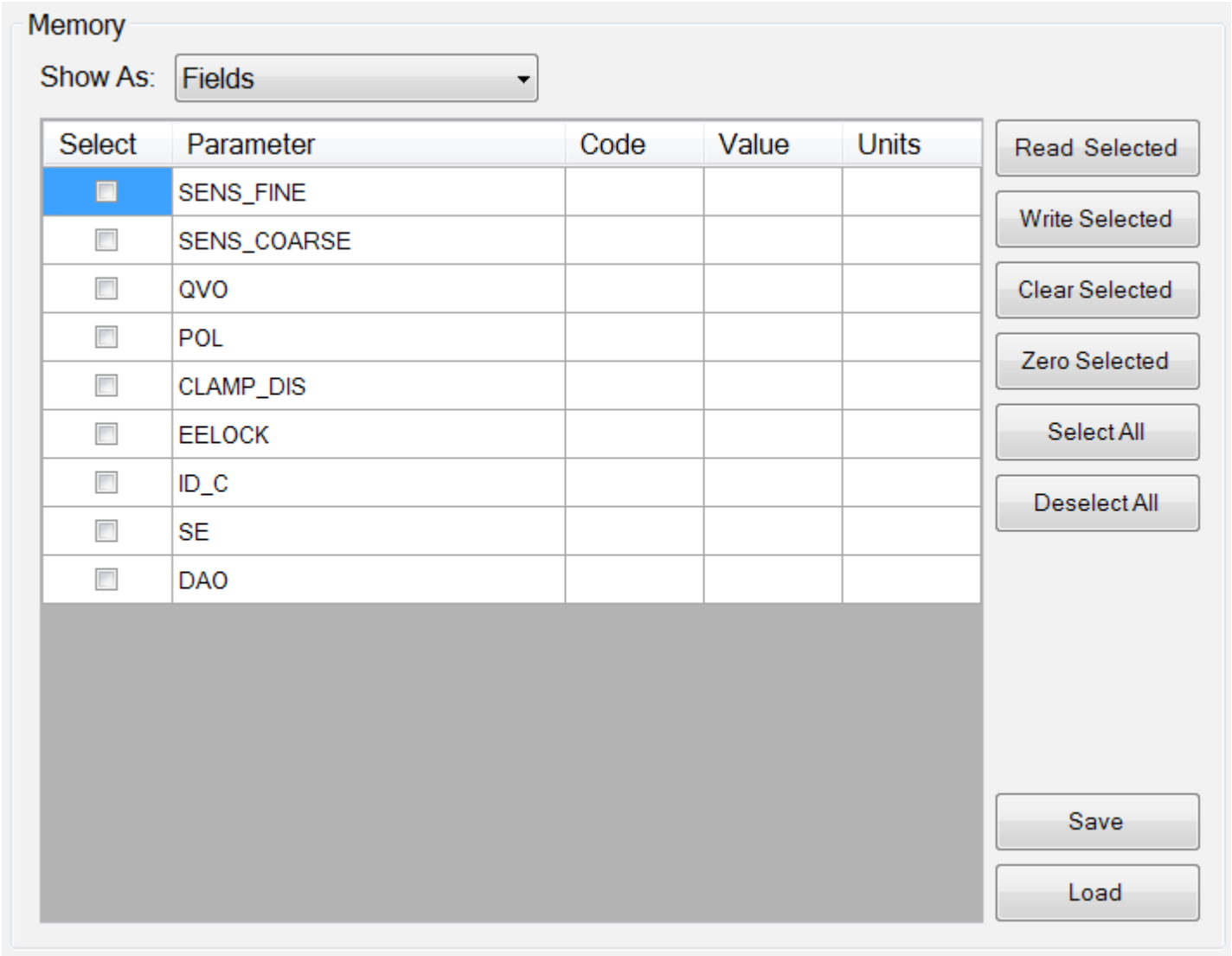

Figure 16: Memory Field

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

TWO-POINT PROGRAMMING FIELD

POSITION 1 VOLTAGE TEXT BOX

The voltage reading for position 1 is displayed in this text box.

POSITION 1 BUTTON

Once the field has been established at the desired level, clicking the Position 1 button causes a series of readings that will be saved and used when performing the two-point programming calculations.

POSITION 2 VOLTAGE TEXT BOX

The voltage reading for position 2 is displayed in this text box.

POSITION 2 AND CALCULATE BUTTON

Once the field has been established at the desired level, clicking the Position 2 and Calculate button causes a series of readings that will be used along with the readings that were performed when pressing the Position 1 button to perform the two-point programming calculations. When the calculations are done, the sensitivity and quiescent output values are written to the device.

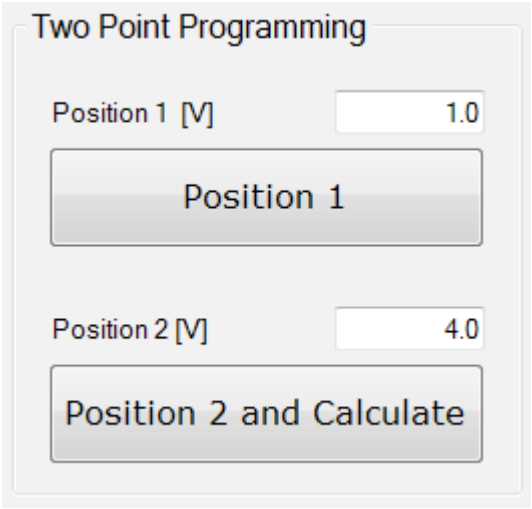

Figure 17: Two-Point Programming Field

SETUP COMMUNICATIONS DIALOG BOX

MANCHESTER FIELD

Serial Pulse High Level [V] Text Entry Box

Clicking the up and down arrows sets the voltage that will be used for the high level of the Manchester signal.

Serial Pulse Low Level [V] Text Entry Box

Clicking the up and down arrows sets the voltage that will be used for the low level of the Manchester signal.

Slew Rate [V/µs] Text Entry Box

Clicking the up and down arrows determines the speed at which the Manchester signal will travel from one voltage to another.

Speed [kbs] Text Entry Box

Clicking the up and down arrows sets the bit rate that will be used when communicating with the device.

REGISTER READ FIELD

Threshold [V] Text Entry Box

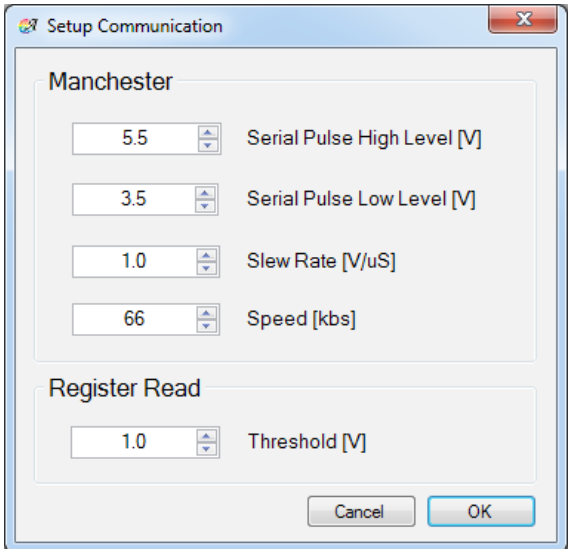

Figure 18: Setup Communications Dialog Box

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

Scripting

TEXT SCRIPTING LANGUAGE

Commands

Click

Simulate a click on a button

```
click UpdateOutput
```

Select

Simulate the selection of a combo box.

```
select ShowViewAs "Memory Locations"
```

Set

Set the value of the parameter or item.

```
set VoltageSupply=5.0
set "ID_C@EEPROM" code=1234
```

Display

Shows a message box with the given character string.

```
display "This will be displayed"
```

verify_same

Verifies that two strings are the same. If they are not the same, the script will return with a message indicating this.

```
verify_same "ID_C@EEPROM" code=0x1234
```

verify_int_eq

Verifies that two integers are equal. If they are not equal, the script will return with a message indicating this.

```
verify_int_eq Output Voltage 3
```

verify_int_ne

Verifies that two integers are not equal. If they are equal, the script will return with a message indicating this.

```
verify_int_ne Output Voltage 5
```

verify_int_lt

Verifies that left integer is less than the right integer. If it is not less, the script will return with a message indicating this.

```
verify_int_lt Output Voltage 4
```

verify_int_le

Verifies that left integer is less than or equal to the right integer. If it is not less than or equal, the script will return with a message indicating this.

```
verify_int_le Output Voltage 4
```

verify_int_gt

Verifies that left integer is greater than the right integer. If it is not greater, the script will return with a message indicating this.

```
verify_int_gt Output Voltage 4
```

verify_int_ge

Verifies that left integer is greater than or equal to the right integer. If it is not greater than or equal, the script will return with a message indicating this.

```
verify_int_ge Output Voltage 5.0
```

verify_double_eq

Verifies that two doubles are equal. If they are not equal, the script will return with a message indicating this.

```
verify_double_eq Output Voltage 5.0
```

verify_double_ne

Verifies that two doubles are not equal. If they are equal, the script will return with a message indicating this.

```
verify_double_ne Output Voltage 5.0
```

verify_double_lt

Verifies that left double is less than the right double. If it is not less, the script will return with a message indicating this.

```
verify_double_lt Output Voltage 5.0
```

verify_double_le

Verifies that left double is less than or equal to the right double. If it is not less than or equal to it, the script will return with a message indicating this.

```
verify_double_le Output Voltage 5.0
```

verify_double_gt

Verifies that left double is greater than the right double. If it is not greater, the script will return with a message indicating this.

```
verify_double_lt Output Voltage 5.0
```


Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

verify_double_ge

Verifies that left double is greater than or equal to the right double. If it is not greater than or equal to it, the script will return with a message indicating this.

```
verify_double_ge Output Voltage 5.0
```

compare_files

Compares the contents of two files. If they are not the same, the script will return with a message indicating this.

Example

```
click PowerOn
verify_double VCC 5.0
set "ID_C@EEPROM" code=4321
verify_int_eq "ID_C @ EEPROM" code=4321
```

C#

To create a script using C# that runs in the Allegro A1363 Sample Programmer, use the following template:

```
using System;
using Allegro.Script;

namespace ScriptTemplate
{
  public class ScriptTemplate: IScript-
Source
  {
    public int ScriptType()
    {
          return 0;
    }

    public string ScriptName()
    {
          return "ScriptTemplate";
    }

    public string
RunScript(IScriptPerformer performer)
{

return "This script does nothing";
    }
  }
}
```

The return value of the script type is not used at the present time so leave it as is. The script name should match the name of the file without the extension and can include spaces. RunScript is the method that will be executed when the script is run. The performer parameter is the scripts access to the application. The full C# language and all of the .Net libraries are available for use in the scripting language.

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

ISciptPerformer Methods

SetScriptVariable

Sets a string into the script variable dictionary with the given name.

```
void SetScriptVariable(string name, string value);
```

GetScriptVariable

Gets a name string from the script variable dictionary. If the name is not in the dictionary, a null is returned. The only predefined string is “scripting_path”. This string allows the script to set which file is to be used, instead of opening the file load or save dialog boxes and asking the user.

```
bool GetScriptVariable(string name, out string value);
```

SetScriptObject

Sets an object into the script object dictionary with the given name.

```
void SetScriptObject(string name, object value);
```

GetScriptObject

Gets a name object from the script object dictionary. If the name is not in the dictionary, a null is returned. Currently the only predefined object is “ASEK”, which is the ASEK programmer the application is currently using.

```
object GetScriptObject(string name);
```

PerformAction

This method performs the equivalent of a mouse click on an action keyword. There are action keywords for each of the buttons in the user interface.

```
bool PerformAction(string action);
```

PerformParameterAction

Sets or resets the select checkbox in the memory grid.

```
bool PerformParameterAction(string parameter, string action);
```

SetValue

Sets the value of the given item.

```
bool SetValue(string target, string value);
```

SetParameterValue

Sets the value of the given parameter.

```
bool SetParameterValue(string parameter, string parameterType, string value);
```

GetValue

Gets the value of the given item.

```
string GetValue(string target);
```

GetDoubleValue

Gets the value of the given item and tries to convert it into a double.

```
double GetDoubleValue(string target);
```

GetIntegerValue

Gets the value of the given item and tries to convert it into an integer.

```
int GetIntegerValue(string target);
```

GetParameterValue

Gets the string value of the given parameter.

```
string GetParameterValue(string parameter, string parameterType);
```

GetDoubleParameterValue

Gets the double value of the given parameter.

```
double GetDoubleParameterValue(string parameter, string parameterType);
```

GetIntegerParameterValue

Gets the integer value of the given parameter.

```
int GetIntegerParameterValue(string parameter, string parameterType);
```


Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

Example

```
using System;
using Allegro.ASEK;
using Allegro.Script;

namespace VerifyOperational
{
  public class VerifyOperational : IScriptSource
  {
          public int ScriptType()
          {
                return 0;
          }
          public string ScriptName()
          {
                return "VerifyOperational";
          }
          public string RunScript(IScriptPerformer performer)
    {
                ASEK05_A1363 asek05 = (ASEK05_A1363)performer.GetScriptObject("ASEK");
                if (asek05 == null)
                {
                      return "Communication not intialized.";
                }
                // Check to see if the ASEK05 is available to program with.
                if (!asek05.VerifyASEK05Present())
                {
                      return "Unable to verify the ASEK05 is connected and powered on.
Make sure it is on the local network, connected it and powered on.";
                }
                // Make sure the test starts in a known point, power off.
                if (!performer.PerformAction("PowerOff"))
                {
                      return "Unable to perform the power off action. The ASEK05 is not
responding to commands. Could be a bad ASEK05.";
                }
                // Make sure the voltage supply is really off
                if (performer.GetDoubleValue("VCC") > 0.1)
                {
                      return "Unable to verify that the supply voltage is off. Could be
a bad ASEK05.";
                }
                // Power up the device.
                if (!performer.PerformAction("PowerOn"))
                {
                      return "Unable to perform the power on action. Could be a bad
ASEK05.";
                }
                // Make sure the supply is close to the desired voltage.
```


Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

```
      double desiredVoltage = performer.GetDoubleValue("VoltageSupply");
                double voltageReading = performer.GetDoubleValue("VCC");
                if ((voltageReading < (desiredVoltage - 0.1)) || (voltageReading > (de-
siredVoltage + 0.1)))
                {
                      return string.Format("The supply voltage is not close to the de-
sired voltage. Could be a bad ASEK05. desired={0}, supply={1}", desiredVoltage, volt-
ageReading);
                }
                // Write to a scratch register to check to see if the device is present
and operational
                int readResults;
                Random randObj = new Random();
                int data = randObj.Next(0x0FFFFFF);
                try
                {
        asek05.WriteRegister(0x01, (uint)data, 0);
                }
                catch
                {
                      return "Error while writing to the A1363, verify that there is an
A1363 in the socket.\r\nIf there is one then it could be a bad A1363 or a different de-
vice.";
                }
                // Read the scratch register
                try
                {
        readResults = (int)asek05.ReadRegister(0x01);
                }
                catch
                {
                      return "Error while reading from the device, could be a bad A1363
or a different device.";
                }
                // Check to make sure the number that was written is the same as what
was read.
                if (data != readResults)
                {
                      return string.Format("Data read from the A1363 does not match what
was written, bad device. written={0}, read={1}", data, readResults);
                }
      return "The set-up is operational.";
    }
  }
}
```


Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

VISUAL BASIC

To create a script using Visual Basic that runs in the Allegro A1363 Samples programmer, use the following template:

```
Imports System
Imports Allegro.Script

Public Class ScriptTemplate
     Implements IScriptSource

     Public Function ScriptType() As Integer Implements IScriptSource.ScriptType
           Return 0
     End Function

     Public Function ScriptName() As String Implements IScriptSource.ScriptName
           Return " ScriptTemplate "
     End Function

     Public Function RunScript(performer As IScriptPerformer ) As String Implements IS-
criptSource.RunScript
           Return "The script is done."
     End Function
End Class
```

The return value of the script type is not used at the present time so leave it as is. The script name should match the name of the file without the extension and can include spaces. RunScript is the method that will be executed when the script is run. The performer parameter is the scripts access to the application. The full Visual Basic language and all of the .Net libraries are available for use in the scripting language.

IScriptPerformer Methods

SetScriptVariable

Sets a string into the script variable dictionary with the given name.

```
void SetScriptVariable(string name, string
value);
```

GetScriptVariable

Gets a name string from the script variable dictionary. If the name is not in the dictionary, a null is returned. The only predefined string is “scripting_path”. This string is how the script can set which file is to be used, instead of opening the file load or save dialogs and asking the user.

```
bool GetScriptVariable(string name, out
string value);
```

SetScriptObject

Sets an object into the script object dictionary with the given name.

```
void SetScriptObject(string name, object
value);
```

GetScriptObject

Gets a name object from the script object dictionary. If the name is not in the dictionary, a null is returned. Currently the only predefined object is “ASEK”, which is the ASEK programmer the application is currently using.

```
object GetScriptObject(string name);
```

Perform Action

Performs the equivalent of a mouse click on an action keyword. There are action keywords for each of the buttons in the user interface.

```
bool PerformAction(string action);
```


Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

PerformParameterAction

Performs parameter action which will set or reset the select checkbox in the memory grid.

```
bool PerformParameterAction(string param-
eter, string action);
```

SetValue

Sets the value of the given item.

```
bool SetValue(string target, string val-
ue);
```

SetParameterValue

Sets the value of the given parameter.

```
bool SetParameterValue(string parameter,
string parameterType, string value);
```

GetValue

Gets the value of the given item.

```
string GetValue(string target);
```

GetDoubleValue

Gets the value of the given item and tries to convert it into a double.

```
double GetDoubleValue(string target);
```

GetIntegerValue

Gets the value of the given item and tries to convert it into an integer.

```
int GetIntegerValue(string target);
```

GetParameterValue

Gets the string value of the given parameter.

```
string GetParameterValue(string parameter,
string parameterType);
```

GetDoubleParameterValue

Gets the double value of the given parameter.

```
double GetDoubleParameterValue(string pa-
rameter, string parameterType);
```

GetIntegerParameterValue

Gets the integer value of the given parameter.

```
int GetIntegerParameterValue(string param-
eter, string parameterType);
```

Example

```
Imports System
Imports Allegro.ASEK
Imports Allegro.Script

Public Class VerifyOperational
  Implements IScriptSource
  Public Function ScriptType() As Integer Implements IScriptSource.ScriptType
    Return 0
  End Function
  Public Function ScriptName() As String Implements IScriptSource.ScriptName
    Return "VerifyOperational"
  End Function
  Public Function RunScript(performer As IScriptPerformer) As String Implements IScript-
Source.RunScript
    Dim asek As ASEK05_A1363 = DirectCast(performer.GetScriptObject("ASEK"), ASEK05_
A1363)
    If asek = null Then
      Return "Communication not intialized."
    End If
    ' Check to see if the ASEK05 is available to program with.
    If asek.VerifyASEK05Present() = False Then
```


Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

```
      Return "Unable to verify the ASEK05 is connected and powered on. Make sure it is
on the local network, connected it and powered on."
    End If
    ' Make sure the test starts in a known point, power off.
    If performer.PerformAction("PowerOff") = False Then
      Return "Unable to perform the power off action. The ASEK05 is not responding to
commands. Could be a bad ASEK05."
    End If
    ' Make sure the voltage supply is really off
    If performer.GetDoubleValue("voltage_supply_reading") > 0.1 Then
      Return "Unable to verify that the supply voltage is off. Could be a bad ASEK05."
    End If
    ' Power up the device.
    If performer.PerformAction("PowerOn") = False Then
      Return "Unable to perform the power on action. Could be a bad ASEK05."
    End If
    ' Make sure the supply is close to the desired voltage.
    Dim desiredVoltage As Double = performer.GetDoubleValue("VoltageSupply")
    Dim voltageReading As Double = performer.GetDoubleValue("VCC")
    If ((voltageReading < (desiredVoltage - 0.1)) Or (voltageReading > (desiredVoltage +
0.1))) Then
      Return String.Format("The supply voltage is not close to the desired voltage.
Could be a bad ASEK05. desired={0}, supply={1}", desiredVoltage, voltageReading)
    End If
    ' Write to a scratch register to check to see if the device is present and opera-
tional
    Dim readResults As UInteger
    Dim address As UInteger = 4
    Dim randObj As Random = New Random()
    Dim data As UInteger = randObj.Next(&HFFF)
    Try
      asek.WriteRegister(1, data, 0)
    Catch ex As Exception
      Return "Error while writing to the A1363, verify that there is an A1363 in the
socket.\r\nIf there is one then it could be a bad A1363 or a different device."
    End Try
    ' Read the scratch register
    Try
      readResults = asek.ReadRegister(1)
    Catch ex As Exception
      Return "Error while reading from the device, could be a bad A1363 or a different
device."
    End Try
    ' Check to make sure the number that was written is the same as what was read.
    If data <> readResults Then
      Return String.Format("Data read from the A1363 does not match what was written,
bad device. written={0}, read={1}", data, readResults)
    End If
    Return "The set-up is operational."
  End Function
End Class
```


Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

KEYWORDS

Action Keywords

UpdateDataGrid

Makes sure the display and the values of the parameters are in sync.

LoadMemoryFile

Loads the memory grid from a file. If the script variable “scripting_path” is set, it is used instead of opening up a file browser and asking the user where the file is.

SaveMemoryFile

Saves the selected parameters in the memory grid to a file. If the script variable “scripting_path” is set, it is used instead of opening up a file browser and asking the user where the file is.

ExitApplication

Quits the application.

PowerOn

Sets the voltage supplied to the device to the value contained in VoltageSupply, then turns the supply on.

PowerOff

Turns the power off for the device.

UpdateOutput

Updates the Vcc and Output text boxes.

SelectAll

Selects all of the parameters. Note: It will select both the memory locations and the fields.

DeselectAll

Deselects all of the parameters.

ZeroSelected

Replaces the values associated with the selected parameters with zeros.

ClearSelected

Removes the values associated with the selected parameters.

ReadSelected

Reads all of the selected parameters to the A1363.

WriteSelected

Writes all of the selected parameters to the A1363. They can be a mix of memory locations and fields.

LoadFile

Loads the fields and/or memory locations from a text or csv file and selects the fields and memory locations that were loaded. If the scripting variable “scripting_path” is set, this path will be used instead of opening a file browser and asking the user for the file.

SaveFile

Saves the selected fields and/or memory locations to a text or csv. If the scripting variable “scripting_path” is set, this path will be used instead of opening a file browser and asking the user for the file.

Position 1

Performs the series of readings to establish the first position for the two-point programming.

Position 2

Performs the series of readings to establish the second position for the two-point programming and then perform the calculations.

Value Keywords

VoltageSupply

This is the desired voltage supply. It will be used when the power is turned on.

OutputVoltage

The output of the A1363 in volts. (Read Only)

VCC

VCC is the voltage that is currently being supplied to the device. (Read Only)

ShowViewAs

Selects if the fields or memory locations will be displayed. Possible values are “Memory Locations” or “Fields”.

Point1

The voltage desired for position 1.

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

Point2

The voltage desired for position 2.

PARAMETERS

The parameters are identified by the parameter name, an @, then the group name. For example, to specify the SENS_FINE field in the EEPROM, the string would be **SENS_FINE@EEPROM**.

EEPROM

0x00	QVO	EELOCK	0x10
SENS_FINE	POL	0x01	SE
SENS_COARSE	CLAMP_DIS	ID_C	DAO

File Formats

MEMORY

CSV or Comma Separated Value files use the file extension ".csv". Lines can be blank or if they start with a # then they are comments.

Each set of parameters will start with the group name. The group name is "EEPROM. The parameters are one per line with the name of the parameter, a comma, and then the value of the parameter.

Text files use the file extension ".txt" and are the same as csv files except instead of commas the parameter name and value are separated by an equal sign.

Example of CSV file:

```
# this is a comment line
EEPROM,
SENS_FINE,-246
SENS_COARSE,2
QVO,125
POL,0
CLAMP_DIS,1
EELOCK,0
```

Example of Text file

```
# this is a comment line
EEPROM
SENS_FINE=-246
SENS_COARSE=2
QVO=125
POL=0
CLAMP_DIS=1
EELOCK=0
```


Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

Programming the A1363 with the ASEK05 DLL

INITIALIZING

To initialize the programmer, create an object of the type you want to communicate with.

```
ASEK05_A1363 Device = new ASEK05_A1363();
```

POWERING UP THE DEVICE

To initialize the communication with the device and to power it up.

```
Device.InitDevice(double manchesterLow-
Voltage, double manchesterHighVoltage,
double supplyVoltage, double threshold-
Voltage, uint unlockAddress, uint unlock-
Code);
```

This will set up the ASEK05 to talk to the device.

The defaults used by the A1363 Samples Programmer application:

```
Device.InitDevice(3.5, 5.5, 5.0, 1.0,
0x24, 0x27811F77);
```

READING FROM THE DEVICE

There are three main commands to read from the device.

```
uint data = Device. ReadRegister(int ad-
dress);
```

Reads from the address given and returns a 24-bit value.

```
uint data = Device.
ReadPartialRegister(int address, int high-
Bit, int lowBit);
```

Reads from address given and extracts the bit field and right justifies it.

```
int data = Device.
ReadPartialRegisterSigned(int address, int
highBit, int lowBit);
```

Reads from address given and extracts the bit field and right justifies it and sign extends the leftmost bit of the field.

There is a special command to read the output voltage from the device.

```
double output = Device.GetOutputVoltage();
```

WRITING TO THE DEVICE

There are two main commands to write to the device.

```
Device.WriteRegister(int address, uint
data, int type);
```

Writes the data to the address. If type is 0 the data is written to the EEPROM otherwise it needs to be 1 and it will be written to the volatile register.

```
Device.WritePartialRegister(int address,
uint data, int highBit, int lowBit int
type);
```

Inserts the given data into the register. Does not modify the other bits of the register.

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

Example

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Allegro.ASEK;
namespace ASEK20_A1363Example
{
  class Program
   {
    static void Main(string[] args)
    {
      ASEK05_A1363 asek05_A1363 = new ASEK05_A1363();
      // Initialize the ASEK20
      try
      {
        asek05_A1363.InitDevice(3.5, 5.5, 5.0, 1.0, 0x24, 0x27811F77);
      }
      catch (Exception ex)
      {
        // there was an error with the command
        Console.WriteLine("Unable to Intialize the Device. message = " + ex.Message);
        return;
      }
      try
      {
        // Read the output voltage of the device
        double outputVoltage = asek05_A1363.GetOutput();
        Console.WriteLine("The output voltage = " + outputVoltage.ToString());
        // Read the QVO from the device
        int qvo = asek05_A1363.ReadPartialRegisterSigned(0x0, 19, 11);
        Console.WriteLine("QVO = " + qvo.ToString());
        // Write the customer field in the EEPROM to the device
        asek05_A1363.WriteRegister(0x01, 0x123456, 0);
        // Set the bit to reverse the output polarity
        asek05_A1363.WritePartialRegister(0x0, 1, 21, 21, 0);
        // Turn the device off
        asek05_A1363.OffVcc();
      }
      catch (Exception ex)
      {
        // there was an error with one of the commands
        Console.WriteLine("Error. message = " + ex.Message());
      }
    }
  }
}
```


Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com

Revision History

Number	Date	Description
–	July 9, 2014	Initial release
1	November 17, 2020	Minor editorial updates

Allegro MicroSystems
955 Perimeter Road
Manchester, NH 03103-3353 U.S.A.
www.allegromicro.com