How to Use Silego's Arduino Library with GreenPAK

Author: David Riedell
Date: May 29, 2016

Introduction

This app note will explain how to use Silego's Arduino Library to interact with GreenPAK devices using I2C communication. An Arduino Library allows a user to create a simpler program by offloading common functions and declarations into a separate library folder. In this case, we have created a library and a set of macros for each I2C-compatible GreenPAK device. That library is available on this App Note's website page. For more information about using I2C with GreenPAK, read **AN-1090**. This app note will assume the user has basic knowledge of the Arduino IDE, but is unfamiliar with Arduino Libraries.

Installing the Library

The first thing you need to do is download the "SilegoLibrary.zip" folder. Unzip this folder into your Arduino libraries directory. The default location for this directory is:

C:\Program Files (x86)\Arduino\libraries

Inside the "SilegoLibrary" folder you will find five items:

- Silego.h – header file that includes definitions for the library
- Silego.cpp – C++ file that includes the library code
- library.properties – includes meta information for the Arduino libraries manager
- macros directory – holds several header files which define shortcut macros to simplify code
- examples directory – includes a few example programs to help you get the hang of using the library

Once your library is in the correct directory, you will need to include it by opening up an Arduino sketch, then navigating to Sketch -> Include Library -> Manage Libraries. You should see a progress bar appear briefly at the bottom of the "Library Manager" window. Once the progress bar disappears, you can use the search field to verify that "Silego" appears among your libraries.

Using the Library

Within your Arduino sketch you need to include your header files and create an instance of the Silego class. In this case, we'll call the class "silego" with a lowercase "s".

```
// Include Silego header file
#include "Silego.h"

// Include macros for SLG46531
#include "macros/SLG46531.h"

// Create an instance of Silego
class called "silego" with device
address 0x00
Silego silego(0x00);
```

When you create the instance of the Silego class, you will need to pass it a byte-long parameter, which will give your class the device address you're working with. The device address in Silego's SLG46531V chip can be configured by editing its control code within the I2C properties sidebar as shown in Figure 1.

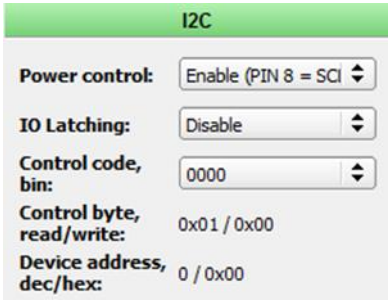

Figure 1. I2C Properties

The SLG46531V has 16 possible device addresses, shown in Table 1. For this example we will use address 0x00.

Control Code, bin:	Device address, dec:	Device address, hex:
0000	0	0x00
0001	8	0x08
0010	16	0x10
0011	24	0x18
0100	32	0x20
0101	40	0x28
0110	48	0x30
0111	56	0x38
1000	64	0x40
1001	72	0x48
1010	80	0x50
1011	88	0x58
1100	96	0x60
1101	104	0x68
1110	112	0x70
1111	120	0x78

Table 1. Silego SLG46531V Device addresses Addresses

WriteI2C

There are two functions available in version 0.0.1 of the Silego library: writeI2C and readI2C. The syntax to call one of these functions in an Arduino sketch is silego.function(parameters);. WriteI2C is defined in Silego.h with three variants:

```
void writeI2C(byte byte_address,
byte data);
void writeI2C(byte byte_address,
bool data, byte bit_location);
void writeI2C(byte byte_address,
byte data, byte bit_location, byte
length);
```

The first variant allows the user to write a byte of data to the register address given in the first parameter. The second variant allows the user to write a single bit of data to the byte address in the first parameter and the bit location given as the third parameter without altering the rest of the bits in the byte. For instance, if you wanted to write "xxx1 xxxx" to byte_address 0xCC (where "x's" are bits you don't wish to change), your command would be:

```
writeI2C(0xCC, 1, 0x10);
```

Where:

- 0xCC = the register's byte_address
- 1 = the Boolean value you wish to write
- 0x10 = the bit_location offset, since the desired bit in "xxx1 xxxx" occurs at the 2^4 bit location

The last writeI2C command allows you to write several consecutive bits to a register by adding in a "length" parameter.

If you wanted to use this function to write "x100 1xxx" to byte address 0xCC, the command would be:

```
writeI2C(0xCC, 1001, 0x08, 0x04);
```

Where:

- 0xCC = the register's byte_address
- 1001 = the value you wish to write
- 0x08 = the bit_location offset of the written value's LSB, which occurs at "xxxx 1xxx", the 2^3 bit location
- 0x04 = the length of the value you wish to write (1001 is 4 bits long)

ReadI2C

Like writeI2C, the readI2C function has multiple variants that can be called depending on how many parameters are included by the user:

```
uint8_t readI2C(byte byte_address);
bool readI2C(byte byte_address, byte bit_location);
```

The first variant only includes one parameter, the byte address you wish to read. This variant will return the value of the byte stored in the GreenPAK's register at the specified byte address. If you wish to read the value stored at address 0xCC, you would need a command that looks something like this:

```
uint8_t myVariable = readI2C(0xCC);
```

The second variant of readI2C returns a Boolean which represents the value of a single bit within a byte located at byte_address.

If the value stored at 0xCC is "1111 1011", then the following command would return 0 because the bit value at the 2^2 bit location is 0:

```
readI2C(0xCC, 0x02);
```

Macros

Back in Section 3 we had you include a file called "SLG46531.h" at the top of your Arduino sketch. If you open up that file you'll see that there are about 100 pre-defined macros to simplify your I2C commands. Each macro's value is taken from the product's datasheet which can be downloaded from the Silego website.

Example Arduino Sketch

In the code below, we have our included files at the top, followed by our instantiation of the Silego library with the device address 0x00. We then declare byte "a" and call Serial.begin(9600) to start serial data transmission at 9600 baud. Byte "a" is written to byte_address RAM_BYTE_0 using writeI2C. Then we read back the value of "a" and store it in "myData." Finally, we print out "myData" to the Arduino Serial Monitor and increment "a."

```
#include "Silego.h"              //
Include Silego header file
#include "macros/SLG46531.h"     //
Include macros for SLG46531

// Create an instance of Silego
class called
// "silego" with device address 0x00
Silego silego(0x00);

byte a = 0;

void setup() {
  Serial.begin(9600);
  Serial.print("GreenPAK: ");
  Serial.println(GreenPAK);
}

void loop() {
  // write the value of variable "a"
to byte_address RAM_BYTE_0
  silego.writeI2C(RAM_BYTE_0, a);

  // read the data in RAM_BYTE_0 and
store it in variable "myData"
  byte myData =
silego.readI2C(RAM_BYTE_0);

  Serial.print("myData: ");       //
print the value of "myData" to
  Serial.print(myData);           //
the Arduino Serial Monitor
  Serial.println();
  a++;
  delay(100);
}
```

Conclusion

In this App Note we explained how to use Silego's Arduino Library to simplify testing and prototyping with I2C-capable GreenPAK devices. Using I2C with GreenPAK allows the user to see what's going on inside the GreenPAK in real time, and also allows them to change settings and connections on the fly. This capability makes Silego's GreenPAK products extremely flexible IC's that can be used in a variety of applications.

About the Author

Name: David Riedell

Background: David attained a BS in Computer Engineering from North Carolina State University. He is currently working with CMICs as an applications engineer at Silego Technology.

Contact: **appnotes@silego.com**

Document History

Document Title: How to Use Silego's Arduino Library with GreenPAK

Document Number: AN-1107

Revision	Orig. of Change	Submission Date	Description of Change
A	David Riedell	05/29/2016	New application note

Worldwide Sales and Design Support

Silego Technology maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the sales person closest to you, visit us at **Sales Representatives and Distributors.**

About Silego Technology

Silego Technology, Inc. is a fabless semiconductor company headquartered in Santa Clara, California, with operations in Taiwan, and additional design/technology centers in China, Korea and Ukraine.

Silego Technology Inc.
1515 Wyatt Drive
Santa Clara, CA 95054

Phone: 408-327-8800
Fax: 408-988-3800
Website: www.silego.com