
ARBOR DIO

User's Guide

Version 2.1.0



Revision History

Version	Date	Descriptions
1.0.0	2020/08/14	Initial release
2.0.0	2023/10/20	Modify architecture for all in one case
2.0.1	2025/02/26	Add screenshot for install and DIO configuration tool.
2.1.0	2025/04/29	Add Test Tool description and modify description for each section.

Contents

- 1. Setting Environment.....3
- 2. SDK Index.....3
- 3. Install4
 - 3.1 Standard installation.....4
 - 3.2 Silent installation5
- 4. DIO Library6
- 5. Function7
 - 5.1 DIO Version7
 - 5.2 DIO Open.....7
 - 5.3 Initial Digital I/O Mode17
 - 5.4 Initial Digital I/O Mode27
 - 5.5 Set Digital Output Data1.....7
 - 5.6 Set Digital Output Data2.....8
 - 5.7 Get Digital Input Status18
 - 5.8 Get Digital Input Status28
 - 5.9 DIO Pin Mode8
 - 5.10 DIO Pin Output.....8
 - 5.11 DIO Pin Input8
- 6. Sample Code9
- 7. Test Tool.....9
- Appendix.....11
 - Appendix A. DIOConfigCreator11

- Figure 14
- Figure 24
- Figure 35
- Figure 45
- Figure 56
- Figure 610
- Figure 711
- Figure 811
- Figure 912

1. Setting Environment

Operating System	Windows7 above
Required SW	.Net framework 4.5.1

2. SDK Index

```
/
├─ SampleCode
├─ TestTool
├─ dll
│   ├── x64
│   └─ x86
├─ doc
│   ├── ARBOR_DIO_User_Guide.pdf
│   └─ CHANGELOG.md
└─ setup
    ├── ARBOR-DIO-Setup.exe
    ├── ARBOR-DIO-Silent-Setup.exe
    └─ DIOConfigCreator.exe
```

3. Install

The Arbor DIO SDK offers two installation methods: a standard installation and a silent installation.

3.1 Standard installation

Go to 'setup' folder and execute ARBOR-DIO-Setup.exe .

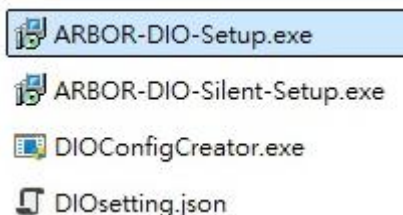


Figure 1

After the installer finishes, it will automatically launch DIOConfigCreator.exe. This tool will detect your system information and automatically match the appropriate settings. All you need to do is click the 'Apply' button and then click 'Exit'.

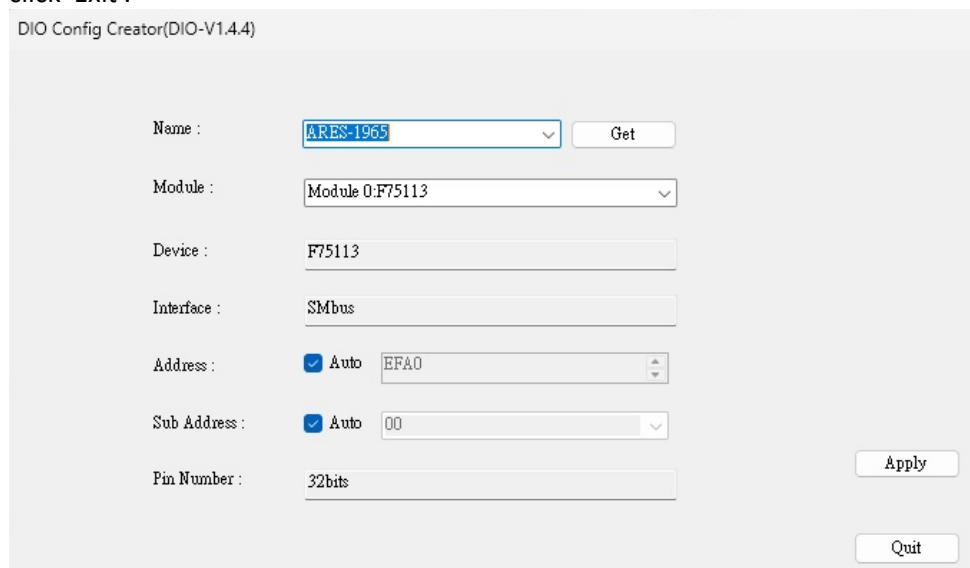


Figure 2

You will see that a 'dll' folder and a 'SampleCode' folder have been created under the 'setup' directory.

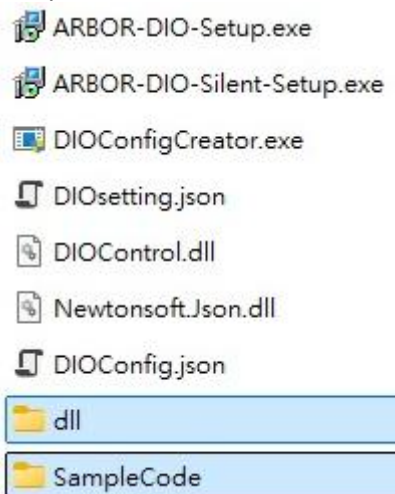


Figure 3

Please restart your computer to apply the installation settings.

3.2 Silent installation

Go to 'setup' folder and execute ARBOR-DIO-Silent-Setup.exe

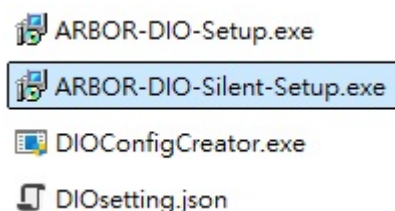


Figure 4

After the installation is complete, a 'dll' folder and a 'SampleCode' folder will be created under the 'setup' directory.

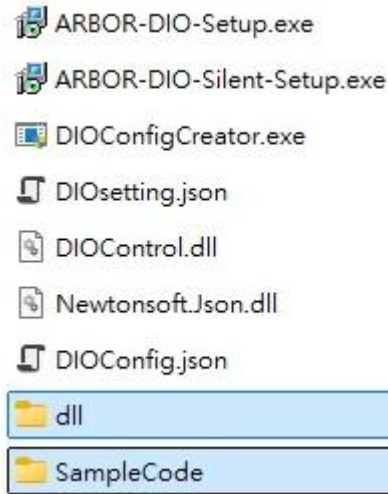


Figure 5

Please restart your computer to apply the installation settings.

4. DIO Library

This SDK provides both 32-bit and 64-bit DLLs.

```
/dll
├─ x64
│   └─ DIO64.dll
└─ x86
    └─ DIO.dll
```

For the functions available in the DLLs, please refer to Chapter 5. Developers should ensure that the DLL is placed in the same folder as their application; otherwise, the application may not run properly.

5. Function

This section describes the DIO and HWM APIs provided by the SDK. The supported APIs vary depending on the number of DIO pins in different products. For example, products with 8-pin DIO support DIO_Data1() and DIO_Status1(), while products with 16-pin DIO may also support DIO_Data2() and DIO_Status2(). For details on which products support which APIs, please refer to the sample code. Here, we only provide explanations for each API.

5.1 DIO Version

This function gets the version of the DLL file.

`char*` DIO_Ver()

Example:

```
char* sVersion = DIO_Ver();
```

5.2 DIO Open

This function enables the ARBOR-DIO functionality. The return value is true if the enable was successful.

`bool` DIO_Open()

Example:

```
bool Result =DIO_Open();
```

5.3 Initial Digital I/O Mode1

This function sets the input/output mode. The parameter "iMode" bit 0 ~ 7 mapping to pin 0 ~ 7. If the bit set to 1, then the pin is set to output mode.

`void` DIO_Mode1(int iMode)

5.4 Initial Digital I/O Mode2

This function sets the input/output mode. The parameter "iMode" bit 0 ~ 7 mapping to pin 8 ~ 15. If the bit set to 1, then the pin is set to output mode.

`void` DIO_Mode2(int iMode)

5.5 Set Digital Output Data1

This function sets the output level High/Low. The parameter "iValue" bit 0 ~ 7 mapping to pin 0 ~ 7. If the bit set to 1, then the pin is set to high level.

`void` DIO_Data1(int iValue)

5.6 Set Digital Output Data2

This function sets the output level High/Low. The parameter "iValue" bit 0 ~ 7 mapping to pin 8 ~ 15. If the bit set to 1, then the pin is set to high level.

`void DIO_Data2(int iValue)`

5.7 Get Digital Input Status1

This function gets the input level High/Low. The return value bit 0 ~ 7 mapping to pin 0 ~ 7. If the bit is "1", then the pin is high level.

`int DIO_Status1()`

5.8 Get Digital Input Status2

This function gets the input level High/Low. The return value bit 0 ~ 7 mapping to pin 8 ~ 15. If the bit is "1", then the pin is high level.

`int DIO_Status2()`

5.9 DIO Pin Mode

This function sets the input/output mode of a single pin. The parameter "index" corresponds to the pin number. The parameter "bMode" sets the input/output mode of the pin, with "0" for input and "1" for output.

`void Pin_Mode(int iIndex, bool bMode)`

5.10 DIO Pin Output

This function sets the output level of a single pin. The parameter "index" corresponds to the pin number. The parameter "bValue" sets the High/Low of the pin, with "0" for Low and "1" for High.

`void Pin_Output(int iIndex, bool bValue)`

5.11 DIO Pin Input

This function gets the level of a single pin. The parameter "index" corresponds to the pin number. The return value is the level of the pin, with "0" for low level and "1" for high level.

`int Pin_Input(int iIndex)`

6. Sample Code

This SDK provides C++ and C# sample code (Visual Studio projects).

```
/SampleCode
├─ Module0
│   ├── DIOAppCS
│   └─ DIOAppVC
├─ Module1
│   ├── DIOAppCS
│   └─ DIOAppCS
.....
```

You can find all the examples in the 'SampleCode' folder. During installation, the SDK will output the matching sample code to the 'setup' directory. The sample code demonstrates how to import the DLL and call the relevant functions.

7. Test Tool

This SDK provides a 32-bit test program to demonstrate related features.

```
/TestTool
└─ DIOTestApp.exe
```

Before using it, please make sure to follow the instructions in Chapter 3 to install the SDK on your Arbor product.

The test tool detects the DIO configuration and adjusts the UI components accordingly. For example, if the device has 8 DIO pins, the application will enable operations for DIO0 through DIO7.

This dynamic adjustment ensures that only the relevant DIO controls are displayed and accessible, providing a user interface that matches the specific hardware configuration.

Arbor-STD V1.1.0 (DLL: V1.0.0)

ARES-5310

DIO

	Mode	Output	Status		Mode	Output	Status
DIO0	<input checked="" type="radio"/> input <input type="radio"/> output	<input checked="" type="radio"/> high <input type="radio"/> low	●	DIO16	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO1	<input checked="" type="radio"/> input <input type="radio"/> output	<input checked="" type="radio"/> high <input type="radio"/> low	●	DIO17	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO2	<input checked="" type="radio"/> input <input type="radio"/> output	<input checked="" type="radio"/> high <input type="radio"/> low	●	DIO18	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO3	<input checked="" type="radio"/> input <input type="radio"/> output	<input checked="" type="radio"/> high <input type="radio"/> low	●	DIO19	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO4	<input type="radio"/> input <input checked="" type="radio"/> output	<input checked="" type="radio"/> high <input type="radio"/> low	●	DIO20	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO5	<input type="radio"/> input <input checked="" type="radio"/> output	<input checked="" type="radio"/> high <input type="radio"/> low	●	DIO21	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO6	<input type="radio"/> input <input checked="" type="radio"/> output	<input checked="" type="radio"/> high <input type="radio"/> low	●	DIO22	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO7	<input type="radio"/> input <input checked="" type="radio"/> output	<input checked="" type="radio"/> high <input type="radio"/> low	●	DIO23	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO8	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●	DIO24	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO9	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●	DIO25	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO10	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●	DIO26	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO11	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●	DIO27	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO12	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●	DIO28	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO13	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●	DIO29	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO14	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●	DIO30	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●
DIO15	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●	DIO31	<input type="radio"/> input <input type="radio"/> output	<input type="radio"/> high <input type="radio"/> low	●

Figure 6

Appendix

Appendix A. DIOConfigCreator

DIOConfigCreator.exe is a tool that can automatically detect Arbor products and export the corresponding configuration settings.

DIO Config Creator(DIO-V1.4.4)

The screenshot shows the DIO Config Creator application window. It contains several input fields and buttons. The 'Name' field is a dropdown menu showing 'ARES-5310'. The 'Module' field is a dropdown menu showing 'Module 9:F81866D-GPIO8X'. The 'Device' field is a text box containing 'F81866D'. The 'Interface' field is a text box containing 'LPC'. The 'Address' field has a checked 'Auto' checkbox and a dropdown menu showing '2E'. The 'Sub Address' field is empty. The 'Pin Number' field is a text box containing '8bits'. There are 'Get', 'Apply', and 'Quit' buttons.

Figure 7

In most cases, after launching the application, users will see that the Product Name matches the System Model under System Information.

The screenshot shows two windows. The top window is the DIO Config Creator application, with the 'Name' field highlighted by a red box, showing 'ARES-5310'. The bottom window is the Windows System Information application, with the 'System Summary' tab selected. The 'System Model' entry is highlighted by a red box, showing 'ARES-5310'.

Item	Value
OS Name	Microsoft Windows 10 Io
Version	10.0.19044 Build 19044
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	DESKTOP-SKDCA65
System Manufacturer	ARBOR
System Model	ARES-5310
System Type	x64-based PC

Figure 8

If the device is an Arbor product with a DIO interface, the tool will automatically display the correct Model setting. Users simply need to click 'Apply' and then 'Exit' in order to complete the configuration.

If users want to export configuration settings for a different model, they can manually select the desired model and then click 'Apply' and 'Exit' to finish the setup.

The screenshot shows a configuration window with the following fields and values:

- Name :** ARES-5310 (dropdown menu)
- Module :** Module 9:F81866D-GPIO8X (dropdown menu, currently open)
- Device :** (empty)
- Interface :** (empty)
- Address :** (empty)
- Sub Address :** (empty)
- Pin Number :** (empty)

The open dropdown menu for 'Module' contains the following options:

- Module 0:F75113
- Module 1:F81866D-GPIO5X-GPIO0X
- Module 2:F75111-GPIO2X-GPIO1X
- Module 3:F75111-GPIO1X-GPIO2X
- Module 4:TCA6408A
- Module 5:F75111-GPIO2X
- Module 6:F75111-GPIO1X
- Module 7:F75111-GPIO1X-Isolation
- Module 8:IT8528Ex8 (EC)
- Module 9:F81866D-GPIO8X
- Module 10:F81866D-GPIO5X
- Module 11:F81966-GPIO5X
- Module 12:F71869ED-GPIO3X
- Module 13:F81866D-GPIO5X-GPIO8X
- Module 14:F81866D-GPIO8X-GPIO7X-Isolation
- Module 15:F81966-GPIO8X
- Module 16:F81966D-GPIO8X-GPIO7X-Isolation
- Module 17:F81966Dx32-GP9GP7-GP6GP5GP0-Isolation
- Module 18:F75111x16-GP1GP2GP3
- Module 19:F81966Dx16_GP5GP6GP0

Figure 9

However, please note that if the manually selected configuration does not match the actual hardware device, the SDK may not function properly.