

# IO.Bindings.libsimpleio Namespace

Wrapper Classes for the Linux Simple I/O Library

(<http://git.munts.com/libsimpleio>)

## ↳ Classes

	Class	Description
	<a href="#">libADC</a>	Wrapper for libsimpleio A/D converter services.
	<a href="#">libDAC</a>	Wrapper for libsimpleio D/A converter services.
	<a href="#">libEvent</a>	Wrapper for libsimpleio epoll services.
	<a href="#">libGPIO</a>	Wrapper for libsimpleio GPIO services.
	<a href="#">libHIDRaw</a>	Wrapper for libsimpleio raw HID services.
	<a href="#">libI2C</a>	Wrapper for libsimpleio I <sup>2</sup> C bus controller services.
	<a href="#">libIPv4</a>	Wrapper for libsimpleio IPv4 network services.

	<a href="#">libLinux</a>	Wrapper for libsimpleio Linux system call services.
	<a href="#">libPWM</a>	Wrapper for libsimpleio PWM output services.
	<a href="#">libSerial</a>	Wrapper for libsimpleio serial port services.
	<a href="#">libSPI</a>	Wrapper for libsimpleio SPI device services.
	<a href="#">libStream</a>	Wrapper for libsimpleio Stream Framing Protocol services.
	<a href="#">libWatchdog</a>	Wrapper for libsimpleio watchdog timer services.

# Munts Technologies Linux Simple I/O Library .Net Standard 2.0 Class Library 2.2020.135.1

## libADC Class

Wrapper for libsimpleio A/D converter services.

### ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibADC](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

### ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libADC
```

The [libADC](#) type exposes the following members.

### ▪ Constructors

	Name	Description
	<a href="#">libADC</a>	Initializes a new instance of the <a href="#">libADC</a> class

[Top](#)

### ▪ Methods

	Name	Description
 	<a href="#">ADC_close</a>	Close a Linux IIO A/D converter

input device.

---

	<a href="#">ADC_get_name</a>	Get the subsystem name for the specified Linux IIO A/D converter device.
	<a href="#">ADC_open</a>	Open a Linux IIO A/D converter input device.
	<a href="#">ADC_read</a>	Read a Linux IIO A/D converter input device.

---

[Top](#)

## See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libADC Constructor

Initializes a new instance of the [libADC](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libADC()
```

## ◀ See Also

### Reference

[libADC Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libADC Methods

The [libADC](#) type exposes the following members.

## ▪ Methods

	Name	Description
 	<a href="#">ADC_close</a>	Close a Linux IIO A/D converter input device.
 	<a href="#">ADC_get_name</a>	Get the subsystem name for the specified Linux IIO A/D converter device.
 	<a href="#">ADC_open</a>	Open a Linux IIO A/D converter input device.
 	<a href="#">ADC_read</a>	Read a Linux IIO A/D converter input device.

[Top](#)

## ▪ See Also

[Reference](#)

[libADC Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libADCADC\_close Method

Close a Linux IIO A/D converter input device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void ADC_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libADC Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libADCADC\_get\_name Method

Get the subsystem name for the specified Linux IIO A/D converter device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void ADC_get_name(
    int chip,
    StringBuilder name,
    int size,
    out int error
)
```

## Parameters

*chip*

Type: [System.Int32](#)

Linux IIO device number.

*name*

Type: [System.Text.StringBuilder](#)

Destination buffer.

*size*

Type: [System.Int32](#)

Size of destination buffer.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libADC Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libADCADC\_open Method

Open a Linux IIO A/D converter input device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void ADC_open(
    int chip,
    int channel,
    out int fd,
    out int error
)
```

## Parameters

*chip*

Type: [SystemInt32](#)

Linux IIO device number.

*channel*

Type: [SystemInt32](#)

Input channel number.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ↳ See Also

[Reference](#)

[libADC Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libADCADC\_read Method

Read a Linux IIO A/D converter input device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void ADC_read(
    int fd,
    out int sample,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*sample*

Type: [SystemInt32](#)

Analog sample data.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libADC Class](#)

## IO.Bindings.libsimpleio Namespace

---

# Munts Technologies Linux Simple I/O Library .Net Standard 2.0 Class Library 2.2020.135.1

## libDAC Class

Wrapper for libsimpleio D/A converter services.

### ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibDAC](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

### ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libDAC
```

The [libDAC](#) type exposes the following members.

### ▪ Constructors

	Name	Description
	<a href="#">libDAC</a>	Initializes a new instance of the <a href="#">libDAC</a> class

[Top](#)

### ▪ Methods

	Name	Description
	<a href="#">DAC_close</a>	Close a Linux IIO D/A converter

output device.

---

≡   S	<a href="#">DAC_get_name</a>	Get the subsystem name for the specified Linux IIO D/A converter device.
≡   S	<a href="#">DAC_open</a>	Open a Linux IIO D/A converter output device.
≡   S	<a href="#">DAC_write</a>	Write to a Linux IIO D/A converter output device.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libDAC Constructor

Initializes a new instance of the [libDAC](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libDAC()
```

## ◀ See Also

**Reference**

[libDAC Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libDAC Methods

The [libDAC](#) type exposes the following members.

## Methods

	Name	Description
 	<a href="#">DAC_close</a>	Close a Linux IIO D/A converter output device.
 	<a href="#">DAC_get_name</a>	Get the subsystem name for the specified Linux IIO D/A converter device.
 	<a href="#">DAC_open</a>	Open a Linux IIO D/A converter output device.
 	<a href="#">DAC_write</a>	Write to a Linux IIO D/A converter output device.

[Top](#)

## See Also

[Reference](#)

[libDAC Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libDACDAC\_close Method

Close a Linux IIO D/A converter output device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void DAC_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libDAC Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libDACDAC\_get\_name Method

Get the subsystem name for the specified Linux IIO D/A converter device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void DAC_get_name(
    int chip,
    StringBuilder name,
    int size,
    out int error
)
```

## Parameters

*chip*

Type: [SystemInt32](#)

Linux IIO device number.

*name*

Type: [System.TextStringBuilder](#)

Destination buffer.

*size*

Type: [SystemInt32](#)

Size of destination buffer.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

**Reference**

[libDAC Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libDACDAC\_open Method

Open a Linux IIO D/A converter output device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void DAC_open(
    int chip,
    int channel,
    out int fd,
    out int error
)
```

## Parameters

*chip*

Type: [SystemInt32](#)

Linux IIO device number.

*channel*

Type: [SystemInt32](#)

Output channel number.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ↳ See Also

[Reference](#)

[libDAC Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libDACDAC\_write Method

Write to a Linux IIO D/A converter output device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void DAC_write(
    int fd,
    int sample,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*sample*

Type: [SystemInt32](#)

Analog sample data.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libDAC Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libEvent Class

Wrapper for libsimpleio epoll services.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibEvent](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libEvent
```

The [libEvent](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">libEvent</a>	Initializes a new instance of the <a href="#">libEvent</a> class

[Top](#)

## ▪ Methods

	Name	Description
≡ S	<a href="#">EVENT_close</a>	Close an <a href="#">epoll</a> event

dispatcher.

---

≡	EVENT_modify_fd	Modify a file registration.
≡	EVENT_open	Open an <code>epoll</code> event dispatcher.
≡	EVENT_register_fd	Register a file descriptor with an <code>epoll</code> event dispatcher.
≡	EVENT_unregister_fd	Unregister a file from an <code>epoll</code> dispatcher.
≡	EVENT_wait	Wait for events from an <code>epoll</code> dispatcher.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libEvent Constructor

Initializes a new instance of the [libEvent](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libEvent()
```

## ◀ See Also

### Reference

[libEvent Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libEvent Methods

The [libEvent](#) type exposes the following members.

## Methods

	Name	Description
 	<a href="#">EVENT_close</a>	Close an <code>epoll</code> event dispatcher.
 	<a href="#">EVENT_modify_fd</a>	Modify a file registration.
 	<a href="#">EVENT_open</a>	Open an <code>epoll</code> event dispatcher.
 	<a href="#">EVENT_register_fd</a>	Register a file descriptor with an <code>epoll</code> event dispatcher.
 	<a href="#">EVENT_unregister_fd</a>	Unregister a file from an <code>epoll</code> dispatcher.
 	<a href="#">EVENT_wait</a>	Wait for events from an <code>epoll</code> dispatcher.

[Top](#)

## See Also

[Reference](#)

[libEvent Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libEventEVENT\_close Method

Close an `epoll` event dispatcher.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void EVENT_close(
    int epfd,
    out int error
)
```

## Parameters

*epfd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an `errno` value upon failure.

## ◀ See Also

[Reference](#)

[libEvent Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libEventEVENT\_modify\_fd Method

Modify a file registration.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public static void EVENT_modify_fd(
    int epfd,
    int fd,
    int events,
    int handle,
    out int error
)
```

## Parameters

*epfd*

Type: [SystemInt32](#)

File descriptor for the dispatcher.

*fd*

Type: [SystemInt32](#)

File descriptor to register for events.

*events*

Type: [SystemInt32](#)

Events to register for. May be a sum of the individual event flags.

*handle*

Type: [SystemInt32](#)

Event handle.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## See Also

[Reference](#)

[libEvent Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libEventEVENT\_open Method

Open an `epoll` event dispatcher.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void EVENT_open(
    out int epfd,
    out int error
)
```

## Parameters

*epfd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an `errno` value upon failure.

## ◀ See Also

[Reference](#)

[libEvent Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libEventEVENT\_register\_fd Method

Register a file descriptor with an `epoll` event dispatcher.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public static void EVENT_register_fd(  
    int epfd,  
    int fd,  
    int events,  
    int handle,  
    out int error  
)
```

## Parameters

*epfd*

Type: [SystemInt32](#)

File descriptor for the dispatcher.

*fd*

Type: [SystemInt32](#)

File descriptor to register for events.

*events*

Type: [SystemInt32](#)

Events to register for. May be a sum of the individual event flags.

*handle*

Type: [SystemInt32](#)

Event handle.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## See Also

**Reference**

[libEvent Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libEventEVENT\_unregister\_fd Method

Unregister a file from an `epoll` dispatcher.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static void EVENT_unregister_fd(
    int epfd,
    int fd,
    out int error
)
```

## Parameters

*epfd*

Type: [SystemInt32](#)

File descriptor for the dispatcher.>

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an `errno` value upon failure.

## ◀ See Also

## Reference

[libEvent Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libEventEVENT\_wait Method

Wait for events from an `epoll` dispatcher.

**Namespace:** `IO.Bindings.libsimpleio`

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void EVENT_wait(
    int epfd,
    out int fd,
    out int events,
    out int handle,
    int timeouts,
    out int error
)
```

## Parameters

*epfd*

Type: `SystemInt32`

File descriptor for the dispatcher.

*fd*

Type: `SystemInt32`

File descriptor the event is applicable to.

*events*

Type: `SystemInt32`

Events that occurred. May be a sum of the individual event flags.

*handle*

Type: [SystemInt32](#)

Event handle provided when the file descriptor was registered.

*timeoutms*

Type: [SystemInt32](#)

Time in milliseconds to wait for an event.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

**Reference**

[libEvent Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# Munts Technologies Linux Simple I/O Library .Net Standard 2.0 Class Library 2.2020.135.1

## libGPIO Class

Wrapper for libsimpleio GPIO services.

### ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibGPIO](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

### ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libGPIO
```

The [libGPIO](#) type exposes the following members.

### ▪ Constructors

	Name	Description
	<a href="#">libGPIO</a>	Initializes a new instance of the <a href="#">libGPIO</a> class

[Top](#)

### ▪ Methods

	Name	Description
 	<a href="#">GPIO_chip_info</a>	Get GPIO chip information.

≡  	<a href="#">GPIO_close</a>	Close a Linux GPIO pin device.
≡  	<a href="#">GPIO_configure</a>	Configure a Linux GPIO pin.
≡  	<a href="#">GPIO_line_close</a>	Close a single GPIO line.
≡  	<a href="#">GPIO_line_event</a>	Read an edge trigger event from single GPIO line.
≡  	<a href="#">GPIO_line_info</a>	Get GPIO line information.
≡  	<a href="#">GPIO_line_open</a>	Open a single GPIO line.
≡  	<a href="#">GPIO_line_read</a>	Read the state of a single GPIO line.
≡  	<a href="#">GPIO_line_write</a>	Write the state of a single GPIO line.
≡  	<a href="#">GPIO_open</a>	Open a Linux GPIO pin device.
≡  	<a href="#">GPIO_read</a>	Read a Linux GPIO pin.
≡  	<a href="#">GPIO_write</a>	Write a Linux GPIO pin.

[Top](#)

## Fields

	Name	Description
♦  	<a href="#">DIRECTION_INPUT</a>	Input data direction.
♦  	<a href="#">DIRECTION_OUTPUT</a>	Out data direction.
♦  		

	<code>DRIVER_OPENDRAIN</code>	Open drain (sink only) output driver.
♦ <b>s</b>	<code>DRIVER_OPENSOURCE</code>	Open source (source only) output driver
♦ <b>s</b>	<code>DRIVER_PUSHPULL</code>	Push-pull (source and sink) output driver.
♦ <b>s</b>	<code>EDGE_BOTH</code>	Interrupt on both edges.
♦ <b>s</b>	<code>EDGE_FALLING</code>	Interrupt on falling edge.
♦ <b>s</b>	<code>EDGE_NONE</code>	Interrupts are disabled.
♦ <b>s</b>	<code>EDGE_RISING</code>	Interrupt on rising edge.
♦ <b>s</b>	<code>EVENT_REQUEST_BOTH</code>	Enable GPIO input interrupt on both edges.
♦ <b>s</b>	<code>EVENT_REQUEST_FALLING</code>	Enable GPIO input interrupt on falling edge.
♦ <b>s</b>	<code>EVENT_REQUEST_NONE</code>	Disable GPIO input interrupt.

	EVENT_REQUEST_RISING	Enable GPIO input interrupt on rising edge.
♦ S	LINE_INFO_ACTIVE_LOW	GPIO line is configured as active low (inverted).
♦ S	LINE_INFO_KERNEL	GPIO line is being used by the kernel.
♦ S	LINE_INFO_OPEN_DRAIN	GPIO line is configured as open drain (current sink only).
♦ S	LINE_INFO_OPEN_SOURCE	GPIO line is configured as open source (current source only).
♦ S	LINE_INFO_OUTPUT	GPIO line is configured as an output.
♦ S	LINE_REQUEST_ACTIVE_HIGH	Select GPIO line polarity active high (normal).
♦ S	LINE_REQUEST_ACTIVE_LOW	Select GPIO line polarity active low (inverted).

• <b>S</b>	LINE_REQUEST_INPUT	Select GPIO line direction input.
• <b>S</b>	LINE_REQUEST_OPEN_DRAIN	Select GPIO line driver open drain (current sink only).
• <b>S</b>	LINE_REQUEST_OPEN_SOURCE	Select GPIO line driver open source (current source only).
• <b>S</b>	LINE_REQUEST_OUTPUT	Select GPIO line direction output.
• <b>S</b>	LINE_REQUEST_PUSH_PULL	Select GPIO line driver push-pull (current source and sink).
• <b>S</b>	POLARITY_ACTIVEHIGH	Active high (normal) polarity.
• <b>S</b>	POLARITY_ACTIVELOW	Active low (inverted) polarity.

[Top](#)

## See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO Constructor

Initializes a new instance of the [libGPIO](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libGPIO()
```

## ◀ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO Methods

The [libGPIO](#) type exposes the following members.

## Methods

	Name	Description
 	<a href="#">GPIO_chip_info</a>	Get GPIO chip information.
 	<a href="#">GPIO_close</a>	Close a Linux GPIO pin device.
 	<a href="#">GPIO_configure</a>	Configure a Linux GPIO pin.
 	<a href="#">GPIO_line_close</a>	Close a single GPIO line.
 	<a href="#">GPIO_line_event</a>	Read an edge trigger event from single GPIO line.
 	<a href="#">GPIO_line_info</a>	Get GPIO line information.
 	<a href="#">GPIO_line_open</a>	Open a single GPIO line.
 	<a href="#">GPIO_line_read</a>	Read the state of a single GPIO line.
 	<a href="#">GPIO_line_write</a>	Write the state of a single GPIO line.
 	<a href="#">GPIO_open</a>	Open a Linux GPIO pin device.
 	<a href="#">GPIO_read</a>	Read a Linux GPIO pin.



[GPIO\\_write](#)

Write a Linux GPIO pin.

---

[Top](#)

## ◀ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

---

# libGPIOGPIO\_chip\_info Method

Get GPIO chip information.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_chip_info(
    int chip,
    StringBuilder name,
    int namesize,
    StringBuilder Label,
    int Labelsize,
    out int Lines,
    out int error
)
```

## Parameters

*chip*

Type: [SystemInt32](#)

GPIO chip number.

*name*

Type: [System.TextStringBuilder](#)

GPIO chip name.

*namesize*

Type: [SystemInt32](#)

Maximum size of name.

*label*

Type: [System.TextStringBuilder](#)

GPIO chip label.

*labelsize*

Type: [SystemInt32](#)

Maximum size of label.

*lines*

Type: [SystemInt32](#)

Number of GPIO lines.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

**Reference**

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOGPIO\_close Method

Close a Linux GPIO pin device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOGPIO\_configure Method

Configure a Linux GPIO pin.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_configure(
    int pin,
    int direction,
    int state,
    int edge,
    int polarity,
    out int error
)
```

## Parameters

*pin*

Type: [SystemInt32](#)

Pin number.

*direction*

Type: [SystemInt32](#)

Data direction.

*state*

Type: [SystemInt32](#)

Initial GPIO output state.

*edge*

Type: [SystemInt32](#)

Interrupt edge for input pin.

*polarity*

Type: [SystemInt32](#)

Polarity

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

**Reference**

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOGPIO\_line\_close Method

Close a single GPIO line.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_line_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

Linux file descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOGPIO\_line\_event Method

Read an edge trigger event from single GPIO line.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_line_event(
    int fd,
    out int state,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

Linux file descriptor.

*state*

Type: [SystemInt32](#)

State of the GPIO line after the edge trigger event.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libGPIO Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libGPIOGPIO\_line\_info Method

Get GPIO line information.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_line_info(
    int chip,
    int line,
    StringBuilder name,
    int namesize,
    StringBuilder Label,
    int labelsize,
    out int error
)
```

## Parameters

*chip*

Type: [System.Int32](#)

GPIO chip number.

*line*

Type: [System.Int32](#)

GPIO line number.

*name*

Type: [System.Text.StringBuilder](#)

GPIO line name.

*namesize*

Type: [SystemInt32](#)

Maximum size of name.

*label*

Type: [System.TextStringBuilder](#)

GPIO line label.

*labelsize*

Type: [SystemInt32](#)

Maximum size of label.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOGPIO\_line\_open Method

Open a single GPIO line.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_line_open(
    int chip,
    int line,
    int flags,
    int events,
    int state,
    out int fd,
    out int error
)
```

## Parameters

*chip*

Type: [SystemInt32](#)

GPIO chip number.

*line*

Type: [SystemInt32](#)

GPIO line number.

*flags*

Type: [SystemInt32](#)

GPIO line configuration flags.

*events*

Type: [SystemInt32](#)

GPIO line event flags.

*state*

Type: [SystemInt32](#)

Initial GPIO output state.

*fd*

Type: [SystemInt32](#)

Linux file descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOGPIO\_line\_read Method

Read the state of a single GPIO line.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_line_read(
    int fd,
    out int state,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

Linux file descriptor.

*state*

Type: [SystemInt32](#)

State of the GPIO line.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libGPIO Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libGPIOGPIO\_line\_write Method

Write the state of a single GPIO line.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_line_write(  
    int fd,  
    int state,  
    out int error  
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

Linux file descriptor.

*state*

Type: [SystemInt32](#)

State of the GPIO line.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libGPIO Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libGPIOGPIO\_open Method

Open a Linux GPIO pin device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_open(
    int pin,
    out int fd,
    out int error
)
```

## Parameters

*pin*

Type: [SystemInt32](#)

Pin number.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libGPIO Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libGPIOGPIO\_read Method

Read a Linux GPIO pin.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_read(
    int fd,
    out int state,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*state*

Type: [SystemInt32](#)

Pin state.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libGPIO Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libGPIOGPIO\_write Method

Write a Linux GPIO pin.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void GPIO_write(
    int fd,
    int state,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*state*

Type: [SystemInt32](#)

Pin state.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libGPIO Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libGPIO Fields

The [libGPIO](#) type exposes the following members.

## Fields

Name	Description
<a href="#">DIRECTION_INPUT</a>	Input data direction.
<a href="#">DIRECTION_OUTPUT</a>	Out data direction.
<a href="#">DRIVER_OPENDRAIN</a>	Open drain (sink only) output driver.
<a href="#">DRIVER_OPENSOURCE</a>	Open source (source only) output driver
<a href="#">DRIVER_PUSHPULL</a>	Push-pull (source and sink) output driver.
<a href="#">EDGE_BOTH</a>	Interrupt on both edges.
<a href="#">EDGE_FALLING</a>	Interrupt on falling edge.
<a href="#">EDGE_NONE</a>	Interrupts are

disabled.

• s	EDGE_RISING	Interrupt on rising edge.
• s	EVENT_REQUEST_BOTH	Enable GPIO input interrupt on both edges.
• s	EVENT_REQUEST_FALLING	Enable GPIO input interrupt on falling edge.
• s	EVENT_REQUEST_NONE	Disable GPIO input interrupt.
• s	EVENT_REQUEST_RISING	Enable GPIO input interrupt on rising edge.
• s	LINE_INFO_ACTIVE_LOW	GPIO line is configured as active low (inverted).
• s	LINE_INFO_KERNEL	GPIO line is being used by the kernel.
• s	LINE_INFO_OPEN_DRAIN	GPIO line is configured as open drain (current sink only).
• s	LINE_INFO_OPEN_SOURCE	GPIO line is

		configured as open source (current source only).
♦ <b>s</b>	LINE_INFO_OUTPUT	GPIO line is configured as an output.
♦ <b>s</b>	LINE_REQUEST_ACTIVE_HIGH	Select GPIO line polarity active high (normal).
♦ <b>s</b>	LINE_REQUEST_ACTIVE_LOW	Select GPIO line polarity active low (inverted).
♦ <b>s</b>	LINE_REQUEST_INPUT	Select GPIO line direction input.
♦ <b>s</b>	LINE_REQUEST_OPEN_DRAIN	Select GPIO line driver open drain (current sink only).
♦ <b>s</b>	LINE_REQUEST_OPEN_SOURCE	Select GPIO line driver open source (current source only).
♦ <b>s</b>	LINE_REQUEST_OUTPUT	Select GPIO line direction output.
♦ <b>s</b>	LINE_REQUEST_PUSH_PULL	Select GPIO line driver push-pull

(current source  
and sink).

• <b>s</b>	<a href="#">POLARITY_ACTIVEHIGH</a>	Active high (normal) polarity.
• <b>s</b>	<a href="#">POLARITY_ACTIVELOW</a>	Active low (inverted) polarity.

[Top](#)

## ◀ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIODIRECTION\_INPUT Field

Input data direction.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int DIRECTION_INPUT = 0
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIODIRECTION\_OUTPUT Field

Out data direction.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int DIRECTION_OUTPUT = 1
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIODRIVER\_OPENDRAIN

## Field

Open drain (sink only) output driver.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

### ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int DRIVER_OPENDRAIN = 1
```

### Field Value

Type: [Int32](#)

### ▪ See Also

#### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIODRIVER\_OPENSOURCE Field

Open source (source only) output driver

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int DRIVER_OPENSOURCE = 2
```

## Field Value

Type: [Int32](#)

## ▪ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO.DRIVER\_PUSH\_PULL Field

Push-pull (source and sink) output driver.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int DRIVER_PUSH_PULL = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOEDGE\_BOTH Field

Interrupt on both edges.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int EDGE_BOTH = 3
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOEDGE\_FALLING Field

Interrupt on falling edge.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int EDGE_FALLING = 2
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOEDGE\_NONE Field

Interrupts are disabled.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int EDGE_NONE = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOEDGE\_RISING Field

Interrupt on rising edge.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int EDGE_RISING = 1
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOEVENT\_REQUEST\_BOTH Field

Enable GPIO input interrupt on both edges.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int EVENT_REQUEST_BOTH = 3
```

## Field Value

Type: [Int32](#)

## ▪ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOEVENT\_REQUEST\_FALLING Field

Enable GPIO input interrupt on falling edge.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int EVENT_REQUEST_FALLING = 2
```

## Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOEVENT\_REQUEST\_NONE

## Field

Disable GPIO input interrupt.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

### ▪ Syntax

C#    VB    F#

Copy

```
public const int EVENT_REQUEST_NONE = 0
```

### Field Value

Type: [Int32](#)

### ▪ See Also

#### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOEVENT\_REQUEST\_RISING

## Field

Enable GPIO input interrupt on rising edge.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

### ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int EVENT_REQUEST_RISING = 1
```

### Field Value

Type: [Int32](#)

### ▪ See Also

#### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_INFO\_ACTIVE\_LOW Field

GPIO line is configured as active low (inverted).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int LINE_INFO_ACTIVE_LOW = 4
```

## Field Value

Type: [Int32](#)

## ▪ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_INFO\_KERNEL Field

GPIO line is being used by the kernel.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LINE_INFO_KERNEL = 1
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_INFO\_OPEN\_DRAIN Field

GPIO line is configured as open drain (current sink only).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int LINE_INFO_OPEN_DRAIN = 8
```

## Field Value

Type: [Int32](#)

## ↳ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_INFO\_OPEN\_SOURCE Field

GPIO line is configured as open source (current source only).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int LINE_INFO_OPEN_SOURCE = 16
```

## Field Value

Type: [Int32](#)

## ▪ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_INFO\_OUTPUT Field

GPIO line is configured as an output.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public const int LINE_INFO_OUTPUT = 2
```

### Field Value

Type: [Int32](#)

## ↳ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_REQUEST\_ACTIVE\_HIGH Field

Select GPIO line polarity active high (normal).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public const int LINE_REQUEST_ACTIVE_HIGH = 0
```

### Field Value

Type: [Int32](#)

## ▪ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_REQUEST\_ACTIVE\_LOW Field

Select GPIO line polarity active low (inverted).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int LINE_REQUEST_ACTIVE_LOW = 4
```

## Field Value

Type: [Int32](#)

## ▪ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_REQUEST\_INPUT Field

Select GPIO line direction input.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int LINE_REQUEST_INPUT = 1
```

## Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_REQUEST\_OPEN\_DRAIN Field

Select GPIO line driver open drain (current sink only).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

Copy

```
public const int LINE_REQUEST_OPEN_DRAIN = 8
```

### Field Value

Type: [Int32](#)

## ↳ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO.LINE\_REQUEST\_OPEN\_SOURCE Field

Select GPIO line driver open source (current source only).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LINE_REQUEST_OPEN_SOURCE = 16
```

## Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_REQUEST\_OUTPUT Field

Select GPIO line direction output.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int LINE_REQUEST_OUTPUT = 2
```

## Field Value

Type: [Int32](#)

## ▪ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO LINE\_REQUEST\_PUSH\_PULL Field

Select GPIO line driver push-pull (current source and sink).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int LINE_REQUEST_PUSH_PULL = 0
```

## Field Value

Type: [Int32](#)

## ▪ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIOPOLARITY\_ACTIVEHIGH Field

Active high (normal) polarity.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int POLARITY_ACTIVEHIGH = 1
```

## Field Value

Type: [Int32](#)

## ▪ See Also

### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libGPIO.POLARITY\_ACTIVELOW

## Field

Active low (inverted) polarity.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

### ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int POLARITY_ACTIVELOW = 0
```

### Field Value

Type: [Int32](#)

### ▪ See Also

#### Reference

[libGPIO Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libHIDRaw Class

Wrapper for libsimpleio raw HID services.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibHIDRaw](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libHIDRaw
```

The [libHIDRaw](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">libHIDRaw</a>	Initializes a new instance of the <a href="#">libHIDRaw</a> class

[Top](#)

## ▪ Methods

	Name	Description
≡ S	<a href="#">HIDRAW_close</a>	Close a Linux raw HID.

---

≡  	<a href="#">HIDRAW_get_info</a>	Get Linux raw HID bus type, vendor ID, and product ID.
≡  	<a href="#">HIDRAW_get_name</a>	Get Linux raw HID name string.
≡  	<a href="#">HIDRAW_open</a>	Open a Linux raw HID device by device node name.
≡  	<a href="#">HIDRAW_open_id</a>	Open a Linux raw HID device by vendor ID and product ID.
≡  	<a href="#">HIDRAW_receive</a>	Get a 64-byte report from a Linux HID.
≡  	<a href="#">HIDRAW_send</a>	Send a 64-byte report to a Linux HID.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libHIDRaw Constructor

Initializes a new instance of the [libHIDRaw](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libHIDRaw()
```

## ◀ See Also

### Reference

[libHIDRaw Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libHIDRaw Methods

The [libHIDRaw](#) type exposes the following members.

## ▪ Methods

	Name	Description
 	<a href="#">HIDRAW_close</a>	Close a Linux raw HID.
 	<a href="#">HIDRAW_get_info</a>	Get Linux raw HID bus type, vendor ID, and product ID.
 	<a href="#">HIDRAW_get_name</a>	Get Linux raw HID name string.
 	<a href="#">HIDRAW_open</a>	Open a Linux raw HID device by device node name.
 	<a href="#">HIDRAW_open_id</a>	Open a Linux raw HID device by vendor ID and product ID.
 	<a href="#">HIDRAW_receive</a>	Get a 64-byte report from a Linux HID.
 	<a href="#">HIDRAW_send</a>	Send a 64-byte report to a Linux HID.

[Top](#)

## ▪ See Also

## Reference

[libHIDRaw Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libHIDRawHIDRAW\_close Method

Close a Linux raw HID.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void HIDRAW_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libHIDRaw Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libHIDRawHIDRAW\_get\_info Method

Get Linux raw HID bus type, vendor ID, and product ID.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public static void HIDRAW_get_info(
    int fd,
    out int bustype,
    out int vendor,
    out int product,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*bustype*

Type: [SystemInt32](#)

Bus type.

*vendor*

Type: [SystemInt32](#)

Vendor ID.

*product*

Type: [SystemInt32](#)

Product ID.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## See Also

**Reference**

[libHIDRaw Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libHIDRawHIDRAW\_get\_name

## Method

Get Linux raw HID name string.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public static void HIDRAW_get_name(
    int fd,
    StringBuilder name,
    int size,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*name*

Type: [System.TextStringBuilder](#)

Destination buffer.

*size*

Type: [SystemInt32](#)

Size of destination buffer.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libHIDRaw Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libHIDRawHIDRAW\_open Method

Open a Linux raw HID device by device node name.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void HIDRAW_open(  
    string devname,  
    out int fd,  
    out int error  
)
```

## Parameters

*devname*

Type: [SystemString](#)

Device node name.

*fd*

Type: [SystemInt32](#)

Device node name.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libHIDRaw Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libHIDRawHIDRAW\_open\_id

## Method

Open a Linux raw HID device by vendor ID and product ID.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static void HIDRAW_open_id(
    int VID,
    int PID,
    [out] int fd,
    [out] int error
)
```

## Parameters

*VID*

Type: [SystemInt32](#)

Vendor ID.

*PID*

Type: [SystemInt32](#)

Product ID.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libHIDRaw Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libHIDRawHIDRAW\_receive Method

Get a 64-byte report from a Linux HID.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public static void HIDRAW_receive(
    int fd,
    byte[] buf,
    int bufsize,
    out int count,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*buf*

Type: [SystemByte](#)

Destination buffer.

*bufsize*

Type: [SystemInt32](#)

Destination buffer size.

*count*

Type: [SystemInt32](#)

Number of bytes actually received.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## See Also

**Reference**

[libHIDRaw Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libHIDRawHIDRAW\_send Method

Send a 64-byte report to a Linux HID.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void HIDRAW_send(
    int fd,
    byte[] buf,
    int bufsize,
    out int count,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*buf*

Type: [SystemByte](#)

Source buffer.

*bufsize*

Type: [SystemInt32](#)

Source buffer size.

*count*

Type: [SystemInt32](#)

Number of bytes actually sent.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libHIDRaw Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libI2C Class

Wrapper for libsimpleio I<sup>2</sup>C bus controller services.

## ► Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibI2C](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ► Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class libI2C
```

The [libI2C](#) type exposes the following members.

## ► Constructors

	Name	Description
≡	<a href="#">libI2C</a>	Initializes a new instance of the <a href="#">libI2C</a> class

[Top](#)

## ► Methods

	Name	Description
≡ S	<a href="#">I2C_close</a>	Close a Linux I <sup>2</sup> C bus controller

device.



## I2C\_open

Open a Linux I<sup>2</sup>C bus controller device.



## I2C\_transaction

Send bytes to and/or receive bytes from an I<sup>2</sup>C slave device.

[Top](#)

## ◀ See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libI2C Constructor

Initializes a new instance of the [libI2C](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public libI2C()
```

## ▪ See Also

**Reference**

[libI2C Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libI2C Methods

The [libI2C](#) type exposes the following members.

## ▪ Methods

	<b>Name</b>	<b>Description</b>
 	<a href="#">I2C_close</a>	Close a Linux I <sup>2</sup> C bus controller device.
 	<a href="#">I2C_open</a>	Open a Linux I <sup>2</sup> C bus controller device.
 	<a href="#">I2C_transaction</a>	Send bytes to and/or receive bytes from an I <sup>2</sup> C slave device.

[Top](#)

## ▪ See Also

### Reference

[libI2C Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libI2C\_I2C\_close Method

Close a Linux I<sup>2</sup>C bus controller device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void I2C_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libI2C Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libI2C\_I2C\_open Method

Open a Linux I<sup>2</sup>C bus controller device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void I2C_open(
    string devname,
    out int fd,
    out int error
)
```

## Parameters

*devname*

Type: [SystemString](#)

Device node name.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libI2C Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libI2CI2C\_transaction Method

Send bytes to and/or receive bytes from an I<sup>2</sup>C slave device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void I2C_transaction(  
    int fd,  
    int slaveaddr,  
    byte[] cmd,  
    int cmdLen,  
    byte[] resp,  
    int resplen,  
    out int error  
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*slaveaddr*

Type: [SystemInt32](#)

Slave address.

*cmd*

Type: [SystemByte](#)

Source buffer.

*cmdlen*

Type: [SystemInt32](#)

Source buffer size.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Response buffer size.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## See Also

[Reference](#)

[libI2C Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4 Class

Wrapper for libsimpleio IPv4 network services.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibIPV4](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libIPV4
```

The [libIPV4](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">libIPV4</a>	Initializes a new instance of the <a href="#">libIPV4</a> class

[Top](#)

## ▪ Methods

	Name	Description
≡ S	<a href="#">IPv4_ntoa</a>	Convert an IPv4 address to a

dotted notation string (e.g. 1.2.3.4).

---

≡  S	<a href="#">IPV4_resolve</a>	Resolve a domain name to an IPv4 host address.
≡  S	<a href="#">TCP4_accept</a>	Start TCP server and wait for a single connection.
≡  S	<a href="#">TCP4_close</a>	Close a TCP connection.
≡  S	<a href="#">TCP4_connect</a>	Connect to a TCP server.
≡  S	<a href="#">TCP4_receive</a>	Receive bytes from TCP peer.
≡  S	<a href="#">TCP4_send</a>	Send bytes to TCP peer.
≡  S	<a href="#">TCP4_server</a>	Start a TCP server and fork for each connection.
≡  S	<a href="#">UDP4_close</a>	Close a UDP socket.
≡  S	<a href="#">UDP4_open</a>	Open a UDP socket.
≡  S	<a href="#">UDP4_receive</a>	Receive a UDP datagram.
≡  S	<a href="#">UDP4_send</a>	Send a UDP datagram.

---

[Top](#)

## Fields

	Name	Description
♦  S	<a href="#">INADDR_ANY</a>	IPv4 address for binding to all network interfaces.
♦  S	<a href="#">INADDR_BROADCAST</a>	IPv4 broadcast address.

---

• <b>s</b>	<a href="#">INADDR_LOOPBACK</a>	IPv4 address for binding to the loopback interface (aka <a href="#">localhost</a> ).
• <b>s</b>	<a href="#">MSG_DONTROUTE</a>	Don't use a gateway to send out the packet, send to hosts only on directly connected networks.
• <b>s</b>	<a href="#">MSG_DONTWAIT</a>	Enables nonblocking operation; if the operation would block, <a href="#">EAGAIN</a> or <a href="#">EWOULDBLOCK</a> is returned.
• <b>s</b>	<a href="#">MSG_MORE</a>	The caller has more data to send. This flag informs the kernel to package all of the data sent in calls with this flag set into a single datagram which is transmitted only when a call is performed that does not specify this flag.

---

[Top](#)

## ↳ See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4 Constructor

Initializes a new instance of the [libIPV4](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libIPV4()
```

## ◀ See Also

### Reference

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPv4 Methods

The [libIPv4](#) type exposes the following members.

## Methods

	Name	Description
≡  	<a href="#">IPV4_ntoa</a>	Convert an IPv4 address to a dotted notation string (e.g. 1.2.3.4).
≡  	<a href="#">IPV4_resolve</a>	Resolve a domain name to an IPv4 host address.
≡  	<a href="#">TCP4_accept</a>	Start TCP server and wait for a single connection.
≡  	<a href="#">TCP4_close</a>	Close a TCP connection.
≡  	<a href="#">TCP4_connect</a>	Connect to a TCP server.
≡  	<a href="#">TCP4_receive</a>	Receive bytes from TCP peer.
≡  	<a href="#">TCP4_send</a>	Send bytes to TCP peer.
≡  	<a href="#">TCP4_server</a>	Start a TCP server and fork for each connection.
≡  	<a href="#">UDP4_close</a>	Close a UDP socket.
≡  	<a href="#">UDP4_open</a>	Open a UDP socket.
≡  	<a href="#">UDP4_receive</a>	Receive a UDP datagram.



[UDP4\\_send](#)

Send a UDP datagram.

[Top](#)

## ◀ See Also

**Reference**

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4IPV4\_ntoa Method

Convert an IPv4 address to a dotted notation string (e.g. 1.2.3.4).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void IPV4_ntoa(
    int host,
    StringBuilder buf,
    int bufsize,
    out int error
)
```

## Parameters

*host*

Type: [SystemInt32](#)

IPv4 host address

*buf*

Type: [System.TextStringBuilder](#)

Destination buffer.

*bufsize*

Type: [SystemInt32](#)

Destination buffer size.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ↳ See Also

**Reference**

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

---

# libIPV4IPv4\_resolve Method

Resolve a domain name to an IPv4 host address.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void IPV4_resolve(
    string hostname,
    out int host,
    out int error
)
```

## Parameters

*hostname*

Type: [SystemString](#)

Host name to resolve.

*host*

Type: [SystemInt32](#)

IPv4 host address.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libIPV4 Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libIPV4TCP4\_accept Method

Start TCP server and wait for a single connection.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void TCP4_accept(
    int host,
    int port,
    out int fd,
    out int error
)
```

## Parameters

*host*

Type: [SystemInt32](#)

IPv4 address, of the interface to listen on. Use 0.0.0.0 to listen on all interfaces.

*port*

Type: [SystemInt32](#)

TCP port number.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4TCP4\_close Method

Close a TCP connection.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void TCP4_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4TCP4\_connect Method

Connect to a TCP server.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void TCP4_connect(
    int host,
    int port,
    out int fd,
    out int error
)
```

## Parameters

*host*

Type: [SystemInt32](#)

IPv4 host address.

*port*

Type: [SystemInt32](#)

TCP port number.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ↳ See Also

**Reference**

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4TCP4\_receive Method

Receive bytes from TCP peer.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void TCP4_receive(  
    int fd,  
    byte[] buf,  
    int bufsize,  
    out int count,  
    out int error  
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*buf*

Type: [SystemByte](#)

Destination buffer.

*bufsize*

Type: [SystemInt32](#)

Destination buffer size.

*count*

Type: [SystemInt32](#)

Number of bytes actually received.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

**Reference**

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4TCP4\_send Method

Send bytes to TCP peer.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void TCP4_send(
    int fd,
    byte[] buf,
    int bufsize,
    out int count,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*buf*

Type: [SystemByte](#)

Source buffer.

*bufsize*

Type: [SystemInt32](#)

Source buffer size.

*count*

Type: [SystemInt32](#)

Number of bytes actually sent.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

**Reference**

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4TCP4\_server Method

Start a TCP server and fork for each connection.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void TCP4_server(  
    int host,  
    int port,  
    out int fd,  
    out int error  
)
```

## Parameters

*host*

Type: [SystemInt32](#)

IPv4 address, of the interface to listen on. Use 0.0.0.0 to listen on all interfaces.

*port*

Type: [SystemInt32](#)

TCP port number.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4UDP4\_close Method

Close a UDP socket.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void UDP4_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4UDP4\_open Method

Open a UDP socket.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void UDP4_open(
    int host,
    int port,
    out int fd,
    out int error
)
```

## Parameters

*host*

Type: [SystemInt32](#)

IPv4 host address.

*port*

Type: [SystemInt32](#)

UDP port number.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ↳ See Also

**Reference**

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

---

# libIPV4UDP4\_receive Method

Receive a UDP datagram.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void UDP4_receive(
    int fd,
    out int host,
    out int port,
    byte[] buf,
    int bufsize,
    int flags,
    out int count,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*host*

Type: [SystemInt32](#)

Source IPv4 host address.

*port*

Type: [SystemInt32](#)

Source UDP port number.

*buf*

Type: [SystemByte](#)

Destination buffer.

*bufsize*

Type: [SystemInt32](#)

Destination buffer size.

*flags*

Type: [SystemInt32](#)

Flags for the Linux `recvfrom()` system call.

*count*

Type: [SystemInt32](#)

Number of bytes actually received.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an `errno` value upon failure.

## See Also

**Reference**

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4UDP4\_send Method

Send a UDP datagram.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void UDP4_send(
    int fd,
    int host,
    int port,
    byte[] buf,
    int bufsize,
    int flags,
    out int count,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*host*

Type: [SystemInt32](#)

Destination IPv4 host address.

*port*

Type: [SystemInt32](#)

Destination UDP port number.

*buf*

Type: [SystemByte](#)

Source buffer.

*bufsize*

Type: [SystemInt32](#)

Source buffer size.

*flags*

Type: [SystemInt32](#)

Flags for the Linux `sendto()` system call.

*count*

Type: [SystemInt32](#)

Number of bytes actually sent.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an `errno` value upon failure.

## See Also

**Reference**

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPv4 Fields

The [libIPv4](#) type exposes the following members.

## Fields

	Name	Description
• <a href="#">S</a>	<a href="#">INADDR_ANY</a>	IPv4 address for binding to all network interfaces.
• <a href="#">S</a>	<a href="#">INADDR_BROADCAST</a>	IPv4 broadcast address.
• <a href="#">S</a>	<a href="#">INADDR_LOOPBACK</a>	IPv4 address for binding to the loopback interface (aka <a href="#">localhost</a> ).
• <a href="#">S</a>	<a href="#">MSG_DONTROUTE</a>	Don't use a gateway to send out the packet, send to hosts only on directly connected networks.
• <a href="#">S</a>	<a href="#">MSG_DONTWAIT</a>	Enables nonblocking operation; if the operation would block, <a href="#">EAGAIN</a> or <a href="#">EWOULDBLOCK</a> is returned.
• <a href="#">S</a>	<a href="#">MSG_MORE</a>	The caller has more data to send. This flag informs the kernel to package all of the data sent in calls with this

flag set into a single datagram which is transmitted only when a call is performed that does not specify this flag.

---

[Top](#)

## ◀ See Also

**Reference**

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4INADDR\_ANY Field

IPv4 address for binding to all network interfaces.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int INADDR_ANY = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4INADDR\_BROADCAST Field

IPv4 broadcast address.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int INADDR_BROADCAST = -1
```

## Field Value

Type: [Int32](#)

## ▪ See Also

### Reference

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4INADDR\_LOOPBACK Field

IPv4 address for binding to the loopback interface (aka `localhost`).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int INADDR_LOOPBACK = 2130706433
```

## Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4MSG\_DONTROUTE Field

Don't use a gateway to send out the packet, send to hosts only on directly connected networks.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MSG_DONTROUTE = 4
```

## Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4MSG\_DONTWAIT Field

Enables nonblocking operation; if the operation would block, [EAGAIN](#) or [EWOULDBLOCK](#) is returned.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MSG_DONTWAIT = 64
```

## Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libIPV4MSG\_MORE Field

The caller has more data to send. This flag informs the kernel to package all of the data sent in calls with this flag set into a single datagram which is transmitted only when a call is performed that does not specify this flag.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public const int MSG_MORE = 32768
```

## Field Value

Type: [Int32](#)

## ↳ See Also

### Reference

[libIPV4 Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinux Class

Wrapper for libsimpleio Linux system call services.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibLinux](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class libLinux
```

The [libLinux](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">libLinux</a>	Initializes a new instance of the <a href="#">libLinux</a> class

[Top](#)

## ▪ Methods

	Name	Description
	<a href="#">LINUX_command</a>	Execute a shell command

string.

---

≡  	LINUX_detach	Detach the process and run it in the background.
≡  	LINUX_drop_privileges	Drop process privileges to those of the specified user.
<h2>▲ Remarks</h2>		
		Only a process running at superuser privilege is allowed to drop privileges.
≡  	LINUX_errno	Fetch the value of <code>errno</code> .
≡  	LINUX_openlog	Open a connection to the <code>syslog</code> service.
≡  	LINUX_poll	Wait for an event on one or more files.
≡  	LINUX_strerror	Retrieve the error message for a particular <code>errno</code> error code.
≡  	LINUX_syslog	Send a message to the <code>syslog</code> service.
≡  	LINUX_usleep	Sleep for the specified number of microseconds.

---

[Top](#)

## ▲ Fields

	<b>Name</b>	<b>Description</b>
♦ s	<a href="#">LOG_ALERT</a>	Action must be taken immediately.
♦ s	<a href="#">LOG_AUTH</a>	Security/authorization messages.
♦ s	<a href="#">LOG_AUTHPRIV</a>	Securit/authorization messages.
♦ s	<a href="#">LOG_CRIT</a>	Critical condition.
♦ s	<a href="#">LOG_CRON</a>	<code>cron</code> daemon messages.
♦ s	<a href="#">LOG_DAEMON</a>	System daemons.
♦ s	<a href="#">LOG_DEBUG</a>	Debug message.
♦ s	<a href="#">LOG_EMERG</a>	System is unusable.
♦ s	<a href="#">LOG_ERR</a>	Error condition.
♦ s	<a href="#">LOG_FTP</a>	<code>FTP</code> daemon messages.
♦ s	<a href="#">LOG_INFO</a>	Informational message.
♦ s	<a href="#">LOG_KERN</a>	Kernel messages.
♦ s	<a href="#">LOG_LOCAL0</a>	Reserved for local use.
♦ s	<a href="#">LOG_LOCAL1</a>	Reserved for local use.
♦ s	<a href="#">LOG_LOCAL2</a>	Reserved for local use.
♦ s	<a href="#">LOG_LOCAL3</a>	Reserved for local use.
♦ s	<a href="#">LOG_LOCAL4</a>	Reserved for local use.

♦ <b>s</b>	<a href="#">LOG_LOCAL5</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LOCAL6</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LOCAL7</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LPR</a>	Line printer subsystem
♦ <b>s</b>	<a href="#">LOG_MAIL</a>	Mail system.
♦ <b>s</b>	<a href="#">LOG_NEWS</a>	Network news subsystem
♦ <b>s</b>	<a href="#">LOG_NOTICE</a>	Normal but significant condition.
♦ <b>s</b>	<a href="#">LOG_PROGNAME</a>	Use the program name for the identity string.
♦ <b>s</b>	<a href="#">LOG_SYSLOG</a>	Messages generated internally by syslogd
♦ <b>s</b>	<a href="#">LOG_USER</a>	Random user-level messages.
♦ <b>s</b>	<a href="#">LOG_UUCP</a>	UUCP subsystem
♦ <b>s</b>	<a href="#">LOG_WARNING</a>	Warning condition.
♦ <b>s</b>	<a href="#">POLLERR</a>	An error occurred.
♦ <b>s</b>	<a href="#">POLLHUP</a>	Peer closed connection.
♦ <b>s</b>	<a href="#">POLLIN</a>	There is data to read.
♦ <b>s</b>	<a href="#">POLLNVAL</a>	File descriptor is invalid.
♦ <b>s</b>	<a href="#">POLLOUT</a>	Writing is now possible.



POLLPRI

There is urgent data to read.

---

[Top](#)

## ◀ See Also

**Reference**

[IO.Bindings.libsimpleio Namespace](#)

---

# libLinux Constructor

Initializes a new instance of the [libLinux](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libLinux()
```

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinux Methods

The [libLinux](#) type exposes the following members.

## Methods

Name	Description
   <a href="#">LINUX_command</a>	Execute a shell command string.
   <a href="#">LINUX_detach</a>	Detach the process and run it in the background.
   <a href="#">LINUX_drop_privileges</a>	Drop process privileges to those of the specified user.
	Only a process running at superuser privilege is allowed to drop privileges.
   <a href="#">LINUX_errno</a>	Fetch the value of <a href="#">errno</a> .
   <a href="#">LINUX_openlog</a>	Open a connection to the <a href="#">syslog</a> service.
   <a href="#">LINUX_poll</a>	Wait for an event on one or more files.

	<a href="#">LINUX_strerror</a>	Retrieve the error message for a particular <code>errno</code> error code.
 <b>S</b>	<a href="#">LINUX_syslog</a>	Send a message to the <code>syslog</code> service.
 <b>S</b>	<a href="#">LINUX_usleep</a>	Sleep for the specified number of microseconds.

[Top](#)

## ◀ See Also

**Reference**

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLINUX\_command Method

Execute a shell command string.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void LINUX_command(  
    string cmd,  
    out int error  
)
```

## Parameters

*cmd*

Type: [SystemString](#)

Command string.

*error*

Type: [SystemInt32](#)

Error code.

## ◀ See Also

[Reference](#)

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLINUX\_detach Method

Detach the process and run it in the background.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void LINUX_detach(  
    out int error  
)
```

## Parameters

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLINUX\_drop\_privileges Method

Drop process privileges to those of the specified user.

## Remarks

Only a process running at superuser privilege is allowed to drop privileges.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## Syntax

C#    VB    F#

[Copy](#)

```
public static void LINUX_drop_privileges(
    string username,
    out int error
)
```

## Parameters

*username*

Type: [SystemString](#)

User privileges to assume.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## See Also

## Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLINUX\_errno Method

Fetch the value of `errno`.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static int LINUX_errno()
```

### Return Value

Type: [Int32](#)

Current value of `errno`.

## ▪ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLINUX\_openlog Method

Open a connection to the `syslog` service.

**Namespace:** `IO.Bindings.libsimpleio`

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public static void LINUX_openlog(
    string id,
    int options,
    int facility,
    out int error
)
```

## Parameters

*id*

Type: `SystemString`

Program identifier.

*options*

Type: `SystemInt32`

Logging options.

*facility*

Type: `SystemInt32`

Logging facility identifier.

*error*

Type: `SystemInt32`

Error code. Zero upon success or an `errno` value upon failure.

## ↳ See Also

**Reference**

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLINUX\_poll Method

Wait for an event on one or more files.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void LINUX_poll(
    int numfiles,
    int[] files,
    int[] events,
    int[] results,
    int timeouts,
    out int error
)
```

## Parameters

*numfiles*

Type: [SystemInt32](#)

Number elements in each of the following arrays.

*files*

Type: [SystemInt32](#)

File descriptors.

*events*

Type: [SystemInt32](#)

Events to wait for on each file descriptor.

*results*

Type: [SystemInt32](#)

Events that occurred on each file descriptor.

*timeoutms*

Type: [SystemInt32](#)

Milliseconds to wait for an event to occur. A value of -1 means wait forever and a value of 0 means do not wait at all.

*error*

Type: [SystemInt32](#)

Error code.

## ◀ See Also

**Reference**

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLINUX\_strerror Method

Retrieve the error message for a particular `errno` error code.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void LINUX_strerror(
    int error,
    StringBuilder buf,
    int bufsize
)
```

## Parameters

*error*

Type: [SystemInt32](#)

Error code.

*buf*

Type: [System.TextStringBuilder](#)

Destination buffer.

*bufsize*

Type: [SystemInt32](#)

Destination buffer size.

## ◀ See Also

[Reference](#)

[libLinux Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libLinuxLINUX\_syslog Method

Send a message to the `syslog` service.

**Namespace:** `IO.Bindings.libsimpleio`

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public static void LINUX_syslog(  
    int priority,  
    string msg,  
    out int error  
)
```

## Parameters

*priority*

Type: `SystemInt32`

Message priority

*msg*

Type: `SystemString`

Message to send.

*error*

Type: `SystemInt32`

Error code. Zero upon success or an `errno` value upon failure.

## ◀ See Also

[Reference](#)

[libLinux Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libLinuxLINUX\_usleep Method

Sleep for the specified number of microseconds.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void LINUX_usleep(
    int microsecs,
    out int error
)
```

## Parameters

*microsecs*

Type: [SystemInt32](#)

Number of microseconds to sleep.

*error*

Type: [SystemInt32](#)

Error code.

## ◀ See Also

[Reference](#)

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinux Fields

The `libLinux` type exposes the following members.

## Fields

	Name	Description
• 	<a href="#">LOG_ALERT</a>	Action must be taken immediately.
• 	<a href="#">LOG_AUTH</a>	Security/authorization messages.
• 	<a href="#">LOG_AUTHPRIV</a>	Securit/authorization messages.
• 	<a href="#">LOG_CRIT</a>	Critical condition.
• 	<a href="#">LOG_CRON</a>	<code>cron</code> daemon messages.
• 	<a href="#">LOG_DAEMON</a>	System daemons.
• 	<a href="#">LOG_DEBUG</a>	Debug message.
• 	<a href="#">LOG_EMERG</a>	System is unusable.
• 	<a href="#">LOG_ERR</a>	Error condition.
• 	<a href="#">LOG_FTP</a>	<code>FTP</code> daemon messages.
• 	<a href="#">LOG_INFO</a>	Informational message.
• 	<a href="#">LOG_KERN</a>	Kernel messages.

♦ <b>s</b>	<a href="#">LOG_LOCAL0</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LOCAL1</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LOCAL2</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LOCAL3</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LOCAL4</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LOCAL5</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LOCAL6</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LOCAL7</a>	Reserved for local use.
♦ <b>s</b>	<a href="#">LOG_LPR</a>	Line printer subsystem
♦ <b>s</b>	<a href="#">LOG_MAIL</a>	Mail system.
♦ <b>s</b>	<a href="#">LOG_NEWS</a>	Network news subsystem
♦ <b>s</b>	<a href="#">LOG_NOTICE</a>	Normal but significant condition.
♦ <b>s</b>	<a href="#">LOG_PROGNAME</a>	Use the program name for the identity string.
♦ <b>s</b>	<a href="#">LOG_SYSLOG</a>	Messages generated internally by syslogd
♦ <b>s</b>	<a href="#">LOG_USER</a>	Random user-level messages.
♦ <b>s</b>	<a href="#">LOG_UUCP</a>	UUCP subsystem
♦ <b>s</b>	<a href="#">LOG_WARNING</a>	Warning condition.

• S	POLLERR	An error occurred.
• S	POLLHUP	Peer closed connection.
• S	POLLIN	There is data to read.
• S	POLLNVAL	File descriptor is invalid.
• S	POLLOUT	Writing is now possible.
• S	POLLPRI	There is urgent data to read.

[Top](#)

## See Also

[Reference](#)

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_ALERT Field

Action must be taken immediately.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_ALERT = 1
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_AUTH Field

Security/authorization messages.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_AUTH = 32
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_AUTHPRIV Field

Securit/authorization messages.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_AUTHPRIV = 80
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_CRIT Field

Critical condition.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int LOG_CRIT = 2
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_CRON Field

[cron](#) daemon messages.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_CRON = 72
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_DAEMON Field

System daemons.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_DAEMON = 24
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_DEBUG Field

Debug message.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_DEBUG = 7
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_EMERG Field

System is unusable.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_EMERG = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_ERR Field

Error condition.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_ERR = 3
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_FTP Field

**FTP** daemon messages.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_FTP = 88
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_INFO Field

Informational message.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_INFO = 6
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_KERN Field

Kernel messages.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_KERN = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_LOCAL0 Field

Reserved for local use.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_LOCAL0 = 128
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_LOCAL1 Field

Reserved for local use.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_LOCAL1 = 136
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_LOCAL2 Field

Reserved for local use.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_LOCAL2 = 144
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_LOCAL3 Field

Reserved for local use.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_LOCAL3 = 152
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_LOCAL4 Field

Reserved for local use.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_LOCAL4 = 160
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_LOCAL5 Field

Reserved for local use.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_LOCAL5 = 168
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_LOCAL6 Field

Reserved for local use.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_LOCAL6 = 176
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_LOCAL7 Field

Reserved for local use.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_LOCAL7 = 184
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_LPR Field

Line printer subsystem

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_LPR = 48
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_MAIL Field

Mail system.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_MAIL = 16
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_NEWS Field

Network news subsystem

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_NEWS = 56
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_NOTICE Field

Normal but significant condition.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_NOTICE = 5
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_PROGNAME Field

Use the program name for the identity string.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const string LOG_PROGNAME = ""
```

### Field Value

Type: [String](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_SYSLOG Field

Messages generated internally by syslogd

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_SYSLOG = 40
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_USER Field

Random user-level messages.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_USER = 8
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_UUCP Field

UUCP subsystem

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_UUCP = 64
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxLOG\_WARNING Field

Warning condition.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int LOG_WARNING = 4
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxPOLLERR Field

An error occurred.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int POLLERR = 8
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxPOLLHUP Field

Peer closed connection.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int POLLHUP = 16
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxPOLLIN Field

There is data to read.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int POLLIN = 1
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxPOLLNVAL Field

File descriptor is invalid.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int POLLNVAL = 32
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxPOLLOUT Field

Writing is now possible.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int POLLOUT = 4
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libLinuxPOLLPRI Field

There is urgent data to read.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int POLLPRI = 2
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libLinux Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libPWM Class

Wrapper for libsimpleio PWM output services.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibPWM](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libPWM
```

The [libPWM](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">libPWM</a>	Initializes a new instance of the <a href="#">libPWM</a> class

[Top](#)

## ▪ Methods

	Name	Description
	<a href="#">PWM_close</a>	Close a Linux PWM output

device.

---

≡  	<a href="#">PWM_configure</a>	Configure a Linux PWM output device.
≡  	<a href="#">PWM_open</a>	Open a Linux PWM output device.
≡  	<a href="#">PWM_write</a>	Set a Linux PWM output device duty cycle.

---

[Top](#)

## Fields

	Name	Description
≡  	<a href="#">ActiveHigh</a>	Configure the PWM output as active high (normal).
≡  	<a href="#">ActiveLow</a>	Configure the PWM output as active low (inverted).

---

[Top](#)

## See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libPWM Constructor

Initializes a new instance of the [libPWM](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libPWM()
```

## ◀ See Also

### Reference

[libPWM Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libPWM Methods

The [libPWM](#) type exposes the following members.

## Methods

	Name	Description
 	<a href="#">PWM_close</a>	Close a Linux PWM output device.
 	<a href="#">PWM_configure</a>	Configure a Linux PWM output device.
 	<a href="#">PWM_open</a>	Open a Linux PWM output device.
 	<a href="#">PWM_write</a>	Set a Linux PWM output device duty cycle.

[Top](#)

## See Also

[Reference](#)

[libPWM Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libPWM\_PWM\_close Method

Close a Linux PWM output device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void PWM_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libPWM Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libPWMPWM\_configure Method

Configure a Linux PWM output device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void PWM_configure(
    int chip,
    int channel,
    int period,
    int ontime,
    int polarity,
    out int error
)
```

## Parameters

*chip*

Type: [SystemInt32](#)

Chip number.

*channel*

Type: [SystemInt32](#)

Channel number.

*period*

Type: [SystemInt32](#)

Pulse period in microseconds.

*ontime*

Type: [SystemInt32](#)

Initial on time in microseconds.

*polarity*

Type: [SystemInt32](#)

PWM output polarity (0 for active low/inverted or 1 for active high/normal).

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ▪ Remarks

On many platforms two or more PWM outputs may share the same clock generator, so configuring different PWM pulse periods may not be possible.

## ▪ Remarks

Not all platforms support active low (inverted) PWM outputs.

## ▪ See Also

**Reference**

[libPWM Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libPWMPWM\_open Method

Open a Linux PWM output device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void PWM_open(
    int chip,
    int channel,
    [out] int fd,
    [out] int error
)
```

## Parameters

*chip*

Type: [SystemInt32](#)

Chip number.

*channel*

Type: [SystemInt32](#)

Channel number.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ↳ See Also

**Reference**

[libPWM Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libPWMPWM\_write Method

Set a Linux PWM output device duty cycle.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void PWM_write(
    int fd,
    int ontime,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*ontime*

Type: [SystemInt32](#)

On time in microseconds.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libPWM Class](#)

## IO.Bindings.libsimpleio Namespace

---

# libPWM Fields

The [libPWM](#) type exposes the following members.

## Fields

Name	Description
<a href="#">  ActiveHigh</a>	Configure the PWM output as active high (normal).
<a href="#">  ActiveLow</a>	Configure the PWM output as active low (inverted).

[Top](#)

## See Also

### Reference

[libPWM Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libPWMActiveHigh Field

Configure the PWM output as active high (normal).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int ActiveHigh = 1
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[libPWM Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libPWMActiveLow Field

Configure the PWM output as active low (inverted).

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int ActiveLow = 0
```

### Field Value

Type: [Int32](#)

## « Remarks

Not all platforms support active low (inverted) PWM outputs.

## « See Also

### Reference

[libPWM Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSerial Class

Wrapper for libsimpleio serial port services.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibSerial](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libSerial
```

The [libSerial](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">libSerial</a>	Initializes a new instance of the <a href="#">libSerial</a> class

[Top](#)

## ▪ Methods

	Name	Description
≡ S	<a href="#">SERIAL_close</a>	Close a Linux serial port device.

---

≡ 	<a href="#">SERIAL_open</a>	Open a Linux serial port device.
≡ 	<a href="#">SERIAL_receive</a>	Receive data from a Linux serial port device.
≡ 	<a href="#">SERIAL_send</a>	Send data to a Linux serial port device.

---

[Top](#)

## ↳ Fields

	Name	Description
≡ 	<a href="#">PARITY_EVEN</a>	Request even parity checking.
≡ 	<a href="#">PARITY_NONE</a>	Disable parity checking.
≡ 	<a href="#">PARITY_ODD</a>	Request odd parity checking.

[Top](#)

## ↳ See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libSerial Constructor

Initializes a new instance of the [libSerial](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libSerial()
```

## ◀ See Also

### Reference

[libSerial Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSerial Methods

The [libSerial](#) type exposes the following members.

## ▪ Methods

	Name	Description
 S	<a href="#">SERIAL_close</a>	Close a Linux serial port device.
 S	<a href="#">SERIAL_open</a>	Open a Linux serial port device.
 S	<a href="#">SERIAL_receive</a>	Receive data from a Linux serial port device.
 S	<a href="#">SERIAL_send</a>	Send data to a Linux serial port device.

[Top](#)

## ▪ See Also

### Reference

[libSerial Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSerialSERIAL\_close Method

Close a Linux serial port device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void SERIAL_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libSerial Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSerialSERIAL\_open Method

Open a Linux serial port device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void SERIAL_open(
    string devname,
    int baudrate,
    int parity,
    int databits,
    int stopbits,
    out int fd,
    out int error
)
```

## Parameters

*devname*

Type: [SystemString](#)

Device node name.

*baudrate*

Type: [SystemInt32](#)

Baud rate.

*parity*

Type: [SystemInt32](#)

Parity setting (0 to 2).

*databits*

Type: [SystemInt32](#)

Word size setting (5 to 8).

*stopbits*

Type: [SystemInt32](#)

Number of stop bits (1 or 2).

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libSerial Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSerialSERIAL\_receive Method

Receive data from a Linux serial port device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void SERIAL_receive(
    int fd,
    byte[] buf,
    int bufsize,
    out int count,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*buf*

Type: [SystemByte](#)

Destination buffer.

*bufsize*

Type: [SystemInt32](#)

Destination buffer size.

*count*

Type: [SystemInt32](#)

Number of bytes actually received.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

**Reference**

[libSerial Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSerialSERIAL\_send Method

Send data to a Linux serial port device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void SERIAL_send(
    int fd,
    byte[] buf,
    int bufsize,
    out int count,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*buf*

Type: [SystemByte](#)

Source buffer.

*bufsize*

Type: [SystemInt32](#)

Source buffer size.

*count*

Type: [SystemInt32](#)

Number of bytes actually sent.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

**Reference**

[libSerial Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSerial Fields

The [libSerial](#) type exposes the following members.

## Fields

Name	Description
<a href="#">   PARITY_EVEN</a>	Request even parity checking.
<a href="#">  PARITY_NONE</a>	Disable parity checking.
<a href="#">  PARITY_ODD</a>	Request odd parity checking.

[Top](#)

## See Also

### Reference

[libSerial Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSerialPARITY\_EVEN Field

Request even parity checking.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int PARITY_EVEN = 1
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libSerial Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSerialPARITY\_NONE Field

Disable parity checking.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int PARITY_NONE = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libSerial Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSerialPARITY\_ODD Field

Request odd parity checking.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int PARITY_ODD = 2
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libSerial Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSPI Class

Wrapper for libsimpleio SPI device services.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibSPI](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libSPI
```

The [libSPI](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">libSPI</a>	Initializes a new instance of the <a href="#">libSPI</a> class

[Top](#)

## ▪ Methods

	Name	Description
	<a href="#">SPI_close</a>	Close a Linux SPI device.



[SPI\\_open](#) Open a Linux SPI device.



[SPI\\_transaction](#) Send bytes to and/or receive bytes from a Linux SPI device.

[Top](#)

## ↳ Fields

	Name	Description
	<a href="#">SPI_AUTO_CS</a>	Use hardware slave select.

[Top](#)

## ↳ See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libSPI Constructor

Initializes a new instance of the [libSPI](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libSPI()
```

## ◀ See Also

### Reference

[libSPI Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSPI Methods

The [libSPI](#) type exposes the following members.

## Methods

	Name	Description
 	<a href="#">SPI_close</a>	Close a Linux SPI device.
 	<a href="#">SPI_open</a>	Open a Linux SPI device.
 	<a href="#">SPI_transaction</a>	Send bytes to and/or receive bytes from a Linux SPI device.

[Top](#)

## See Also

### Reference

[libSPI Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSPI\_SPI\_close Method

Close a Linux SPI device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void SPI_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

[Reference](#)

[libSPI Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSPISPI\_open Method

Open a Linux SPI device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void SPI_open(
    string devname,
    int mode,
    int wordsize,
    int speed,
    out int fd,
    out int error
)
```

## Parameters

*devname*

Type: [SystemString](#)

Device node name.

*mode*

Type: [SystemInt32](#)

SPI transfer mode (0 .. 3)

*wordsize*

Type: [SystemInt32](#)

SPI transfer word size (8, 16, or 32).

*speed*

Type: [SystemInt32](#)

SPI transfer speed in Hz.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## Remarks

The Linux kernel creates device nodes for each SPI slave device, of the form `/dev/spidevX.Y` where `X` is the SPI bus controller number and `Y` is the SPI slave select number.

## See Also

[Reference](#)

[libSPI Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSPISPI\_transaction Method

Send bytes to and/or receive bytes from a Linux SPI device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static void SPI_transaction(  
    int fd,  
    int csfd,  
    byte[] cmd,  
    int cmdLen,  
    int delayus,  
    byte[] resp,  
    int resplen,  
    out int error  
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*csfd*

Type: [SystemInt32](#)

Chip select file descriptor.

*cmd*

Type: [SystemByte](#)

Source buffer.

*cmdlen*

Type: [SystemInt32](#)

Source buffer size.

*delayus*

Type: [SystemInt32](#)

Delay in microseconds between the write and read operations.

*resp*

Type: [SystemByte](#)

Destination buffer.

*resplen*

Type: [SystemInt32](#)

Destination buffer size.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## See Also

**Reference**

[libSPI Class](#)

[IO.Bindings.libsimpleio Namespace](#)

Munts Technologies Linux Simple I/O Library .Net Standard 2.0 Class  
Library 2.2020.135.1

# libSPI Fields

The [libSPI](#) type exposes the following members.

## Fields

	Name	Description
	<a href="#">SPI_AUTO_CS</a>	Use hardware slave select.

[Top](#)

## See Also

**Reference**

[libSPI Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libSPI.SPI\_AUTO\_CS Field

Use hardware slave select.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int SPI_AUTO_CS = -1
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[libSPI Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libStream Class

Wrapper for libsimpleio Stream Framing Protocol services.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibStream](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libStream
```

The [libStream](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">libStream</a>	Initializes a new instance of the <a href="#">libStream</a> class

[Top](#)

## ▪ Methods

	Name	Description
≡ S	<a href="#">STREAM_decode_frame</a>	Decode a frame.

---

≡  	<a href="#">STREAM_encode_frame</a>	Encode a frame.
≡  	<a href="#">STREAM_receive_frame</a>	Receive an encoded frame.
≡  	<a href="#">STREAM_send_frame</a>	Send an encoded frame.

---

[Top](#)

## ↳ See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libStream Constructor

Initializes a new instance of the [libStream](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libStream()
```

## ◀ See Also

### Reference

[libStream Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libStream Methods

The [libStream](#) type exposes the following members.

## ▪ Methods

Name	Description
 <a href="#">STREAM_decode_frame</a>	Decode a frame.
 <a href="#">STREAM_encode_frame</a>	Encode a frame.
 <a href="#">STREAM_receive_frame</a>	Receive an encoded frame.
 <a href="#">STREAM_send_frame</a>	Send an encoded frame.

[Top](#)

## ▪ See Also

### Reference

[libStream Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libStreamSTREAM\_decode\_frame Method

Decode a frame.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static void STREAM_decode_frame(
    byte[] src,
    int srcLen,
    byte[] dst,
    int dstSize,
    out int dstLen,
    out int error
)
```

## Parameters

*src*

Type: [SystemByte](#)

Source buffer.

*srcLen*

Type: [SystemInt32](#)

Source buffer size.

*dst*

Type: [SystemByte](#)

Destination buffer.

*dstsize*

Type: [SystemInt32](#)

Destination buffer size.

*dstlen*

Type: [SystemInt32](#)

Size of decoded frame.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

**Reference**

[libStream Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libStreamSTREAM\_encode\_frame Method

Encode a frame.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static void STREAM_encode_frame(
    byte[] src,
    int srcLen,
    byte[] dst,
    int dstSize,
    out int dstLen,
    out int error
)
```

## Parameters

*src*

Type: [SystemByte](#)

Source buffer.

*srcLen*

Type: [SystemInt32](#)

Source buffer size.

*dst*

Type: [SystemByte](#)

Destination buffer.

*dstsize*

Type: [SystemInt32](#)

Destination buffer size.

*dstlen*

Type: [SystemInt32](#)

Size of encoded frame.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libStream Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libStreamSTREAM\_receive\_frame Method

Receive an encoded frame.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public static void STREAM_receive_frame(
    int fd,
    byte[] buf,
    int bufsize,
    out int count,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptior.

*buf*

Type: [SystemByte](#)

Destination buffer.

*bufsize*

Type: [SystemInt32](#)

Destination buffer size.

*count*

Type: [SystemInt32](#)

Number of bytes actually received.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

**Reference**

[libStream Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libStreamSTREAM\_send\_frame Method

Send an encoded frame.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public static void STREAM_send_frame(
    int fd,
    byte[] buf,
    int bufsize,
    out int count,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*buf*

Type: [SystemByte](#)

Source buffer.

*bufsize*

Type: [SystemInt32](#)

Source buffer size.

*count*

Type: [SystemInt32](#)

Number of bytes actually sent.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## See Also

**Reference**

[libStream Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libWatchdog Class

Wrapper for libsimpleio watchdog timer services.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Bindings.libsimpleiolibWatchdog](#)

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class libWatchdog
```

The [libWatchdog](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">libWatchdog</a>	Initializes a new instance of the <a href="#">libWatchdog</a> class

[Top](#)

## ▪ Methods

	Name	Description
≡ S	<a href="#">WATCHDOG_close</a>	Close a Linux watchdog

timer device.

---

 S	<a href="#">WATCHDOG_get_timeout</a>	Query a Linux watchdog timer device.
 S	<a href="#">WATCHDOG_kick</a>	Reset the watchdog timer.
 S	<a href="#">WATCHDOG_open</a>	Open a Linux watchdog timer device.
 S	<a href="#">WATCHDOG_set_timeout</a>	Change the watchdog timer period.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Bindings.libsimpleio Namespace](#)

# libWatchdog Constructor

Initializes a new instance of the [libWatchdog](#) class

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public libWatchdog()
```

## ◀ See Also

### Reference

[libWatchdog Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libWatchdog Methods

The [libWatchdog](#) type exposes the following members.

## Methods

	Name	Description
 	<a href="#">WATCHDOG_close</a>	Close a Linux watchdog timer device.
 	<a href="#">WATCHDOG_get_timeout</a>	Query a Linux watchdog timer device.
 	<a href="#">WATCHDOG_kick</a>	Reset the watchdog timer.
 	<a href="#">WATCHDOG_open</a>	Open a Linux watchdog timer device.
 	<a href="#">WATCHDOG_set_timeout</a>	Change the watchdog timer period.

[Top](#)

## See Also

### Reference

[libWatchdog Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libWatchdogWATCHDOG\_close Method

Close a Linux watchdog timer device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static void WATCHDOG_close(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libWatchdog Class](#)

[IO.Bindings.libsimpleio Namespace](#)



# libWatchdogWATCHDOG\_get\_timeout Method

Query a Linux watchdog timer device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static void WATCHDOG_get_timeout(
    int fd,
    out int timeout,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*timeout*

Type: [SystemInt32](#)

Timeout period in seconds.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ▪ See Also

### Reference

libWatchdog Class  
IO.Bindings.libsimpleio Namespace

---

# libWatchdogWATCHDOG\_kick Method

Reset the watchdog timer.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static void WATCHDOG_kick(
    int fd,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

### Reference

[libWatchdog Class](#)

[IO.Bindings.libsimpleio Namespace](#)



# libWatchdogWATCHDOG\_open Method

Open a Linux watchdog timer device.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public static void WATCHDOG_open(
    string devname,
    out int fd,
    out int error
)
```

## Parameters

*devname*

Type: [SystemString](#)

Device node name.

*fd*

Type: [SystemInt32](#)

File descriptor.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ◀ See Also

## Reference

[libWatchdog Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# libWatchdogWATCHDOG\_set\_timeout Method

Change the watchdog timer period.

**Namespace:** [IO.Bindings.libsimpleio](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public static void WATCHDOG_set_timeout(
    int fd,
    int newtimeout,
    out int timeout,
    out int error
)
```

## Parameters

*fd*

Type: [SystemInt32](#)

File descriptor.

*newtimeout*

Type: [SystemInt32](#)

Requested timeout period in seconds.

*timeout*

Type: [SystemInt32](#)

Actual timeout period in seconds.

*error*

Type: [SystemInt32](#)

Error code. Zero upon success or an [errno](#) value upon failure.

## ▪ Remarks

Not all platforms allow changing the timeout period. Some platforms may not allow *increasing* the period.

## ▪ See Also

### Reference

[libWatchdog Class](#)

[IO.Bindings.libsimpleio Namespace](#)

# IO.Devices.AD5593R Namespace

AD5593 Analog/Digital I/O Device Services.

## Classes

	Class	Description
	<a href="#">Device</a>	Encapsulates the AD5593R I <sup>2</sup> C Analog/Digital I/O device.

## Enumerations

	Enumeration	Description
	<a href="#">PinMode</a>	AD5593R I/O Pin Modes.
	<a href="#">ReferenceMode</a>	ADC5593R ADC and DAC reference settings.

# Device Class

Encapsulates the AD5593R I<sup>2</sup>C Analog/Digital I/O device.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.AD5593RDevice](#)

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Device</a>	Constructor for a single AD5593R device.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">ADC_Reference</a>	Write-only property for setting the AD5593R ADC reference

mode.

---

	<a href="#">DAC_Reference</a>	Write-only property for setting the AD5593R DAC reference mode.
	<a href="#">GPIO_Inputs</a>	GPIO input register state. Any I/O pin that is not configured as a GPIO input will read as zero.
	<a href="#">GPIO_Outputs</a>	GPIO output register state. Any I/O pin that is not configured as a GPIO output will be written as zero.

---

[Top](#)

## ◀ Methods

	Name	Description
	<a href="#">ADC_Create</a>	Create an AD5593R ADC input object.
	<a href="#">ConfigureChannel</a>	Configure a single AD5593R I/O pin.
	<a href="#">DAC_Create</a>	Create an AD5593R DAC output object.
	<a href="#">GPIO_Create</a>	Create an AD5593R GPIO pin object.
	<a href="#">Read_ADC</a>	Read from an ADC channel.
	<a href="#">Write_DAC</a>	Write to a DAC channel.

---

[Top](#)

## Fields

Name	Description
  <a href="#">ADC_Resolution</a>	ADC resolution in bits.
  <a href="#">DAC_Resolution</a>	DAC resolution in bits.
  <a href="#">MaxChannel</a>	Maximum I/O channel number.
  <a href="#">MinChannel</a>	Minimum I/O channel number.

[Top](#)

## See Also

### Reference

[IO.Devices.AD5593R Namespace](#)

---

# Device Constructor

Constructor for a single AD5593R device.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus,  
    int addr  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup>C bus controller object.

*addr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">ADC_Reference</a>	Write-only property for setting the AD5593R ADC reference mode.
	<a href="#">DAC_Reference</a>	Write-only property for setting the AD5593R DAC reference mode.
	<a href="#">GPIO_Inputs</a>	GPIO input register state. Any I/O pin that is not configured as a GPIO input will read as zero.
	<a href="#">GPIO_Outputs</a>	GPIO output register state. Any I/O pin that is not configured as a GPIO output will be written as zero.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

## IO.Devices.AD5593R Namespace

---

# DeviceADC\_Reference Property

Write-only property for setting the AD5593R ADC reference mode.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public ReferenceMode ADC_Reference { set; }
```

### Property Value

Type: [ReferenceMode](#)

## ◀ See Also

### [Reference](#)

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceDAC\_Reference Property

Write-only property for setting the AD5593R DAC reference mode.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public ReferenceMode DAC_Reference { set; }
```

### Property Value

Type: [ReferenceMode](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceGPIO\_Inputs Property

GPIO input register state. Any I/O pin that is not configured as a GPIO input will read as zero.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte GPIO_Inputs { get; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceGPIO\_Outputs Property

GPIO output register state. Any I/O pin that is not configured as a GPIO output will be written as zero.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public byte GPIO_Outputs { get; set; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## Methods

	Name	Description
≡	<a href="#">ADC_Create</a>	Create an AD5593R ADC input object.
≡	<a href="#">ConfigureChannel</a>	Configure a single AD5593R I/O pin.
≡	<a href="#">DAC_Create</a>	Create an AD5593R DAC output object.
≡	<a href="#">GPIO_Create</a>	Create an AD5593R GPIO pin object.
≡	<a href="#">Read_ADC</a>	Read from an ADC channel.
≡	<a href="#">Write_DAC</a>	Write to a DAC channel.

[Top](#)

## See Also

### Reference

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceADC\_Create Method

Create an AD5593R ADC input object.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample ADC_Create(  
    int channel  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

AD5593R ADC channel number (0 to 7).

## Return Value

Type: [Sample](#)

ADC input object.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceConfigureChannel Method

Configure a single ADC5593R I/O pin.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void ConfigureChannel(  
    int channel,  
    PinMode mode  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

ADC5593R I/O channel number (0 to 7).

*mode*

Type: [IO.Devices.AD5593RPinMode](#)

ADC5593R I/O pin mode.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceDAC\_Create Method

Create an AD5593R DAC output object.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample DAC_Create(  
    int channel,  
    int sample = 0  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

AD5593R DAC channel number (0 to 7).

*sample* (Optional)

Type: [SystemInt32](#)

Initial DAC output sample.

## Return Value

Type: [Sample](#)

DAC output object.

## ◀ See Also

**Reference**

[Device Class](#)

## IO.Devices.AD5593R Namespace

---

# DeviceGPIO\_Create Method

Create an AD5593R GPIO pin object.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO_Create(  
    int channel,  
    Direction dir,  
    bool state = false  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

AD5593R GPIO channel number (0 to 7).

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin data direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO pin object.

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceRead\_ADC Method

Read from an ADC channel.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int Read_ADC(  
    int channel  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

ADC channel number (0 to 7).

## Return Value

Type: [Int32](#)

ADC input sample data (0 to 4095).

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceWrite\_DAC Method

Write to a DAC channel.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Write_DAC(
    int channel,
    int data
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

DAC channel number (0 to 7).

*data*

Type: [SystemInt32](#)

DAC output sample data (0 to 4095).

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# Device Fields

The [Device](#) type exposes the following members.

## Fields

	Name	Description
• 	<a href="#">ADC_Resolution</a>	ADC resolution in bits.
• 	<a href="#">DAC_Resolution</a>	DAC resolution in bits.
• 	<a href="#">MaxChannel</a>	Maximum I/O channel number.
• 	<a href="#">MinChannel</a>	Minimum I/O channel number.

[Top](#)

## See Also

### Reference

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceADC\_Resolution Field

ADC resolution in bits.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int ADC_Resolution = 12
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceDAC\_Resolution Field

DAC resolution in bits.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int DAC_Resolution = 12
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceMaxChannel Field

Maximum I/O channel number.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MaxChannel = 7
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# DeviceMinChannel Field

Minimum I/O channel number.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MinChannel = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.AD5593R Namespace](#)

# PinMode Enumeration

AD5593R I/O Pin Modes.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum PinMode
```

## ▪ Members

Member name	Value	Description
ADC_Input	0	Analog input.
DAC_Output	1	Analog output.
GPIO_Input	2	GPIO input.
GPIO_Output	3	GPIO output.
GPIO_Output_OpenDrain	4	GPIO open drain output.

## ▪ See Also

[Reference](#)

## IO.Devices.AD5593R Namespace

---

# ReferenceMode Enumeration

ADC5593R ADC and DAC reference settings.

**Namespace:** [IO.Devices.AD5593R](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum ReferenceMode
```

## ▪ Members

Member name	Value	Description
Internalx1	0	The reference voltage is 2.5V using the internal reference.
Internalx2	1	The reference voltage is 5.0V using the internal reference.
Externalx1	2	The reference voltage is 1.0*Vref, using an external reference.
Externalx2	3	The reference voltage is 2.0*Vref, using an external reference.

## See Also

### Reference

[IO.Devices.AD5593R Namespace](#)

---

# IO.Devices.AD5593R.ADC Namespace

AD5593 Analog/Digital I/O Device ADC Input Services.

## ◀ Classes

Class	Description
 Sample	Encapsulates AD5593R ADC inputs.

# Sample Class

Encapsulates AD5593R ADC inputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.AD5593R.ADCSample](#)

**Namespace:** [IO.Devices.AD5593R.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Sample : Sample
```

The [Sample](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Sample</a>	Create an AD5593R ADC input.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.



[sample](#)

Read-only property returning an integer analog sample value.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.AD5593R.ADC Namespace](#)

# Sample Constructor

Create an AD5593R ADC input.

**Namespace:** [IO.Devices.AD5593R.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample(  
    Device dev,  
    int channel  
)
```

## Parameters

*dev*

Type: [IO.Devices.AD5593RDevice](#)

AD5593R device object.

*channel*

Type: [SystemInt32](#)

AD5593R I/O channel number (0 to 7).

## ◀ See Also

[Reference](#)

[Sample Class](#)

[IO.Devices.AD5593R.ADC Namespace](#)

# Sample Properties

The [Sample](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.
	<a href="#">sample</a>	Read-only property returning an integer analog sample value.

[Top](#)

## See Also

### Reference

[Sample Class](#)

[IO.Devices.AD5593R.ADC Namespace](#)

# Sampleresolution Property

Read-only property returning the number of bits of resolution.

**Namespace:** [IO.Devices.AD5593R.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public int resolution { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Sampleresolution](#)

## ▪ See Also

[Reference](#)

[Sample Class](#)

[IO.Devices.AD5593R.ADC Namespace](#)

# Samplesample Property

Read-only property returning an integer analog sample value.

**Namespace:** [IO.Devices.AD5593R.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public int sample { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Samplesample](#)

## ▪ See Also

[Reference](#)

[Sample Class](#)

[IO.Devices.AD5593R.ADC Namespace](#)

# IO.Devices.AD5593R.DAC Namespace

AD5593 Analog/Digital I/O Device DAC Output Services.

## ◀ Classes

Class	Description
 Sample	Encapsulates AD5593R DAC outputs.

# Sample Class

Encapsulates AD5593R DAC outputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.AD5593R.DACSample](#)

**Namespace:** [IO.Devices.AD5593R.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Sample : Sample
```

The [Sample](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Sample</a>	Create an AD5593R DAC output.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.



## sample

Write-only property for writing an integer analog sample to a DAC output.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.AD5593R.DAC Namespace](#)

# Sample Constructor

Create an AD5593R DAC output.

**Namespace:** [IO.Devices.AD5593R.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public Sample(  
    Device dev,  
    int channel,  
    int sample = 0  
)
```

## Parameters

*dev*

Type: [IO.Devices.AD5593RDevice](#)

AD5593R device object.

*channel*

Type: [SystemInt32](#)

AD5593R I/O channel number (0 to 7).

*sample (Optional)*

Type: [SystemInt32](#)

Initial DAC output sample.

## « See Also

[Reference](#)

[Sample Class](#)

## IO.Devices.AD5593R.DAC Namespace

---

# Sample Properties

The [Sample](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.
	<a href="#">sample</a>	Write-only property for writing an integer analog sample to a DAC output.

[Top](#)

## See Also

### Reference

[Sample Class](#)

[IO.Devices.AD5593R.DAC Namespace](#)

# Sampleresolution Property

Read-only property returning the number of bits of resolution.

**Namespace:** [IO.Devices.AD5593R.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public int resolution { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Sampleresolution](#)

## ▪ See Also

[Reference](#)

[Sample Class](#)

[IO.Devices.AD5593R.DAC Namespace](#)

# Samplesample Property

Write-only property for writing an integer analog sample to a DAC output.

**Namespace:** [IO.Devices.AD5593R.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▲ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public int sample { set; }
```

### Property Value

Type: [Int32](#)

### Implements

[Samplesample](#)

## ▲ See Also

### Reference

[Sample Class](#)

[IO.Devices.AD5593R.DAC Namespace](#)

# IO.Devices.AD5593R.GPIO Namespace

AD5593 Analog/Digital I/O Device GPIO Pin Services.

## ◀ Classes

Class	Description
 Pin	Encapsulates AD5593R GPIO pins.

# Pin Class

Encapsulates AD5593R GPIO pins.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.AD5593R.GPIOPin](#)

**Namespace:** [IO.Devices.AD5593R.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Pin : Pin
```

The [Pin](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Pin</a>	Create an AD5593R GPIO pin.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## ↳ See Also

### Reference

[IO.Devices.AD5593R.GPIO Namespace](#)

# Pin Constructor

Create an AD5593R GPIO pin.

**Namespace:** [IO.Devices.AD5593R.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin(  
    Device dev,  
    int channel,  
    Direction dir,  
    bool state = false  
)
```

## Parameters

*dev*

Type: [IO.Devices.AD5593RDevice](#)

AD5593R device object.

*channel*

Type: [SystemInt32](#)

AD5593R I/O channel number (0 to 7).

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin data direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## See Also

**Reference**

[Pin Class](#)

[IO.Devices.AD5593R.GPIO Namespace](#)

# Pin Properties

The [Pin](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## See Also

### Reference

[Pin Class](#)

[IO.Devices.AD5593R.GPIO Namespace](#)

# Pinstate Property

Read/Write GPIO state property.

**Namespace:** [IO.Devices.AD5593R.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public bool state { get; set; }
```

### Property Value

Type: [Boolean](#)

### Implements

[Pinstate](#)

## « See Also

[Reference](#)

[Pin Class](#)

[IO.Devices.AD5593R.GPIO Namespace](#)

# IO.Devices.ADC121C021

## Namespace

ADC121C021 I<sup>2</sup>C A/D Converter Services

### ↳ Classes

Class	Description
 Sample	Encapsulates the ADC121C021 I <sup>2</sup> C A/D converter.

# Sample Class

Encapsulates the ADC121C021 I<sup>2</sup>C A/D converter.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ADC121C021Sample](#)

**Namespace:** [IO.Devices.ADC121C021](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Sample : Sample
```

The [Sample](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Sample</a>	Constructor for an ADC121C021 analog input.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">resolution</a>	Return the number of bits of A/D

resolution.



[sample](#) Returns a single 12-bit analog sample.

[Top](#)

## See Also

### Reference

[IO.Devices.ADC121C021 Namespace](#)

# Sample Constructor

Constructor for an ADC121C021 analog input.

**Namespace:** [IO.Devices.ADC121C021](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample(  
    Bus bus,  
    byte addr  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup>C bus controller.

*addr*

Type: [SystemByte](#)

I<sup>2</sup>C slave address.

## ◀ See Also

[Reference](#)

[Sample Class](#)

[IO.Devices.ADC121C021 Namespace](#)

# Sample Properties

The [Sample](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">resolution</a>	Return the number of bits of A/D resolution.
	<a href="#">sample</a>	Returns a single 12-bit analog sample.

[Top](#)

## See Also

**Reference**

[Sample Class](#)

[IO.Devices.ADC121C021 Namespace](#)

# Sampleresolution Property

Return the number of bits of A/D resolution.

**Namespace:** [IO.Devices.ADC121C021](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public int resolution { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Sampleresolution](#)

## ▪ See Also

[Reference](#)

[Sample Class](#)

[IO.Devices.ADC121C021 Namespace](#)

# Samplesample Property

Returns a single 12-bit analog sample.

**Namespace:** [IO.Devices.ADC121C021](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int sample { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Samplesample](#)

## ◀ See Also

[Reference](#)

[Sample Class](#)

[IO.Devices.ADC121C021 Namespace](#)

# IO.Devices.ClickBoards.RemoteIO.ADAC Namespace

Mikroelektronika ADAC Click MIKROE-2690 Services

## ▪ Classes

	Class	Description
	<a href="#">Board</a>	Encapsulates the Mikroelektronika ADAC Click Board. <a href="#">MIKROE-2690</a> .

# Munts Technologies Linux Simple I/O Library .Net Standard 2.0 Class Library 2.2020.135.1

## Board Class

Encapsulates the Mikroelektronika ADAC Click Board. [MIKROE-2690](#).

### ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ClickBoards.RemoteIO.ADACBoard](#)

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

### ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Board
```

The [Board](#) type exposes the following members.

### ▪ Constructors

	Name	Description
	<a href="#">Board</a>	Constructor for a single ADAC click.

[Top](#)

### ▪ Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying AD5593R device object.

[Top](#)

## ◀ Methods

	Name	Description
≡	<a href="#">ADC</a>	Factory function for creating ADC inputs.
≡	<a href="#">DAC</a>	Factory function for creating DAC outputs.
≡	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
≡	<a href="#">Reset</a>	Issue hardware reset to the AD5593R.

[Top](#)

## ◀ Fields

	Name	Description
◆ <b>S</b>	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.ClickBoards.RemoteIO.ADAC Namespace](#)

# Board Constructor

Constructor for a single ADAC click.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Board(  
    int socknum,  
    int addr = 16,  
    Device remdev = null  
)
```

## Parameters

*socknum*

Type: [SystemInt32](#)

mikroBUS socket number.

*addr* (Optional)

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*remdev* (Optional)

Type: [IO.RemoteDevice](#)

Remote I/O server device object.

## ◀ See Also

[Reference](#)

[Board Class](#)

## IO.Devices.ClickBoards.RemoteIO.ADAC Namespace

---

# Board Properties

The [Board](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying AD5593R device object.

[Top](#)

## See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.ADAC Namespace](#)

# Boarddevice Property

Returns the underlying AD5593R device object.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device device { get; }
```

### Property Value

Type: [Device](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.ADAC Namespace](#)

# Board Methods

The [Board](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">ADC</a>	Factory function for creating ADC inputs.
	<a href="#">DAC</a>	Factory function for creating DAC outputs.
	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
	<a href="#">Reset</a>	Issue hardware reset to the AD5593R.

[Top](#)

## See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.ADAC Namespace](#)

# BoardADC Method

Factory function for creating ADC inputs.

**Namespace:** [IO.Devices.ClickBoards.RemotelIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample ADC(  
    int channel  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

AD5593R I/O channel number (0 to 7).

## Return Value

Type: [Sample](#)

ADC input object.

## ◀ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.RemotelIO.ADAC Namespace](#)

# BoardDAC Method

Factory function for creating DAC outputs.

**Namespace:** [IO.Devices.ClickBoards.RemotelIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample DAC(  
    int channel,  
    int sample = 0  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

AD5593R I/O channel number (0 to 7).

*sample* (Optional)

Type: [SystemInt32](#)

Initial DAC output sample.

## Return Value

Type: [Sample](#)

DAC output object.

## ◀ See Also

**Reference**

[Board Class](#)

## IO.Devices.ClickBoards.RemoteIO.ADAC Namespace

---

# BoardGPIO Method

Factory function for creating GPIO pins.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO(
    int channel,
    Direction dir,
    bool state = false
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

AD5593R I/O channel number (0 to 7).

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO pin object.

## ↳ See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.ADAC Namespace](#)

---

# BoardReset Method

Issue hardware reset to the AD5593R.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Reset()
```

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.ADAC Namespace](#)

# Board Fields

The [Board](#) type exposes the following members.

## Fields

	Name	Description
	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.ADAC Namespace](#)

# BoardDefaultAddress Field

Default I<sup>2</sup>C slave address.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const byte DefaultAddress = 16
```

### Field Value

Type: [Byte](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.ADAC Namespace](#)

# IO.Devices.ClickBoards.RemoteIO.Expand Namespace

Mikroelektronika Expand Click MIKROE-951 Services

## ▪ Classes

Class	Description
 <a href="#">Board</a>	Encapsulates the Mikroelektronika Expand Click Board.

# Board Class

Encapsulates the Mikroelektronika Expand Click Board.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ClickBoards.RemoteIO.ExpandBoard](#)

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.Expand](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Board
```

The [Board](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Board</a>	Constructor for a single Expand click.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying MCP23S17 device object.

[Top](#)

## ◀ Methods

	Name	Description
	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
	<a href="#">Reset</a>	Issue hardware reset to the MCP23S17.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.ClickBoards.RemoteIO.Expand Namespace](#)

# Board Constructor

Constructor for a single Expand click.

**Namespace:** [IO.Devices.ClickBoards.RemotelO.Expand](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Board(  
    int socknum,  
    Device remdev = null  
)
```

## Parameters

*socknum*

Type: [SystemInt32](#)

mikroBUS socket number.

*remdev* (Optional)

Type: [IO.RemoteDevice](#)

Remote I/O server device object.

## ◀ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.RemotelO.Expand Namespace](#)

# Board Properties

The [Board](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying MCP23S17 device object.

[Top](#)

## See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand Namespace](#)

# Boarddevice Property

Returns the underlying MCP23S17 device object.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.Expand](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device device { get; }
```

### Property Value

Type: [Device](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand Namespace](#)

# Board Methods

The [Board](#) type exposes the following members.

## ▪ Methods

	<b>Name</b>	<b>Description</b>
	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
	<a href="#">Reset</a>	Issue hardware reset to the MCP23S17.

[Top](#)

## ▪ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand Namespace](#)

# BoardGPIO Method

Factory function for creating GPIO pins.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.Expand](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO(
    int channel,
    Direction dir,
    bool state = false
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

MCP23S17 channel number (0 to 15).

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO pin object.

## ↳ See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand Namespace](#)

---

# BoardReset Method

Issue hardware reset to the MCP23S17.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.Expand](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Reset()
```

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand Namespace](#)

# IO.Devices.ClickBoards.RemotelIO.Expand2 Namespace

Mikroelektronika Expand 2 Click MIKROE-1838 Services

## ▪ Classes

Class	Description
 <a href="#">Board</a>	Encapsulates the Mikroelektronika Expand 2 Click Board.

# Board Class

Encapsulates the Mikroelektronika Expand 2 Click Board.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ClickBoards.RemoteIO.Expand2Board](#)

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Board
```

The [Board](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Board</a>	Constructor for a single Expand 2 click.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying MCP23017 device object.

[Top](#)

## ◀ Methods

	Name	Description
≡	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
≡	<a href="#">Reset</a>	Issue hardware reset to the MCP23017.

[Top](#)

## ◀ Fields

	Name	Description
◆ <b>S</b>	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.ClickBoards.RemoteIO.Expand2 Namespace](#)

# Board Constructor

Constructor for a single Expand 2 click.

**Namespace:** [IO.Devices.ClickBoards.RemotelIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Board(  
    int socknum,  
    int addr = 32,  
    Device remdev = null  
)
```

## Parameters

*socknum*

Type: [SystemInt32](#)

mikroBUS socket number.

*addr* (Optional)

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*remdev* (Optional)

Type: [IO.RemoteDevice](#)

Remote I/O server device object.

## ◀ See Also

[Reference](#)

[Board Class](#)

## IO.Devices.ClickBoards.RemoteIO.Expand2 Namespace

---

# Board Properties

The [Board](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying MCP23017 device object.

[Top](#)

## See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand2 Namespace](#)

# Boarddevice Property

Returns the underlying MCP23017 device object.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device device { get; }
```

### Property Value

Type: [Device](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand2 Namespace](#)

# Board Methods

The [Board](#) type exposes the following members.

## ▪ Methods

	Name	Description
	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
	<a href="#">Reset</a>	Issue hardware reset to the MCP23017.

[Top](#)

## ▪ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand2 Namespace](#)

# BoardGPIO Method

Factory function for creating GPIO pins.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO(
    int channel,
    Direction dir,
    bool state = false
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

MCP23017 channel number (0 to 15).

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO pin object.

## ↳ See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.RemoTeIO.Expand2 Namespace](#)

---

# BoardReset Method

Issue hardware reset to the MCP23017.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public void Reset()
```

## ▪ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand2 Namespace](#)

# Board Fields

The [Board](#) type exposes the following members.

## ↳ Fields

	Name	Description
	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand2 Namespace](#)

# BoardDefaultAddress Field

Default I<sup>2</sup>C slave address.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const int DefaultAddress = 32
```

### Field Value

Type: [Int32](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.Expand2 Namespace](#)

# IO.Devices.ClickBoards.RemoteIO.PWM Namespace

Mikroelektronika PWM Click MIKROE-1898 Services

## ▪ Classes

Class	Description
 <a href="#">Board</a>	Encapsulates the Mikroelektronika PWM Click Board. <a href="#">MIKROE-1898</a> .

# Board Class

Encapsulates the Mikroelektronika PWM Click Board. [MIKROE-1898](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ClickBoards.RemoteIO.PWMBoard](#)

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Board
```

The [Board](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Board</a>	Constructor for a single PWM click.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">dev</a>	Returns the underlying PCA9685 device object.

[Top](#)

## ◀ Methods

	Name	Description
≡	<a href="#">GPIO</a>	Factory function for creating GPIO output pins.
≡	<a href="#">PWM</a>	Factory function for creating PWM outputs.
≡	<a href="#">Servo</a>	Factory function for creating servo outputs.

[Top](#)

## ◀ Fields

	Name	Description
• s	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.ClickBoards.RemoteIO.PWM Namespace](#)

# Board Constructor

Constructor for a single PWM click.

**Namespace:** [IO.Devices.ClickBoards.RemotelIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Board(  
    int socknum,  
    int freq,  
    int addr = 64,  
    Device remdev = null  
)
```

## Parameters

*socknum*

Type: [SystemInt32](#)

mikroBUS socket number.

*freq*

Type: [SystemInt32](#)

PWM pulse frequency in Hz.

*addr* (Optional)

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*remdev* (Optional)

Type: [IO.RemoteDevice](#)

Remote I/O device object.

## ↳ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.RemoTeIO.PWM Namespace](#)

---

# Board Properties

The [Board](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">dev</a>	Returns the underlying PCA9685 device object.

[Top](#)

## See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.PWM Namespace](#)

# Boarddev Property

Returns the underlying PCA9685 device object.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device dev { get; }
```

### Property Value

Type: [Device](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.PWM Namespace](#)

# Board Methods

The [Board](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">GPIO</a>	Factory function for creating GPIO output pins.
	<a href="#">PWM</a>	Factory function for creating PWM outputs.
	<a href="#">Servo</a>	Factory function for creating servo outputs.

[Top](#)

## See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.PWM Namespace](#)

# BoardGPIO Method

Factory function for creating GPIO output pins.

**Namespace:** [IO.Devices.ClickBoards.RemotelIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO(  
    int channel,  
    bool state = false  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

PCA9685 output channel number.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO output pin object.

## ◀ See Also

**Reference**

[Board Class](#)

## IO.Devices.ClickBoards.RemoteIO.PWM Namespace

---

# BoardPWM Method

Factory function for creating PWM outputs.

**Namespace:** [IO.Devices.ClickBoards.RemotelIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Output PWM(  
    int channel,  
    double dutycycle = 0  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

PCA9685 output channel number.

*dutycycle* (Optional)

Type: [SystemDouble](#)

Initial PWM output duty cycle.

## Return Value

Type: [Output](#)

PWM output object.

## ◀ See Also

**Reference**

[Board Class](#)

## IO.Devices.ClickBoards.RemoteIO.PWM Namespace

---

# BoardServo Method

Factory function for creating servo outputs.

**Namespace:** [IO.Devices.ClickBoards.RemotelIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public Output Servo(  
    int channel,  
    double position = 0  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

PCA9685 output channel number.

*position* (Optional)

Type: [SystemDouble](#)

Initial servo position.>

## Return Value

Type: [Output](#)

Servo output object.

## ↳ See Also

**Reference**

[Board Class](#)

## IO.Devices.ClickBoards.RemoteIO.PWM Namespace

---

# Board Fields

The [Board](#) type exposes the following members.

## Fields

	Name	Description
	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.PWM Namespace](#)

# BoardDefaultAddress Field

Default I<sup>2</sup>C slave address.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const byte DefaultAddress = 64
```

### Field Value

Type: [Byte](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.PWM Namespace](#)

# IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace

Mikroelektronika 7Seg Click MIKROE-1201 Services

## Classes

Class	Description
 <a href="#">Board</a>	Encapsulates the Mikroelektronika 7Seg Click Board. <a href="#">MIKROE-1201</a> .

## Enumerations

Enumeration	Description
 <a href="#">BoardBase</a>	Numeral systems.
 <a href="#">BoardZeroBlanking</a>	Zero blanking modes.

# Board Class

Encapsulates the Mikroelektronika 7Seg Click Board. [MIKROE-1201](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ClickBoards.RemoteIO.SevenSegmentBoard](#)

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Board
```

The [Board](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Board</a>	Constructor for a single 7seg click.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">blanking</a>	Zero blanking mode. Allowed values are <a href="#">None</a> , <a href="#">Leading</a> , and <a href="#">Full</a> .



**brightness** Write-only property for setting the brightness of the display. Allowed values are 0.0 to 100.0 percent.



**leftdp** Write-only property for setting the left digit decimal point.



**radix** Numerical base or radix. Allowed values are **Decimal** and **Hexadecimal**.



**rightdp** Write-only property for setting the right digit decimal point.



**state** Write-only property for setting the state of the display. Allowed values are 0 to 99 for decimal mode and 0 to 255 for hexadecimal mode.

[Top](#)

## ◀ Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">Clear</a>	Clear the display.
≡	<a href="#">Reset</a>	Issue hardware reset to the 74HC595 shift register chain.

[Top](#)

## ◀ Remarks

The **MISOakaSDI** pin should be removed from the 7seg click, because it is not tri-state and will interfere with other devices on the same SPI

bus.

## See Also

### Reference

[IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace](#)

# Board Constructor

Constructor for a single 7seg click.

**Namespace:** [IO.Devices.ClickBoards.RemotelO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Board(  
    int socket,  
    BoardBase radix = BoardBase.Decimal,  
    BoardZeroBlanking blanking = BoardZeroBlanking  
    int pwmfreq = 100,  
    Device remdev = null  
)
```

## Parameters

*socket*

Type: [SystemInt32](#)

mikroBUS socket number.

*radix* (Optional)

Type: [IO.Devices.ClickBoards.RemotelO.SevenSegmentBoardBase](#)

Numerical base or radix. Allowed values are [Decimal](#) and [Hexadecimal](#).

*blanking* (Optional)

Type: [IO.Devices.ClickBoards.RemotelO.SevenSegmentBoardZeroBlanking](#)

Zero blanking. Allowed values are [None](#), [Leading](#), and [Full](#).

*pwmfreq* (Optional)

Type: [SystemInt32](#)

PWM frequency. Set to zero to use GPIO instead of PWM.

*remdev* (Optional)

Type: [IO.RemoteDevice](#)

Remote I/O server device object.

## See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace](#)

# Board Properties

The [Board](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">blinking</a>	Zero blanking mode. Allowed values are <a href="#">None</a> , <a href="#">Leading</a> , and <a href="#">Full</a> .
	<a href="#">brightness</a>	Write-only property for setting the brightness of the display. Allowed values are 0.0 to 100.0 percent.
	<a href="#">leftdp</a>	Write-only property for setting the left digit decimal point.
	<a href="#">radix</a>	Numerical base or radix. Allowed values are <a href="#">Decimal</a> and <a href="#">Hexadecimal</a> .
	<a href="#">rightdp</a>	Write-only property for setting the right digit decimal point.
	<a href="#">state</a>	Write-only property for setting the state of the display. Allowed values are 0 to 99 for decimal mode and 0 to 255 for hexadecimal mode.

[Top](#)

## ↳ See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.RemoTeIO.SevenSegment Namespace](#)

# Boardblanking Property

Zero blanking mode. Allowed values are `None`, `Leading`, and `Full`.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public BoardZeroBlanking blanking { get; set; }
```

## Property Value

Type: [BoardZeroBlanking](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace](#)

# Boardbrightness Property

Write-only property for setting the brightness of the display. Allowed values are 0.0 to 100.0 percent.

**Namespace:** [IO.Devices.ClickBoards.RemotelO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public double brightness { set; }
```

## Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemotelO.SevenSegment Namespace](#)

# Boardleftdp Property

Write-only property for setting the left digit decimal point.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public bool leftdp { set; }
```

### Property Value

Type: [Boolean](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace](#)

# Boardradix Property

Numerical base or radix. Allowed values are [Decimal](#) and [Hexadecimal](#).

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public BoardBase radix { get; set; }
```

### Property Value

Type: [BoardBase](#)

## ▪ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace](#)

# Boardrightdp Property

Write-only property for setting the right digit decimal point.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public bool rightdp { set; }
```

### Property Value

Type: [Boolean](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace](#)

# Boardstate Property

Write-only property for setting the state of the display. Allowed values are 0 to 99 for decimal mode and 0 to 255 for hexadecimal mode.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public int state { set; }
```

## Property Value

Type: [Int32](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace](#)

# Board Methods

The [Board](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Clear</a>	Clear the display.
	<a href="#">Reset</a>	Issue hardware reset to the 74HC595 shift register chain.

[Top](#)

## See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace](#)

# BoardClear Method

Clear the display.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Clear()
```

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace](#)

# BoardReset Method

Issue hardware reset to the 74HC595 shift register chain.

**Namespace:** [IO.Devices.ClickBoards.RemoteIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Reset()
```

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.RemoteIO.SevenSegment Namespace](#)

# BoardBase Enumeration

Numerical systems.

**Namespace:** [IO.Devices.ClickBoards.RemotelO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum Base
```

## ▪ Members

Member name	Value	Description
Decimal	0	Base 10.
Hexadecimal	1	Base 16.

## ▪ See Also

### Reference

[IO.Devices.ClickBoards.RemotelO.SevenSegment Namespace](#)

# BoardZeroBlanking Enumeration

Zero blanking modes.

**Namespace:** [IO.Devices.ClickBoards.RemotelO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum ZeroBlanking
```

## ▪ Members

Member name	Value	Description
None	0	No zero blanking.
Leading	1	Leading zero blanking.
Full	2	Full zero blanking.

## ▪ See Also

### Reference

[IO.Devices.ClickBoards.RemotelO.SevenSegment Namespace](#)

# IO.Devices.ClickBoards.SimpleIO.ADAC Namespace

Mikroelektronika ADAC Click MIKROE-2690 Services

## ↳ Classes

Class	Description
 <a href="#">Board</a>	Encapsulates the Mikroelektronika ADAC Click Board. <a href="#">MIKROE-2690</a> .

# Board Class

Encapsulates the Mikroelektronika ADAC Click Board. [MIKROE-2690](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ClickBoards.SimpleIO.ADACBoard](#)

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Board
```

The [Board](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Board</a>	Constructor for a single ADAC click.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying AD5593R device object.

[Top](#)

## ◀ Methods

	Name	Description
≡	<a href="#">ADC</a>	Factory function for creating ADC inputs.
≡	<a href="#">DAC</a>	Factory function for creating DAC outputs.
≡	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
≡	<a href="#">Reset</a>	Issue hardware reset to the AD5593R.

[Top](#)

## ◀ Fields

	Name	Description
◆ <b>S</b>	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.ClickBoards.SimpleIO.ADAC Namespace](#)

# Board Constructor

Constructor for a single ADAC click.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Board(  
    int socknum,  
    int addr = 16  
)
```

## Parameters

*socknum*

Type: [SystemInt32](#)

mikroBUS socket number.

*addr* (Optional)

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

## ◀ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.ADAC Namespace](#)

# Board Properties

The [Board](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying AD5593R device object.

[Top](#)

## See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.ADAC Namespace](#)

# Boarddevice Property

Returns the underlying AD5593R device object.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device device { get; }
```

### Property Value

Type: [Device](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.ADAC Namespace](#)

# Board Methods

The [Board](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">ADC</a>	Factory function for creating ADC inputs.
	<a href="#">DAC</a>	Factory function for creating DAC outputs.
	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
	<a href="#">Reset</a>	Issue hardware reset to the AD5593R.

[Top](#)

## See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.ADAC Namespace](#)

# BoardADC Method

Factory function for creating ADC inputs.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample ADC(  
    int channel  
)
```

### Parameters

*channel*

Type: [SystemInt32](#)

AD5593R I/O channel number (0 to 7).

### Return Value

Type: [Sample](#)

ADC input object.

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.ADAC Namespace](#)

# BoardDAC Method

Factory function for creating DAC outputs.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample DAC(  
    int channel,  
    int sample = 0  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

AD5593R I/O channel number (0 to 7).

*sample* (Optional)

Type: [SystemInt32](#)

Initial DAC output sample.

## Return Value

Type: [Sample](#)

DAC output object.

## ◀ See Also

**Reference**

[Board Class](#)

## IO.Devices.ClickBoards.SimpleIO.ADAC Namespace

---

# BoardGPIO Method

Factory function for creating GPIO pins.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO(
    int channel,
    Direction dir,
    bool state = false
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

AD5593R I/O channel number (0 to 7).

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO pin object.

## ↳ See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.ADAC Namespace](#)

---

# BoardReset Method

Issue hardware reset to the AD5593R.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public void Reset()
```

## ▪ See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.ADAC Namespace](#)

# Board Fields

The [Board](#) type exposes the following members.

## ↳ Fields

	Name	Description
	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.ADAC Namespace](#)

# BoardDefaultAddress Field

Default I<sup>2</sup>C slave address.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.ADAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public const byte DefaultAddress = 16
```

### Field Value

Type: [Byte](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.ADAC Namespace](#)

# IO.Devices.ClickBoards.SimpleIO.Expand Namespace

Mikroelektronika Expand Click MIKROE-951 Services

## ▪ Classes

Class	Description
 <a href="#">Board</a>	Encapsulates the Mikroelektronika Expand Click Board.

# Board Class

Encapsulates the Mikroelektronika Expand Click Board.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ClickBoards.SimpleIO.ExpandBoard](#)

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Board
```

The [Board](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Board</a>	Constructor for a single Expand 2 click.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying MCP23S17 device object.

[Top](#)

## ◀ Methods

	Name	Description
	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
	<a href="#">Reset</a>	Issue hardware reset to the MCP23S17.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.ClickBoards.SimpleIO.Expand Namespace](#)

# Board Constructor

Constructor for a single Expand 2 click.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Board(  
            int socknum  
)
```

## Parameters

*socknum*

Type: [SystemInt32](#)

mikroBUS socket number.

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand Namespace](#)

# Board Properties

The [Board](#) type exposes the following members.

## ▪ Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying MCP23S17 device object.

[Top](#)

## ▪ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand Namespace](#)

# Boarddevice Property

Returns the underlying MCP23S17 device object.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device device { get; }
```

### Property Value

Type: [Device](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand Namespace](#)

# Board Methods

The [Board](#) type exposes the following members.

## ▪ Methods

	Name	Description
	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
	<a href="#">Reset</a>	Issue hardware reset to the MCP23S17.

[Top](#)

## ▪ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand Namespace](#)

# BoardGPIO Method

Factory function for creating GPIO pins.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO(
    int channel,
    Direction dir,
    bool state = false
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

MCP23S17 channel number (0 to 15).

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO pin object.

## ↳ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand Namespace](#)

---

# BoardReset Method

Issue hardware reset to the MCP23S17.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public void Reset()
```

## ▪ See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand Namespace](#)

# IO.Devices.ClickBoards.SimpleIO.Expand2 Namespace

Mikroelektronika Expand 2 Click MIKROE-1838 Services

## Classes

Class	Description
 Board	Encapsulates the Mikroelektronika Expand 2 Click Board.

# Board Class

Encapsulates the Mikroelektronika Expand 2 Click Board.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ClickBoards.SimpleIO.Expand2Board](#)

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Board
```

The [Board](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Board</a>	Constructor for a single Expand 2 click.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying MCP23017 device object.

[Top](#)

## ◀ Methods

	Name	Description
≡	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
≡	<a href="#">Reset</a>	Issue hardware reset to the MCP23017.

[Top](#)

## ◀ Fields

	Name	Description
◆ <b>S</b>	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.ClickBoards.SimpleIO.Expand2 Namespace](#)

# Board Constructor

Constructor for a single Expand 2 click.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Board(  
    int socknum,  
    int addr = 32  
)
```

## Parameters

*socknum*

Type: [SystemInt32](#)

mikroBUS socket number.

*addr* (Optional)

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

## ◀ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand2 Namespace](#)

# Board Properties

The [Board](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">device</a>	Returns the underlying MCP23017 device object.

[Top](#)

## See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand2 Namespace](#)

# Boarddevice Property

Returns the underlying MCP23017 device object.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device device { get; }
```

### Property Value

Type: [Device](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand2 Namespace](#)

# Board Methods

The [Board](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">GPIO</a>	Factory function for creating GPIO pins.
	<a href="#">Reset</a>	Issue hardware reset to the MCP23017.

[Top](#)

## See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand2 Namespace](#)

# BoardGPIO Method

Factory function for creating GPIO pins.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO(
    int channel,
    Direction dir,
    bool state = false
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

MCP23017 channel number (0 to 15).

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO pin object.

## ↳ See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand2 Namespace](#)

---

# BoardReset Method

Issue hardware reset to the MCP23017.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Reset()
```

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand2 Namespace](#)

# Board Fields

The [Board](#) type exposes the following members.

## ↳ Fields

	Name	Description
	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand2 Namespace](#)

# BoardDefaultAddress Field

Default I<sup>2</sup>C slave address.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.Expand2](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public const int DefaultAddress = 32
```

## Field Value

Type: [Int32](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.Expand2 Namespace](#)

# IO.Devices.ClickBoards.SimpleIO.PWM Namespace

Mikroelektronika PWM Click MIKROE-1898 Services

## ↳ Classes

Class	Description
 Board	Encapsulates the Mikroelektronika PWM Click Board. <a href="#">MIKROE-1898</a> .

# Board Class

Encapsulates the Mikroelektronika PWM Click Board. [MIKROE-1898](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ClickBoards.SimpleIO.PWMBoard](#)

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Board
```

The [Board](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Board</a>	Constructor for a single PWM click.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">dev</a>	Returns the underlying PCA9685 device object.

[Top](#)

## ◀ Methods

	Name	Description
≡	<a href="#">GPIO</a>	Factory function for creating GPIO output pins.
≡	<a href="#">PWM</a>	Factory function for creating PWM outputs.
≡	<a href="#">Servo</a>	Factory function for creating servo outputs.

[Top](#)

## ◀ Fields

	Name	Description
• s	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.ClickBoards.SimpleIO.PWM Namespace](#)

# Board Constructor

Constructor for a single PWM click.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Board(  
    int socknum,  
    int freq,  
    int addr = 64  
)
```

## Parameters

*socknum*

Type: [SystemInt32](#)

mikroBUS socket number.

*freq*

Type: [SystemInt32](#)

PWM pulse frequency in Hz.

*addr* (Optional)

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

## ◀ See Also

[Reference](#)

[Board Class](#)

## IO.Devices.ClickBoards.SimpleIO.PWM Namespace

---

# Board Properties

The [Board](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">dev</a>	Returns the underlying PCA9685 device object.

[Top](#)

## See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.PWM Namespace](#)

# Boarddev Property

Returns the underlying PCA9685 device object.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device dev { get; }
```

### Property Value

Type: [Device](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.PWM Namespace](#)

# Board Methods

The [Board](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">GPIO</a>	Factory function for creating GPIO output pins.
	<a href="#">PWM</a>	Factory function for creating PWM outputs.
	<a href="#">Servo</a>	Factory function for creating servo outputs.

[Top](#)

## See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.PWM Namespace](#)

# BoardGPIO Method

Factory function for creating GPIO output pins.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO(  
    int channel,  
    bool state = false  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

PCA9685 output channel number.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO output pin object.

## ↳ See Also

**Reference**

[Board Class](#)

## IO.Devices.ClickBoards.SimpleIO.PWM Namespace

---

# BoardPWM Method

Factory function for creating PWM outputs.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Output PWM(  
    int channel,  
    double dutycycle = 0  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

PCA9685 output channel number.

*dutycycle* (Optional)

Type: [SystemDouble](#)

Initial PWM output duty cycle.

## Return Value

Type: [Output](#)

PWM output object.

## ◀ See Also

**Reference**

[Board Class](#)

## IO.Devices.ClickBoards.SimpleIO.PWM Namespace

---

# BoardServo Method

Factory function for creating servo outputs.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public Output Servo(  
    int channel,  
    double position = 0  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

PCA9685 output channel number.

*position* (Optional)

Type: [SystemDouble](#)

Initial servo position.>

## Return Value

Type: [Output](#)

Servo output object.

## ↳ See Also

**Reference**

[Board Class](#)

## IO.Devices.ClickBoards.SimpleIO.PWM Namespace

---

# Board Fields

The [Board](#) type exposes the following members.

## Fields

	Name	Description
	<a href="#">DefaultAddress</a>	Default I <sup>2</sup> C slave address.

[Top](#)

## See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.PWM Namespace](#)

# BoardDefaultAddress Field

Default I<sup>2</sup>C slave address.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public const byte DefaultAddress = 64
```

### Field Value

Type: [Byte](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.PWM Namespace](#)

# IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace

Mikroelektronika 7Seg Click MIKROE-1201 Services

## Classes

Class	Description
 <a href="#">Board</a>	Encapsulates the Mikroelektronika 7Seg Click Board. <a href="#">MIKROE-1201</a> .

## Enumerations

Enumeration	Description
 <a href="#">BoardBase</a>	Numeral systems.
 <a href="#">BoardZeroBlanking</a>	Zero blanking modes.

# Board Class

Encapsulates the Mikroelektronika 7Seg Click Board. [MIKROE-1201](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ClickBoards.SimpleIO.SevenSegmentBoard](#)

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Board
```

The [Board](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Board</a>	Constructor for a single 7seg click.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">blanking</a>	Zero blanking mode. Allowed values are <a href="#">None</a> , <a href="#">Leading</a> , and <a href="#">Full</a> .

	brightness	Write-only property for setting the brightness of the display. Allowed values are 0.0 to 100.0 percent.
	leftdp	Write-only property for setting the left digit decimal point.
	radix	Numerical base or radix. Allowed values are <b>Decimal</b> and <b>Hexadecimal</b> .
	rightdp	Write-only property for setting the right digit decimal point.
	state	Write-only property for setting the state of the display. Allowed values are 0 to 99 for decimal mode and 0 to 255 for hexadecimal mode.

[Top](#)

## Methods

	Name	Description
	Clear	Clear the display.

[Top](#)

## Remarks

The **MISOakaSDI** pin should be removed from the 7seg click, because it is not tri-state and will interfere with other devices on the same SPI bus.

## See Also

## Reference

### [IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# Board Constructor

Constructor for a single 7seg click.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Board(  
    int socket,  
    BoardBase radix = BoardBase.Decimal,  
    BoardZeroBlanking blanking = BoardZeroBlanking  
    int pwmfreq = 100  
)
```

## Parameters

*socket*

Type: [SystemInt32](#)

mikroBUS socket number.

*radix* (Optional)

Type: [IO.Devices.ClickBoards.SimpleIO.SevenSegmentBoardBase](#)

Numerical base or radix. Allowed values are [Decimal](#) and

[Hexadecimal](#).

*blanking* (Optional)

Type: [IO.Devices.ClickBoards.SimpleIO.SevenSegmentBoardZeroBlanking](#)

Zero blanking. Allowed values are [None](#), [Leading](#), and [Full](#).

*pwmfreq* (Optional)

Type: [SystemInt32](#)

PWM frequency. Set to zero to use GPIO instead of PWM.

## ◀ See Also

**Reference**

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# Board Properties

The [Board](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">blinking</a>	Zero blanking mode. Allowed values are <a href="#">None</a> , <a href="#">Leading</a> , and <a href="#">Full</a> .
	<a href="#">brightness</a>	Write-only property for setting the brightness of the display. Allowed values are 0.0 to 100.0 percent.
	<a href="#">leftdp</a>	Write-only property for setting the left digit decimal point.
	<a href="#">radix</a>	Numerical base or radix. Allowed values are <a href="#">Decimal</a> and <a href="#">Hexadecimal</a> .
	<a href="#">rightdp</a>	Write-only property for setting the right digit decimal point.
	<a href="#">state</a>	Write-only property for setting the state of the display. Allowed values are 0 to 99 for decimal mode and 0 to 255 for hexadecimal mode.

[Top](#)

## ↳ See Also

[Reference](#)

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

---

# Boardblanking Property

Zero blanking mode. Allowed values are `None`, `Leading`, and `Full`.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public BoardZeroBlanking blanking { get; set; }
```

## Property Value

Type: [BoardZeroBlanking](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# Boardbrightness Property

Write-only property for setting the brightness of the display. Allowed values are 0.0 to 100.0 percent.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public double brightness { set; }
```

## Property Value

Type: [Double](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# Boardleftdp Property

Write-only property for setting the left digit decimal point.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public bool leftdp { set; }
```

### Property Value

Type: [Boolean](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# Boardradix Property

Numerical base or radix. Allowed values are [Decimal](#) and [Hexadecimal](#).

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public BoardBase radix { get; set; }
```

## Property Value

Type: [BoardBase](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# Boardrightdp Property

Write-only property for setting the right digit decimal point.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public bool rightdp { set; }
```

### Property Value

Type: [Boolean](#)

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# Boardstate Property

Write-only property for setting the state of the display. Allowed values are 0 to 99 for decimal mode and 0 to 255 for hexadecimal mode.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public int state { set; }
```

## Property Value

Type: [Int32](#)

## ↳ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# Board Methods

The [Board](#) type exposes the following members.

## ▪ Methods

	Name	Description
	<a href="#">Clear</a>	Clear the display.

[Top](#)

## ▪ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# BoardClear Method

Clear the display.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Clear()
```

## ◀ See Also

### Reference

[Board Class](#)

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# BoardBase Enumeration

Numerical systems.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum Base
```

## ▪ Members

Member name	Value	Description
Decimal	0	Base 10.
Hexadecimal	1	Base 16.

## ▪ See Also

### Reference

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# BoardZeroBlanking Enumeration

Zero blanking modes.

**Namespace:** [IO.Devices.ClickBoards.SimpleIO.SevenSegment](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum ZeroBlanking
```

## ▪ Members

Member name	Value	Description
None	0	No zero blanking.
Leading	1	Leading zero blanking.
Full	2	Full zero blanking.

## ▪ See Also

### Reference

[IO.Devices.ClickBoards.SimpleIO.SevenSegment Namespace](#)

# IO.Devices.Grove.ADC

## Namespace

Seeed Studio Grove I<sup>2</sup>C ADC (ADC121C021) Services

### ◀ Classes

Class	Description
 Device	Encapsulates the Seeed Studio Grove I <sup>2</sup> C ADC (ADC121C021).

# Device Class

Encapsulates the Seeed Studio Grove I<sup>2</sup>C ADC (ADC121C021).

## ► Inheritance Hierarchy

[SystemObject](#) [IO.Devices.Grove.ADCDevice](#)

**Namespace:** [IO.Devices.Grove.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ► Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Device
```

The [Device](#) type exposes the following members.

## ► Constructors

	Name	Description
	<a href="#">Device</a>	Constructor for a Seeed Studio Grove I <sup>2</sup> C ADC (ADC121C021).

[Top](#)

## ► Properties

	Name	Description
	<a href="#">voltage</a>	Read-only property returning an analog

input voltage measurement.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.Grove.ADC Namespace](#)

---

# Device Constructor

Constructor for a Seeed Studio Grove I<sup>2</sup>C ADC (ADC121C021).

**Namespace:** [IO.Devices.Grove.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus,  
    byte addr = 80  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup>C bus object.

*addr* (Optional)

Type: [SystemByte](#)

I<sup>2</sup>C device address.

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.Grove.ADC Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">voltage</a>	Read-only property returning an analog input voltage measurement.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.Grove.ADC Namespace](#)

# Devicevoltage Property

Read-only property returning an analog input voltage measurement.

**Namespace:** [IO.Devices.Grove.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public double voltage { get; }
```

### Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.Grove.ADC Namespace](#)

# IO.Devices.Grove.Temperature Namespace

Seeed Studio Grove Temperature Sensor (thermistor) Services

## ↳ Classes

Class	Description
 Device	Encapsulates the Seeed Studio Grove Temperature Sensor (thermistor).

# Device Class

Encapsulates the Seeed Studio Grove Temperature Sensor (thermistor).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.Grove.TemperatureDevice](#)

**Namespace:** [IO.Devices.Grove.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Device : Sensor
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Device</a>	Constructor for a Seeed Studio Grove Temperature Sensor (thermistor).

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">Celsius</a>	Read-only property returning the

temperature in degrees Celsius.

---



**Fahrenheit** Read-only property returning the temperature in degrees Fahrenheit.

---



**Kelvins** Read-only property returning the temperature in Kelvins.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.Grove.Temperature Namespace](#)

# Device Constructor

Constructor for a Seeed Studio Grove Temperature Sensor (thermistor).

**Namespace:** [IO.Devices.Grove.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Voltage Vin,  
    double Vcc = 3.3  
)
```

## Parameters

*Vin*

Type: [IO.Interfaces.ADCVoltage](#)

Voltage input object.

*Vcc* (Optional)

Type: [SystemDouble](#)

Reference voltage.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.Grove.Temperature Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius.
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit.
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins.

[Top](#)

## See Also

### Reference

[Device Class](#)

[IO.Devices.Grove.Temperature Namespace](#)

# DeviceCelsius Property

Read-only property returning the temperature in degrees Celsius.

**Namespace:** [IO.Devices.Grove.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public double Celsius { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorCelsius](#)

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.Grove.Temperature Namespace](#)

# DeviceFahrenheit Property

Read-only property returning the temperature in degrees Fahrenheit.

**Namespace:** [IO.Devices.Grove.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public double Fahrenheit { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorFahrenheit](#)

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.Grove.Temperature Namespace](#)

# DeviceKelvins Property

Read-only property returning the temperature in Kelvins.

**Namespace:** [IO.Devices.Grove.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public double Kelvins { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorKelvins](#)

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.Grove.Temperature Namespace](#)

# IO.Devices.Grove.Temperature\_Humidity Namespace

Seeed Studio Grove Temperature and Humdity Sensor (TH02) Services.

## ▪ Classes

	Class	Description
	<a href="#">Device</a>	Encapsulate the Seeed Studio Grove Temperature and Humidity Sensor (TH02).

# Device Class

Encapsulate the Seeed Studio Grove Temperature and Humidity Sensor (TH02).

## ▪ Inheritance Hierarchy

```
SystemObject IO.Devices.TH02Device  
IO.Devices.Grove.Temperature_HumidityDevice
```

**Namespace:** [IO.Devices.Grove.Temperature\\_Humidity](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Device : Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

Name	Description
 <a href="#">Device</a>	Constructor for a Seeed Studio Grove Temperature and Humidity Sensor (TH02).

[Top](#)

## ▪ Properties

	<b>Name</b>	<b>Description</b>
	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius. (Inherited from <a href="#">Device</a> .)
	<a href="#">DeviceID</a>	Read-only property returning the device ID. (Inherited from <a href="#">Device</a> .)
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit. (Inherited from <a href="#">Device</a> .)
	<a href="#">Humidity</a>	Read-only property returning the percentage relative humidity. (Inherited from <a href="#">Device</a> .)
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins. (Inherited from <a href="#">Device</a> .)

[Top](#)

## See Also

### Reference

[IO.Devices.Grove.Temperature\\_Humidity Namespace](#)

# Device Constructor

Constructor for a Seeed Studio Grove Temperature and Humidity Sensor (TH02).

**Namespace:** [IO.Devices.Grove.Temperature\\_Humidity](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)  
I<sup>2</sup> bus object.

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.Grove.Temperature\\_Humidity Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius. (Inherited from <a href="#">Device</a> .)
	<a href="#">DeviceID</a>	Read-only property returning the device ID. (Inherited from <a href="#">Device</a> .)
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit. (Inherited from <a href="#">Device</a> .)
	<a href="#">Humidity</a>	Read-only property returning the percentage relative humidity. (Inherited from <a href="#">Device</a> .)
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins. (Inherited from <a href="#">Device</a> .)

[Top](#)

## See Also

## Reference

Device Class

IO.Devices.Grove.Temperature\_Humidity Namespace

---

# IO.Devices.HDC1080 Namespace

HDC1080 I<sup>2</sup>C Temperature/Humidity Sensor Services

## » Classes

Class	Description
 <a href="#">Device</a>	Encapsulates the HDC1080 temperature and humidity sensor.

# Device Class

Encapsulates the HDC1080 temperature and humidity sensor.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.HDC1080Device](#)  
[IO.Devices.Pmod.HYGRODevice](#)

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public class Device : Sensor, Sensor
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Device</a>	Constructor for an HDC1080 temperature and humidity sensor object.

[Top](#)

## ▪ Properties

	Name	Description
--	------	-------------

	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius.
	<a href="#">DeviceID</a>	Read-only property returning the device ID.
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit.
	<a href="#">Humidity</a>	Read-only property returning the percentage relative humidity.
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins.
	<a href="#">ManufacturerID</a>	Read-only property returning the manufacturer ID.

[Top](#)

## Methods

	Name	Description
	<a href="#">Read</a>	Read from an HDC1080 device register.
	<a href="#">Write</a>	Write to an HDC1080 device register.

[Top](#)

## Fields

	Name	Description
	<a href="#">RegConfiguration</a>	Configuration Register

address.

• S	RegDeviceID	Device ID Register address.
• S	RegHumidity	Humidity Register address.
• S	RegManufacturerID	Manufacturer ID Register address.
• S	RegSerialNumberFirst	Serial Number First Bits Register address.
• S	RegSerialNumberLast	Serial Number Last Bits Register address.
• S	RegSerialNumberMid	Serial Number Middle Bits Register address.
• S	RegTemperature	Temperature Register address.

[Top](#)

## See Also

### Reference

[IO.Devices.HDC1080 Namespace](#)

# Device Constructor

Constructor for an HDC1080 temperature and humidity sensor object.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup>C bus controller.

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius.
	<a href="#">DeviceID</a>	Read-only property returning the device ID.
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit.
	<a href="#">Humidity</a>	Read-only property returning the percentage relative humidity.
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins.
	<a href="#">ManufacturerID</a>	Read-only property returning the manufacturer ID.

[Top](#)

## See Also

### Reference

Device Class  
IO.Devices.HDC1080 Namespace

---

# DeviceCelsius Property

Read-only property returning the temperature in degrees Celsius.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public double Celsius { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorCelsius](#)

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceDeviceID Property

Read-only property returning the device ID.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public ushort DeviceID { get; }
```

### Property Value

Type: [UInt16](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceFahrenheit Property

Read-only property returning the temperature in degrees Fahrenheit.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public double Fahrenheit { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorFahrenheit](#)

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceHumidity Property

Read-only property returning the percentage relative humidity.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public double Humidity { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorHumidity](#)

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceKelvins Property

Read-only property returning the temperature in Kelvins.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public double Kelvins { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorKelvins](#)

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceManufacturerID Property

Read-only property returning the manufacturer ID.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public ushort ManufacturerID { get; }
```

### Property Value

Type: [UInt16](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Read</a>	Read from an HDC1080 device register.
	<a href="#">Write</a>	Write to an HDC1080 device register.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceRead Method

Read from an HDC1080 device register.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public ushort Read(  
    byte reg  
)
```

## Parameters

*reg*

Type: [SystemByte](#)

8-bit register address.

## Return Value

Type: [UInt16](#)

16-bit register data

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceWrite Method

Write to an HDC1080 device register.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Write(  
    byte reg,  
    ushort data  
)
```

## Parameters

*reg*

Type: [SystemByte](#)

8-bit register address.

*data*

Type: [SystemUInt16](#)

16-bit register data.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# Device Fields

The [Device](#) type exposes the following members.

## Fields

Name	Description
<a href="#"><b>RegConfiguration</b></a>	Configuration Register address.
<a href="#"><b>RegDeviceID</b></a>	Device ID Register address.
<a href="#"><b>RegHumidity</b></a>	Humidity Register address.
<a href="#"><b>RegManufacturerID</b></a>	Manufacturer ID Register address.
<a href="#"><b>RegSerialNumberFirst</b></a>	Serial Number First Bits Register address.
<a href="#"><b>RegSerialNumberLast</b></a>	Serial Number Last Bits Register address.
<a href="#"><b>RegSerialNumberMid</b></a>	Serial Number Middle Bits Register address.
<a href="#"><b>RegTemperature</b></a>	Temperature Register address.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceRegConfiguration Field

Configuration Register address.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte RegConfiguration = 2
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceRegDeviceID Field

Device ID Register address.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte RegDeviceID = 255
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceRegHumidity Field

Humidity Register address.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte RegHumidity = 1
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceRegManufacturerID Field

Manufacturer ID Register address.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte RegManufacturerID = 254
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceRegSerialNumberFirst Field

Serial Number First Bits Register address.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte RegSerialNumberFirst = 251
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceRegSerialNumberLast Field

Serial Number Last Bits Register address.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte RegSerialNumberLast = 253
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceRegSerialNumberMid Field

Serial Number Middle Bits Register address.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte RegSerialNumberMid = 252
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# DeviceRegTemperature Field

Temperature Register address.

**Namespace:** [IO.Devices.HDC1080](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte RegTemperature = 0
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.HDC1080 Namespace](#)

# IO.Devices.MCP23017

## Namespace

MCP23017 I<sup>2</sup>C GPIO Expander Device Services.

### ↳ Classes

Class	Description
 Device	Encapsulates the MCP23017 I <sup>2</sup> C I/O GPIO Expander.

# Device Class

Encapsulates the MCP23017 I<sup>2</sup>C I/O GPIO Expander.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.MCP23017Device](#)

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Device</a>	Constructor for a single MCP23017 device.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">Direction</a>	Data Direction Property (16 bits). Bits 0

to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.

	<a href="#">DirectionA</a>	Port A Data Direction Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">DirectionB</a>	Port B Data Direction Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">Polarity</a>	Data Polarity Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.
	<a href="#">PolarityA</a>	Port A Data Polarity Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">PolarityB</a>	Port B Data Polarity Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">Port</a>	Port Data Property (16 bites). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B.
	<a href="#">PortA</a>	Port A Data Property (8 bits).
	<a href="#">PortB</a>	Port B Data Property (8 bits).
	<a href="#">Pullups</a>	Input Pullup Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=high impedance and 1=100k pullup.

	<a href="#">PullupsA</a>	Port A Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.
	<a href="#">PullupsB</a>	Port B Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.

[Top](#)

## ◀ Methods

	Name	Description
	<a href="#">GPIO_Create</a>	Create an MCP23017 GPIO pin object.

[Top](#)

## ◀ Fields

	Name	Description
	<a href="#">MaxChannel</a>	Maximum I/O channel number.
	<a href="#">MinChannel</a>	Minimum I/O channel number.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.MCP23017 Namespace](#)

# Device Constructor

Constructor for a single MCP23017 device.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus,  
    int addr  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup>C bus controller object.

*addr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Direction</a>	Data Direction Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.
	<a href="#">DirectionA</a>	Port A Data Direction Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">DirectionB</a>	Port B Data Direction Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">Polarity</a>	Data Polarity Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.
	<a href="#">PolarityA</a>	Port A Data Polarity Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">PolarityB</a>	Port B Data Polarity Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">Port</a>	Port Data Property (16 bites). Bits 0 to 7 correspond to PORT A and bits 8 to

15 correspond to PORT B.

	<a href="#">PortA</a>	Port A Data Property (8 bits).
	<a href="#">PortB</a>	Port B Data Property (8 bits).
	<a href="#">Pullups</a>	Input Pullup Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=high impedance and 1=100k pullup.
	<a href="#">PullupsA</a>	Port A Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.
	<a href="#">PullupsB</a>	Port B Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DeviceDirection Property

Data Direction Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public uint Direction { get; set; }
```

## Property Value

Type: [UInt32](#)

## ▪ Remarks

This property follows the industry standard convention for data direction bit polarity (1=output) rather than the MCP23017 [IODIR](#) register polarity (0=output).

## ▪ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DeviceDirectionA Property

Port A Data Direction Property (8 bits). For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public byte DirectionA { get; set; }
```

## Property Value

Type: [Byte](#)

## ▪ Remarks

This property follows the industry standard convention for data direction bit polarity (1=output) rather than the MCP23017 [IODIRA](#) register polarity (0=output).

## ▪ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DeviceDirectionB Property

Port B Data Direction Property (8 bits). For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ► Syntax

C#    VB    F#

[Copy](#)

```
public byte DirectionB { get; set; }
```

### Property Value

Type: [Byte](#)

## ► Remarks

This property follows the industry standard convention for data direction bit polarity (1=output) rather than the MCP23017 [IODIRA](#) register polarity (0=output).

## ► See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DevicePolarity Property

Data Polarity Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public uint Polarity { get; set; }
```

## Property Value

Type: [UInt32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DevicePolarityA Property

Port A Data Polarity Property (8 bits). For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public byte PolarityA { get; set; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DevicePolarityB Property

Port B Data Polarity Property (8 bits). For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte PolarityB { get; set; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DevicePort Property

Port Data Property (16 bites). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public uint Port { get; set; }
```

## Property Value

Type: [UInt32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DevicePortA Property

Port A Data Property (8 bits).

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte PortA { get; set; }
```

### Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DevicePortB Property

Port B Data Property (8 bits).

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte PortB { get; set; }
```

### Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DevicePullups Property

Input Pullup Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=high impedance and 1=100k pullup.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public uint Pullups { get; set; }
```

## Property Value

Type: [UInt32](#)

## ↳ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DevicePullupsA Property

Port A Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public byte PullupsA { get; set; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DevicePullupsB Property

Port B Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte PullupsB { get; set; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## ▪ Methods

	Name	Description
	<a href="#">GPIO_Create</a>	Create an MCP23017 GPIO pin object.

[Top](#)

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DeviceGPIO\_Create Method

Create an MCP23017 GPIO pin object.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO_Create(  
    int channel,  
    Direction dir,  
    bool state = false  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

MCP23017 channel number (0 to 15).

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin data direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO pin object.

## ↳ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

---

# Device Fields

The [Device](#) type exposes the following members.

## Fields

	Name	Description
• 	<a href="#">MaxChannel</a>	Maximum I/O channel number.
• 	<a href="#">MinChannel</a>	Minimum I/O channel number.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DeviceMaxChannel Field

Maximum I/O channel number.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MaxChannel = 15
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# DeviceMinChannel Field

Minimum I/O channel number.

**Namespace:** [IO.Devices.MCP23017](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MinChannel = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23017 Namespace](#)

# IO.Devices.MCP23017.GPIO Namespace

MCP23017 I<sup>2</sup>C GPIO Expander GPIO Pin Services.

## « Classes

Class	Description
 <a href="#">Pin</a>	Encapsulates MCP23017 GPIO pins.

# Pin Class

Encapsulates MCP23017 GPIO pins.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.MCP23017.GPIOPin](#)

**Namespace:** [IO.Devices.MCP23017.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Pin : Pin
```

The [Pin](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Pin</a>	Create a single MCP23017 GPIO pin.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.MCP23017.GPIO Namespace](#)

---

# Pin Constructor

Create a single MCP23017 GPIO pin.

**Namespace:** [IO.Devices.MCP23017.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin(  
    Device dev,  
    int channel,  
    Direction dir,  
    bool state = false  
)
```

## Parameters

*dev*

Type: [IO.Devices.MCP23017Device](#)

MCP23017 device object.

*channel*

Type: [SystemInt32](#)

MCP23017 I/O channel number.

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin data direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## See Also

**Reference**

[Pin Class](#)

[IO.Devices.MCP23017.GPIO Namespace](#)

---

# Pin Properties

The [Pin](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## See Also

### Reference

[Pin Class](#)

[IO.Devices.MCP23017.GPIO Namespace](#)

# Pinstate Property

Read/Write GPIO state property.

**Namespace:** [IO.Devices.MCP23017.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public bool state { get; set; }
```

### Property Value

Type: [Boolean](#)

### Implements

[Pinstate](#)

## ▪ See Also

[Reference](#)

[Pin Class](#)

[IO.Devices.MCP23017.GPIO Namespace](#)

# IO.Devices.MCP23S17

## Namespace

MCP23S17 SPI GPIO Expander Device Services.

### ↳ Classes

Class	Description
 Device	Encapsulates the MCP23S17 SPI I/O GPIO Expander.

# Device Class

Encapsulates the MCP23S17 SPI I/O GPIO Expander.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.MCP23S17Device](#)

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Device</a>	Constructor for a single MCP23S17 device.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">Direction</a>	Data Direction Property (16 bits). Bits 0

to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.

	<a href="#">DirectionA</a>	Port A Data Direction Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">DirectionB</a>	Port B Data Direction Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">Polarity</a>	Data Polarity Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.
	<a href="#">PolarityA</a>	Port A Data Polarity Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">PolarityB</a>	Port B Data Polarity Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">Port</a>	Port Data Property (16 bites). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B.
	<a href="#">PortA</a>	Port A Data Property (8 bits).
	<a href="#">PortB</a>	Port B Data Property (8 bits).
	<a href="#">Pullups</a>	Input Pullup Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=high impedance and 1=100k pullup.

---

	<a href="#">PullupsA</a>	Port A Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.
	<a href="#">PullupsB</a>	Port B Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.

---

[Top](#)

## ◀ Methods

Name	Description
 <a href="#">GPIO_Create</a>	Create an MCP23S17 GPIO pin object.

[Top](#)

## ◀ Fields

Name	Description
 <a href="#">MaxChannel</a>	Maximum I/O channel number.
 <a href="#">MinChannel</a>	Minimum I/O channel number.
 <a href="#">SPI_Frequency</a>	SPI maximum clock frequency in Hz.
 <a href="#">SPI_Mode</a>	SPI transfer mode.
 <a href="#">SPI_WordSize</a>	SPI transaction word size.

[Top](#)

## See Also

### Reference

[IO.Devices.MCP23S17 Namespace](#)

---

# Device Constructor

Constructor for a single MCP23S17 device.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Device dev  
)
```

## Parameters

*dev*

Type: [IO.Interfaces.SPIDevice](#)

SPI slave device object.

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Direction</a>	Data Direction Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.
	<a href="#">DirectionA</a>	Port A Data Direction Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">DirectionB</a>	Port B Data Direction Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">Polarity</a>	Data Polarity Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.
	<a href="#">PolarityA</a>	Port A Data Polarity Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">PolarityB</a>	Port B Data Polarity Property (8 bits). For each bit, 0=input and 1=output.
	<a href="#">Port</a>	Port Data Property (16 bites). Bits 0 to 7 correspond to PORT A and bits 8 to

15 correspond to PORT B.

	<a href="#">PortA</a>	Port A Data Property (8 bits).
	<a href="#">PortB</a>	Port B Data Property (8 bits).
	<a href="#">Pullups</a>	Input Pullup Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=high impedance and 1=100k pullup.
	<a href="#">PullupsA</a>	Port A Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.
	<a href="#">PullupsB</a>	Port B Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DeviceDirection Property

Data Direction Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public uint Direction { get; set; }
```

### Property Value

Type: [UInt32](#)

## ▪ Remarks

This property follows the industry standard convention for data direction bit polarity (1=output) rather than the MCP23S17 [IODIR](#) register polarity (0=output).

## ▪ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DeviceDirectionA Property

Port A Data Direction Property (8 bits). For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ► Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public byte DirectionA { get; set; }
```

## Property Value

Type: [Byte](#)

## ► Remarks

This property follows the industry standard convention for data direction bit polarity (1=output) rather than the MCP23S17 [IODIRA](#) register polarity (0=output).

## ► See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DeviceDirectionB Property

Port B Data Direction Property (8 bits). For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public byte DirectionB { get; set; }
```

### Property Value

Type: [Byte](#)

## ▪ Remarks

This property follows the industry standard convention for data direction bit polarity (1=output) rather than the MCP23S17 [IODIRA](#) register polarity (0=output).

## ▪ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DevicePolarity Property

Data Polarity Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public uint Polarity { get; set; }
```

## Property Value

Type: [UInt32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DevicePolarityA Property

Port A Data Polarity Property (8 bits). For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public byte PolarityA { get; set; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DevicePolarityB Property

Port B Data Polarity Property (8 bits). For each bit, 0=input and 1=output.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte PolarityB { get; set; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DevicePort Property

Port Data Property (16 bites). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▲ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public uint Port { get; set; }
```

## Property Value

Type: [UInt32](#)

## ▲ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DevicePortA Property

Port A Data Property (8 bits).

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public byte PortA { get; set; }
```

### Property Value

Type: [Byte](#)

## « See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DevicePortB Property

Port B Data Property (8 bits).

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte PortB { get; set; }
```

### Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DevicePullups Property

Input Pullup Property (16 bits). Bits 0 to 7 correspond to PORT A and bits 8 to 15 correspond to PORT B. For each bit, 0=high impedance and 1=100k pullup.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public uint Pullups { get; set; }
```

## Property Value

Type: [UInt32](#)

## ↳ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DevicePullupsA Property

Port A Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte PullupsA { get; set; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DevicePullupsB Property

Port B Input Pullup Property (16 bits). For each bit, 0=high impedance and 1=100k pullup.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte PullupsB { get; set; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## ▪ Methods

	Name	Description
	<a href="#">GPIO_Create</a>	Create an MCP23S17 GPIO pin object.

[Top](#)

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DeviceGPIO\_Create Method

Create an MCP23S17 GPIO pin object.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO_Create(  
    int channel,  
    Direction dir,  
    bool state = false  
)
```

## Parameters

*channel*

Type: [SystemInt32](#)

MCP23S17 channel number (0 to 15).

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin data direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO pin object.

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# Device Fields

The [Device](#) type exposes the following members.

## Fields

	Name	Description
• 	<a href="#">MaxChannel</a>	Maximum I/O channel number.
• 	<a href="#">MinChannel</a>	Minimum I/O channel number.
• 	<a href="#">SPI_Frequency</a>	SPI maximum clock frequency in Hz.
• 	<a href="#">SPI_Mode</a>	SPI transfer mode.
• 	<a href="#">SPI_WordSize</a>	SPI transaction word size.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DeviceMaxChannel Field

Maximum I/O channel number.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MaxChannel = 15
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DeviceMinChannel Field

Minimum I/O channel number.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MinChannel = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DeviceSPI\_Frequency Field

SPI maximum clock frequency in Hz.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int SPI_Frequency = 10000000
```

## Field Value

Type: [Int32](#)

## « Remarks

Guaranteed only for 2.7V to 5.5V and -40°C to +85°C.

## « See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DeviceSPI\_Mode Field

SPI transfer mode.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int SPI_Mode = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# DeviceSPI\_WordSize Field

SPI transaction word size.

**Namespace:** [IO.Devices.MCP23S17](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int SPI_WordSize = 8
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.MCP23S17 Namespace](#)

# IO.Devices.MCP23S17.GPIO Namespace

MCP23S17 SPI GPIO Expander GPIO Pin Services.

## ◀ Classes

Class	Description
 Pin	Encapsulates MCP23S17 GPIO pins.

# Pin Class

Encapsulates MCP23S17 GPIO pins.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.MCP23S17.GPIOPin](#)

**Namespace:** [IO.Devices.MCP23S17.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Pin : Pin
```

The [Pin](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Pin</a>	Create a single MCP23S17 GPIO pin.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.MCP23S17.GPIO Namespace](#)

# Pin Constructor

Create a single MCP23S17 GPIO pin.

**Namespace:** [IO.Devices.MCP23S17.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin(  
    Device dev,  
    int channel,  
    Direction dir,  
    bool state = false  
)
```

## Parameters

*dev*

Type: [IO.Devices.MCP23S17Device](#)

MCP23S17 device object.

*channel*

Type: [SystemInt32](#)

MCP23S17 I/O channel number.

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin data direction.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## See Also

**Reference**

[Pin Class](#)

[IO.Devices.MCP23S17.GPIO Namespace](#)

---

# Pin Properties

The [Pin](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## See Also

### Reference

[Pin Class](#)

[IO.Devices.MCP23S17.GPIO Namespace](#)

# Pinstate Property

Read/Write GPIO state property.

**Namespace:** [IO.Devices.MCP23S17.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public bool state { get; set; }
```

### Property Value

Type: [Boolean](#)

### Implements

[Pinstate](#)

## ▪ See Also

[Reference](#)

[Pin Class](#)

[IO.Devices.MCP23S17.GPIO Namespace](#)

# IO.Devices.PCA8574 Namespace

PCA8574 (and similar) I<sup>2</sup>C GPIO Expander Device Services

## ► Classes

Class	Description
 <a href="#">Device</a>	Encapsulates PCA8574 (and similar) I <sup>2</sup> C GPIO Expanders.

# Device Class

Encapsulates PCA8574 (and similar) I<sup>2</sup>C GPIO Expanders.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.PCA8574Device](#)

**Namespace:** [IO.Devices.PCA8574](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Device</a>	Constructor for a PCA8574 (or similar) GPIO Expander.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">Latch</a>	This read-only property returns the last

value written to the output latch.

[Top](#)

## ◀ Methods

	Name	Description
	<a href="#">Read</a>	Return actual state of the GPIO pins.
	<a href="#">Write</a>	Write all GPIO pins.

[Top](#)

## ◀ Fields

	Name	Description
 	<a href="#">MAX_PINS</a>	The number of available GPIO pins per chip.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.PCA8574 Namespace](#)

# Device Constructor

Constructor for a PCA8574 (or similar) GPIO Expander.

**Namespace:** [IO.Devices.PCA8574](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus,  
    int addr,  
    byte states = 255  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup>C bus controller.

*addr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*states (Optional)*

Type: [SystemByte](#)

Initial output states.

## ◀ See Also

### Reference

Device Class  
IO.Devices.PCA8574 Namespace

---

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Latch</a>	This read-only property returns the last value written to the output latch.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.PCA8574 Namespace](#)

# DeviceLatch Property

This read-only property returns the last value written to the output latch.

**Namespace:** [IO.Devices.PCA8574](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte Latch { get; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA8574 Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Read</a>	Return actual state of the GPIO pins.
	<a href="#">Write</a>	Write all GPIO pins.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.PCA8574 Namespace](#)

# DeviceRead Method

Return actual state of the GPIO pins.

**Namespace:** [IO.Devices.PCA8574](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte Read()
```

### Return Value

Type: [Byte](#)

Pin states (MSB = GPIO7).

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA8574 Namespace](#)

# DeviceWrite Method

Write all GPIO pins.

**Namespace:** [IO.Devices.PCA8574](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Write(  
    byte data  
)
```

## Parameters

*data*

Type: [SystemByte](#)

Data to write to pins (MSB = GPIO7).

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA8574 Namespace](#)

# Device Fields

The [Device](#) type exposes the following members.

## Fields

Name	Description
 <a href="#">MAX_PINS</a>	The number of available GPIO pins per chip.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.PCA8574 Namespace](#)

# DeviceMAX\_PINS Field

The number of available GPIO pins per chip.

**Namespace:** [IO.Devices.PCA8574](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MAX_PINS = 8
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA8574 Namespace](#)

# IO.Devices.PCA8574.GPIO Namespace

PCA8574 (and similar) I<sup>2</sup>C GPIO Expander GPIO Pin Services

## ↳ Classes

Class	Description
 Pin	Encapsulates PCA8574 (and similar) I <sup>2</sup> C GPIO Expander pins.

Munts Technologies Linux Simple I/O Library .Net Standard 2.0 Class  
Library 2.2020.135.1

# Pin Class

Encapsulates PCA8574 (and similar) I<sup>2</sup>C GPIO Expander pins.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.PCA8574.GPIOPin](#)

**Namespace:** [IO.Devices.PCA8574.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Pin : Pin
```

The [Pin](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Pin</a>	Constructor for a single GPIO pin.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## ↳ See Also

**Reference**

[IO.Devices.PCA8574.GPIO Namespace](#)

# Pin Constructor

Constructor for a single GPIO pin.

**Namespace:** [IO.Devices.PCA8574.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin(  
    Device dev,  
    int num,  
    Direction dir,  
    bool state = false  
)
```

## Parameters

*dev*

Type: [IO.Devices.PCA8574Device](#)  
PCA8574 (or similar) device.

*num*

Type: [SystemInt32](#)  
GPIO pin number.

*dir*

Type: [IO.Interfaces.GPIODirection](#)  
Data direction.

*state (Optional)*

Type: [SystemBoolean](#)  
Initial GPIO output state.

## See Also

**Reference**

[Pin Class](#)

[IO.Devices.PCA8574.GPIO Namespace](#)

---

Munts Technologies Linux Simple I/O Library .Net Standard 2.0 Class  
Library 2.2020.135.1

# Pin Properties

The [Pin](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## See Also

### Reference

[Pin Class](#)

[IO.Devices.PCA8574.GPIO Namespace](#)

# Pinstate Property

Read/Write GPIO state property.

**Namespace:** [IO.Devices.PCA8574.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public bool state { get; set; }
```

### Property Value

Type: [Boolean](#)

### Implements

[Pinstate](#)

## « See Also

[Reference](#)

[Pin Class](#)

[IO.Devices.PCA8574.GPIO Namespace](#)

# IO.Devices.PCA9534 Namespace

PCA9534 (and similar) I<sup>2</sup>C GPIO Expander Device Services

## » Classes

Class	Description
 <a href="#">Device</a>	Encapsulates PCA9534 (and similar) I <sup>2</sup> C GPIO Expanders.

# Device Class

Encapsulates PCA9534 (and similar) I<sup>2</sup>C GPIO Expanders.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.PCA9534Device](#)

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Device</a>	Constructor for a PCA9534 (or similar) GPIO Expander.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">Config</a>	This read-only property returns the last

value written to the configuration register.



**Latch** This read-only property returns the last value written to the output port register.

[Top](#)

## Methods

	Name	Description
≡	<a href="#">Read</a>	Return actual state of the GPIO pins.
≡	<a href="#">Read(Byte)</a>	Read from the specified PCA9534 device register.
≡	<a href="#">Write(Byte)</a>	Write all GPIO pins.
≡	<a href="#">Write(Byte, Byte)</a>	Write to the specified PCA9534 device register.

[Top](#)

## Fields

	Name	Description
• <b>s</b>	<a href="#">AllInputs</a>	Configure all pins as inputs.
• <b>s</b>	<a href="#">AllNormal</a>	Configure all inputs as normal polarity.
• <b>s</b>	<a href="#">AllOff</a>	Turn all outputs off.
• <b>s</b>	<a href="#">AllOutputs</a>	Configure all pins as outputs.

• <b>S</b>	<a href="#">ConfigurationReg</a>	Configuration Register address.
• <b>S</b>	<a href="#">InputPolarityReg</a>	Input Port Polarity Register address.
• <b>S</b>	<a href="#">InputPortReg</a>	Input Port Register address.
• <b>S</b>	<a href="#">MAX_PINS</a>	The number of available GPIO pins per chip.
• <b>S</b>	<a href="#">OutputPortReg</a>	Output Port Register address.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.PCA9534 Namespace](#)

# Device Constructor

Constructor for a PCA9534 (or similar) GPIO Expander.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus,  
    int addr,  
    byte config = 255,  
    byte states = 0  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup>C bus controller.

*addr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*config* (Optional)

Type: [SystemByte](#)

GPIO pin configuration.

*states* (Optional)

Type: [SystemByte](#)

Initial output states.

## See Also

**Reference**

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Config</a>	This read-only property returns the last value written to the configuration register.
	<a href="#">Latch</a>	This read-only property returns the last value written to the output port register.

[Top](#)

## See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceConfig Property

This read-only property returns the last value written to the configuration register.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte Config { get; }
```

### Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceLatch Property

This read-only property returns the last value written to the output port register.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte Latch { get; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# Device Methods

## ◀ Methods

	Name	Description
≡	<a href="#">Read</a>	Return actual state of the GPIO pins.
≡	<a href="#">Read(Byte)</a>	Read from the specified PCA9534 device register.
≡	<a href="#">Write(Byte)</a>	Write all GPIO pins.
≡	<a href="#">Write(Byte, Byte)</a>	Write to the specified PCA9534 device register.

[Top](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceRead Method

## ↳ Overload List

	Name	Description
	<a href="#">Read</a>	Return actual state of the GPIO pins.
	<a href="#">Read(Byte)</a>	Read from the specified PCA9534 device register.

[Top](#)

## ↳ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceRead Method

Return actual state of the GPIO pins.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte Read()
```

### Return Value

Type: [Byte](#)

Pin states (MSB = GPIO7).

## ◀ See Also

### Reference

[Device Class](#)

[Read Overload](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceRead Method (Byte)

Read from the specified PCA9534 device register.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte Read(  
    byte addr  
)
```

## Parameters

*addr*

Type: [SystemByte](#)

Register address.

## Return Value

Type: [Byte](#)

Register contents.

## ◀ See Also

### Reference

[Device Class](#)

[Read Overload](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceWrite Method

## ↳ Overload List

	Name	Description
≡	<a href="#">Write(Byte)</a>	Write all GPIO pins.
≡	<a href="#">Write(Byte, Byte)</a>	Write to the specified PCA9534 device register.

[Top](#)

## ↳ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceWrite Method (Byte)

Write all GPIO pins.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Write(  
    byte data  
)
```

## Parameters

*data*

Type: [SystemByte](#)

Data to write to pins (MSB = GPIO7).

## ◀ See Also

### Reference

[Device Class](#)

[Write Overload](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceWrite Method (Byte, Byte)

Write to the specified PCA9534 device register.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Write(  
    byte addr,  
    byte data  
)
```

## Parameters

*addr*

Type: [SystemByte](#)

Register address.

*data*

Type: [SystemByte](#)

Data to written.

## ◀ See Also

### Reference

[Device Class](#)

[Write Overload](#)

[IO.Devices.PCA9534 Namespace](#)

# Device Fields

The [Device](#) type exposes the following members.

## Fields

	Name	Description
• <a href="#"><b>s</b></a>	<a href="#">AllInputs</a>	Configure all pins as inputs.
• <a href="#"><b>s</b></a>	<a href="#">AllNormal</a>	Configure all inputs as normal polarity.
• <a href="#"><b>s</b></a>	<a href="#">AllOff</a>	Turn all outputs off.
• <a href="#"><b>s</b></a>	<a href="#">AllOutputs</a>	Configure all pins as outputs.
• <a href="#"><b>s</b></a>	<a href="#">ConfigurationReg</a>	Configuration Register address.
• <a href="#"><b>s</b></a>	<a href="#">InputPolarityReg</a>	Input Port Polarity Register address.
• <a href="#"><b>s</b></a>	<a href="#">InputPortReg</a>	Input Port Register address.
• <a href="#"><b>s</b></a>	<a href="#">MAX_PINS</a>	The number of available GPIO pins per chip.
• <a href="#"><b>s</b></a>	<a href="#">OutputPortReg</a>	Output Port Register address.

[Top](#)

## See Also

## Reference

### Device Class

#### IO.Devices.PCA9534 Namespace

---

# DeviceAllInputs Field

Configure all pins as inputs.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte AllInputs = 255
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceAllNormal Field

Configure all inputs as normal polarity.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte AllNormal = 0
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceAllOff Field

Turn all outputs off.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte AllOff = 0
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceAllOutputs Field

Configure all pins as outputs.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte AllOutputs = 0
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceConfigurationReg Field

Configuration Register address.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte ConfigurationReg = 3
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceInputPolarityReg Field

Input Port Polarity Register address.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte InputPolarityReg = 2
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceInputPortReg Field

Input Port Register address.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte InputPortReg = 0
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceMAX\_PINS Field

The number of available GPIO pins per chip.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MAX_PINS = 8
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# DeviceOutputPortReg Field

Output Port Register address.

**Namespace:** [IO.Devices.PCA9534](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const byte OutputPortReg = 1
```

### Field Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9534 Namespace](#)

# IO.Devices.PCA9534.GPIO Namespace

PCA9534 (and similar) I<sup>2</sup>C GPIO Expander GPIO Pin Services

## ↳ Classes

Class	Description
 Pin	Encapsulates PCA9534 (and similar) I <sup>2</sup> C GPIO Expander pins.

# Pin Class

Encapsulates PCA9534 (and similar) I<sup>2</sup>C GPIO Expander pins.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.PCA9534.GPIOPin](#)

**Namespace:** [IO.Devices.PCA9534.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Pin : Pin
```

The [Pin](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Pin</a>	Constructor for a single GPIO pin.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## ↳ See Also

### Reference

[IO.Devices.PCA9534.GPIO Namespace](#)

---

# Pin Constructor

Constructor for a single GPIO pin.

**Namespace:** [IO.Devices.PCA9534.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin(  
    Device dev,  
    int num,  
    Direction dir,  
    bool state = false  
)
```

## Parameters

*dev*

Type: [IO.Devices.PCA9534Device](#)  
PCA9534 (or similar) device.

*num*

Type: [SystemInt32](#)  
GPIO pin number.

*dir*

Type: [IO.Interfaces.GPIODirection](#)  
Data direction.

*state (Optional)*

Type: [SystemBoolean](#)  
Initial GPIO output state.

## See Also

**Reference**

[Pin Class](#)

[IO.Devices.PCA9534.GPIO Namespace](#)

---

# Pin Properties

The [Pin](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## See Also

### Reference

[Pin Class](#)

[IO.Devices.PCA9534.GPIO Namespace](#)

# Pinstate Property

Read/Write GPIO state property.

**Namespace:** [IO.Devices.PCA9534.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public bool state { get; set; }
```

### Property Value

Type: [Boolean](#)

### Implements

[Pinstate](#)

## ◀ See Also

[Reference](#)

[Pin Class](#)

[IO.Devices.PCA9534.GPIO Namespace](#)

# IO.Devices.PCA9685 Namespace

PCA9685 I<sup>2</sup>C PWM Controller Device Services

## » Classes

Class	Description
 <a href="#">Device</a>	Encapsulates the PCA9685 I <sup>2</sup> C PWM Controller.

# Device Class

Encapsulates the PCA9685 I<sup>2</sup>C PWM Controller.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.PCA9685Device](#)

**Namespace:** [IO.Devices.PCA9685](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Device</a>	Constructor for a single PCA9685 device.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">Frequency</a>	Read-only property returning the configured PWM pulse frequency.

[Top](#)

## ◀ Methods

	Name	Description
≡	<a href="#">ReadChannel</a>	Read PCA9685 output channel data.
≡	<a href="#">WriteChannel</a>	Write PCA9685 output channel data.

[Top](#)

## ◀ Fields

	Name	Description
♦ <b>s</b>	<a href="#">INTERNAL_CLOCK</a>	Select internal 25 MHz clock oscillator.
♦ <b>s</b>	<a href="#">MAX_CHANNEL</a>	Maximum PCA9685 output channel number.
♦ <b>s</b>	<a href="#">MAX_CLOCK</a>	Maximum clock frequency.
♦ <b>s</b>	<a href="#">MIN_CHANNEL</a>	Minimum PCA9685 output channel number.
♦ <b>s</b>	<a href="#">MIN_CLOCK</a>	Minimum clock frequency.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.PCA9685 Namespace](#)



# Device Constructor

Constructor for a single PCA9685 device.

**Namespace:** [IO.Devices.PCA9685](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus,  
    int addr,  
    int freq = 50,  
    int clock = 0  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup>C bus controller object.

*addr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*freq* (Optional)

Type: [SystemInt32](#)

PWM pulse frequency. Default is 50 Hz.

*clock* (Optional)

Type: [SystemInt32](#)

PCA9685 clock source. Use [INTERNAL\\_CLOCK](#) to select the

internal 25 MHz clock generator.

## ◀ See Also

**Reference**

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Frequency</a>	Read-only property returning the configured PWM pulse frequency.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# DeviceFrequency Property

Read-only property returning the configured PWM pulse frequency.

**Namespace:** [IO.Devices.PCA9685](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int Frequency { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">ReadChannel</a>	Read PCA9685 output channel data.
	<a href="#">WriteChannel</a>	Write PCA9685 output channel data.

[Top](#)

## See Also

**Reference**

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# DeviceReadChannel Method

Read PCA9685 output channel data.

**Namespace:** [IO.Devices.PCA9685](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void ReadChannel(  
    byte channel,  
    ref byte[] data  
)
```

## Parameters

*channel*

Type: [SystemByte](#)

Output channel number.

*data*

Type: [SystemByte](#)

Output channel data (4 bytes).

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# DeviceWriteChannel Method

Write PCA9685 output channel data.

**Namespace:** [IO.Devices.PCA9685](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void WriteChannel(  
    byte channel,  
    byte[] data  
)
```

## Parameters

*channel*

Type: [SystemByte](#)

Output channel number.

*data*

Type: [SystemByte](#)

Output channel data.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# Device Fields

The [Device](#) type exposes the following members.

## Fields

	Name	Description
• S	<a href="#">INTERNAL_CLOCK</a>	Select internal 25 MHz clock oscillator.
• S	<a href="#">MAX_CHANNEL</a>	Maximum PCA9685 output channel number.
• S	<a href="#">MAX_CLOCK</a>	Maximum clock frequency.
• S	<a href="#">MIN_CHANNEL</a>	Minimum PCA9685 output channel number.
• S	<a href="#">MIN_CLOCK</a>	Minimum clock frequency.

[Top](#)

## See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# DeviceINTERNAL\_CLOCK Field

Select internal 25 MHz clock oscillator.

**Namespace:** [IO.Devices.PCA9685](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int INTERNAL_CLOCK = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# DeviceMAX\_CHANNEL Field

Maximum PCA9685 output channel number.

**Namespace:** [IO.Devices.PCA9685](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MAX_CHANNEL = 15
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# DeviceMAX\_CLOCK Field

Maximum clock frequency.

**Namespace:** [IO.Devices.PCA9685](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MAX_CLOCK = 50000000
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# DeviceMIN\_CHANNEL Field

Minimum PCA9685 output channel number.

**Namespace:** [IO.Devices.PCA9685](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MIN_CHANNEL = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# DeviceMIN\_CLOCK Field

Minimum clock frequency.

**Namespace:** [IO.Devices.PCA9685](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MIN_CLOCK = 1
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.PCA9685 Namespace](#)

# IO.Devices.PCA9685.GPIO Namespace

PCA9685 I<sup>2</sup>C PWM Controller GPIO Pin Services

## ↳ Classes

Class	Description
 Pin	Encapsulates PCA9685 GPIO outputs.

# Pin Class

Encapsulates PCA9685 GPIO outputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.PCA9685.GPIOPin](#)

**Namespace:** [IO.Devices.PCA9685.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Pin : Pin
```

The [Pin](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Pin</a>	Constructor for a single GPIO output pin.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO output state property.

[Top](#)

## ↳ See Also

### Reference

[IO.Devices.PCA9685.GPIO Namespace](#)

---

# Pin Constructor

Constructor for a single GPIO output pin.

**Namespace:** [IO.Devices.PCA9685.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin(  
    Device dev,  
    int channel,  
    bool state = false  
)
```

## Parameters

*dev*

Type: [IO.Devices.PCA9685Device](#)  
PCA9685 device object.

*channel*

Type: [SystemInt32](#)  
Output channel number.

*state (Optional)*

Type: [SystemBoolean](#)  
Initial GPIO output state.

## ◀ See Also

**Reference**

[Pin Class](#)

## IO.Devices.PCA9685.GPIO Namespace

---

# Pin Properties

The [Pin](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO output state property.

[Top](#)

## See Also

### Reference

[Pin Class](#)

[IO.Devices.PCA9685.GPIO Namespace](#)

# Pinstate Property

Read/Write GPIO output state property.

**Namespace:** [IO.Devices.PCA9685.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public bool state { get; set; }
```

### Property Value

Type: [Boolean](#)

### Implements

[Pinstate](#)

## ▪ See Also

[Reference](#)

[Pin Class](#)

[IO.Devices.PCA9685.GPIO Namespace](#)

# IO.Devices.PCA9685.PWM Namespace

PCA9685 I<sup>2</sup>C PWM Controller PWM Output Services

## ↳ Classes

Class	Description
 <a href="#">Output</a>	Encapsulates PCA9685 PWM outputs.

# Output Class

Encapsulates PCA9685 PWM outputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.PCA9685.PWMOutput](#)

**Namespace:** [IO.Devices.PCA9685.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Output : Output
```

The [Output](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Output</a>	Constructor for a single PWM output.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">dutycycle</a>	Write-only property for setting the PWM output duty cycle. Allowed values

are 0.0 to 100.0 percent.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.PCA9685.PWM Namespace](#)

---

# Output Constructor

Constructor for a single PWM output.

**Namespace:** [IO.Devices.PCA9685.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Output(  
    Device dev,  
    int channel,  
    double dutycycle = 0  
)
```

## Parameters

*dev*

Type: [IO.Devices.PCA9685Device](#)  
PCA9685 device object.

*channel*

Type: [SystemInt32](#)  
Output channel number.

*dutycycle (Optional)*

Type: [SystemDouble](#)  
Initial PWM output duty cycle.

## ◀ See Also

[Reference](#)  
[Output Class](#)

## IO.Devices.PCA9685.PWM Namespace

---

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">dutycycle</a>	Write-only property for setting the PWM output duty cycle. Allowed values are 0.0 to 100.0 percent.

[Top](#)

## See Also

[Reference](#)

[Output Class](#)

[IO.Devices.PCA9685.PWM Namespace](#)

# Outputdutycycle Property

Write-only property for setting the PWM output duty cycle. Allowed values are 0.0 to 100.0 percent.

**Namespace:** [IO.Devices.PCA9685.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▲ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public double dutycycle { set; }
```

### Property Value

Type: [Double](#)

### Implements

[Outputdutycycle](#)

## ▲ See Also

### Reference

[Output Class](#)

[IO.Devices.PCA9685.PWM Namespace](#)

# IO.Devices.PCA9685.Servo Namespace

PCA9685 I<sup>2</sup>C PWM Controller Servo Output Services

## ↳ Classes

Class	Description
 <a href="#">Output</a>	Encapsulates PCA9685 servo outputs.

# Output Class

Encapsulates PCA9685 servo outputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.PCA9685.ServoOutput](#)

**Namespace:** [IO.Devices.PCA9685.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Output : Output
```

The [Output](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Output</a>	Constructor for a single servo output.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">position</a>	Write-only property for setting the normalized servo position. Allowed

values are -0.0 to +1.0.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.PCA9685.Servo Namespace](#)

---

# Output Constructor

Constructor for a single servo output.

**Namespace:** [IO.Devices.PCA9685.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Output(  
    Device dev,  
    int channel,  
    double position = 0  
)
```

## Parameters

*dev*

Type: [IO.Devices.PCA9685Device](#)  
PCA9685 device object.

*channel*

Type: [SystemInt32](#)  
Output channel number.

*position (Optional)*

Type: [SystemDouble](#)  
Initial servo position.

## ◀ See Also

[Reference](#)

[Output Class](#)

## IO.Devices.PCA9685.Servo Namespace

---

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">position</a>	Write-only property for setting the normalized servo position. Allowed values are -0.0 to +1.0.

[Top](#)

## See Also

[Reference](#)

[Output Class](#)

[IO.Devices.PCA9685.Servo Namespace](#)

# Outputposition Property

Write-only property for setting the normalized servo position. Allowed values are -0.0 to +1.0.

**Namespace:** [IO.Devices.PCA9685.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▲ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public double position { set; }
```

### Property Value

Type: [Double](#)

### Implements

[Outputposition](#)

## ▲ See Also

### [Reference](#)

[Output Class](#)

[IO.Devices.PCA9685.Servo Namespace](#)

# IO.Devices.Pmod.HYGRO

## Namespace

Digilent Pmod HYGRO Temperature and Humdity Sensor (HDC1080) Services.

### ↳ Classes

Class	Description
 Device	Encapsulate the Digilent Pmod HYGRO Temperature and Humidity Sensor (HDC1080).

# Device Class

Encapsulate the Digilent Pmod HYGRO Temperature and Humidity Sensor (HDC1080).

## ▪ Inheritance Hierarchy

SystemObject IO.Devices.HDC1080Device  
IO.Devices.Pmod.HYGRODevice

**Namespace:** [IO.Devices.Pmod.HYGRO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

Copy

```
public class Device : Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

Name	Description
 <a href="#">Device</a>	Constructor for a Digilent Pmod HYGRO Temperature and Humidity Sensor (HDC1080).

[Top](#)

## ▪ Properties

	<b>Name</b>	<b>Description</b>
	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius. (Inherited from <a href="#">Device</a> .)
	<a href="#">DeviceID</a>	Read-only property returning the device ID. (Inherited from <a href="#">Device</a> .)
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit. (Inherited from <a href="#">Device</a> .)
	<a href="#">Humidity</a>	Read-only property returning the percentage relative humidity. (Inherited from <a href="#">Device</a> .)
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins. (Inherited from <a href="#">Device</a> .)
	<a href="#">ManufacturerID</a>	Read-only property returning the manufacturer ID. (Inherited from <a href="#">Device</a> .)

[Top](#)

## ◀ Methods

	<b>Name</b>	<b>Description</b>
	<a href="#">Read</a>	Read from an HDC1080 device register. (Inherited from <a href="#">Device</a> .)



## Write

Write to an HDC1080 device register.  
(Inherited from [Device](#).)

---

[Top](#)

## See Also

### Reference

[IO.Devices.Pmod.HYGRO Namespace](#)

---

# Device Constructor

Constructor for a Digilent Pmod HYGRO Temperature and Humidity Sensor (HDC1080).

**Namespace:** [IO.Devices.Pmod.HYGRO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup> bus object.

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.Pmod.HYGRO Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius. (Inherited from <a href="#">Device</a> .)
	<a href="#">DeviceID</a>	Read-only property returning the device ID. (Inherited from <a href="#">Device</a> .)
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit. (Inherited from <a href="#">Device</a> .)
	<a href="#">Humidity</a>	Read-only property returning the percentage relative humidity. (Inherited from <a href="#">Device</a> .)
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins. (Inherited from <a href="#">Device</a> .)
	<a href="#">ManufacturerID</a>	Read-only property returning the manufacturer ID.

(Inherited from [Device](#).)

---

[Top](#)

## ◀ See Also

**Reference**

[Device Class](#)

[IO.Devices.Pmod.HYGRO Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Read</a>	Read from an HDC1080 device register. (Inherited from <a href="#">Device</a> .)
	<a href="#">Write</a>	Write to an HDC1080 device register. (Inherited from <a href="#">Device</a> .)

[Top](#)

## See Also

**Reference**

[Device Class](#)

[IO.Devices.Pmod.HYGRO Namespace](#)

# IO.Devices.SN74HC595

## Namespace

SN74HC595 8-Bit Shift Register Device Services

### Classes

Class	Description
 Device	Encapsulates a chain of one or more SN74HC595 8-bit shift registers.

# Device Class

Encapsulates a chain of one or more SN74HC595 8-bit shift registers.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.SN74HC595Device](#)

**Namespace:** [IO.Devices.SN74HC595](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Device</a>	Constructor for a chain of one or more SN74HC595 shift registers.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">Length</a>	Read-only property returning the

number of stages in the chain.



**state** Read/Write shift register chain state property.

[Top](#)

## ◀ Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">ClrBit</a>	Clear a single bit in the shift register chain.
≡	<a href="#">ReadBit</a>	Read a single bit in the shift register chain.
≡	<a href="#">SetBit</a>	Set a single bit in the shift register chain.

[Top](#)

## ◀ Fields

	<b>Name</b>	<b>Description</b>
♦ <b>S</b>	<a href="#">SPI_MaxFreq</a>	SPI maximum clock frequency for the SNHC74HC595 shift register. (Most pessimistic datasheet limit at 2V.)
♦ <b>S</b>	<a href="#">SPI_Mode</a>	SPI clock mode for the SNHC74HC595 shift register.

[Top](#)

## ◀ See Also

## Reference

### IO.Devices.SN74HC595 Namespace

---

# Device Constructor

Constructor for a chain of one or more SN74HC595 shift registers.

**Namespace:** [IO.Devices.SN74HC595](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Device dev,  
    int stages = 1,  
    byte[] initialstate = null  
)
```

## Parameters

*dev*

Type: [IO.Interfaces.SPIDevice](#)

SPI device object.

*stages* (Optional)

Type: [SystemInt32](#)

Number of stages in the chain.

*initialstate* (Optional)

Type: [SystemByte](#)

Initial shift register chain state.

## ◀ See Also

[Reference](#)

[Device Class](#)

## IO.Devices.SN74HC595 Namespace

---

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Length</a>	Read-only property returning the number of stages in the chain.
	<a href="#">state</a>	Read/Write shift register chain state property.

[Top](#)

## See Also

### Reference

[Device Class](#)

[IO.Devices.SN74HC595 Namespace](#)

# DeviceLength Property

Read-only property returning the number of stages in the chain.

**Namespace:** [IO.Devices.SN74HC595](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int Length { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.SN74HC595 Namespace](#)

# Devicestate Property

Read/Write shift register chain state property.

**Namespace:** [IO.Devices.SN74HC595](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte[] state { get; set; }
```

## Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.SN74HC595 Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## Methods

	Name	Description
≡	<a href="#">ClrBit</a>	Clear a single bit in the shift register chain.
≡	<a href="#">ReadBit</a>	Read a single bit in the shift register chain.
≡	<a href="#">SetBit</a>	Set a single bit in the shift register chain.

[Top](#)

## See Also

### Reference

[Device Class](#)

[IO.Devices.SN74HC595 Namespace](#)

# DeviceClrBit Method

Clear a single bit in the shift register chain.

**Namespace:** [IO.Devices.SN74HC595](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void ClrBit(  
    int index,  
    byte mask  
)
```

## Parameters

*index*

Type: [SystemInt32](#)

Shift register stage number. Zero indicates the first register stage.

*mask*

Type: [SystemByte](#)

Shift register bit mask.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.SN74HC595 Namespace](#)

# DeviceReadBit Method

Read a single bit in the shift register chain.

**Namespace:** [IO.Devices.SN74HC595](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public bool ReadBit(  
    int index,  
    byte mask  
)
```

## Parameters

*index*

Type: [SystemInt32](#)

Shift register stage number. Zero indicates the first register stage.

*mask*

Type: [SystemByte](#)

Shift register bit mask.

## Return Value

Type: [Boolean](#)

Boolean bit value.

## ◀ See Also

### Reference

[Device Class](#)

## IO.Devices.SN74HC595 Namespace

---

# DeviceSetBit Method

Set a single bit in the shift register chain.

**Namespace:** [IO.Devices.SN74HC595](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void SetBit(  
    int index,  
    byte mask  
)
```

## Parameters

*index*

Type: [SystemInt32](#)

Shift register stage number. Zero indicates the first register stage.

*mask*

Type: [SystemByte](#)

Shift register bit mask.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.SN74HC595 Namespace](#)

# Device Fields

The [Device](#) type exposes the following members.

## Fields

Name	Description
 <a href="#">SPI_MaxFreq</a>	SPI maximum clock frequency for the SNHC74HC595 shift register. (Most pessimistic datasheet limit at 2V.)
 <a href="#">SPI_Mode</a>	SPI clock mode for the SNHC74HC595 shift register.

[Top](#)

## See Also

### Reference

[Device Class](#)

[IO.Devices.SN74HC595 Namespace](#)

# DeviceSPI\_MaxFreq Field

SPI maximum clock frequency for the SNHC74HC595 shift register.  
(Most pessimistic datasheet limit at 2V.)

**Namespace:** [IO.Devices.SN74HC595](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int SPI_MaxFreq = 4000000
```

## Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.SN74HC595 Namespace](#)

# DeviceSPI\_Mode Field

SPI clock mode for the SNHC74HC595 shift register.

**Namespace:** [IO.Devices.SN74HC595](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int SPI_Mode = 0
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[Device Class](#)

[IO.Devices.SN74HC595 Namespace](#)

# IO.Devices.SN74HC595.GPIO Namespace

SN74HC595 8-Bit Shift Register GPIO Pin Services

## ◀ Classes

Class	Description
 Pin	Encapsulates SN74HC595 GPIO outputs.

# Pin Class

Encapsulates SN74HC595 GPIO outputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.SN74HC595.GPIOPin](#)

**Namespace:** [IO.Devices.SN74HC595.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Pin : Pin
```

The [Pin](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Pin</a>	Constructor for a single GPIO output pin.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO pin state property.

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.SN74HC595.GPIO Namespace](#)

---

# Pin Constructor

Constructor for a single GPIO output pin.

**Namespace:** [IO.Devices.SN74HC595.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin(  
    Device dev,  
    int pos,  
    bool state = false  
)
```

## Parameters

*dev*

Type: [IO.Devices.SN74HC595Device](#)  
SN74HC595 device object.

*pos*

Type: [SystemInt32](#)  
Bit position, numbered left to right. Zero indicates the most significant bit of the first shift register stage.

*state (Optional)*

Type: [SystemBoolean](#)  
Initial GPIO output state.

## ◀ See Also

### Reference

Pin Class

IO.Devices.SN74HC595.GPIO Namespace

---

# Pin Properties

The [Pin](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO pin state property.

[Top](#)

## See Also

### Reference

[Pin Class](#)

[IO.Devices.SN74HC595.GPIO Namespace](#)

# Pinstate Property

Read/Write GPIO pin state property.

**Namespace:** [IO.Devices.SN74HC595.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public bool state { get; set; }
```

### Property Value

Type: [Boolean](#)

### Implements

[Pinstate](#)

## « See Also

[Reference](#)

[Pin Class](#)

[IO.Devices.SN74HC595.GPIO Namespace](#)

# IO.Devices.TH02 Namespace

TH02 I<sup>2</sup>C Temperature/Humidity Sensor Services

## ↳ Classes

Class	Description
 Device	Encapsulates the TH02 temperature and humidity sensor.

# Device Class

Encapsulates the TH02 temperature and humidity sensor.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.TH02Device](#)  
[IO.Devices.Grove.Temperature\\_HumidityDevice](#)

**Namespace:** [IO.Devices.TH02](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public class Device : Sensor, Sensor
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Device</a>	Constructor for an TH02 temperature and humidity sensor object.

[Top](#)

## ▪ Properties

	Name	Description
--	------	-------------

	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius.
	<a href="#">DeviceID</a>	Read-only property returning the device ID.
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit.
	<a href="#">Humidity</a>	Read-only property returning the percentage relative humidity.
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins.

[Top](#)

## See Also

### Reference

[IO.Devices.TH02 Namespace](#)

# Device Constructor

Constructor for an TH02 temperature and humidity sensor object.

**Namespace:** [IO.Devices.TH02](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup>C bus controller.

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.TH02 Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius.
	<a href="#">DeviceID</a>	Read-only property returning the device ID.
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit.
	<a href="#">Humidity</a>	Read-only property returning the percentage relative humidity.
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Devices.TH02 Namespace](#)

# DeviceCelsius Property

Read-only property returning the temperature in degrees Celsius.

**Namespace:** [IO.Devices.TH02](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public double Celsius { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorCelsius](#)

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.TH02 Namespace](#)

# DeviceDeviceID Property

Read-only property returning the device ID.

**Namespace:** [IO.Devices.TH02](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public byte DeviceID { get; }
```

### Property Value

Type: [Byte](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Devices.TH02 Namespace](#)

# DeviceFahrenheit Property

Read-only property returning the temperature in degrees Fahrenheit.

**Namespace:** [IO.Devices.TH02](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public double Fahrenheit { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorFahrenheit](#)

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.TH02 Namespace](#)

# DeviceHumidity Property

Read-only property returning the percentage relative humidity.

**Namespace:** [IO.Devices.TH02](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public double Humidity { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorHumidity](#)

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.TH02 Namespace](#)

# DeviceKelvins Property

Read-only property returning the temperature in Kelvins.

**Namespace:** [IO.Devices.TH02](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public double Kelvins { get; }
```

### Property Value

Type: [Double](#)

### Implements

[SensorKelvins](#)

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Devices.TH02 Namespace](#)

# IO.Devices.Thermistor Namespace

Thermistor Modeling Services

## ↳ Classes

Class	Description
 <a href="#">NTC_B</a>	Encapsulate an NTC thermistor.

# NTC\_B Class

Encapsulate an NTC thermistor.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.ThermistorNTC\\_B](#)

**Namespace:** [IO.Devices.Thermistor](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class NTC_B
```

The [NTC\\_B](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">NTC_B</a>	Constructor for a single NTC thermistor object instance.

[Top](#)

## ▪ Methods

	Name	Description
≡	<a href="#">Kelvins</a>	Kelvin temperature as a function of the

thermistor resistance.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Devices.Thermistor Namespace](#)

---

# NTC\_B Constructor

Constructor for a single NTC thermistor object instance.

**Namespace:** [IO.Devices.Thermistor](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public NTC_B(  
    double B,  
    double R0,  
    double T0 = 298.15  
)
```

## Parameters

*B*

Type: [SystemDouble](#)

Thermistor B parameter.

*R0*

Type: [SystemDouble](#)

Thermistor resistance in ohms at the specified reference temperature.

*T0* (Optional)

Type: [SystemDouble](#)

Thermistor reference temperature in Kelvins.

## ◀ See Also

### Reference

NTC\_B Class  
IO.Devices.Thermistor Namespace

---

# NTC\_B Methods

The [NTC\\_B](#) type exposes the following members.

## ▪ Methods

	Name	Description
	<a href="#">Kelvins</a>	Kelvin temperature as a function of the thermistor resistance.

[Top](#)

## ▪ See Also

[Reference](#)

[NTC\\_B Class](#)

[IO.Devices.Thermistor Namespace](#)

# NTC\_BKelvins Method

Kelvin temperature as a function of the thermistor resistance.

**Namespace:** [IO.Devices.Thermistor](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public double Kelvins(  
    double R  
)
```

### Parameters

R

Type: [SystemDouble](#)

Thermistor resistance in ohms.

### Return Value

Type: [Double](#)

Temperature in Kelvins.

## ◀ See Also

### Reference

[NTC\\_B Class](#)

[IO.Devices.Thermistor Namespace](#)

# IO.Devices.USB.Munts

## Namespace

Vendor and Product Identifiers for Munts Technologies

(<http://tech.munts.com>) USB devices

### Classes

	Class	Description
	<a href="#">HID</a>	USB device constants for Munts Technologies USB HID devices.
	<a href="#">Serial</a>	USB device constants for Munts Technologies USB serial port devices.

# HID Class

USB device constants for Munts Technologies USB HID devices.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.USB.MuntsHID](#)

**Namespace:** [IO.Devices.USB.Munts](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static class HID
```

The HID type exposes the following members.

## ▪ Fields

	Name	Description
• <a href="#">S</a>	<a href="#">Product</a>	Product ID for Munts Technologies USB hid devices.
• <a href="#">S</a>	<a href="#">Vendor</a>	Vendor ID for Munts Technologies

[Top](#)

## ▪ See Also

[Reference](#)

## IO.Devices.USB.Munts Namespace

---

# HID Fields

The [HID](#) type exposes the following members.

## Fields

Name	Description
 <a href="#">Product</a>	Product ID for Munts Technologies USB hid devices.
 <a href="#">Vendor</a>	Vendor ID for Munts Technologies

[Top](#)

## See Also

[Reference](#)

[HID Class](#)

[IO.Devices.USB.Munts Namespace](#)

# HIDProduct Field

Product ID for Munts Technologies USB hid devices.

**Namespace:** [IO.Devices.USB.Munts](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int Product = 2810
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[HID Class](#)

[IO.Devices.USB.Munts Namespace](#)

# HIDVendor Field

Vendor ID for Munts Technologies

**Namespace:** [IO.Devices.USB.Munts](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int Vendor = 5840
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[HID Class](#)

[IO.Devices.USB.Munts Namespace](#)

# Serial Class

USB device constants for Munts Technologies USB serial port devices.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Devices.USB.MuntsSerial](#)

**Namespace:** [IO.Devices.USB.Munts](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static class Serial
```

The [Serial](#) type exposes the following members.

## ▪ Fields

	Name	Description
❖ <a href="#">S</a>	<a href="#">Product</a>	Product ID for Munts Technologies USB serial port devices.
❖ <a href="#">S</a>	<a href="#">Vendor</a>	Vendor ID for Munts Technologies

[Top](#)

## ▪ See Also

[Reference](#)

## IO.Devices.USB.Munts Namespace

---

# Serial Fields

The [Serial](#) type exposes the following members.

## Fields

Name	Description
  <a href="#">Product</a>	Product ID for Munts Technologies USB serial port devices.
  <a href="#">Vendor</a>	Vendor ID for Munts Technologies

[Top](#)

## See Also

[Reference](#)

[Serial Class](#)

[IO.Devices.USB.Munts Namespace](#)

# SerialProduct Field

Product ID for Munts Technologies USB serial port devices.

**Namespace:** [IO.Devices.USB.Munts](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int Product = 2811
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Serial Class](#)

[IO.Devices.USB.Munts Namespace](#)

# SerialVendor Field

Vendor ID for Munts Technologies

**Namespace:** [IO.Devices.USB.Munts](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int Vendor = 5840
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[Serial Class](#)

[IO.Devices.USB.Munts Namespace](#)

# IO.Interfaces.ADC Namespace

Abstract Interface for ADC (Analog to Digital Converter) Inputs

## Classes

Class	Description
 <a href="#">Input</a>	Encapsulates ADC voltage inputs.

## Interfaces

Interface	Description
 <a href="#">Sample</a>	Abstract interface for ADC inputs returning an integer sample value.
 <a href="#">Voltage</a>	Abstract interface for ADC inputs returning a floating point voltage value.

# Input Class

Encapsulates ADC voltage inputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Interfaces.ADCInput](#)

**Namespace:** [IO.Interfaces.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Input : Voltage
```

The [Input](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Input</a>	Create an ADC voltage input.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">voltage</a>	Read-only property returning the analog input voltage.

[Top](#)

## ↳ See Also

**Reference**

[IO.Interfaces.ADC Namespace](#)

# Input Constructor

Create an ADC voltage input.

**Namespace:** [IO.Interfaces.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Input(  
    Sample input,  
    double reference,  
    double gain = 1  
)
```

## Parameters

*input*

Type: [IO.Interfaces.ADCSample](#)

ADC sample object.

*reference*

Type: [SystemDouble](#)

ADC reference in volts.

*gain* (Optional)

Type: [SystemDouble](#)

ADC input gain in volts per volt.

## ◀ See Also

[Reference](#)

[Input Class](#)

## IO.Interfaces.ADC Namespace

---

# Input Properties

The [Input](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">voltage</a>	Read-only property returning the analog input voltage.

[Top](#)

## See Also

[Reference](#)

[Input Class](#)

[IO.Interfaces.ADC Namespace](#)

# InputVoltage Property

Read-only property returning the analog input voltage.

**Namespace:** [IO.Interfaces.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public double voltage { get; }
```

### Property Value

Type: [Double](#)

### Implements

[VoltageVoltage](#)

## ▪ See Also

[Reference](#)

[Input Class](#)

[IO.Interfaces.ADC Namespace](#)

# Sample Interface

Abstract interface for ADC inputs returning an integer sample value.

**Namespace:** [IO.Interfaces.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public interface Sample
```

The [Sample](#) type exposes the following members.

## ▪ Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.
	<a href="#">sample</a>	Read-only property returning an integer analog sample value.

[Top](#)

## ▪ See Also

[Reference](#)

[IO.Interfaces.ADC Namespace](#)

# Sample Properties

The [Sample](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.
	<a href="#">sample</a>	Read-only property returning an integer analog sample value.

[Top](#)

## See Also

### Reference

[Sample Interface](#)

[IO.Interfaces.ADC Namespace](#)

# Sampleresolution Property

Read-only property returning the number of bits of resolution.

**Namespace:** [IO.Interfaces.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
int resolution { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Sample Interface](#)

[IO.Interfaces.ADC Namespace](#)

# Samplesample Property

Read-only property returning an integer analog sample value.

**Namespace:** [IO.Interfaces.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
int sample { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Sample Interface](#)

[IO.Interfaces.ADC Namespace](#)

# Voltage Interface

Abstract interface for ADC inputs returning a floating point voltage value.

**Namespace:** [IO.Interfaces.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public interface Voltage
```

The [Voltage](#) type exposes the following members.

## ▀ Properties

	Name	Description
	<a href="#">voltage</a>	Read-only property returning a floating point analog voltage value.

[Top](#)

## ▀ See Also

[Reference](#)

[IO.Interfaces.ADC Namespace](#)

# Voltage Properties

The [Voltage](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">voltage</a>	Read-only property returning a floating point analog voltage value.

[Top](#)

## See Also

[Reference](#)

[Voltage Interface](#)

[IO.Interfaces.ADC Namespace](#)

# VoltageVoltage Property

Read-only property returning a floating point analog voltage value.

**Namespace:** [IO.Interfaces.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
double voltage { get; }
```

### Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Voltage Interface](#)

[IO.Interfaces.ADC Namespace](#)

# IO.Interfaces.DAC Namespace

Abstract Interface for DAC (Digital to Analog Converter) Outputs

## Classes

Class	Description
 <a href="#">Output</a>	Encapsulates DAC voltage outputs.

## Interfaces

Interface	Description
 <a href="#">Sample</a>	Abstract interface for DAC outputs accepting an integer output sample value.
 <a href="#">Voltage</a>	Abstract interface for DAC outputs accepting a floating point output voltage value.

# Output Class

Encapsulates DAC voltage outputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Interfaces.DACOutput](#)

**Namespace:** [IO.Interfaces.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Output : Voltage
```

The [Output](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Output</a>	Create an DAC voltage output.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">voltage</a>	Write-only for setting the DAC output voltage.

[Top](#)

## ↳ See Also

**Reference**

[IO.Interfaces.DAC Namespace](#)

# Output Constructor

Create an DAC voltage output.

**Namespace:** [IO.Interfaces.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public Output(  
    Sample output,  
    double reference,  
    double gain = 1  
)
```

## Parameters

*output*

Type: [IO.Interfaces.DACSample](#)

DAC output object.

*reference*

Type: [SystemDouble](#)

DAC output reference in volts.

*gain* (Optional)

Type: [SystemDouble](#)

DAC output gain in volts per volt.

## ▪ See Also

[Reference](#)

[Output Class](#)

## IO.Interfaces.DAC Namespace

---

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">voltage</a>	Write-only for setting the DAC output voltage.

[Top](#)

## See Also

[Reference](#)

[Output Class](#)

[IO.Interfaces.DAC Namespace](#)

# Outputvoltage Property

Write-only for setting the DAC output voltage.

**Namespace:** [IO.Interfaces.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public double voltage { set; }
```

### Property Value

Type: [Double](#)

### Implements

[Voltagevoltage](#)

## ◀ See Also

[Reference](#)

[Output Class](#)

[IO.Interfaces.DAC Namespace](#)

# Sample Interface

Abstract interface for DAC outputs accepting an integer output sample value.

**Namespace:** [IO.Interfaces.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ► Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public interface Sample
```

The [Sample](#) type exposes the following members.

## ► Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.
	<a href="#">sample</a>	Write-only property for setting the DAC output level.

[Top](#)

## ► See Also

### Reference

[IO.Interfaces.DAC Namespace](#)



# Sample Properties

The [Sample](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.
	<a href="#">sample</a>	Write-only property for setting the DAC output level.

[Top](#)

## See Also

### Reference

[Sample Interface](#)

[IO.Interfaces.DAC Namespace](#)

# Sampleresolution Property

Read-only property returning the number of bits of resolution.

**Namespace:** [IO.Interfaces.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
int resolution { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Sample Interface](#)

[IO.Interfaces.DAC Namespace](#)

# Samplesample Property

Write-only property for setting the DAC output level.

**Namespace:** [IO.Interfaces.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
int sample { set; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Sample Interface](#)

[IO.Interfaces.DAC Namespace](#)

# Voltage Interface

Abstract interface for DAC outputs accepting a floating point output voltage value.

**Namespace:** [IO.Interfaces.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public interface Voltage
```

The [Voltage](#) type exposes the following members.

## ▀ Properties

	Name	Description
	<a href="#">voltage</a>	Write-only property for setting the DAC output voltage.

[Top](#)

## ▀ See Also

[Reference](#)

[IO.Interfaces.DAC Namespace](#)

# Voltage Properties

The [Voltage](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">voltage</a>	Write-only property for setting the DAC output voltage.

[Top](#)

## See Also

[Reference](#)

[Voltage Interface](#)

[IO.Interfaces.DAC Namespace](#)

# Voltagevoltage Property

Write-only property for setting the DAC output voltage.

**Namespace:** [IO.Interfaces.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
double voltage { set; }
```

### Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Voltage Interface](#)

[IO.Interfaces.DAC Namespace](#)

# IO.Interfaces.GPIO Namespace

Abstract Interface for GPIO (General Purpose Input/Output) Pins

## ▪ Interfaces

	Interface	Description
	<a href="#">Pin</a>	Abstract interface for GPIO pins.

## ▪ Enumerations

	Enumeration	Description
	<a href="#">Direction</a>	GPIO pin data direction settings.

# Direction Enumeration

GPIO pin data direction settings.

**Namespace:** [IO.Interfaces.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum Direction
```

## ▪ Members

Member name	Value	Description
Input	0	Input pin (read only)
Output	1	Output pin (read or write)

## ▪ See Also

### Reference

[IO.Interfaces.GPIO Namespace](#)

# Pin Interface

Abstract interface for GPIO pins.

**Namespace:** [IO.Interfaces.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public interface Pin
```

The [Pin](#) type exposes the following members.

## ▪ Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## ▪ See Also

[Reference](#)

[IO.Interfaces.GPIO Namespace](#)

# Pin Properties

The [Pin](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## See Also

[Reference](#)

[Pin Interface](#)

[IO.Interfaces.GPIO Namespace](#)

# Pinstate Property

Read/Write GPIO state property.

**Namespace:** [IO.Interfaces.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
bool state { get; set; }
```

### Property Value

Type: [Boolean](#)

## « See Also

### Reference

[Pin Interface](#)

[IO.Interfaces.GPIO Namespace](#)

# IO.Interfaces.Humidity Namespace

Abstract Interface for Humidity Sensors

## ► Interfaces

Interface	Description
 Sensor	Abstract interface for humidity sensors.

# Sensor Interface

Abstract interface for humidity sensors.

**Namespace:** [IO.Interfaces.Humidity](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public interface Sensor
```

The [Sensor](#) type exposes the following members.

## ▪ Properties

Name	Description
 <a href="#">Humidity</a>	Read-only property returning the percentage relative humidity.

[Top](#)

## ▪ See Also

### Reference

[IO.Interfaces.Humidity Namespace](#)

# Sensor Properties

The [Sensor](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Humidity</a>	Read-only property returning the percentage relative humidity.

[Top](#)

## See Also

[Reference](#)

[Sensor Interface](#)

[IO.Interfaces.Humidity Namespace](#)

# SensorHumidity Property

Read-only property returning the percentage relative humidity.

**Namespace:** [IO.Interfaces.Humidity](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
double Humidity { get; }
```

### Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Sensor Interface](#)

[IO.Interfaces.Humidity Namespace](#)

# IO.Interfaces.I2C Namespace

Abstract Interface for I<sup>2</sup>C (Inter-Integrated Circuit) Bus Controllers

## ► Classes

	<b>Class</b>	<b>Description</b>
	<a href="#">Device</a>	Encapsulates a single I <sup>2</sup> C slave device.
	<a href="#">Speeds</a>	I <sup>2</sup> C bus speed constants.

## ► Interfaces

	<b>Interface</b>	<b>Description</b>
	<a href="#">Bus</a>	Abstract interface for I <sup>2</sup> C bus controllers.

# Bus Interface

Abstract interface for I<sup>2</sup>C bus controllers.

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public interface Bus
```

The [Bus](#) type exposes the following members.

## ▪ Methods

	Name	Description
≡	<a href="#">Read</a>	Read bytes from an I <sup>2</sup> C slave device.
≡	<a href="#">Transaction</a>	Write and read bytes to and from an I <sup>2</sup> C slave device.
≡	<a href="#">Write</a>	Write bytes to an I <sup>2</sup> C slave device.

[Top](#)

## ▪ See Also

[Reference](#)

[IO.Interfaces.I2C Namespace](#)



# Bus Methods

The [Bus](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Read</a>	Read bytes from an I <sup>2</sup> C slave device.
	<a href="#">Transaction</a>	Write and read bytes to and from an I <sup>2</sup> C slave device.
	<a href="#">Write</a>	Write bytes to an I <sup>2</sup> C slave device.

[Top](#)

## See Also

### Reference

[Bus Interface](#)

[IO.Interfaces.I2C Namespace](#)

# BusRead Method

Read bytes from an I<sup>2</sup>C slave device.

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
void Read(  
    int slaveaddr,  
    byte[] resp,  
    int resplen  
)
```

## Parameters

*slaveaddr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

## « See Also

### Reference

## Bus Interface

### IO.Interfaces.I2C Namespace

# BusTransaction Method

Write and read bytes to and from an I<sup>2</sup>C slave device.

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
void Transaction(  
    int slaveaddr,  
    byte[] cmd,  
    int cmdlen,  
    byte[] resp,  
    int resplen,  
    int delayus = 0  
)
```

## Parameters

*slaveaddr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

*delayus* (Optional)

Type: [SystemInt32](#)

Delay in microseconds between the I<sup>2</sup>C write and read cycles.

Allowed values are 0 to 65535 microseconds.

## See Also

**Reference**

[Bus Interface](#)

[IO.Interfaces.I2C Namespace](#)

# BusWrite Method

Write bytes to an I<sup>2</sup>C slave device.

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
void Write(  
    int slaveaddr,  
    byte[] cmd,  
    int cmdlen  
)
```

## Parameters

*slaveaddr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

## « See Also

### Reference

## Bus Interface

### IO.Interfaces.I2C Namespace

# Device Class

Encapsulates a single I<sup>2</sup>C slave device.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Interfaces.I2CDevice](#)

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Device</a>	Create an I <sup>2</sup> C slave device.

[Top](#)

## ▪ Methods

	Name	Description
≡	<a href="#">Read</a>	Read bytes from an I <sup>2</sup> C slave device.
≡		

**Transaction** Write and read bytes to and from an I<sup>2</sup>C slave device.



**Write** Write bytes to an I<sup>2</sup>C slave device.

[Top](#)

## ◀ See Also

**Reference**

[IO.Interfaces.I2C Namespace](#)

# Device Constructor

Create an I<sup>2</sup>C slave device.

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Bus bus,  
    int slaveaddr  
)
```

## Parameters

*bus*

Type: [IO.Interfaces.I2CBus](#)

I<sup>2</sup>C bus controller object.

*slaveaddr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

## ▪ See Also

[Reference](#)

[Device Class](#)

[IO.Interfaces.I2C Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Read</a>	Read bytes from an I <sup>2</sup> C slave device.
	<a href="#">Transaction</a>	Write and read bytes to and from an I <sup>2</sup> C slave device.
	<a href="#">Write</a>	Write bytes to an I <sup>2</sup> C slave device.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Interfaces.I2C Namespace](#)

# DeviceRead Method

Read bytes from an I<sup>2</sup>C slave device.

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Read(  
    byte[] resp,  
    int resplen  
)
```

## Parameters

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Interfaces.I2C Namespace](#)

# DeviceTransaction Method

Write and read bytes to and from an I<sup>2</sup>C slave device.

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Transaction(  
    byte[] cmd,  
    int cmdLen,  
    byte[] resp,  
    int resplen,  
    int delayus = 0  
)
```

## Parameters

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

*delayus* (Optional)

Type: [SystemInt32](#)

Delay in microseconds between the I<sup>2</sup>C write and read cycles.

Allowed values are 0 to 65535 microseconds.

## See Also

**Reference**

[Device Class](#)

[IO.Interfaces.I2C Namespace](#)

# DeviceWrite Method

Write bytes to an I<sup>2</sup>C slave device.

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Write(  
    byte[] cmd,  
    int cmdLen  
)
```

## Parameters

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Interfaces.I2C Namespace](#)

# Speeds Class

I<sup>2</sup>C bus speed constants.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Interfaces.I2CSpeeds](#)

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static class Speeds
```

The [Speeds](#) type exposes the following members.

## ▪ Fields

	Name	Description
• <a href="#">S</a>	<a href="#">FastMode</a>	Fast Mode
• <a href="#">S</a>	<a href="#">FastModePlus</a>	Fast Mode Plus
• <a href="#">S</a>	<a href="#">StandardMode</a>	Standard Mode

[Top](#)

## ▪ See Also

## Reference

### IO.Interfaces.I2C Namespace

---

# Speeds Fields

The [Speeds](#) type exposes the following members.

## Fields

Name	Description
<a href="#">  FastMode</a>	Fast Mode
<a href="#">  FastModePlus</a>	Fast Mode Plus
<a href="#">  StandardMode</a>	Standard Mode

[Top](#)

## See Also

[Reference](#)

[Speeds Class](#)

[IO.Interfaces.I2C Namespace](#)

# SpeedsFastMode Field

Fast Mode

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int FastMode = 400000
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[Speeds Class](#)

[IO.Interfaces.I2C Namespace](#)

# SpeedsFastModePlus Field

Fast Mode Plus

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int FastModePlus = 1000000
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Speeds Class](#)

[IO.Interfaces.I2C Namespace](#)

# SpeedsStandardMode Field

Standard Mode

**Namespace:** [IO.Interfaces.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int StandardMode = 100000
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[Speeds Class](#)

[IO.Interfaces.I2C Namespace](#)

# IO.Interfaces.Message64

## Namespace

Abstract Interface for 64-Byte Message Services

### Classes

Class	Description
 <a href="#">Message</a>	Encapsulates 64-byte messages.

### Interfaces

Interface	Description
 <a href="#">Messenger</a>	Abstract interface for sending and receiving 64-byte messages.

# Message Class

Encapsulates 64-byte messages.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Interfaces.Message64Message](#)

**Namespace:** [IO.Interfaces.Message64](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Message
```

The [Message](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Message</a>	Create a message object without initializing the payload.
≡	<a href="#">Message(Byte)</a>	Create a message object with an initialized payload.

[Top](#)

## ▪ Fields

Name	Description
• <a href="#">payload</a>	Message payload.
• <a href="#">S</a> <a href="#">Size</a>	Message payload size.

[Top](#)

## See Also

### Reference

[IO.Interfaces.Message64 Namespace](#)

# Message Constructor

## ↳ Overload List

	Name	Description
≡	<a href="#">Message</a>	Create a message object without initializing the payload.
≡	<a href="#">Message(Byte)</a>	Create a message object with an initialized payload.

[Top](#)

## ↳ See Also

### Reference

[Message Class](#)

[IO.Interfaces.Message64 Namespace](#)

# Message Constructor

Create a message object without initializing the payload.

**Namespace:** [IO.Interfaces.Message64](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public Message()
```

## ▪ See Also

### Reference

[Message Class](#)

[Message Overload](#)

[IO.Interfaces.Message64 Namespace](#)

# Message Constructor (Byte)

Create a message object with an initialized payload.

**Namespace:** [IO.Interfaces.Message64](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▀ Syntax

C#    VB    F#

[Copy](#)

```
public Message(  
    byte fill  
)
```

### Parameters

*fill*

Type: [System.Byte](#)

Value to initialize the payload with.

## ▀ See Also

### Reference

[Message Class](#)

[Message Overload](#)

[IO.Interfaces.Message64 Namespace](#)

# Message Fields

The [Message](#) type exposes the following members.

## Fields

Name	Description
• <a href="#">payload</a>	Message payload.
• <b>s</b> <a href="#">Size</a>	Message payload size.

[Top](#)

## See Also

### Reference

[Message Class](#)

[IO.Interfaces.Message64 Namespace](#)

# Messagepayload Field

Message payload.

**Namespace:** [IO.Interfaces.Message64](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public byte[] payload
```

### Field Value

Type: [Byte](#)

## « See Also

### Reference

[Message Class](#)

[IO.Interfaces.Message64 Namespace](#)

# MessageSize Field

Message payload size.

**Namespace:** [IO.Interfaces.Message64](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const int Size = 64
```

### Field Value

Type: [Int32](#)

## « See Also

### Reference

[Message Class](#)

[IO.Interfaces.Message64 Namespace](#)

# Messenger Interface

Abstract interface for sending and receiving 64-byte messages.

**Namespace:** [IO.Interfaces.Message64](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public interface Messenger
```

The [Messenger](#) type exposes the following members.

## ▪ Methods

	Name	Description
≡	<a href="#">Receive</a>	Receive a 64-byte message.
≡	<a href="#">Send</a>	Send a 64-byte message.
≡	<a href="#">Transaction</a>	Send a 64-byte command and receive a 64-byte response.

[Top](#)

## ▪ See Also

### Reference

[IO.Interfaces.Message64 Namespace](#)



# Messenger Methods

The [Messenger](#) type exposes the following members.

## ▪ Methods

	Name	Description
	<a href="#">Receive</a>	Receive a 64-byte message.
	<a href="#">Send</a>	Send a 64-byte message.
	<a href="#">Transaction</a>	Send a 64-byte command and receive a 64-byte response.

[Top](#)

## ▪ See Also

### Reference

[Messenger Interface](#)

[IO.Interfaces.Message64 Namespace](#)

# MessengerReceive Method

Receive a 64-byte message.

**Namespace:** [IO.Interfaces.Message64](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
void Receive(  
    Message resp  
)
```

## Parameters

*resp*

Type: [IO.Interfaces.Message64Message](#)

Message received.

## ◀ See Also

### Reference

[Messenger Interface](#)

[IO.Interfaces.Message64 Namespace](#)

# MessengerSend Method

Send a 64-byte message.

**Namespace:** [IO.Interfaces.Message64](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
void Send(  
    Message cmd  
)
```

## Parameters

*cmd*

Type: [IO.Interfaces.Message64Message](#)

Message to be sent.

## ◀ See Also

### Reference

[Messenger Interface](#)

[IO.Interfaces.Message64 Namespace](#)

# MessengerTransaction Method

Send a 64-byte command and receive a 64-byte response.

**Namespace:** [IO.Interfaces.Message64](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
void Transaction(  
    Message cmd,  
    Message resp  
)
```

## Parameters

*cmd*

Type: [IO.Interfaces.Message64Message](#)

Command to be sent.

*resp*

Type: [IO.Interfaces.Message64Message](#)

Response to be received.

## ◀ See Also

### Reference

[Messenger Interface](#)

[IO.Interfaces.Message64 Namespace](#)

# IO.Interfaces.Motor Namespace

Abstract Interface For Variable Speed Motor Outputs

## ▪ Classes

	Class	Description
	<a href="#">Velocities</a>	Motor velocity constants.

## ▪ Interfaces

	Interface	Description
	<a href="#">Output</a>	Abstract interface for variable speed motor outputs.

# Output Interface

Abstract interface for variable speed motor outputs.

**Namespace:** [IO.Interfaces.Motor](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public interface Output
```

The [Output](#) type exposes the following members.

## ▪ Properties

Name	Description
 <a href="#">velocity</a>	Write-only motor velocity property.

[Top](#)

## ▪ See Also

[Reference](#)

[IO.Interfaces.Motor Namespace](#)

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">velocity</a>	Write-only motor velocity property.

[Top](#)

## See Also

### Reference

[Output Interface](#)

[IO.Interfaces.Motor Namespace](#)

# Outputvelocity Property

Write-only motor velocity property.

**Namespace:** [IO.Interfaces.Motor](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
double velocity { set; }
```

### Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Output Interface](#)

[IO.Interfaces.Motor Namespace](#)

# Velocities Class

Motor velocity constants.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Interfaces.MotorVelocities](#)

**Namespace:** [IO.Interfaces.Motor](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static class Velocities
```

The [Velocities](#) type exposes the following members.

## ▪ Fields

	Name	Description
• <a href="#"><b>S</b></a>	<a href="#">Maximum</a>	Maximum velocity (full speed forward).
• <a href="#"><b>S</b></a>	<a href="#">Minimum</a>	Minimum velocity (full speed reverse).
• <a href="#"><b>S</b></a>	<a href="#">Stop</a>	Zero velocity (motor stopped).

[Top](#)

## ▪ See Also

## Reference

### IO.Interfaces.Motor Namespace

---

# Velocities Fields

The [Velocities](#) type exposes the following members.

## Fields

	Name	Description
• 	<a href="#">Maximum</a>	Maximum velocity (full speed forward).
• 	<a href="#">Minimum</a>	Minimum velocity (full speed reverse).
• 	<a href="#">Stop</a>	Zero velocity (motor stopped).

[Top](#)

## See Also

### Reference

[Velocities Class](#)

[IO.Interfaces.Motor Namespace](#)

# VelocitiesMaximum Field

Maximum velocity (full speed forward).

**Namespace:** [IO.Interfaces.Motor](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const double Maximum = 1
```

### Field Value

Type: [Double](#)

## « See Also

### Reference

[Velocities Class](#)

[IO.Interfaces.Motor Namespace](#)

# VelocitiesMinimum Field

Minimum velocity (full speed reverse).

**Namespace:** [IO.Interfaces.Motor](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const double Minimum = -1
```

### Field Value

Type: [Double](#)

## « See Also

### Reference

[Velocities Class](#)

[IO.Interfaces.Motor Namespace](#)

# VelocitiesStop Field

Zero velocity (motor stopped).

**Namespace:** [IO.Interfaces.Motor](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const double Stop = 0
```

### Field Value

Type: [Double](#)

## ◀ See Also

### Reference

[Velocities Class](#)

[IO.Interfaces.Motor Namespace](#)

# IO.Interfaces.PWM Namespace

Abstract Interface for PWM (Pulse Width Modulated) Outputs

## ▪ Classes

	Class	Description
	<a href="#">DutyCycles</a>	PWM dutycycle constants.

## ▪ Interfaces

	Interface	Description
	<a href="#">Output</a>	Abstract interface for PWM outputs.

# DutyCycles Class

PWM dutycycle constants.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Interfaces.PWMDutyCycles](#)

**Namespace:** [IO.Interfaces.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static class DutyCycles
```

The [DutyCycles](#) type exposes the following members.

## ▪ Fields

	Name	Description
• <a href="#"><b>S</b></a>	<a href="#">Maximum</a>	Maximum duty cycle (percent).
• <a href="#"><b>S</b></a>	<a href="#">Minimum</a>	Minimum duty cycle (percent).

[Top](#)

## ▪ See Also

**Reference**

[IO.Interfaces.PWM Namespace](#)



# DutyCycles Fields

The [DutyCycles](#) type exposes the following members.

## Fields

	Name	Description
• 	<a href="#">Maximum</a>	Maximum duty cycle (percent).
• 	<a href="#">Minimum</a>	Minimum duty cycle (percent).

[Top](#)

## See Also

### Reference

[DutyCycles Class](#)

[IO.Interfaces.PWM Namespace](#)

# DutyCyclesMaximum Field

Maximum duty cycle (percent).

**Namespace:** [IO.Interfaces.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const double Maximum = 100
```

### Field Value

Type: [Double](#)

## ◀ See Also

### Reference

[DutyCycles Class](#)

[IO.Interfaces.PWM Namespace](#)

# DutyCyclesMinimum Field

Minimum duty cycle (percent).

**Namespace:** [IO.Interfaces.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const double Minimum = 0
```

### Field Value

Type: [Double](#)

## ◀ See Also

### Reference

[DutyCycles Class](#)

[IO.Interfaces.PWM Namespace](#)

# Output Interface

Abstract interface for PWM outputs.

**Namespace:** [IO.Interfaces.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public interface Output
```

The [Output](#) type exposes the following members.

## ▪ Properties

	Name	Description
	<a href="#">dutycycle</a>	Write-only PWM duty cycle property.

[Top](#)

## ▪ See Also

[Reference](#)

[IO.Interfaces.PWM Namespace](#)

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">dutycycle</a>	Write-only PWM duty cycle property.

[Top](#)

## See Also

### Reference

[Output Interface](#)

[IO.Interfaces.PWM Namespace](#)

# Outputdutycycle Property

Write-only PWM duty cycle property.

**Namespace:** [IO.Interfaces.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
double dutycycle { set; }
```

### Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Output Interface](#)

[IO.Interfaces.PWM Namespace](#)

# IO.Interfaces.Servo Namespace

Abstract Interface for Servo Outputs

## ▪ Classes

	Class	Description
	<a href="#">Positions</a>	Servo position constants.

## ▪ Interfaces

	Interface	Description
	<a href="#">Output</a>	Abstract interface for servo outputs.

# Output Interface

Abstract interface for servo outputs.

**Namespace:** [IO.Interfaces.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public interface Output
```

The [Output](#) type exposes the following members.

## ▪ Properties

Name	Description
 <a href="#">position</a>	Write-only servo position property.

[Top](#)

## ▪ See Also

[Reference](#)

[IO.Interfaces.Servo Namespace](#)

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">position</a>	Write-only servo position property.

[Top](#)

## See Also

### Reference

[Output Interface](#)

[IO.Interfaces.Servo Namespace](#)

# Outputposition Property

Write-only servo position property.

**Namespace:** [IO.Interfaces.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
double position { set; }
```

### Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Output Interface](#)

[IO.Interfaces.Servo Namespace](#)

# Positions Class

Servo position constants.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Interfaces.ServoPositions](#)

**Namespace:** [IO.Interfaces.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static class Positions
```

The [Positions](#) type exposes the following members.

## ▪ Fields

	Name	Description
• <a href="#"><b>S</b></a>	<a href="#">Maximum</a>	Maximum displacement position.
• <a href="#"><b>S</b></a>	<a href="#">Minimum</a>	Minimum displacement position.
• <a href="#"><b>S</b></a>	<a href="#">Neutral</a>	Zero displacement (neutral) position.

[Top](#)

## ▪ See Also

## Reference

### [IO.Interfaces.Servo Namespace](#)

# Positions Fields

The [Positions](#) type exposes the following members.

## Fields

	Name	Description
• 	<a href="#">Maximum</a>	Maximum displacement position.
• 	<a href="#">Minimum</a>	Minimum displacement position.
• 	<a href="#">Neutral</a>	Zero displacement (neutral) position.

[Top](#)

## See Also

### Reference

[Positions Class](#)

[IO.Interfaces.Servo Namespace](#)

# PositionsMaximum Field

Maximum displacement position.

**Namespace:** [IO.Interfaces.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const double Maximum = 1
```

### Field Value

Type: [Double](#)

## « See Also

### Reference

[Positions Class](#)

[IO.Interfaces.Servo Namespace](#)

# PositionsMinimum Field

Minimum displacement position.

**Namespace:** [IO.Interfaces.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const double Minimum = -1
```

### Field Value

Type: [Double](#)

## ◀ See Also

### Reference

[Positions Class](#)

[IO.Interfaces.Servo Namespace](#)

# PositionsNeutral Field

Zero displacement (neutral) position.

**Namespace:** [IO.Interfaces.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public const double Neutral = 0
```

### Field Value

Type: [Double](#)

## « See Also

### Reference

[Positions Class](#)

[IO.Interfaces.Servo Namespace](#)

# IO.Interfaces.SPI Namespace

Abstract Interface for SPI (Serial Peripheral Interconnect) Slave Devices

## ▪ Interfaces

Interface	Description
<a href="#">Device</a>	Abstract interface for SPI slave devices.

# Device Interface

Abstract interface for SPI slave devices.

**Namespace:** [IO.Interfaces.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public interface Device
```

The [Device](#) type exposes the following members.

## ▪ Methods

	Name	Description
≡	<a href="#">Read</a>	Read bytes from an SPI slave device.
≡	<a href="#">Transaction</a>	Write bytes to and read bytes from an SPI slave device.
≡	<a href="#">Write</a>	Write bytes to an SPI slave device.

[Top](#)

## ▪ See Also

**Reference**

[IO.Interfaces.SPI Namespace](#)



# Device Methods

The [Device](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Read</a>	Read bytes from an SPI slave device.
	<a href="#">Transaction</a>	Write bytes to and read bytes from an SPI slave device.
	<a href="#">Write</a>	Write bytes to an SPI slave device.

[Top](#)

## See Also

### Reference

[Device Interface](#)

[IO.Interfaces.SPI Namespace](#)

# DeviceRead Method

Read bytes from an SPI slave device.

**Namespace:** [IO.Interfaces.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
void Read(  
    byte[] resp,  
    int resplen  
)
```

## Parameters

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

## ◀ See Also

### Reference

[Device Interface](#)

[IO.Interfaces.SPI Namespace](#)

# DeviceTransaction Method

Write bytes to and read bytes from an SPI slave device.

**Namespace:** [IO.Interfaces.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
void Transaction(  
    byte[] cmd,  
    int cmdLen,  
    byte[] resp,  
    int resplen,  
    int delayus = 0  
)
```

## Parameters

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

*delayus* (Optional)

Type: [SystemInt32](#)

Delay in microseconds between write and read operations.

## See Also

**Reference**

[Device Interface](#)

[IO.Interfaces.SPI Namespace](#)

# DeviceWrite Method

Write bytes to an SPI slave device.

**Namespace:** [IO.Interfaces.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
void Write(  
    byte[] cmd,  
    int cmdLen  
)
```

## Parameters

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

## ◀ See Also

### Reference

[Device Interface](#)

[IO.Interfaces.SPI Namespace](#)

# IO.Interfaces.Temperature Namespace

Abstract Interface for Temperature Sensors

## ▪ Classes

Class	Description
 <a href="#">Conversions</a>	Temperature conversion functions.

## ▪ Interfaces

Interface	Description
 <a href="#">Sensor</a>	Abstract interface for temperature sensors.

# Conversions Class

Temperature conversion functions.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Interfaces.TemperatureConversions](#)

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Conversions
```

The [Conversions](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Conversions</a>	Initializes a new instance of the <a href="#">Conversions</a> class

[Top](#)

## ▪ Methods

	Name	Description
≡ S	<a href="#">CelsiusToFahrenheit</a>	Convert degrees Celsius to

degrees Fahrenheit.

---

 S	<a href="#">CelsiusToKelvins</a>	Convert degrees Celsius to Kelvins.
 S	<a href="#">FahrenheitToCelsius</a>	Convert degrees Fahrenheit to degrees Celsius.
 S	<a href="#">FahrenheitToKelvins</a>	Convert degrees Fahrenheit to Kelvins.
 S	<a href="#">KelvinsToCelsius</a>	Convert Kelvins to degrees Celsius.
 S	<a href="#">KelvinsToFahrenheit</a>	Convert Kelvins to degrees Fahrenheit.

---

[Top](#)

## See Also

### Reference

[IO.Interfaces.Temperature Namespace](#)

# Conversions Constructor

Initializes a new instance of the [Conversions](#) class

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public Conversions()
```

## ▪ See Also

### Reference

[Conversions Class](#)

[IO.Interfaces.Temperature Namespace](#)

# Conversions Methods

The [Conversions](#) type exposes the following members.

## ▪ Methods

	Name	Description
 	<a href="#">CelsiusToFahrenheit</a>	Convert degrees Celsius to degrees Fahrenheit.
 	<a href="#">CelsiusToKelvins</a>	Convert degrees Celsius to Kelvins.
 	<a href="#">FahrenheitToCelsius</a>	Convert degrees Fahrenheit to degrees Celsius.
 	<a href="#">FahrenheitToKelvins</a>	Convert degrees Fahrenheit to Kelvins.
 	<a href="#">KelvinsToCelsius</a>	Convert Kelvins to degrees Celsius.
 	<a href="#">KelvinsToFahrenheit</a>	Convert Kelvins to degrees Fahrenheit.

[Top](#)

## ▪ See Also

[Reference](#)

[Conversions Class](#)

## IO.Interfaces.Temperature Namespace

---

# ConversionsCelsiusToFahrenheit Method

Convert degrees Celsius to degrees Fahrenheit.

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static double CelsiusToFahrenheit(  
    double celsius  
)
```

## Parameters

*celsius*

Type: [SystemDouble](#)

Temperature in degrees Celsius.

## Return Value

Type: [Double](#)

Temperature in degrees Fahrenheit.

## ◀ See Also

### Reference

[Conversions Class](#)

[IO.Interfaces.Temperature Namespace](#)

# ConversionsCelsiusToKelvins Method

Convert degrees Celsius to Kelvins.

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static double CelsiusToKelvins(  
    double celsius  
)
```

## Parameters

*celsius*

Type: [SystemDouble](#)

Temperature in degrees Celsius.

## Return Value

Type: [Double](#)

Temperature in Kelvins.

## ◀ See Also

### Reference

[Conversions Class](#)

[IO.Interfaces.Temperature Namespace](#)

# ConversionsFahrenheitToCelsius Method

Convert degrees Fahrenheit to degrees Celsius.

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static double FahrenheitToCelsius(  
    double fahrenheit  
)
```

## Parameters

*fahrenheit*

Type: [SystemDouble](#)

Temperature in degrees Fahrenheit.

## Return Value

Type: [Double](#)

Temperature in degrees Celsius.

## ◀ See Also

### Reference

[Conversions Class](#)

[IO.Interfaces.Temperature Namespace](#)

# ConversionsFahrenheitToKelvins Method

Convert degrees Fahrenheit to Kelvins.

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static double FahrenheitToKelvins(  
    double fahrenheit  
)
```

## Parameters

*fahrenheit*

Type: [SystemDouble](#)

Temperature in degrees Fahrenheit.

## Return Value

Type: [Double](#)

Temperature in Kelvins.

## ◀ See Also

### Reference

[Conversions Class](#)

[IO.Interfaces.Temperature Namespace](#)

# ConversionsKelvinsToCelsius Method

Convert Kelvins to degrees Celsius.

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static double KelvinsToCelsius(  
    double kelvins  
)
```

## Parameters

*kelvins*

Type: [SystemDouble](#)

Temperature in Kelvins.

## Return Value

Type: [Double](#)

Temperature in degrees Celsius.

## ◀ See Also

### Reference

[Conversions Class](#)

[IO.Interfaces.Temperature Namespace](#)

# ConversionsKelvinsToFahrenheit Method

Convert Kelvins to degrees Fahrenheit.

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static double KelvinsToFahrenheit(  
    double kelvins  
)
```

## Parameters

*kelvins*

Type: [SystemDouble](#)

Temperature in Kelivns.

## Return Value

Type: [Double](#)

Temperature in degrees Fahrenheit.

## ◀ See Also

### Reference

[Conversions Class](#)

[IO.Interfaces.Temperature Namespace](#)

# Sensor Interface

Abstract interface for temperature sensors.

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public interface Sensor
```

The [Sensor](#) type exposes the following members.

## ▪ Properties

	Name	Description
	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius.
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit.
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins.

[Top](#)

## ▪ See Also

[Reference](#)

## IO.Interfaces.Temperature Namespace

---

# Sensor Properties

The [Sensor](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Celsius</a>	Read-only property returning the temperature in degrees Celsius.
	<a href="#">Fahrenheit</a>	Read-only property returning the temperature in degrees Fahrenheit.
	<a href="#">Kelvins</a>	Read-only property returning the temperature in Kelvins.

[Top](#)

## See Also

### Reference

[Sensor Interface](#)

[IO.Interfaces.Temperature Namespace](#)

# SensorCelsius Property

Read-only property returning the temperature in degrees Celsius.

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
double Celsius { get; }
```

### Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Sensor Interface](#)

[IO.Interfaces.Temperature Namespace](#)

# SensorFahrenheit Property

Read-only property returning the temperature in degrees Fahrenheit.

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
double Fahrenheit { get; }
```

### Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Sensor Interface](#)

[IO.Interfaces.Temperature Namespace](#)

# SensorKelvins Property

Read-only property returning the temperature in Kelvins.

**Namespace:** [IO.Interfaces.Temperature](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
double Kelvins { get; }
```

### Property Value

Type: [Double](#)

## ◀ See Also

### Reference

[Sensor Interface](#)

[IO.Interfaces.Temperature Namespace](#)

# IO.Interfaces.Watchdog Namespace

Abstract Interface for Watchdog Timers

## ► Interfaces

Interface	Description
 Timer	Abstract interface for watchdog timers.

# Timer Interface

Abstract interface for watchdog timers.

**Namespace:** [IO.Interfaces.Watchdog](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public interface Timer
```

The [Timer](#) type exposes the following members.

## ▀ Properties

	Name	Description
	<a href="#">timeout</a>	Read/Write watchdog timer period property.

[Top](#)

## ▀ Methods

	Name	Description
	<a href="#">Kick</a>	Reset the watchdog timer.

[Top](#)

## ▀ See Also

## Reference

### IO.Interfaces.Watchdog Namespace

---

# Timer Properties

The [Timer](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">timeout</a>	Read/Write watchdog timer period property.

[Top](#)

## See Also

[Reference](#)

[Timer Interface](#)

[IO.Interfaces.Watchdog Namespace](#)

# Timertimeout Property

Read/Write watchdog timer period property.

**Namespace:** [IO.Interfaces.Watchdog](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
int timeout { get; set; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Timer Interface](#)

[IO.Interfaces.Watchdog Namespace](#)

# Timer Methods

The [Timer](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Kick</a>	Reset the watchdog timer.

[Top](#)

## See Also

### Reference

[Timer Interface](#)

[IO.Interfaces.Watchdog Namespace](#)

# TimerKick Method

Reset the watchdog timer.

**Namespace:** [IO.Interfaces.Watchdog](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
void Kick()
```

## ◀ See Also

**Reference**

[Timer Interface](#)

[IO.Interfaces.Watchdog Namespace](#)

# IO.Objects.libsimpleio.ADC Namespace

ADC (Analog to Digital Converter) Input Services

## ↳ Classes

Class	Description
 <a href="#">Sample</a>	Encapsulates Linux Industrial I/O Subsystem ADC inputs using <a href="#">libsimpleio</a> .

# Sample Class

Encapsulates Linux Industrial I/O Subsystem ADC inputs using [libsimpleio](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.ADCSample](#)

**Namespace:** [IO.Objects.libsimpleio.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Sample : Sample
```

The [Sample](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Sample</a>	Constructor for a single ADC input.

[Top](#)

## ▪ Properties

	Name	Description
☰	<a href="#">fd</a>	Read-only property returning the Linux

file descriptor for the ADC input.



**resolution** Read-only property returning the number of bits of resolution.



**sample** Read-only property returning an integer analog sample from an ADC input.

[Top](#)

## ◀ Methods

	Name	Description
 S	<b>name</b>	Retrieve the subsystem name string for a Linux Industrial I/O Subsystem ADC device.

[Top](#)

## ◀ See Also

### Reference

[IO.Objects.libsimpleio.ADC Namespace](#)

# Sample Constructor

Constructor for a single ADC input.

**Namespace:** [IO.Objects.libsimpleio.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample(  
    Designator desg,  
    int resolution  
)
```

## Parameters

*desg*

Type: [IO.Objects.libsimpleio.DeviceDesignator](#)

ADC input designator.

*resolution*

Type: [SystemInt32](#)

Bits of resolution.

## ◀ See Also

[Reference](#)

[Sample Class](#)

[IO.Objects.libsimpleio.ADC Namespace](#)

# Sample Properties

The [Sample](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the ADC input.
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.
	<a href="#">sample</a>	Read-only property returning an integer analog sample from an ADC input.

[Top](#)

## See Also

### Reference

[Sample Class](#)

[IO.Objects.libsimpleio.ADC Namespace](#)

# Samplefd Property

Read-only property returning the Linux file descriptor for the ADC input.

**Namespace:** [IO.Objects.libsimpleio.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int fd { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Sample Class](#)

[IO.Objects.libsimpleio.ADC Namespace](#)

# Sampleresolution Property

Read-only property returning the number of bits of resolution.

**Namespace:** [IO.Objects.libsimpleio.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public int resolution { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Sampleresolution](#)

## ▪ See Also

[Reference](#)

[Sample Class](#)

[IO.Objects.libsimpleio.ADC Namespace](#)

# Samplesample Property

Read-only property returning an integer analog sample from an ADC input.

**Namespace:** [IO.Objects.libsimpleio.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public int sample { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Samplesample](#)

## ◀ See Also

### Reference

[Sample Class](#)

[IO.Objects.libsimpleio.ADC Namespace](#)

# Sample Methods

The [Sample](#) type exposes the following members.

## ▪ Methods

	Name	Description
 <b>S</b>	<a href="#">name</a>	Retrieve the subsystem name string for a Linux Industrial I/O Subsystem ADC device.

[Top](#)

## ▪ See Also

[Reference](#)

[Sample Class](#)

[IO.Objects.libsimpleio.ADC Namespace](#)

# Samplename Method

Retrieve the subsystem name string for a Linux Industrial I/O Subsystem ADC device.

**Namespace:** [IO.Objects.libsimpleio.ADC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static string name(  
    int chip  
)
```

## Parameters

*chip*

Type: [SystemInt32](#)

ADC chip number.

## Return Value

Type: [String](#)

Subsystem name.

## ◀ See Also

[Reference](#)

[Sample Class](#)

[IO.Objects.libsimpleio.ADC Namespace](#)

# IO.Objects.libsimpleio.DAC Namespace

DAC (Digital to Analog Converter) Output Services

## ↳ Classes

Class	Description
	<a href="#">Sample</a> Encapsulates Linux Industrial I/O Subsystem DAC outputs using <a href="#">libsimpleio</a> .

# Sample Class

Encapsulates Linux Industrial I/O Subsystem DAC outputs using [libsimpleio](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.DACSample](#)

**Namespace:** [IO.Objects.libsimpleio.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public class Sample : Sample
```

The [Sample](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Sample</a>	Constructor for a single DAC output.

[Top](#)

## ▪ Properties

	Name	Description
☰	<a href="#">fd</a>	Read-only property returning the Linux

file descriptor for the DAC output.



**resolution** Read-only property returning the number of bits of resolution.



**sample** Write-only property for writing an integer analog sample to a DAC output.

[Top](#)

## ◀ Methods

	Name	Description
	<b>name</b>	Retrieve the subsystem name string for a Linux Industrial I/O Subsystem DAC device.

[Top](#)

## ◀ See Also

### Reference

[IO.Objects.libsimpleio.DAC Namespace](#)

# Sample Constructor

Constructor for a single DAC output.

**Namespace:** [IO.Objects.libsimpleio.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample(  
    Designator desg,  
    int resolution,  
    int sample = 0  
)
```

## Parameters

*desg*

Type: [IO.Objects.libsimpleio.DeviceDesignator](#)

DAC output designator.

*resolution*

Type: [SystemInt32](#)

Bits of resolution.

*sample* (Optional)

Type: [SystemInt32](#)

Initial DAC output sample.

## ◀ See Also

[Reference](#)

[Sample Class](#)

## IO.Objects.libsimpleio.DAC Namespace

---

# Sample Properties

The [Sample](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the DAC output.
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.
	<a href="#">sample</a>	Write-only property for writing an integer analog sample to a DAC output.

[Top](#)

## See Also

### Reference

[Sample Class](#)

[IO.Objects.libsimpleio.DAC Namespace](#)

# Samplefd Property

Read-only property returning the Linux file descriptor for the DAC output.

**Namespace:** [IO.Objects.libsimpleio.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public int fd { get; }
```

## Property Value

Type: [Int32](#)

## ◀ See Also

[Reference](#)

[Sample Class](#)

[IO.Objects.libsimpleio.DAC Namespace](#)

# Sampleresolution Property

Read-only property returning the number of bits of resolution.

**Namespace:** [IO.Objects.libsimpleio.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public int resolution { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Sampleresolution](#)

## ▪ See Also

[Reference](#)

[Sample Class](#)

[IO.Objects.libsimpleio.DAC Namespace](#)

# Samplesample Property

Write-only property for writing an integer analog sample to a DAC output.

**Namespace:** [IO.Objects.libsimpleio.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▲ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public int sample { set; }
```

### Property Value

Type: [Int32](#)

### Implements

[Samplesample](#)

## ▲ See Also

### Reference

[Sample Class](#)

[IO.Objects.libsimpleio.DAC Namespace](#)

# Sample Methods

The [Sample](#) type exposes the following members.

## ▪ Methods

	Name	Description
 	<a href="#">name</a>	Retrieve the subsystem name string for a Linux Industrial I/O Subsystem DAC device.

[Top](#)

## ▪ See Also

[Reference](#)

[Sample Class](#)

[IO.Objects.libsimpleio.DAC Namespace](#)

# Samplename Method

Retrieve the subsystem name string for a Linux Industrial I/O Subsystem DAC device.

**Namespace:** [IO.Objects.libsimpleio.DAC](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static string name(  
    int chip  
)
```

## Parameters

*chip*

Type: [SystemInt32](#)

DAC chip number.

## Return Value

Type: [String](#)

Subsystem name.

## ◀ See Also

[Reference](#)

[Sample Class](#)

[IO.Objects.libsimpleio.DAC Namespace](#)

# IO.Objects.libsimpleio.Device Namespace

Common device declarations

## ▪ Structures

	Structure	Description
	<a href="#">Designator</a>	Linux kernel I/O device designator..

# Designator Structure

Linux kernel I/O device designator..

**Namespace:** [IO.Objects.libsimpleio.Device](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public struct Designator
```

The [Designator](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Designator</a>	device pin designator constructor.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">available</a>	Returns <code>true</code> if this device designator is not equal to <a href="#">Unavailable</a> .

[Top](#)

## ▪ Fields

	Name	Description
◆	<a href="#">chan</a>	Linux kernel I/O device channel number.
◆	<a href="#">chip</a>	Linux kernel I/O device chip number.
◆ <b>S</b>	<a href="#">Unavailable</a>	Linux kernel I/O device designator for an unavailable device.

[Top](#)

## ◀ Remarks

Many Linux kernel I/O devices, including ADC inputs, DAC outputs, GPIO pins, I2C buses, PWM outputs, and SPI devices, are selected by a tuple of integers: chip and channel.

## ◀ See Also

### Reference

[IO.Objects.libsimpleio.Device Namespace](#)

# Designator Constructor

device pin designator constructor.

**Namespace:** [IO.Objects.libsimpleio.Device](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator(  
    uint chip,  
    uint chan  
)
```

## Parameters

*chip*

Type: [SystemUInt32](#)

Linux kernel I/O device chip number.

*chan*

Type: [SystemUInt32](#)

Linux kernel I/O device channel number.

## ◀ See Also

### Reference

[Designator Structure](#)

[IO.Objects.libsimpleio.Device Namespace](#)

# Designator Properties

The [Designator](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">available</a>	Returns <code>true</code> if this device designator is not equal to <a href="#">Unavailable</a> .

[Top](#)

## See Also

[Reference](#)

[Designator Structure](#)

[IO.Objects.libsimpleio.Device Namespace](#)

# Designatoravailable Property

Returns `true` if this device designator is not equal to `Unavailable`.

**Namespace:** [IO.Objects.libsimpleio.Device](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public bool available { get; }
```

## Property Value

Type: [Boolean](#)

## « See Also

### Reference

[Designator Structure](#)

[IO.Objects.libsimpleio.Device Namespace](#)

# Designator Fields

The [Designator](#) type exposes the following members.

## Fields

	Name	Description
◆	<a href="#">chan</a>	Linux kernel I/O device channel number.
◆	<a href="#">chip</a>	Linux kernel I/O device chip number.
◆ <b>S</b>	<a href="#">Unavailable</a>	Linux kernel I/O device designator for an unavailable device.

[Top](#)

## See Also

### Reference

[Designator Structure](#)

[IO.Objects.libsimpleio.Device Namespace](#)

# Designatorchan Field

Linux kernel I/O device channel number.

**Namespace:** [IO.Objects.libsimpleio.Device](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public uint chan
```

### Field Value

Type: [UInt32](#)

## ◀ See Also

### Reference

[Designator Structure](#)

[IO.Objects.libsimpleio.Device Namespace](#)

# Designatorchip Field

Linux kernel I/O device chip number.

**Namespace:** [IO.Objects.libsimpleio.Device](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public uint chip
```

### Field Value

Type: [UInt32](#)

## ◀ See Also

### Reference

[Designator Structure](#)

[IO.Objects.libsimpleio.Device Namespace](#)

# DesignatorUnavailable Field

Linux kernel I/O device designator for an unavailable device.

**Namespace:** [IO.Objects.libsimpleio.Device](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator Unavailable
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Designator Structure](#)

[IO.Objects.libsimpleio.Device Namespace](#)

# IO.Objects.libsimpleio.Exceptions Namespace

Exception Services

## ↳ Classes

Class	Description
 <a href="#">Exception</a>	Encapsulates exceptions that may include an optional <code>errno</code> value.

# Exception Class

Encapsulates exceptions that may include an optional `errno` value.

## ▪ Inheritance Hierarchy

[SystemObject](#) [SystemException](#)  
[IO.Objects.libsimpleio.ExceptionsException](#)

**Namespace:** [IO.Objects.libsimpleio.Exceptions](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Exception : Exception
```

## ▪ Constructors

	Name	Description
≡	<a href="#">Exception(String)</a>	Constructor for an exception that writes an error message to standard output.
≡	<a href="#">Exception(String, Int32)</a>	Constructor for an exception that writes an error message including an <code>errno</code> value.

[Top](#)

## ↳ See Also

### Reference

[IO.Objects.libsimpleio.Exceptions Namespace](#)

---

# Exception Constructor

## ↳ Overload List

	Name	Description
≡	<a href="#">Exception(String)</a>	Constructor for an exception that writes an error message to standard output.
≡	<a href="#">Exception(String, Int32)</a>	Constructor for an exception that writes an error message including an <code>errno</code> value.

[Top](#)

## ↳ See Also

### Reference

[Exception Class](#)

[IO.Objects.libsimpleio.Exceptions Namespace](#)

# Exception Constructor (String)

Constructor for an exception that writes an error message to standard output.

**Namespace:** [IO.Objects.libsimpleio.Exceptions](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public Exception(  
    string message  
)
```

## Parameters

*message*

Type: [System.String](#)

Error message.

## ◀ See Also

### Reference

[Exception Class](#)

[Exception Overload](#)

[IO.Objects.libsimpleio.Exceptions Namespace](#)

# Exception Constructor (String, Int32)

Constructor for an exception that writes an error message including an `errno` value.

**Namespace:** [IO.Objects.libsimpleio.Exceptions](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

**C#**    **VB**    **F#**

[Copy](#)

```
public Exception(  
    string message,  
    int error  
)
```

## Parameters

*message*

Type: [System.String](#)

Error message.

*error*

Type: [System.Int32](#)

Error code.

## ↳ See Also

### Reference

[Exception Class](#)

[Exception Overload](#)

## IO.Objects.libsimpleio.Exceptions Namespace

---

# IO.Objects.libsimpleio.GPIO Namespace

GPIO (General Purpose Input/Output) Pin Services

## Classes

	Class	Description
	<a href="#">Pin</a>	Encapsulates Linux GPIO pins using <a href="#">libsimpleio</a> .

## Enumerations

	Enumeration	Description
	<a href="#">PinDriver</a>	GPIO output driver settings.
	<a href="#">PinEdge</a>	GPIO input interrupt edge settings.
	<a href="#">PinPolarity</a>	GPIO polarity settings

# Pin Class

Encapsulates Linux GPIO pins using [libsimpleio](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.GPIOPin](#)

**Namespace:** [IO.Objects.libsimpleio.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Pin : Pin
```

The [Pin](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Pin</a>	Constructor for a single GPIO pin.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the GPIO pin.



[state](#)

Read/Write GPIO state property.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Objects.libsimpleio.GPIO Namespace](#)

---

# Pin Constructor

Constructor for a single GPIO pin.

**Namespace:** [IO.Objects.libsimpleio.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin(  
    Designator desg,  
    Direction dir,  
    bool state = false,  
    PinDriver driver = PinDriver.PushPull,  
    PinEdge edge = PinEdge.None,  
    PinPolarity polarity = PinPolarity.ActiveHigh  
)
```

## Parameters

*desg*

Type: [IO.Objects.libsimpleio.DeviceDesignator](#)  
GPIO pin designator.

*dir*

Type: [IO.Interfaces.GPIODirection](#)  
Data direction.

*state* (Optional)

Type: [SystemBoolean](#)  
Initial GPIO output state.

*driver* (Optional)

Type: [IO.Objects.libsimpleio.GPIOPinDriver](#)

Output driver setting

***edge (Optional)***

Type: [IO.Objects.libsimpleio.GPIOPinEdge](#)

Interrupt edge setting.

***polarity (Optional)***

Type: [IO.Objects.libsimpleio.GPIOPinPolarity](#)

Polarity setting.

## ◀ See Also

### Reference

[Pin Class](#)

[IO.Objects.libsimpleio.GPIO Namespace](#)

# Pin Properties

The [Pin](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the GPIO pin.
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## See Also

[Reference](#)

[Pin Class](#)

[IO.Objects.libsimpleio.GPIO Namespace](#)

# Pinfd Property

Read-only property returning the Linux file descriptor for the GPIO pin.

**Namespace:** [IO.Objects.libsimpleio.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int fd { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Pin Class](#)

[IO.Objects.libsimpleio.GPIO Namespace](#)

# Pinstate Property

Read/Write GPIO state property.

**Namespace:** [IO.Objects.libsimpleio.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public bool state { get; set; }
```

### Property Value

Type: [Boolean](#)

### Implements

[Pinstate](#)

## ▪ See Also

[Reference](#)

[Pin Class](#)

[IO.Objects.libsimpleio.GPIO Namespace](#)

# PinDriver Enumeration

GPIO output driver settings.

**Namespace:** [IO.Objects.libsimpleio.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum Driver
```

## ▪ Members

Member name	Value	Description
PushPull	0	Push Pull (current source/sink) output driver.
OpenDrain	1	Open Drain (current sink) output driver.
OpenSource	2	Open Source (current source) output driver.

## ▪ See Also

### Reference

[IO.Objects.libsimpleio.GPIO Namespace](#)

# PinEdge Enumeration

GPIO input interrupt edge settings.

**Namespace:** [IO.Objects.libsimpleio.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum Edge
```

## ▪ Members

Member name	Value	Description
None	0	Configure GPIO input pin with interrupt disabled.
Rising	1	Configure GPIO input pin to interrupt on rising edge.
Falling	2	Configure GPIO pin to interrupt on falling edge.
Both	3	Configure GPIO pin to interrupt on both edges.

## ▪ See Also

## Reference

### [IO.Objects.libsimpleio.GPIO Namespace](#)

# PinPolarity Enumeration

GPIO polarity settings

**Namespace:** [IO.Objects.libsimpleio.GPIO](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum Polarity
```

## ▪ Members

Member name	Value	Description
ActiveLow	0	Configure GPIO pin as active low (inverted logic).
ActiveHigh	1	Configure GPIO pin as active high (normal logic).

## ▪ See Also

### Reference

[IO.Objects.libsimpleio.GPIO Namespace](#)

# IO.Objects.libsimpleio.HID Namespace

Raw HID (Human Interface Device) Services

## ↳ Classes

Class	Description
 <a href="#">Messenger</a>	Encapsulates Linux raw HID devices using <a href="#">libsimpleio</a> .

# Messenger Class

Encapsulates Linux raw HID devices using [libsimpleio](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.HIDMessenger](#)

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Messenger : Messenger
```

The Messenger type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Messenger(String, Int32)</a>	Constructor for a single raw HID device.
≡	<a href="#">Messenger(Int32, Int32, Int32)</a>	Constructor for a single raw HID device.

[Top](#)

## ▪ Properties

	<b>Name</b>	<b>Description</b>
	<a href="#">bustype</a>	Read-only property returning the bus type identifierfor a raw HID device.
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for a raw HID device.
	<a href="#">name</a>	Read-only property returning the device information string for a raw HID device.
	<a href="#">product</a>	Read-only property returning the vendor identifierfor a raw HID device.
	<a href="#">vendor</a>	Read-only property returning the bus type identifierfor a raw HID device.

[Top](#)

## ◀ Methods

	<b>Name</b>	<b>Description</b>
	<a href="#">Receive</a>	Receive a 64-byte response message from a raw HID device.
	<a href="#">Send</a>	Send a 64-byte command message to a raw HID device.
	<a href="#">Transaction</a>	Send a 64-byte command message and receive a 64-byte response message.

[Top](#)

## ◀ See Also

## Reference

### [IO.Objects.libsimpleio.HID Namespace](#)

# Messenger Constructor

## ↳ Overload List

	Name	Description
≡	<a href="#">Messenger(String, Int32)</a>	Constructor for a single raw HID device.
≡	<a href="#">Messenger(Int32, Int32, Int32)</a>	Constructor for a single raw HID device.

[Top](#)

## ↳ See Also

### Reference

[Messenger Class](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# Messenger Constructor (String, Int32)

Constructor for a single raw HID device.

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public Messenger(  
    string devname,  
    int timeoutms = 1000  
)
```

## Parameters

*devname*

Type: [SystemString](#)

Device node name.

*timeoutms* (Optional)

Type: [SystemInt32](#)

Time in milliseconds to wait for read and write operations to complete. Zero means wait forever.

## ◀ See Also

### Reference

[Messenger Class](#)

[Messenger Overload](#)

## IO.Objects.libsimpleio.HID Namespace

---

# Messenger Constructor (Int32, Int32, Int32)

Constructor for a single raw HID device.

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public Messenger(  
    int VID = 5840,  
    int PID = 2810,  
    int timeoutms = 1000  
)
```

## Parameters

### VID (Optional)

Type: [SystemInt32](#)

Vendor ID.

### PID (Optional)

Type: [SystemInt32](#)

Product ID.

### timeoutms (Optional)

Type: [SystemInt32](#)

Time in milliseconds to wait for read and write operations to complete. Zero means wait forever.

## ↳ See Also

### Reference

[Messenger Class](#)

[Messenger Overload](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# Messenger Properties

The [Messenger](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">bustype</a>	Read-only property returning the bus type identifierfor a raw HID device.
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for a raw HID device.
	<a href="#">name</a>	Read-only property returning the device information string for a raw HID device.
	<a href="#">product</a>	Read-only property returning the vendor identifierfor a raw HID device.
	<a href="#">vendor</a>	Read-only property returning the bus type identifierfor a raw HID device.

[Top](#)

## See Also

### Reference

[Messenger Class](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# Messengerbusstype Property

Read-only property returning the bus type identifier for a raw HID device.

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int bustype { get; }
```

## Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Messenger Class](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# Messengerfd Property

Read-only property returning the Linux file descriptor for a raw HID device.

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int fd { get; }
```

## Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Messenger Class](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# Messengername Property

Read-only property returning the device information string for a raw HID device.

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public string name { get; }
```

## Property Value

Type: [String](#)

## ◀ See Also

### Reference

[Messenger Class](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# Messengerproduct Property

Read-only property returning the vendor identifier for a raw HID device.

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int product { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Messenger Class](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# Messengervendor Property

Read-only property returning the bus type identifierfor a raw HID device.

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int vendor { get; }
```

## Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Messenger Class](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# Messenger Methods

The [Messenger](#) type exposes the following members.

## ▪ Methods

	Name	Description
≡	<a href="#">Receive</a>	Receive a 64-byte response message from a raw HID device.
≡	<a href="#">Send</a>	Send a 64-byte command message to a raw HID device.
≡	<a href="#">Transaction</a>	Send a 64-byte command message and receive a 64-byte response message.

[Top](#)

## ▪ See Also

### Reference

[Messenger Class](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# MessengerReceive Method

Receive a 64-byte response message from a raw HID device.

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Receive(  
    Message resp  
)
```

## Parameters

*resp*

Type: [IO.Interfaces.Message64Message](#)

64-byte response message.

## Implements

[MessengerReceive\(Message\)](#)

## ◀ See Also

### Reference

[Messenger Class](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# MessengerSend Method

Send a 64-byte command message to a raw HID device.

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Send(  
    Message cmd  
)
```

## Parameters

*cmd*

Type: [IO.Interfaces.Message64Message](#)

64-byte command message.

## Implements

[MessengerSend\(Message\)](#)

## ◀ See Also

### Reference

[Messenger Class](#)

[IO.Objects.libsimpleio.HID Namespace](#)

# MessengerTransaction Method

Send a 64-byte command message and receive a 64-byte response message.

**Namespace:** [IO.Objects.libsimpleio.HID](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Transaction(  
    Message cmd,  
    Message resp  
)
```

## Parameters

*cmd*

Type: [IO.Interfaces.Message64Message](#)  
64-byte command message.

*resp*

Type: [IO.Interfaces.Message64Message](#)  
64-byte response message.

## Implements

[MessengerTransaction\(Message, Message\)](#)

## ◀ See Also

### Reference

[Messenger Class](#)

## IO.Objects.libsimpleio.HID Namespace

---

# IO.Objects.libsimpleio.I2C Namespace

I<sup>2</sup>C (Inter-Integrated Circuit) Bus Controller Services

## ↳ Classes

	Class	Description
	<a href="#">Bus</a>	Encapsulates Linux I <sup>2</sup> C bus controllers using <a href="#">libsimpleio</a> .

# Bus Class

Encapsulates Linux I<sup>2</sup>C bus controllers using [libsimpleio](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.I2CBus](#)

**Namespace:** [IO.Objects.libsimpleio.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Bus : Bus
```

The [Bus](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Bus(String)</a>	Constructor for a single I <sup>2</sup> C bus controller.
≡	<a href="#">Bus(Designator)</a>	Constructor for a single I <sup>2</sup> C bus controller.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the I <sup>2</sup> C bus controller.

[Top](#)

## ◀ Methods

	Name	Description
	<a href="#">Read</a>	Read bytes from an I <sup>2</sup> C device.
	<a href="#">Transaction</a>	Write and receive bytes to/from an I <sup>2</sup> C device.
	<a href="#">Write</a>	Write bytes to an I <sup>2</sup> C device.

[Top](#)

## ◀ See Also

### Reference

[IO.Objects.libsimpleio.I2C Namespace](#)

# Bus Constructor

## ↳ Overload List

	Name	Description
≡	<a href="#">Bus(String)</a>	Constructor for a single I <sup>2</sup> C bus controller.
≡	<a href="#">Bus(Designator)</a>	Constructor for a single I <sup>2</sup> C bus controller.

[Top](#)

## ↳ See Also

### Reference

[Bus Class](#)

[IO.Objects.libsimpleio.I2C Namespace](#)

# Bus Constructor (String)

Constructor for a single I<sup>2</sup>C bus controller.

**Namespace:** [IO.Objects.libsimpleio.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public Bus(  
    string devname  
)
```

## Parameters

*devname*

Type: [SystemString](#)

Device node name.

## ▪ See Also

### Reference

[Bus Class](#)

[Bus Overload](#)

[IO.Objects.libsimpleio.I2C Namespace](#)

# Bus Constructor (Designator)

Constructor for a single I<sup>2</sup>C bus controller.

**Namespace:** [IO.Objects.libsimpleio.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public Bus(  
    Designator desg  
)
```

## Parameters

*desg*

Type: [IO.Objects.libsimpleio.DeviceDesignator](#)

I<sup>2</sup> bus designator.

## ▪ See Also

### Reference

[Bus Class](#)

[Bus Overload](#)

[IO.Objects.libsimpleio.I2C Namespace](#)

# Bus Properties

The [Bus](#) type exposes the following members.

## ▪ Properties

	Name	Description
	fd	Read-only property returning the Linux file descriptor for the I <sup>2</sup> C bus controller.

[Top](#)

## ▪ See Also

[Reference](#)

[Bus Class](#)

[IO.Objects.libsimpleio.I2C Namespace](#)

# Busfd Property

Read-only property returning the Linux file descriptor for the I<sup>2</sup>C bus controller.

**Namespace:** [IO.Objects.libsimpleio.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public int fd { get; }
```

## Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Bus Class](#)

[IO.Objects.libsimpleio.I2C Namespace](#)

# Bus Methods

The [Bus](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Read</a>	Read bytes from an I <sup>2</sup> C device.
	<a href="#">Transaction</a>	Write and receive bytes to/from an I <sup>2</sup> C device.
	<a href="#">Write</a>	Write bytes to an I <sup>2</sup> C device.

[Top](#)

## See Also

### Reference

[Bus Class](#)

[IO.Objects.libsimpleio.I2C Namespace](#)

# BusRead Method

Read bytes from an I<sup>2</sup>C device.

**Namespace:** [IO.Objects.libsimpleio.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Read(  
    int slaveaddr,  
    byte[] resp,  
    int resplen  
)
```

## Parameters

*slaveaddr*

Type: [SystemInt32](#)

Slave device address.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

## Implements

[BusRead\(Int32, Byte, Int32\)](#)

## ◀ See Also

## Reference

### Bus Class

[IO.Objects.libsimpleio.I2C Namespace](#)

# BusTransaction Method

Write and receive bytes to/from an I<sup>2</sup>C device.

**Namespace:** [IO.Objects.libsimpleio.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Transaction(  
    int slaveaddr,  
    byte[] cmd,  
    int cmdlen,  
    byte[] resp,  
    int resplen,  
    int delayus = 0  
)
```

## Parameters

*slaveaddr*

Type: [SystemInt32](#)

Device slave address.

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

*delayus* (Optional)

Type: [SystemInt32](#)

Delay in microseconds between the I<sup>2</sup>C write and read cycles.

Allowed values are 0 to 65535 microseconds.

## Implements

[BusTransaction\(Int32, Byte, Int32, Byte, Int32, Int32\)](#)

## See Also

### Reference

[Bus Class](#)

[IO.Objects.libsimpleio.I2C Namespace](#)

# BusWrite Method

Write bytes to an I<sup>2</sup>C device.

**Namespace:** [IO.Objects.libsimpleio.I2C](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Write(  
    int slaveaddr,  
    byte[] cmd,  
    int cmdlen  
)
```

## Parameters

*slaveaddr*

Type: [SystemInt32](#)

Slave device address.

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

## Implements

[BusWrite\(Int32, Byte, Int32\)](#)

## ◀ See Also

## Reference

### Bus Class

[IO.Objects.libsimpleio.I2C Namespace](#)

# IO.Objects.libsimpleio.mikroBUS Namespace

Mikroelektronika mikroBUS (<https://www.mikroe.com/mikrobus>) Shield and Socket Services

## ▪ Classes

	Class	Description
	<a href="#">Shield</a>	Encapsulates mikroBUS shields (add-on boards providing mikroBUS sockets).
	<a href="#">Socket</a>	Encapsulates mikroBUS sockets.

## ▪ Enumerations

	Enumeration	Description
	<a href="#">ShieldKinds</a>	Supported mikroBUS shields.

# Shield Class

Encapsulates mikroBUS shields (add-on boards providing [mikroBUS](#) sockets).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.mikroBUSShield](#)

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static class Shield
```

The [Shield](#) type exposes the following members.

## ▪ Properties

	Name	Description
 <a href="#">S</a>	<a href="#">kind</a>	Returns the kind of mikroBUS shield that is installed on the target board, as obtained from the <a href="#">SHIELDNAME</a> environment variable or guessed from the <a href="#">BOARDNAME</a> environment variable. The guessed value for BeagleBone family target boards ( <a href="#">BOARDNAME</a> == "BeagleBone*") is <a href="#">BeagleBoneClick4</a> .

The guessed value for Raspberry Pi family target boards (`BOARDNAME == "RaspberryPi*"`) is [PiClick3](#).

---

[Top](#)

## ↳ Fields

	Name	Description
•  	<a href="#">I2CBus</a>	Shared I <sup>2</sup> C bus that is common to all sockets on this shield.

---

[Top](#)

## ↳ See Also

### Reference

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# Shield Properties

The [Shield](#) type exposes the following members.

## Properties

	Name	Description
 	<a href="#">kind</a>	Returns the kind of mikroBUS shield that is installed on the target board, as obtained from the <code>SHIELDNAME</code> environment variable or guessed from the <code>BOARDNAME</code> environment variable. The guessed value for BeagleBone family target boards ( <code>BOARDNAME == "BeagleBone*"</code> ) is <code>BeagleBoneClick4</code> . The guessed value for Raspberry Pi family target boards ( <code>BOARDNAME == "RaspberryPi*"</code> ) is <code>PiClick3</code> .

[Top](#)

## See Also

### Reference

[Shield Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# Shieldkind Property

Returns the kind of mikroBUS shield that is installed on the target board, as obtained from the `SHIELDNAME` environment variable or guessed from the `BOARDNAME` environment variable. The guessed value for BeagleBone family target boards (`BOARDNAME == "BeagleBone*"`) is `BeagleBoneClick4`. The guessed value for Raspberry Pi family target boards (`BOARDNAME == "RaspberryPi*"`) is `PiClick3`.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public static ShieldKinds kind { get; }
```

## Property Value

Type: [ShieldKinds](#)

## ▪ See Also

[Reference](#)

[Shield Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# Shield Fields

The [Shield](#) type exposes the following members.

## Fields

Name	Description
 <a href="#">I2CBus</a>	Shared I <sup>2</sup> C bus that is common to all sockets on this shield.

[Top](#)

## See Also

[Reference](#)

[Shield Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# ShieldI2CBus Field

Shared I<sup>2</sup>C bus that is common to all sockets on this shield.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public static Bus I2CBus
```

### Field Value

Type: [Bus](#)

## ◀ See Also

### Reference

[Shield Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# ShieldKinds Enumeration

Supported mikroBUS shields.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum Kinds
```

## ▪ Members

Member name	Value	Description
BeagleBoneClick2	0	Mikroelektronika BeagleBone Click Shield <a href="#">MIKROE-1596</a> , with two mikroBUS sockets. (Obsolete, but still useful.)
BeagleBoneClick4	1	Mikroelektronika mikroBUS Cape <a href="#">MIKROE-1857</a> with four mikroBUS sockets.

PiClick1	2	Mikroelektronika Pi Click Shield <a href="#">MIKROE-1512/1513</a> for 26-pin expansion header, with one mikroBUS socket (Obsolete.)
PiClick2	3	Mikroelektronika Pi 2 Click Shield <a href="#">MIKROE-1879</a> for 40-pin expansion header, with two mikroBUS sockets.
PiClick3	4	Mikroelektronika Pi 3 Click Shield <a href="#">MIKROE-2756</a> for 40-pin expansion header, with selectable on- board A/D converter and two mikroBUS sockets.
PocketBeagle	5	<a href="#">PocketBeagle</a> with female headers on top, with two

mikroBUS  
sockets.

Unknown

2147483647

No known  
mikroBUS shield  
installed.

## ↳ See Also

### Reference

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# Socket Class

Encapsulates mikroBUS sockets.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.mikroBUSSocket](#)

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Socket
```

The [Socket](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Socket</a>	Constructor for a single mikroBUS socket.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">AIN</a>	Returns the ADC input designator for AN.

	<a href="#">AN</a>	Returns the GPIO pin designator for AN.
	<a href="#">CS</a>	Returns the GPIO pin designator for CS.
	<a href="#">I2CBus</a>	Returns the I <sup>2</sup> C bus designator for this socket.
	<a href="#">INT</a>	Returns the GPIO pin designator for INT.
	<a href="#">MISO</a>	Returns the GPIO pin designator for MISO.
	<a href="#">MOSI</a>	Returns the GPIO pin designator for MOSI.
	<a href="#">PWM</a>	Returns the GPIO pin designator for PWM.
	<a href="#">PWMOut</a>	Returns the PWM output designator for this socket.
	<a href="#">RST</a>	Returns the GPIO pin designator for RST.
	<a href="#">RX</a>	Returns the GPIO pin designator for RX.
	<a href="#">SCK</a>	Returns the GPIO pin designator for SCK.
	<a href="#">SCL</a>	Returns the GPIO pin designator for SCL.
	<a href="#">SDA</a>	Returns the GPIO pin designator for SDA.
	<a href="#">SPIDev</a>	Returns the SPI device designator for

this socket.



[TX](#)

Returns the GPIO pin designator for TX.



[UART](#)

Returns the UART device name for this socket.

---

[Top](#)

## « See Also

### Reference

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

---

# Socket Constructor

Constructor for a single mikroBUS socket.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Socket(  
    int num,  
    ShieldKinds shield = ShieldKinds.Unknown  
)
```

## Parameters

*num*

Type: [SystemInt32](#)

Socket number.

*shield* (Optional)

Type: [IO.Objects.libsimpleio.mikroBUSShieldKinds](#)

mikroBUS shield kind. `Shield.Kinds.Unknown` indicates automatic detection using the `Shield.kind` property.

## ◀ See Also

[Reference](#)

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# Socket Properties

The [Socket](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">AIN</a>	Returns the ADC input designator for AN.
	<a href="#">AN</a>	Returns the GPIO pin designator for AN.
	<a href="#">CS</a>	Returns the GPIO pin designator for CS.
	<a href="#">I2CBus</a>	Returns the I <sup>2</sup> C bus designator for this socket.
	<a href="#">INT</a>	Returns the GPIO pin designator for INT.
	<a href="#">MISO</a>	Returns the GPIO pin designator for MISO.
	<a href="#">MOSI</a>	Returns the GPIO pin designator for MOSI.
	<a href="#">PWM</a>	Returns the GPIO pin designator for PWM.
	<a href="#">PWMOut</a>	Returns the PWM output designator for this socket.

	RST	Returns the GPIO pin designator for RST.
	RX	Returns the GPIO pin designator for RX.
	SCK	Returns the GPIO pin designator for SCK.
	SCL	Returns the GPIO pin designator for SCL.
	SDA	Returns the GPIO pin designator for SDA.
	SPIDev	Returns the SPI device designator for this socket.
	TX	Returns the GPIO pin designator for TX.
	UART	Returns the UART device name for this socket.

[Top](#)

## See Also

**Reference**

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketAIN Property

Returns the ADC input designator for AN.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public Designator AIN { get; }
```

### Property Value

Type: [Designator](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketAN Property

Returns the GPIO pin designator for AN.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public Designator AN { get; }
```

### Property Value

Type: [Designator](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketCS Property

Returns the GPIO pin designator for CS.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator CS { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketI2CBus Property

Returns the I<sup>2</sup>C bus designator for this socket.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public Designator I2CBus { get; }
```

## Property Value

Type: [Designator](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketINT Property

Returns the GPIO pin designator for INT.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public Designator INT { get; }
```

### Property Value

Type: [Designator](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketMISO Property

Returns the GPIO pin designator for MISO.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator MISO { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketMOSI Property

Returns the GPIO pin designator for MOSI.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator MOSI { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketPWM Property

Returns the GPIO pin designator for PWM.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator PWM { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketPWMOut Property

Returns the PWM output designator for this socket.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator PWMOut { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketRST Property

Returns the GPIO pin designator for RST.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public Designator RST { get; }
```

### Property Value

Type: [Designator](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketRX Property

Returns the GPIO pin designator for RX.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator RX { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketSCK Property

Returns the GPIO pin designator for SCK.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator SCK { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketSCL Property

Returns the GPIO pin designator for SCL.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator SCL { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketSDA Property

Returns the GPIO pin designator for SDA.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator SDA { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketSPIDev Property

Returns the SPI device designator for this socket.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator SPIDev { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketTX Property

Returns the GPIO pin designator for TX.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Designator TX { get; }
```

### Property Value

Type: [Designator](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# SocketUART Property

Returns the UART device name for this socket.

**Namespace:** [IO.Objects.libsimpleio.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public string UART { get; }
```

### Property Value

Type: [String](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Objects.libsimpleio.mikroBUS Namespace](#)

# IO.Objects.libsimpleio.Platforms Namespace

Platform Definition Classes

## ↳ Classes

Class	Description
 <a href="#">BeagleBone</a>	This class defines identifiers for the devices provided by the BeagleBone hardware platform.
 <a href="#">PocketBeagle</a>	This class defines identifiers for the devices provided by the PocketBeagle hardware platform.
 <a href="#">RaspberryPi</a>	This class defines identifiers for the devices provided by the Raspberry Pi hardware platform.

# BeagleBone Class

This class defines identifiers for the devices provided by the BeagleBone hardware platform.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.Platforms](#)[BeagleBone](#)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public static class BeagleBone
```

The [BeagleBone](#) type exposes the following members.

## ▪ Fields

	Name	Description
• <a href="#"><b>S</b></a>	<a href="#">AIN0</a>	ADC input designator for P9.39 (1.8V)
• <a href="#"><b>S</b></a>	<a href="#">AIN1</a>	ADC input designator for P9.40 (1.8V)
• <a href="#"><b>S</b></a>	<a href="#">AIN2</a>	ADC input designator for P9.37 (1.8V)
• <a href="#"><b>S</b></a>	<a href="#">AIN3</a>	ADC input designator for P9.38 (1.8V)
• <a href="#"><b>S</b></a>	<a href="#">AIN4</a>	ADC input designator for P9.33 (1.8V)

♦ S	AIN5	ADC input designator for P9.36 (1.8V)
♦ S	AIN6	ADC input designator for P9.35 (1.8V)
♦ S	GPIO10	Legacy GPIO pin designator for P8.31
♦ S	GPIO11	Legacy GPIO pin designator for P8.32
♦ S	GPIO110	Legacy GPIO pin designator for P9.31
♦ S	GPIO111	Legacy GPIO pin designator for P9.29
♦ S	GPIO112	Legacy GPIO pin designator for P9.30
♦ S	GPIO113	Legacy GPIO pin designator for P9.28
♦ S	GPIO115	Legacy GPIO pin designator for P9.27
♦ S	GPIO117	Legacy GPIO pin designator for P9.25
♦ S	GPIO12	Legacy GPIO pin designator for P9.20
♦ S	GPIO13	Legacy GPIO pin designator for P9.19
♦ S	GPIO14	Legacy GPIO pin designator for P9.26
♦ S	GPIO15	Legacy GPIO pin designator for P9.24
♦ S	GPIO2	Legacy GPIO pin designator for P9.22
♦ S	GPIO20	Legacy GPIO pin designator for P9.41
♦ S	GPIO22	Legacy GPIO pin designator for P8.19
♦ S	GPIO23	Legacy GPIO pin designator for P8.13
♦ S	GPIO26	Legacy GPIO pin designator for P8.14

♦	GPIO27	Legacy GPIO pin designator for P8.17
♦	GPIO3	Legacy GPIO pin designator for P9.21
♦	GPIO30	Legacy GPIO pin designator for P9.11
♦	GPIO31	Legacy GPIO pin designator for P9.13
♦	GPIO32	Legacy GPIO pin designator for P8.25
♦	GPIO33	Legacy GPIO pin designator for P8.24
♦	GPIO34	Legacy GPIO pin designator for P8.5
♦	GPIO35	Legacy GPIO pin designator for P8.6
♦	GPIO36	Legacy GPIO pin designator for P8.23
♦	GPIO37	Legacy GPIO pin designator for P8.22
♦	GPIO38	Legacy GPIO pin designator for P8.3
♦	GPIO39	Legacy GPIO pin designator for P8.4
♦	GPIO4	Legacy GPIO pin designator for P9.18
♦	GPIO44	Legacy GPIO pin designator for P8.12
♦	GPIO45	Legacy GPIO pin designator for P8.11
♦	GPIO46	Legacy GPIO pin designator for P8.16
♦	GPIO47	Legacy GPIO pin designator for P8.15
♦	GPIO48	Legacy GPIO pin designator for P9.15
♦	GPIO49	Legacy GPIO pin designator for P9.23

• S	GPIO5	Legacy GPIO pin designator for P9.17
• S	GPIO50	Legacy GPIO pin designator for P9.14
• S	GPIO51	Legacy GPIO pin designator for P9.16
• S	GPIO60	Legacy GPIO pin designator for P9.12
• S	GPIO61	Legacy GPIO pin designator for P8.26
• S	GPIO62	Legacy GPIO pin designator for P8.21
• S	GPIO63	Legacy GPIO pin designator for P8.20
• S	GPIO65	Legacy GPIO pin designator for P8.18
• S	GPIO66	Legacy GPIO pin designator for P8.7
• S	GPIO67	Legacy GPIO pin designator for P8.8
• S	GPIO68	Legacy GPIO pin designator for P8.10
• S	GPIO69	Legacy GPIO pin designator for P8.9
• S	GPIO7	Legacy GPIO pin designator for P9.42
• S	GPIO70	Legacy GPIO pin designator for P8.45
• S	GPIO71	Legacy GPIO pin designator for P8.46
• S	GPIO72	Legacy GPIO pin designator for P8.43
• S	GPIO73	Legacy GPIO pin designator for P8.44
• S	GPIO74	Legacy GPIO pin designator for P8.41
• S	GPIO75	Legacy GPIO pin designator for P8.42

• <b>S</b>	<a href="#">GPIO76</a>	Legacy GPIO pin designator for P8.39
• <b>S</b>	<a href="#">GPIO77</a>	Legacy GPIO pin designator for P8.40
• <b>S</b>	<a href="#">GPIO78</a>	Legacy GPIO pin designator for P8.37
• <b>S</b>	<a href="#">GPIO79</a>	Legacy GPIO pin designator for P8.38
• <b>S</b>	<a href="#">GPIO8</a>	Legacy GPIO pin designator for P8.35
• <b>S</b>	<a href="#">GPIO80</a>	Legacy GPIO pin designator for P8.36
• <b>S</b>	<a href="#">GPIO81</a>	Legacy GPIO pin designator for P8.34
• <b>S</b>	<a href="#">GPIO86</a>	Legacy GPIO pin designator for P8.27
• <b>S</b>	<a href="#">GPIO87</a>	Legacy GPIO pin designator for P8.29
• <b>S</b>	<a href="#">GPIO88</a>	Legacy GPIO pin designator for P8.28
• <b>S</b>	<a href="#">GPIO89</a>	Legacy GPIO pin designator for P8.30
• <b>S</b>	<a href="#">GPIO9</a>	Legacy GPIO pin designator for P8.33
• <b>S</b>	<a href="#">I2C2</a>	I2C bus designator for P9.19 and P9.20
• <b>S</b>	<a href="#">SPI2_0</a>	SPI slave select designator for P9.28
• <b>S</b>	<a href="#">SPI2_1</a>	SPI slave select designator for P9.42

[Top](#)

## See Also

### Reference

[IO.Objects.libsimpleio.Platforms Namespace](#)



# BeagleBone Fields

The [BeagleBone](#) type exposes the following members.

## Fields

	Name	Description
•  	AIN0	ADC input designator for P9.39 (1.8V)
•  	AIN1	ADC input designator for P9.40 (1.8V)
•  	AIN2	ADC input designator for P9.37 (1.8V)
•  	AIN3	ADC input designator for P9.38 (1.8V)
•  	AIN4	ADC input designator for P9.33 (1.8V)
•  	AIN5	ADC input designator for P9.36 (1.8V)
•  	AIN6	ADC input designator for P9.35 (1.8V)
•  	GPIO10	Legacy GPIO pin designator for P8.31
•  	GPIO11	Legacy GPIO pin designator for P8.32
•  	GPIO110	Legacy GPIO pin designator for P9.31
•  	GPIO111	Legacy GPIO pin designator for P9.29
•  	GPIO112	Legacy GPIO pin designator for P9.30
•  	GPIO113	Legacy GPIO pin designator for P9.28

♦ S	GPIO115	Legacy GPIO pin designator for P9.27
♦ S	GPIO117	Legacy GPIO pin designator for P9.25
♦ S	GPIO12	Legacy GPIO pin designator for P9.20
♦ S	GPIO13	Legacy GPIO pin designator for P9.19
♦ S	GPIO14	Legacy GPIO pin designator for P9.26
♦ S	GPIO15	Legacy GPIO pin designator for P9.24
♦ S	GPIO2	Legacy GPIO pin designator for P9.22
♦ S	GPIO20	Legacy GPIO pin designator for P9.41
♦ S	GPIO22	Legacy GPIO pin designator for P8.19
♦ S	GPIO23	Legacy GPIO pin designator for P8.13
♦ S	GPIO26	Legacy GPIO pin designator for P8.14
♦ S	GPIO27	Legacy GPIO pin designator for P8.17
♦ S	GPIO3	Legacy GPIO pin designator for P9.21
♦ S	GPIO30	Legacy GPIO pin designator for P9.11
♦ S	GPIO31	Legacy GPIO pin designator for P9.13
♦ S	GPIO32	Legacy GPIO pin designator for P8.25
♦ S	GPIO33	Legacy GPIO pin designator for P8.24
♦ S	GPIO34	Legacy GPIO pin designator for P8.5
♦ S	GPIO35	Legacy GPIO pin designator for P8.6

♦ S	GPIO36	Legacy GPIO pin designator for P8.23
♦ S	GPIO37	Legacy GPIO pin designator for P8.22
♦ S	GPIO38	Legacy GPIO pin designator for P8.3
♦ S	GPIO39	Legacy GPIO pin designator for P8.4
♦ S	GPIO4	Legacy GPIO pin designator for P9.18
♦ S	GPIO44	Legacy GPIO pin designator for P8.12
♦ S	GPIO45	Legacy GPIO pin designator for P8.11
♦ S	GPIO46	Legacy GPIO pin designator for P8.16
♦ S	GPIO47	Legacy GPIO pin designator for P8.15
♦ S	GPIO48	Legacy GPIO pin designator for P9.15
♦ S	GPIO49	Legacy GPIO pin designator for P9.23
♦ S	GPIO5	Legacy GPIO pin designator for P9.17
♦ S	GPIO50	Legacy GPIO pin designator for P9.14
♦ S	GPIO51	Legacy GPIO pin designator for P9.16
♦ S	GPIO60	Legacy GPIO pin designator for P9.12
♦ S	GPIO61	Legacy GPIO pin designator for P8.26
♦ S	GPIO62	Legacy GPIO pin designator for P8.21
♦ S	GPIO63	Legacy GPIO pin designator for P8.20
♦ S	GPIO65	Legacy GPIO pin designator for P8.18

♦ S	GPIO66	Legacy GPIO pin designator for P8.7
♦ S	GPIO67	Legacy GPIO pin designator for P8.8
♦ S	GPIO68	Legacy GPIO pin designator for P8.10
♦ S	GPIO69	Legacy GPIO pin designator for P8.9
♦ S	GPIO7	Legacy GPIO pin designator for P9.42
♦ S	GPIO70	Legacy GPIO pin designator for P8.45
♦ S	GPIO71	Legacy GPIO pin designator for P8.46
♦ S	GPIO72	Legacy GPIO pin designator for P8.43
♦ S	GPIO73	Legacy GPIO pin designator for P8.44
♦ S	GPIO74	Legacy GPIO pin designator for P8.41
♦ S	GPIO75	Legacy GPIO pin designator for P8.42
♦ S	GPIO76	Legacy GPIO pin designator for P8.39
♦ S	GPIO77	Legacy GPIO pin designator for P8.40
♦ S	GPIO78	Legacy GPIO pin designator for P8.37
♦ S	GPIO79	Legacy GPIO pin designator for P8.38
♦ S	GPIO8	Legacy GPIO pin designator for P8.35
♦ S	GPIO80	Legacy GPIO pin designator for P8.36
♦ S	GPIO81	Legacy GPIO pin designator for P8.34
♦ S	GPIO86	Legacy GPIO pin designator for P8.27

---

• <b>S</b>	<a href="#">GPIO87</a>	Legacy GPIO pin designator for P8.29
• <b>S</b>	<a href="#">GPIO88</a>	Legacy GPIO pin designator for P8.28
• <b>S</b>	<a href="#">GPIO89</a>	Legacy GPIO pin designator for P8.30
• <b>S</b>	<a href="#">GPIO9</a>	Legacy GPIO pin designator for P8.33
• <b>S</b>	<a href="#">I2C2</a>	I2C bus designator for P9.19 and P9.20
• <b>S</b>	<a href="#">SPI2_0</a>	SPI slave select designator for P9.28
• <b>S</b>	<a href="#">SPI2_1</a>	SPI slave select designator for P9.42

---

[Top](#)

## ↳ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneAIN0 Field

ADC input designator for P9.39 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN0
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneAIN1 Field

ADC input designator for P9.40 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN1
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneAIN2 Field

ADC input designator for P9.37 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN2
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneAIN3 Field

ADC input designator for P9.38 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN3
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneAIN4 Field

ADC input designator for P9.33 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN4
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneAIN5 Field

ADC input designator for P9.36 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN5
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneAIN6 Field

ADC input designator for P9.35 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN6
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO10 Field

Legacy GPIO pin designator for P8.31

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO10
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO11 Field

Legacy GPIO pin designator for P8.32

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO11
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO110 Field

Legacy GPIO pin designator for P9.31

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO110
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO111 Field

Legacy GPIO pin designator for P9.29

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO111
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO112 Field

Legacy GPIO pin designator for P9.30

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO112
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO113 Field

Legacy GPIO pin designator for P9.28

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO113
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO115 Field

Legacy GPIO pin designator for P9.27

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO115
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO117 Field

Legacy GPIO pin designator for P9.25

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO117
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO12 Field

Legacy GPIO pin designator for P9.20

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO12
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO13 Field

Legacy GPIO pin designator for P9.19

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO13
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO14 Field

Legacy GPIO pin designator for P9.26

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO14
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO15 Field

Legacy GPIO pin designator for P9.24

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO15
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO2 Field

Legacy GPIO pin designator for P9.22

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO2
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO20 Field

Legacy GPIO pin designator for P9.41

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO20
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO22 Field

Legacy GPIO pin designator for P8.19

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO22
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO23 Field

Legacy GPIO pin designator for P8.13

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO23
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO26 Field

Legacy GPIO pin designator for P8.14

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO26
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO27 Field

Legacy GPIO pin designator for P8.17

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO27
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO3 Field

Legacy GPIO pin designator for P9.21

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO3
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO30 Field

Legacy GPIO pin designator for P9.11

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO30
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO31 Field

Legacy GPIO pin designator for P9.13

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO31
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO32 Field

Legacy GPIO pin designator for P8.25

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO32
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the BeagleBone White and the [BB-NOEMMC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO33 Field

Legacy GPIO pin designator for P8.24

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO33
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the BeagleBone White and the [BB-NOEMMC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO34 Field

Legacy GPIO pin designator for P8.5

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO34
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the BeagleBone White and the [BB-NOEMMC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO35 Field

Legacy GPIO pin designator for P8.6

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO35
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the BeagleBone White and the [BB-NOEMMC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO36 Field

Legacy GPIO pin designator for P8.23

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO36
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the BeagleBone White and the [BB-NOEMMC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO37 Field

Legacy GPIO pin designator for P8.22

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO37
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the BeagleBone White and the [BB-NOEMMC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO38 Field

Legacy GPIO pin designator for P8.3

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO38
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the BeagleBone White and the [BB-NOEMMC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO39 Field

Legacy GPIO pin designator for P8.4

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO39
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the BeagleBone White and the [BB-NOEMMC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO4 Field

Legacy GPIO pin designator for P9.18

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO4
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO44 Field

Legacy GPIO pin designator for P8.12

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO44
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO45 Field

Legacy GPIO pin designator for P8.11

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO45
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO46 Field

Legacy GPIO pin designator for P8.16

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO46
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO47 Field

Legacy GPIO pin designator for P8.15

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO47
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO48 Field

Legacy GPIO pin designator for P9.15

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO48
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO49 Field

Legacy GPIO pin designator for P9.23

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO49
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO5 Field

Legacy GPIO pin designator for P9.17

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO5
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO50 Field

Legacy GPIO pin designator for P9.14

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO50
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO51 Field

Legacy GPIO pin designator for P9.16

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO51
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO60 Field

Legacy GPIO pin designator for P9.12

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO60
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO61 Field

Legacy GPIO pin designator for P8.26

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO61
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO62 Field

Legacy GPIO pin designator for P8.21

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO62
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the BeagleBone White and the [BB-NOEMMC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO63 Field

Legacy GPIO pin designator for P8.20

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO63
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the BeagleBone White and the [BB-NOEMMC](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO65 Field

Legacy GPIO pin designator for P8.18

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO65
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO66 Field

Legacy GPIO pin designator for P8.7

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO66
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO67 Field

Legacy GPIO pin designator for P8.8

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO67
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO68 Field

Legacy GPIO pin designator for P8.10

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO68
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO69 Field

Legacy GPIO pin designator for P8.9

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO69
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO7 Field

Legacy GPIO pin designator for P9.42

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO7
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO70 Field

Legacy GPIO pin designator for P8.45

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO70
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO71 Field

Legacy GPIO pin designator for P8.46

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO71
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO72 Field

Legacy GPIO pin designator for P8.43

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO72
```

### Field Value

Type: [Designator](#)

## ▀ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▀ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO73 Field

Legacy GPIO pin designator for P8.44

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO73
```

### Field Value

Type: [Designator](#)

## ▀ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▀ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO74 Field

Legacy GPIO pin designator for P8.41

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO74
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO75 Field

Legacy GPIO pin designator for P8.42

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO75
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO76 Field

Legacy GPIO pin designator for P8.39

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO76
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO77 Field

Legacy GPIO pin designator for P8.40

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO77
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO78 Field

Legacy GPIO pin designator for P8.37

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO78
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO79 Field

Legacy GPIO pin designator for P8.38

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO79
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO8 Field

Legacy GPIO pin designator for P8.35

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO8
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO80 Field

Legacy GPIO pin designator for P8.36

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO80
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO81 Field

Legacy GPIO pin designator for P8.34

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO81
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO86 Field

Legacy GPIO pin designator for P8.27

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO86
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO87 Field

Legacy GPIO pin designator for P8.29

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO87
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO88 Field

Legacy GPIO pin designator for P8.28

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO88
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO89 Field

Legacy GPIO pin designator for P8.30

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO89
```

### Field Value

Type: [Designator](#)

## ▀ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▀ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneGPIO9 Field

Legacy GPIO pin designator for P8.33

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO9
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [BB-GPIO](#) device tree overlay.

## ▪ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneI2C2 Field

I2C bus designator for P9.19 and P9.20

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator I2C2
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneSPI2\_0 Field

SPI slave select designator for P9.28

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator SPI2_0
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# BeagleBoneSPI2\_1 Field

SPI slave select designator for P9.42

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static readonly Designator SPI2_1
```

## Field Value

Type: [Designator](#)

## ↳ See Also

### Reference

[BeagleBone Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagle Class

This class defines identifiers for the devices provided by the PocketBeagle hardware platform.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.Platforms](#)[PocketBeagle](#)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static class PocketBeagle
```

The [PocketBeagle](#) type exposes the following members.

## ▪ Fields

	Name	Description
• <a href="#"><b>S</b></a>	AIN0	ADC input designator for P1.19 (1.8V)
• <a href="#"><b>S</b></a>	AIN1	ADC input designator for P1.21 (1.8V)
• <a href="#"><b>S</b></a>	AIN2	ADC input designator for P1.23 (1.8V)
• <a href="#"><b>S</b></a>	AIN3	ADC input designator for P1.25 (1.8V)
• <a href="#"><b>S</b></a>	AIN4	ADC input designator for P1.27 (1.8V)

♦ S	AIN5	ADC input designator for P2.35 (3.6V)
♦ S	AIN6	ADC input designator for P1.2 (3.6V)
♦ S	AIN7	ADC input designator for P2.36 (1.8V)
♦ S	GPIO110	Legacy GPIO pin designator for P1.36
♦ S	GPIO111	Legacy GPIO pin designator for P1.33
♦ S	GPIO112	Legacy GPIO pin designator for P2.32
♦ S	GPIO113	Legacy GPIO pin designator for P2.30
♦ S	GPIO114	Legacy GPIO pin designator for P1.31
♦ S	GPIO115	Legacy GPIO pin designator for P2.34
♦ S	GPIO116	Legacy GPIO pin designator for P2.28
♦ S	GPIO117	Legacy GPIO pin designator for P1.29
♦ S	GPIO12	Legacy GPIO pin designator for P1.26
♦ S	GPIO13	Legacy GPIO pin designator for P1.28
♦ S	GPIO14	Legacy GPIO pin designator for P2.11
♦ S	GPIO15	Legacy GPIO pin designator for P2.9
♦ S	GPIO19	Legacy GPIO pin designator for P2.31
♦ S	GPIO2	Legacy GPIO pin designator for P1.8
♦ S	GPIO20	Legacy GPIO pin designator for P1.20
♦ S	GPIO23	Legacy GPIO pin designator for P2.3

• S	GPIO26	Legacy GPIO pin designator for P1.34
• S	GPIO27	Legacy GPIO pin designator for P2.19
• S	GPIO3	Legacy GPIO pin designator for P1.10
• S	GPIO30	Legacy GPIO pin designator for P2.5
• S	GPIO31	Legacy GPIO pin designator for P2.7
• S	GPIO4	Legacy GPIO pin designator for P1.12
• S	GPIO40	Legacy GPIO pin designator for P2.27
• S	GPIO41	Legacy GPIO pin designator for P2.25
• S	GPIO42	Legacy GPIO pin designator for P1.32
• S	GPIO43	Legacy GPIO pin designator for P1.30
• S	GPIO44	Legacy GPIO pin designator for P2.24
• S	GPIO45	Legacy GPIO pin designator for P2.33
• S	GPIO46	Legacy GPIO pin designator for P2.22
• S	GPIO47	Legacy GPIO pin designator for P2.18
• S	GPIO5	Legacy GPIO pin designator for P1.6
• S	GPIO50	Legacy GPIO pin designator for P2.1
• S	GPIO52	Legacy GPIO pin designator for P2.10
• S	GPIO57	Legacy GPIO pin designator for P2.6
• S	GPIO58	Legacy GPIO pin designator for P2.4

• <b>S</b>	<a href="#">GPIO59</a>	Legacy GPIO pin designator for P2.2
• <b>S</b>	<a href="#">GPIO60</a>	Legacy GPIO pin designator for P2.8
• <b>S</b>	<a href="#">GPIO64</a>	Legacy GPIO pin designator for P2.20
• <b>S</b>	<a href="#">GPIO65</a>	Legacy GPIO pin designator for P2.17
• <b>S</b>	<a href="#">GPIO7</a>	Legacy GPIO pin designator for P2.29
• <b>S</b>	<a href="#">GPIO86</a>	Legacy GPIO pin designator for P2.35
• <b>S</b>	<a href="#">GPIO87</a>	Legacy GPIO pin designator for P1.2
• <b>S</b>	<a href="#">GPIO88</a>	Legacy GPIO pin designator for P1.35
• <b>S</b>	<a href="#">GPIO89</a>	Legacy GPIO pin designator for P1.4
• <b>S</b>	<a href="#">I2C1</a>	I2C bus designator for P1.26 and P1.28
• <b>S</b>	<a href="#">I2C2</a>	I2C bus designator for P2.9 and P2.11
• <b>S</b>	<a href="#">PWM0_0</a>	PWM output designator for P1.36
• <b>S</b>	<a href="#">PWM2_0</a>	PWM output designator for P2.1
• <b>S</b>	<a href="#">SPI0_0</a>	SPI slave select designator for P1.6
• <b>S</b>	<a href="#">SPI1_1</a>	SPI slave select designator for P2.31

[Top](#)

## See Also

### Reference

[IO.Objects.libsimpleio.Platforms Namespace](#)



# PocketBeagle Fields

The [PocketBeagle](#) type exposes the following members.

## Fields

	Name	Description
• <b>S</b>	AIN0	ADC input designator for P1.19 (1.8V)
• <b>S</b>	AIN1	ADC input designator for P1.21 (1.8V)
• <b>S</b>	AIN2	ADC input designator for P1.23 (1.8V)
• <b>S</b>	AIN3	ADC input designator for P1.25 (1.8V)
• <b>S</b>	AIN4	ADC input designator for P1.27 (1.8V)
• <b>S</b>	AIN5	ADC input designator for P2.35 (3.6V)
• <b>S</b>	AIN6	ADC input designator for P1.2 (3.6V)
• <b>S</b>	AIN7	ADC input designator for P2.36 (1.8V)
• <b>S</b>	GPIO110	Legacy GPIO pin designator for P1.36
• <b>S</b>	GPIO111	Legacy GPIO pin designator for P1.33
• <b>S</b>	GPIO112	Legacy GPIO pin designator for P2.32
• <b>S</b>	GPIO113	Legacy GPIO pin designator for P2.30
• <b>S</b>	GPIO114	Legacy GPIO pin designator for P1.31

♦ S	GPIO115	Legacy GPIO pin designator for P2.34
♦ S	GPIO116	Legacy GPIO pin designator for P2.28
♦ S	GPIO117	Legacy GPIO pin designator for P1.29
♦ S	GPIO12	Legacy GPIO pin designator for P1.26
♦ S	GPIO13	Legacy GPIO pin designator for P1.28
♦ S	GPIO14	Legacy GPIO pin designator for P2.11
♦ S	GPIO15	Legacy GPIO pin designator for P2.9
♦ S	GPIO19	Legacy GPIO pin designator for P2.31
♦ S	GPIO2	Legacy GPIO pin designator for P1.8
♦ S	GPIO20	Legacy GPIO pin designator for P1.20
♦ S	GPIO23	Legacy GPIO pin designator for P2.3
♦ S	GPIO26	Legacy GPIO pin designator for P1.34
♦ S	GPIO27	Legacy GPIO pin designator for P2.19
♦ S	GPIO3	Legacy GPIO pin designator for P1.10
♦ S	GPIO30	Legacy GPIO pin designator for P2.5
♦ S	GPIO31	Legacy GPIO pin designator for P2.7
♦ S	GPIO4	Legacy GPIO pin designator for P1.12
♦ S	GPIO40	Legacy GPIO pin designator for P2.27
♦ S	GPIO41	Legacy GPIO pin designator for P2.25

♦ S	GPIO42	Legacy GPIO pin designator for P1.32
♦ S	GPIO43	Legacy GPIO pin designator for P1.30
♦ S	GPIO44	Legacy GPIO pin designator for P2.24
♦ S	GPIO45	Legacy GPIO pin designator for P2.33
♦ S	GPIO46	Legacy GPIO pin designator for P2.22
♦ S	GPIO47	Legacy GPIO pin designator for P2.18
♦ S	GPIO5	Legacy GPIO pin designator for P1.6
♦ S	GPIO50	Legacy GPIO pin designator for P2.1
♦ S	GPIO52	Legacy GPIO pin designator for P2.10
♦ S	GPIO57	Legacy GPIO pin designator for P2.6
♦ S	GPIO58	Legacy GPIO pin designator for P2.4
♦ S	GPIO59	Legacy GPIO pin designator for P2.2
♦ S	GPIO60	Legacy GPIO pin designator for P2.8
♦ S	GPIO64	Legacy GPIO pin designator for P2.20
♦ S	GPIO65	Legacy GPIO pin designator for P2.17
♦ S	GPIO7	Legacy GPIO pin designator for P2.29
♦ S	GPIO86	Legacy GPIO pin designator for P2.35
♦ S	GPIO87	Legacy GPIO pin designator for P1.2
♦ S	GPIO88	Legacy GPIO pin designator for P1.35

---

• <b>S</b>	<a href="#">GPIO89</a>	Legacy GPIO pin designator for P1.4
• <b>S</b>	<a href="#">I2C1</a>	I2C bus designator for P1.26 and P1.28
• <b>S</b>	<a href="#">I2C2</a>	I2C bus designator for P2.9 and P2.11
• <b>S</b>	<a href="#">PWM0_0</a>	PWM output designator for P1.36
• <b>S</b>	<a href="#">PWM2_0</a>	PWM output designator for P2.1
• <b>S</b>	<a href="#">SPI0_0</a>	SPI slave select designator for P1.6
• <b>S</b>	<a href="#">SPI1_1</a>	SPI slave select designator for P2.31

---

[Top](#)

## ↳ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleAIN0 Field

ADC input designator for P1.19 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN0
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [PB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleAIN1 Field

ADC input designator for P1.21 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN1
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [PB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleAIN2 Field

ADC input designator for P1.23 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN2
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [PB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleAIN3 Field

ADC input designator for P1.25 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN3
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [PB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleAIN4 Field

ADC input designator for P1.27 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN4
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [PB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleAIN5 Field

ADC input designator for P2.35 (3.6V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN5
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [PB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleAIN6 Field

ADC input designator for P1.2 (3.6V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN6
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [PB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleAIN7 Field

ADC input designator for P2.36 (1.8V)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN7
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [PB-ADC](#) device tree overlay.

## ▪ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO110 Field

Legacy GPIO pin designator for P1.36

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO110
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO111 Field

Legacy GPIO pin designator for P1.33

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO111
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO112 Field

Legacy GPIO pin designator for P2.32

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO112
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO113 Field

Legacy GPIO pin designator for P2.30

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO113
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO114 Field

Legacy GPIO pin designator for P1.31

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO114
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO115 Field

Legacy GPIO pin designator for P2.34

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO115
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO116 Field

Legacy GPIO pin designator for P2.28

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO116
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO117 Field

Legacy GPIO pin designator for P1.29

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO117
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO12 Field

Legacy GPIO pin designator for P1.26

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO12
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO13 Field

Legacy GPIO pin designator for P1.28

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO13
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO14 Field

Legacy GPIO pin designator for P2.11

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO14
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO15 Field

Legacy GPIO pin designator for P2.9

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO15
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO19 Field

Legacy GPIO pin designator for P2.31

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO19
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO2 Field

Legacy GPIO pin designator for P1.8

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO2
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO20 Field

Legacy GPIO pin designator for P1.20

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO20
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO23 Field

Legacy GPIO pin designator for P2.3

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO23
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO26 Field

Legacy GPIO pin designator for P1.34

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO26
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO27 Field

Legacy GPIO pin designator for P2.19

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO27
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO3 Field

Legacy GPIO pin designator for P1.10

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO3
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO30 Field

Legacy GPIO pin designator for P2.5

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO30
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO31 Field

Legacy GPIO pin designator for P2.7

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO31
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO4 Field

Legacy GPIO pin designator for P1.12

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO4
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO40 Field

Legacy GPIO pin designator for P2.27

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO40
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO41 Field

Legacy GPIO pin designator for P2.25

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO41
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO42 Field

Legacy GPIO pin designator for P1.32

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO42
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO43 Field

Legacy GPIO pin designator for P1.30

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO43
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO44 Field

Legacy GPIO pin designator for P2.24

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO44
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO45 Field

Legacy GPIO pin designator for P2.33

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO45
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO46 Field

Legacy GPIO pin designator for P2.22

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO46
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO47 Field

Legacy GPIO pin designator for P2.18

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO47
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO5 Field

Legacy GPIO pin designator for P1.6

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO5
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO50 Field

Legacy GPIO pin designator for P2.1

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO50
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO52 Field

Legacy GPIO pin designator for P2.10

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO52
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO57 Field

Legacy GPIO pin designator for P2.6

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO57
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO58 Field

Legacy GPIO pin designator for P2.4

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO58
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO59 Field

Legacy GPIO pin designator for P2.2

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO59
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO60 Field

Legacy GPIO pin designator for P2.8

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO60
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO64 Field

Legacy GPIO pin designator for P2.20

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO64
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO65 Field

Legacy GPIO pin designator for P2.17

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO65
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO7 Field

Legacy GPIO pin designator for P2.29

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO7
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO86 Field

Legacy GPIO pin designator for P2.35

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO86
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO87 Field

Legacy GPIO pin designator for P1.2

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO87
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO88 Field

Legacy GPIO pin designator for P1.35

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO88
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleGPIO89 Field

Legacy GPIO pin designator for P1.4

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO89
```

### Field Value

Type: [Designator](#)

## ◀ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleI2C1 Field

I2C bus designator for P1.26 and P1.28

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator I2C1
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleI2C2 Field

I2C bus designator for P2.9 and P2.11

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator I2C2
```

### Field Value

Type: [Designator](#)

## ↳ See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeaglePWM0\_0 Field

PWM output designator for P1.36

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator PWM0_0
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeaglePWM2\_0 Field

PWM output designator for P2.1

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator PWM2_0
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# PocketBeagleSPI0\_0 Field

SPI slave select designator for P1.6

**Namespace:** [IO.Objects.libsimpleioPlatforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator SPI0_0
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleioPlatforms Namespace](#)

# PocketBeagleSPI1\_1 Field

SPI slave select designator for P2.31

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator SPI1_1
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[PocketBeagle Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPi Class

This class defines identifiers for the devices provided by the Raspberry Pi hardware platform.

## Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.Platforms](#)[RaspberryPi](#)

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public static class RaspberryPi
```

The [RaspberryPi](#) type exposes the following members.

## Fields

	Name	Description
• <a href="#"><b>s</b></a>	<a href="#">AIN0</a>	Analog input designator
• <a href="#"><b>s</b></a>	<a href="#">AIN1</a>	Analog input designator
• <a href="#"><b>s</b></a>	<a href="#">GPIO10</a>	Legacy GPIO pin designator
• <a href="#"><b>s</b></a>	<a href="#">GPIO11</a>	Legacy GPIO pin designator
• <a href="#"><b>s</b></a>	<a href="#">GPIO12</a>	Legacy GPIO pin designator

• S	GPIO13	Legacy GPIO pin designator
• S	GPIO14	Legacy GPIO pin designator
• S	GPIO15	Legacy GPIO pin designator
• S	GPIO16	Legacy GPIO pin designator
• S	GPIO17	Legacy GPIO pin designator
• S	GPIO18	Legacy GPIO pin designator
• S	GPIO19	Legacy GPIO pin designator
• S	GPIO2	Legacy GPIO pin designator
• S	GPIO20	Legacy GPIO pin designator
• S	GPIO21	Legacy GPIO pin designator
• S	GPIO22	Legacy GPIO pin designator
• S	GPIO23	Legacy GPIO pin designator
• S	GPIO24	Legacy GPIO pin designator
• S	GPIO25	Legacy GPIO pin designator
• S	GPIO26	Legacy GPIO pin designator
• S	GPIO27	Legacy GPIO pin designator
• S	GPIO3	Legacy GPIO pin designator
• S	GPIO4	Legacy GPIO pin designator
• S	GPIO5	Legacy GPIO pin designator

• <b>S</b>	<a href="#">GPIO6</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO7</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO8</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO9</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">I2C1</a>	I2C bus designator for GPIO2 and GPIO3
• <b>S</b>	<a href="#">PWM0_0</a>	PWM output designator for GPIO18
• <b>S</b>	<a href="#">PWM0_1</a>	PWM output designator for GPIO19
• <b>S</b>	<a href="#">SPI0_0</a>	SPI slave select designator for GPIO8
• <b>S</b>	<a href="#">SPI0_1</a>	SPI slave select designator for GPIO7

[Top](#)

## See Also

### Reference

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPi Fields

The [RaspberryPi](#) type exposes the following members.

## Fields

	Name	Description
◆ S	AIN0	Analog input designator
◆ S	AIN1	Analog input designator
◆ S	GPIO10	Legacy GPIO pin designator
◆ S	GPIO11	Legacy GPIO pin designator
◆ S	GPIO12	Legacy GPIO pin designator
◆ S	GPIO13	Legacy GPIO pin designator
◆ S	GPIO14	Legacy GPIO pin designator
◆ S	GPIO15	Legacy GPIO pin designator
◆ S	GPIO16	Legacy GPIO pin designator
◆ S	GPIO17	Legacy GPIO pin designator
◆ S	GPIO18	Legacy GPIO pin designator
◆ S	GPIO19	Legacy GPIO pin designator
◆ S	GPIO2	Legacy GPIO pin designator

• <b>S</b>	<a href="#">GPIO20</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO21</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO22</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO23</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO24</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO25</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO26</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO27</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO3</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO4</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO5</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO6</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO7</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO8</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">GPIO9</a>	Legacy GPIO pin designator
• <b>S</b>	<a href="#">I2C1</a>	I2C bus designator for GPIO2 and GPIO3
• <b>S</b>	<a href="#">PWM0_0</a>	PWM output designator for GPIO18
• <b>S</b>	<a href="#">PWM0_1</a>	PWM output designator for GPIO19
• <b>S</b>	<a href="#">SPI0_0</a>	SPI slave select designator for GPIO8



[SPI0\\_1](#)

SPI slave select designator for GPIO7

---

[Top](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

---

# RaspberryPiAIN0 Field

Analog input designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN0
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the Mikroelektronika Pi 3 Click Shield and the [Pi3ClickShield](#) device tree overlay.

## ▪ See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiAIN1 Field

Analog input designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator AIN1
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the Mikroelektronika Pi 3 Click Shield and the [Pi3ClickShield](#) device tree overlay.

## ▪ See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO10 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO10
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO11 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO11
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO12 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO12
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO13 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO13
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO14 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO14
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO15 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO15
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO16 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleioPlatforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO16
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleioPlatforms Namespace](#)

# RaspberryPiGPIO17 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO17
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO18 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO18
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO19 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO19
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO2 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO2
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO20 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO20
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO21 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO21
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO22 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO22
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO23 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO23
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO24 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO24
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO25 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO25
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO26 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO26
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO27 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO27
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO3 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO3
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO4 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO4
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO5 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO5
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO6 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO6
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO7 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO7
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO8 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO8
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiGPIO9 Field

Legacy GPIO pin designator

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator GPIO9
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPi2C1 Field

I2C bus designator for GPIO2 and GPIO3

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator I2C1
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiPWM0\_0 Field

PWM output designator for GPIO18

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator PWM0_0
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [pwm](#) device tree overlay.

## ▪ See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiPWM0\_1 Field

PWM output designator for GPIO19

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator PWM0_1
```

### Field Value

Type: [Designator](#)

## ▪ Remarks

Requires the [pwm](#) device tree overlay.

## ▪ See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiSPI0\_0 Field

SPI slave select designator for GPIO8

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator SPI0_0
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# RaspberryPiSPI0\_1 Field

SPI slave select designator for GPIO7

**Namespace:** [IO.Objects.libsimpleio.Platforms](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public static readonly Designator SPI0_1
```

### Field Value

Type: [Designator](#)

## « See Also

### Reference

[RaspberryPi Class](#)

[IO.Objects.libsimpleio.Platforms Namespace](#)

# IO.Objects.libsimpleio.PWM Namespace

PWM (Pulse Width Modulated) Output Services

## ↳ Classes

Class	Description
 <a href="#">Output</a>	Encapsulates Linux PWM outputs using <a href="#">libsimpleio</a> .

# Output Class

Encapsulates Linux PWM outputs using [libsimpleio](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.PWMOutput](#)

**Namespace:** [IO.Objects.libsimpleio.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Output : Output
```

The [Output](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Output</a>	Constructor for a single PWM output.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">dutycycle</a>	Write-only property for setting the PWM output duty cycle. Allowed values

are 0.0 to 100.0 percent.

---



**fd**

Read-only property returning the Linux file descriptor for the PWM output.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Objects.libsimpleio.PWM Namespace](#)

# Output Constructor

Constructor for a single PWM output.

**Namespace:** [IO.Objects.libsimpleio.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Output(  
    Designator desg,  
    int frequency,  
    double dutycycle = 0,  
    int polarity = 1  
)
```

## Parameters

*desg*

Type: [IO.Objects.libsimpleio.DeviceDesignator](#)

PWM output designator.

*frequency*

Type: [SystemInt32](#)

PWM pulse frequency.

*dutycycle* (Optional)

Type: [SystemDouble](#)

Initial PWM output duty cycle.

*polarity* (Optional)

Type: [SystemInt32](#)

PWM output polarity.

## ↳ See Also

[Reference](#)

[Output Class](#)

[IO.Objects.libsimpleio.PWM Namespace](#)

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">dutycycle</a>	Write-only property for setting the PWM output duty cycle. Allowed values are 0.0 to 100.0 percent.
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the PWM output.

[Top](#)

## See Also

**Reference**

[Output Class](#)

[IO.Objects.libsimpleio.PWM Namespace](#)

# Outputdutycycle Property

Write-only property for setting the PWM output duty cycle. Allowed values are 0.0 to 100.0 percent.

**Namespace:** [IO.Objects.libsimpleio.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public double dutycycle { set; }
```

### Property Value

Type: [Double](#)

### Implements

[Outputdutycycle](#)

## ↳ See Also

### Reference

[Output Class](#)

[IO.Objects.libsimpleio.PWM Namespace](#)

# Outputfd Property

Read-only property returning the Linux file descriptor for the PWM output.

**Namespace:** [IO.Objects.libsimpleio.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int fd { get; }
```

## Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Output Class](#)

[IO.Objects.libsimpleio.PWM Namespace](#)

# IO.Objects.libsimpleio.Servo Namespace

Servo Output Services

## ↳ Classes

Class	Description
 <a href="#">Output</a>	Encapsulates Linux servo outputs using <a href="#">libsimpleio</a> .

# Output Class

Encapsulates Linux servo outputs using [libsimpleio](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.ServoOutput](#)

**Namespace:** [IO.Objects.libsimpleio.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Output : Output
```

The [Output](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Output</a>	Constructor for a single servo output.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the servo output.



**position** Write-only property for setting the servo position. Allowed values are -1.0 to +1.0.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Objects.libsimpleio.Servo Namespace](#)

# Output Constructor

Constructor for a single servo output.

**Namespace:** [IO.Objects.libsimpleio.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Output(  
    Designator desg,  
    int frequency = 50,  
    double position = 0  
)
```

## Parameters

*desg*

Type: [IO.Objects.libsimpleio.DeviceDesignator](#)

PWM output designator.

*frequency* (Optional)

Type: [SystemInt32](#)

PWM pulse frequency.

*position* (Optional)

Type: [SystemDouble](#)

Initial servo position.

## ◀ See Also

[Reference](#)

[Output Class](#)

## IO.Objects.libsimpleio.Servo Namespace

---

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the servo output.
	<a href="#">position</a>	Write-only property for setting the servo position. Allowed values are -1.0 to +1.0.

[Top](#)

## See Also

**Reference**

[Output Class](#)

[IO.Objects.libsimpleio.Servo Namespace](#)

# Outputfd Property

Read-only property returning the Linux file descriptor for the servo output.

**Namespace:** [IO.Objects.libsimpleio.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▲ Syntax

C#    VB    F#

[Copy](#)

```
public int fd { get; }
```

## Property Value

Type: [Int32](#)

## ▲ See Also

### Reference

[Output Class](#)

[IO.Objects.libsimpleio.Servo Namespace](#)

# Outputposition Property

Write-only property for setting the servo position. Allowed values are -1.0 to +1.0.

**Namespace:** [IO.Objects.libsimpleio.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▲ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public double position { set; }
```

### Property Value

Type: [Double](#)

### Implements

[Outputposition](#)

## ▲ See Also

### [Reference](#)

[Output Class](#)

[IO.Objects.libsimpleio.Servo Namespace](#)

# IO.Objects.libsimpleio.SPI Namespace

SPI (Serial Peripheral Interconnect) Device Services

## Classes

Class	Description
 Device	Encapsulates Linux SPI devices using <a href="#">libsimpleio</a> .

# Device Class

Encapsulates Linux SPI devices using [libsimpleio](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.SPIDevice](#)

**Namespace:** [IO.Objects.libsimpleio.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Device : Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Device(String, Int32, Int32, Int32, Pin)</a>	Constructor for a single SPI device.
≡	<a href="#">Device(Designator, Int32, Int32, Int32, Pin)</a>	Constructor for a single SPI device.

[Top](#)

## ▪ Properties

	<b>Name</b>	<b>Description</b>
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the SPI slave device.

[Top](#)

## ◀ Methods

	<b>Name</b>	<b>Description</b>
	<a href="#">Read</a>	Read bytes from an SPI slave device.
	<a href="#">Transaction</a>	Write and read bytes to and from an SPI slave device.
	<a href="#">Write</a>	Write bytes to an SPI slave device.

[Top](#)

## ◀ Fields

	<b>Name</b>	<b>Description</b>
 	<a href="#">AUTOCHIPSELECT</a>	Use hardware slave select.

[Top](#)

## ◀ See Also

### Reference

[IO.Objects.libsimpleio.SPI Namespace](#)

# Device Constructor

## ↳ Overload List

	Name	Description
≡	<a href="#">Device(String, Int32, Int32, Int32, Pin)</a>	Constructor for a single SPI device.
≡	<a href="#">Device(Designator, Int32, Int32, Int32, Pin)</a>	Constructor for a single SPI device.

[Top](#)

## ↳ See Also

[Reference](#)

[Device Class](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

# Device Constructor (String, Int32, Int32, Int32, Pin)

Constructor for a single SPI device.

**Namespace:** [IO.Objects.libsimpleio.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public Device(  
    string devname,  
    int mode,  
    int wordsize,  
    int speed,  
    Pin cspin = null  
)
```

## Parameters

*devname*

Type: [SystemString](#)

SPI device node name.

*mode*

Type: [SystemInt32](#)

SPI clock mode.

*wordsize*

Type: [SystemInt32](#)

SPI transfer word size.

*speed*

Type: [SystemInt32](#)

SPI transfer speed.

*cspin* (Optional)

Type: [IO.Objects.libsimpleio.GPIOPin](#)

SPI slave select GPIO pin number, or [AUTOCHIPSELECT](#).

## See Also

**Reference**

[Device Class](#)

[Device Overload](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

# Device Constructor (Designator, Int32, Int32, Int32, Pin)

Constructor for a single SPI device.

**Namespace:** [IO.Objects.libsimpleio.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

Copy

```
public Device(  
    Designator desg,  
    int mode,  
    int wordsize,  
    int speed,  
    Pin cspin = null  
)
```

## Parameters

*desg*

Type: [IO.Objects.libsimpleio.DeviceDesignator](#)

SPI device designator.

*mode*

Type: [SystemInt32](#)

SPI clock mode.

*wordsize*

Type: [SystemInt32](#)

SPI transfer word size.

*speed*

Type: [SystemInt32](#)

SPI transfer speed.

*cspin* (Optional)

Type: [IO.Objects.libsimpleio.GPIOPin](#)

SPI slave select GPIO pin number, or [AUTOCHIPSELECT](#).

## See Also

**Reference**

[Device Class](#)

[Device Overload](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	fd	Read-only property returning the Linux file descriptor for the SPI slave device.

[Top](#)

## See Also

[Reference](#)

[Device Class](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

# Devicefd Property

Read-only property returning the Linux file descriptor for the SPI slave device.

**Namespace:** [IO.Objects.libsimpleio.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public int fd { get; }
```

## Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Read</a>	Read bytes from an SPI slave device.
	<a href="#">Transaction</a>	Write and read bytes to and from an SPI slave device.
	<a href="#">Write</a>	Write bytes to an SPI slave device.

[Top](#)

## See Also

### Reference

[Device Class](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

# DeviceRead Method

Read bytes from an SPI slave device.

**Namespace:** [IO.Objects.libsimpleio.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Read(  
    byte[] resp,  
    int resplen  
)
```

## Parameters

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

## Implements

[DeviceRead\(Byte, Int32\)](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

---

# DeviceTransaction Method

Write and read bytes to and from an SPI slave device.

**Namespace:** [IO.Objects.libsimpleio.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Transaction(  
    byte[] cmd,  
    int cmdLen,  
    byte[] resp,  
    int resplen,  
    int delayus = 0  
)
```

## Parameters

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

*delayus* (Optional)

Type: [SystemInt32](#)

Delay in microseconds between write and read operations.

## Implements

[DeviceTransaction\(Byte, Int32, Byte, Int32, Int32\)](#)

## See Also

### Reference

[Device Class](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

# DeviceWrite Method

Write bytes to an SPI slave device.

**Namespace:** [IO.Objects.libsimpleio.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Write(  
    byte[] cmd,  
    int cmdLen  
)
```

## Parameters

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

## Implements

[DeviceWrite\(Byte, Int32\)](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

---

# Device Fields

The [Device](#) type exposes the following members.

## Fields

Name	Description
 <a href="#">AUTOCHIPSELECT</a>	Use hardware slave select.

[Top](#)

## See Also

### Reference

[Device Class](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

# DeviceAUTOCHIPSELECT Field

Use hardware slave select.

**Namespace:** [IO.Objects.libsimpleio.SPI](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const Pin AUTOCHIPSELECT = null
```

### Field Value

Type: [Pin](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Objects.libsimpleio.SPI Namespace](#)

# IO.Objects.libsimpleio.Watchdog Namespace

Watchdog Timer Services

## Classes

Class	Description
 Timer	Encapsulates Linux watchdog timers using <a href="#">libsimpleio</a> .

# Timer Class

Encapsulates Linux watchdog timers using [libsimpleio](#).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.libsimpleio.WatchdogTimer](#)

**Namespace:** [IO.Objects.libsimpleio.Watchdog](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class Timer : Timer
```

The [Timer](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Timer</a>	Constructor for a single watchdog timer.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the watchdog timer.



`timeout` Get or set the watchdog timeout. Not all platforms may support this. Even if supported, there may be constraints. For example, some platforms allow shortening the timeout but not lengthening it.

---

[Top](#)

## ◀ Methods

	Name	Description
•	<a href="#">Kick</a>	Reset the watchdog timer.

[Top](#)

## ◀ Fields

	Name	Description
• S	<a href="#">DefaultDevice</a>	Default watchdog timer device name.
• S	<a href="#">DefaultTimeout</a>	Default watchdog timer timeout value (disabled).

[Top](#)

## ◀ See Also

### Reference

[IO.Objects.libsimpleio.Watchdog Namespace](#)

# Timer Constructor

Constructor for a single watchdog timer.

**Namespace:** [IO.Objects.libsimpleio.Watchdog](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Timer(  
    string devname = "/dev/watchdog",  
    int timeout = 0  
)
```

## Parameters

*devname* (Optional)

Type: [SystemString](#)

Device node name.

*timeout* (Optional)

Type: [SystemInt32](#)

Watchdog timeout setting in seconds, or [DefaultTimeout](#).

## ◀ See Also

[Reference](#)

[Timer Class](#)

[IO.Objects.libsimpleio.Watchdog Namespace](#)

# Timer Properties

The [Timer](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">fd</a>	Read-only property returning the Linux file descriptor for the watchdog timer.
	<a href="#">timeout</a>	Get or set the watchdog timeout. Not all platforms may support this. Even if supported, there may be constraints. For example, some platforms allow shortening the timeout but not lengthening it.

[Top](#)

## See Also

### Reference

[Timer Class](#)

[IO.Objects.libsimpleio.Watchdog Namespace](#)

# Timerfd Property

Read-only property returning the Linux file descriptor for the watchdog timer.

**Namespace:** [IO.Objects.libsimpleio.Watchdog](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int fd { get; }
```

## Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Timer Class](#)

[IO.Objects.libsimpleio.Watchdog Namespace](#)

# Timertimeout Property

Get or set the watchdog timeout. Not all platforms may support this. Even if supported, there may be constraints. For example, some platforms allow shortening the timeout but not lengthening it.

**Namespace:** [IO.Objects.libsimpleio.Watchdog](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public int timeout { get; set; }
```

### Property Value

Type: [Int32](#)

### Implements

[Timertimeout](#)

## ↳ See Also

### Reference

[Timer Class](#)

[IO.Objects.libsimpleio.Watchdog Namespace](#)

# Timer Methods

The [Timer](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Kick</a>	Reset the watchdog timer.

[Top](#)

## See Also

### Reference

[Timer Class](#)

[IO.Objects.libsimpleio.Watchdog Namespace](#)

# TimerKick Method

Reset the watchdog timer.

**Namespace:** [IO.Objects.libsimpleio.Watchdog](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Kick()
```

## Implements

[TimerKick](#)

## ◀ See Also

### Reference

[Timer Class](#)

[IO.Objects.libsimpleio.Watchdog Namespace](#)

# Timer Fields

The [Timer](#) type exposes the following members.

## Fields

Name	Description
  <a href="#">DefaultDevice</a>	Default watchdog timer device name.
  <a href="#">DefaultTimeout</a>	Default watchdog timer timeout value (disabled).

[Top](#)

## See Also

### Reference

[Timer Class](#)

[IO.Objects.libsimpleio.Watchdog Namespace](#)

# TimerDefaultDevice Field

Default watchdog timer device name.

**Namespace:** [IO.Objects.libsimpleio.Watchdog](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const string DefaultDevice = "/dev/watchdog"
```

### Field Value

Type: [String](#)

## ◀ See Also

### Reference

[Timer Class](#)

[IO.Objects.libsimpleio.Watchdog Namespace](#)

# TimerDefaultTimeout Field

Default watchdog timer timeout value (disabled).

**Namespace:** [IO.Objects.libsimpleio.Watchdog](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int DefaultTimeout = 0
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Timer Class](#)

[IO.Objects.libsimpleio.Watchdog Namespace](#)

# IO.Objects.Message64.UDP Namespace

64-Byte Message Services over UDP

## ↳ Classes

Class	Description
 <a href="#">Messenger</a>	64-Byte Message Transport Client Services using UDP (User Datagram Protocol).

# Messenger Class

64-Byte Message Transport Client Services using UDP (User Datagram Protocol).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.Message64.UDPMessenger](#)

**Namespace:** [IO.Objects.Message64.UDP](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public class Messenger : Messenger
```

The [Messenger](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">Messenger</a>	Constructor for a 64-byte Messenger instance using UDP.

[Top](#)

## ▪ Methods

	Name	Description
≡		

	<a href="#">Receive</a>	Receive a 64-byte response message from a raw HID device.
	<a href="#">Send</a>	Send a 64-byte command message to a raw HID device.
	<a href="#">Transaction</a>	Send a 64-byte command message and receive a 64-byte response message.

[Top](#)

## ◀ See Also

**Reference**

[IO.Objects.Message64.UDP Namespace](#)

# Messenger Constructor

Constructor for a 64-byte Messenger instance using UDP.

**Namespace:** [IO.Objects.Message64.UDP](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Messenger(  
    string host,  
    int port,  
    int timeoutms = 1000  
)
```

## Parameters

*host*

Type: [SystemString](#)

UDP server domain name or IP address.

*port*

Type: [SystemInt32](#)

UDP server port number.

*timeoutms* (**Optional**)

Type: [SystemInt32](#)

Receive timeout in milliseconds. Zero indicates wait forever.

## ◀ See Also

[Reference](#)

[Messenger Class](#)

## IO.Objects.Message64.UDP Namespace

---

# Messenger Methods

The [Messenger](#) type exposes the following members.

## ▪ Methods

	Name	Description
	<a href="#">Receive</a>	Receive a 64-byte response message from a raw HID device.
	<a href="#">Send</a>	Send a 64-byte command message to a raw HID device.
	<a href="#">Transaction</a>	Send a 64-byte command message and receive a 64-byte response message.

[Top](#)

## ▪ See Also

### Reference

[Messenger Class](#)

[IO.Objects.Message64.UDP Namespace](#)

# MessengerReceive Method

Receive a 64-byte response message from a raw HID device.

**Namespace:** [IO.Objects.Message64.UDP](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Receive(  
    Message resp  
)
```

## Parameters

*resp*

Type: [IO.Interfaces.Message64Message](#)

64-byte response message.

## Implements

[MessengerReceive\(Message\)](#)

## ◀ See Also

### Reference

[Messenger Class](#)

[IO.Objects.Message64.UDP Namespace](#)

# MessengerSend Method

Send a 64-byte command message to a raw HID device.

**Namespace:** [IO.Objects.Message64.UDP](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Send(  
    Message cmd  
)
```

## Parameters

*cmd*

Type: [IO.Interfaces.Message64Message](#)

64-byte command message.

## Implements

[MessengerSend\(Message\)](#)

## ◀ See Also

### Reference

[Messenger Class](#)

[IO.Objects.Message64.UDP Namespace](#)

# MessengerTransaction Method

Send a 64-byte command message and receive a 64-byte response message.

**Namespace:** [IO.Objects.Message64.UDP](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Transaction(  
    Message cmd,  
    Message resp  
)
```

## Parameters

*cmd*

Type: [IO.Interfaces.Message64Message](#)

64-byte command message.

*resp*

Type: [IO.Interfaces.Message64Message](#)

64-byte response message.

## Implements

[MessengerTransaction\(Message, Message\)](#)

## ◀ See Also

### Reference

[Messenger Class](#)

## IO.Objects.Message64.UDP Namespace

---

# IO.Objects.Motor.PWM

## Namespace

PWM Controlled Motor Services

### ↳ Classes

Class	Description
 <a href="#">Output</a>	Encapsulates motors controlled by PWM and GPIO outputs.

# Output Class

Encapsulates motors controlled by PWM and GPIO outputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.Motor.PWMOutput](#)

**Namespace:** [IO.Objects.Motor.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Output : Output
```

The [Output](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Output(Output, Output, Double)</a>	Constructor for a single motor, using two PWM outputs for clockwise and counterclockwise rotation control.
	<a href="#">Output(Pin, Output, Double)</a>	Constructor for a single motor, using one GPIO pin for direction control, and one PWM output for speed control.

[Top](#)

## ◀ Properties

	Name	Description
	<a href="#">velocity</a>	Write-only property for setting the normalized motor velocity. Allowed values are -1.0 (full speed reverse) to +1.0 (full speed forward).

[Top](#)

## ◀ See Also

### Reference

[IO.Objects.Motor.PWM Namespace](#)

# Output Constructor

## ↳ Overload List

	Name	Description
	<a href="#">Output(Output, Output, Double)</a>	Constructor for a single motor, using two PWM outputs for clockwise and counterclockwise rotation control.
	<a href="#">Output(Pin, Output, Double)</a>	Constructor for a single motor, using one GPIO pin for direction control, and one PWM output for speed control.

[Top](#)

## ↳ See Also

**Reference**

[Output Class](#)

[IO.Objects.Motor.PWM Namespace](#)

# Output Constructor (Output, Output, Double)

Constructor for a single motor, using two PWM outputs for clockwise and counterclockwise rotation control.

**Namespace:** [IO.Objects.Motor.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public Output(  
    Output clockwise,  
    Output counterclockwise,  
    double velocity = 0  
)
```

## Parameters

*clockwise*

Type: [IO.Interfaces.PWMOutput](#)

PWM output instance (for clockwise rotation control).

*counterclockwise*

Type: [IO.Interfaces.PWMOutput](#)

PWM output instance (for counterclockwise rotation control).

*velocity (Optional)*

Type: [SystemDouble](#)

Initial motor velocity.

## ↳ See Also

### Reference

[Output Class](#)

[Output Overload](#)

[IO.Objects.Motor.PWM Namespace](#)

# Output Constructor (Pin, Output, Double)

Constructor for a single motor, using one GPIO pin for direction control, and one PWM output for speed control.

**Namespace:** [IO.Objects.Motor.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

Copy

```
public Output(  
    Pin direction,  
    Output speed,  
    double velocity = 0  
)
```

## Parameters

*direction*

Type: [IO.Interfaces.GPIOPin](#)

GPIO pin instance (for direction control).

*speed*

Type: [IO.Interfaces.PWMOutput](#)

PWM output instance (for speed control).

*velocity (Optional)*

Type: [SystemDouble](#)

Initial motor velocity.

## ↳ See Also

### Reference

[Output Class](#)

[Output Overload](#)

[IO.Objects.Motor.PWM Namespace](#)

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">velocity</a>	Write-only property for setting the normalized motor velocity. Allowed values are -1.0 (full speed reverse) to +1.0 (full speed forward).

[Top](#)

## See Also

[Reference](#)

[Output Class](#)

[IO.Objects.Motor.PWM Namespace](#)

# Outputvelocity Property

Write-only property for setting the normalized motor velocity. Allowed values are -1.0 (full speed reverse) to +1.0 (full speed forward).

**Namespace:** [IO.Objects.Motor.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public double velocity { set; }
```

### Property Value

Type: [Double](#)

### Implements

[Outputvelocity](#)

## ↳ See Also

### Reference

[Output Class](#)

[IO.Objects.Motor.PWM Namespace](#)

# IO.Objects.Motor.Servo Namespace

Servo Controlled Motor (e.g. continuous rotation servo) Services

## ↳ Classes

Class	Description
 <a href="#">Output</a>	Encapsulates motors controlled by servo outputs (e.g. continuous rotation servos).

# Output Class

Encapsulates motors controlled by servo outputs (e.g. continuous rotation servos).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.Motor.ServoOutput](#)

**Namespace:** [IO.Objects.Motor.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public class Output : Output
```

The [Output](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Output</a>	Constructor for a single motor output.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">velocity</a>	Write-only property for setting the

normalized motor velocity. Allowed values are -1.0 (full speed reverse) to +1.0 (full speed forward).

---

[Top](#)

## ◀ See Also

**Reference**

[IO.Objects.Motor.Servo Namespace](#)

# Output Constructor

Constructor for a single motor output.

**Namespace:** [IO.Objects.Motor.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Output(  
    Output servo,  
    double velocity = 0  
)
```

## Parameters

*servo*

Type: [IO.Interfaces.ServoOutput](#)

Servo output instance.

*velocity* (Optional)

Type: [SystemDouble](#)

Initial motor velocity.

## ◀ See Also

[Reference](#)

[Output Class](#)

[IO.Objects.Motor.Servo Namespace](#)

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">velocity</a>	Write-only property for setting the normalized motor velocity. Allowed values are -1.0 (full speed reverse) to +1.0 (full speed forward).

[Top](#)

## See Also

[Reference](#)

[Output Class](#)

[IO.Objects.Motor.Servo Namespace](#)

# Outputvelocity Property

Write-only property for setting the normalized motor velocity. Allowed values are -1.0 (full speed reverse) to +1.0 (full speed forward).

**Namespace:** [IO.Objects.Motor.Servo](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▲ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public double velocity { set; }
```

### Property Value

Type: [Double](#)

### Implements

[Outputvelocity](#)

## ▲ See Also

### [Reference](#)

[Output Class](#)

[IO.Objects.Motor.Servo Namespace](#)

# IO.Objects.Servo.PWM Namespace

PWM Controlled Servo Services

## ↳ Classes

Class	Description
 <a href="#">Output</a>	Encapsulates servo outputs using PWM outputs.

# Output Class

Encapsulates servo outputs using PWM outputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Objects.Servo.PWMOutput](#)

**Namespace:** [IO.Objects.Servo](#).PWM

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Output : Output
```

The [Output](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Output</a>	Constructor for a single servo output.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">position</a>	Write-only property for setting the servo position. Allowed values are -1.0 to +1.0.

[Top](#)

## ↳ See Also

### Reference

[IO.Objects.Servo.PWM Namespace](#)

# Output Constructor

Constructor for a single servo output.

**Namespace:** [IO.Objects.Servo.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Output(  
    Output pwm,  
    int freq = 50,  
    double position = 0  
)
```

## Parameters

*pwm*

Type: [IO.Interfaces.PWMOutput](#)

PWM output instance.

*freq* (Optional)

Type: [SystemInt32](#)

PWM pulse frequency.

*position* (Optional)

Type: [SystemDouble](#)

Initial servo position.

## ◀ See Also

[Reference](#)

[Output Class](#)

## IO.Objects.Servo.PWM Namespace

---

# Output Properties

The [Output](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">position</a>	Write-only property for setting the servo position. Allowed values are -1.0 to +1.0.

[Top](#)

## See Also

[Reference](#)

[Output Class](#)

[IO.Objects.Servo.PWM Namespace](#)

# Outputposition Property

Write-only property for setting the servo position. Allowed values are -1.0 to +1.0.

**Namespace:** [IO.Objects.Servo.PWM](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public double position { set; }
```

## Property Value

Type: [Double](#)

## Implements

[Outputposition](#)

## ↳ See Also

### [Reference](#)

[Output Class](#)

[IO.Objects.Servo.PWM Namespace](#)

# IO.Remote Namespace

Remote I/O Device Framework, for sending commands and receiving response to/from [Remote I/O Protocol](#) devices.

## Classes

	Class	Description
	<a href="#">ADC</a>	Encapsulates remote A/D inputs.
	<a href="#">DAC</a>	Encapsulates remote D/A outputs.
	<a href="#">Device</a>	Encapsulates a remote I/O device.
	<a href="#">GPIO</a>	Encapsulates remote GPIO pins.
	<a href="#">I2C</a>	Encapsulates remote I <sup>2</sup> C buses.
	<a href="#">PWM</a>	Encapsulates remote PWM outputs.
	<a href="#">SPI</a>	Encapsulates remote SPI slave devices.

## Enumerations

	Enumeration	Description
	<a href="#">MessageTypes</a>	Remote I/O protocol message types
	<a href="#">PeripheralTypes</a>	Types of remote peripherals



# ADC Class

Encapsulates remote A/D inputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.RemoteADC](#)

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class ADC : Sample
```

The [ADC](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">ADC</a>	Create a remote A/D input.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.



## sample

Read-only property returning an integer analog input sample.

[Top](#)

## ◀ See Also

### Reference

[IO.Remote Namespace](#)

# ADC Constructor

Create a remote A/D input.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public ADC(  
    Device dev,  
    int num  
)
```

## Parameters

*dev*

Type: [IO.RemoteDevice](#)

Remote I/O device object.

*num*

Type: [SystemInt32](#)

A/D input number: 0 to 127.

## « Remarks

Use [Device.ADC\\_Create\(\)](#) instead of this constructor.

## « See Also

**Reference**

[ADC Class](#)

## IO.Remote Namespace

---

# ADC Properties

The [ADC](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.
	<a href="#">sample</a>	Read-only property returning an integer analog input sample.

[Top](#)

## See Also

### Reference

[ADC Class](#)

[IO.Remote Namespace](#)

# ADCresolution Property

Read-only property returning the number of bits of resolution.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public int resolution { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Sampleresolution](#)

## « See Also

[Reference](#)

[ADC Class](#)

[IO.Remote Namespace](#)

# ADCsample Property

Read-only property returning an integer analog input sample.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public int sample { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Samplesample](#)

## « See Also

[Reference](#)

[ADC Class](#)

[IO.Remote Namespace](#)

# DAC Class

Encapsulates remote D/A outputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.RemoteDAC](#)

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public class DAC : Sample
```

The [DAC](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">DAC</a>	Create a remote D/A output.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.



## sample

Write-only property for writing an integer analog sample to a DAC output.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Remote Namespace](#)

# DAC Constructor

Create a remote D/A output.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public DAC(  
    Device dev,  
    int num,  
    int sample = 0  
)
```

## Parameters

*dev*

Type: [IO.RemoteDevice](#)

Remote I/O device object.

*num*

Type: [SystemInt32](#)

D/A output number: 0 to 127.

*sample (Optional)*

Type: [SystemInt32](#)

Initial DAC output sample.

## ◀ Remarks

Use [Device.DAC\\_Create\(\)](#) instead of this constructor.

## See Also

[Reference](#)

[DAC Class](#)

[IO.Remote Namespace](#)

# DAC Properties

The [DAC](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">resolution</a>	Read-only property returning the number of bits of resolution.
	<a href="#">sample</a>	Write-only property for writing an integer analog sample to a DAC output.

[Top](#)

## See Also

### Reference

[DAC Class](#)

[IO.Remote Namespace](#)

# DACresolution Property

Read-only property returning the number of bits of resolution.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public int resolution { get; }
```

### Property Value

Type: [Int32](#)

### Implements

[Sampleresolution](#)

## ▪ See Also

[Reference](#)

[DAC Class](#)

[IO.Remote Namespace](#)

# DACsample Property

Write-only property for writing an integer analog sample to a DAC output.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public int sample { set; }
```

### Property Value

Type: [Int32](#)

### Implements

[Samplesample](#)

## ◀ See Also

### Reference

[DAC Class](#)

[IO.Remote Namespace](#)

# Device Class

Encasulates a remote I/O device.

Encasulates a remote I/O device.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.RemoteDevice](#)

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public class Device
```

The [Device](#) type exposes the following members.

## ▪ Constructors

Name	Description
 <a href="#">Device</a>	Create a Remote I/O device object for a Munts Technologies USB raw HID (VID=0x16D0, PID=0x0AFA) device Remote I/O Server.
 <a href="#">Device(Messenger)</a>	Create a Remote I/O device object.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">Capabilities</a>	Capability string from the Remote I/O device.
	<a href="#">Version</a>	Version string from the Remote I/O device.

[Top](#)

## ▪ Methods

	Name	Description
	<a href="#">ADC_Available</a>	Query available A/D inputs.
	<a href="#">ADC_Create</a>	Create a remote A/D input.
	<a href="#">DAC_Available</a>	Query available D/A outputs.
	<a href="#">DAC_Create</a>	Create a remote D/A output.
	<a href="#">Dispatcher</a>	Command dispatcher.
	<a href="#">GPIO_Available</a>	Query available GPIO pins.
	<a href="#">GPIO_Create</a>	Create a remote GPIO pin object.
	<a href="#">I2C_Available</a>	Query available I <sup>2</sup> C buses.
	<a href="#">I2C_Create</a>	Create a remote I <sup>2</sup> C bus controller.

---

≡	<a href="#">PWM_Available</a>	Query available PWM outputs.
≡	<a href="#">PWM_Create</a>	Create a remote PWM output.
≡	<a href="#">SPI_Available</a>	Query available SPI slave devices.
≡	<a href="#">SPI_Create</a>	Create a remote SPI slave device.

---

[Top](#)

## ◀ Fields

	Name	Description
• <b>S</b>	<a href="#">MAX_CHANNELS</a>	Maximum number of channels each subsystem can support.
• <b>S</b>	<a href="#">Unavailable</a>	Designator for an unavailable channel.

---

[Top](#)

## ◀ See Also

### Reference

[IO.Remote Namespace](#)

# Device Constructor

## ↳ Overload List

	Name	Description
	<a href="#">Device</a>	Create a Remote I/O device object for a Munts Technologies USB raw HID (VID=0x16D0, PID=0x0AFA) device Remote I/O Server.
	<a href="#">Device(Messenger)</a>	Create a Remote I/O device object.

[Top](#)

## ↳ See Also

**Reference**

[Device Class](#)

[IO.Remote Namespace](#)

# Device Constructor

Create a Remote I/O device object for a Munts Technologies USB raw HID (VID=0x16D0, PID=0x0AFA) device Remote I/O Server.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)   [VB](#)   [F#](#)

[Copy](#)

```
public Device()
```

## ◀ See Also

### Reference

[Device Class](#)

[Device Overload](#)

[IO.Remote Namespace](#)

# Device Constructor (Messenger)

Create a Remote I/O device object.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device(  
    Messenger m  
)
```

## Parameters

*m*

Type: [IO.Interfaces.Message64Messenger](#)  
Message transport object

## ◀ See Also

### Reference

[Device Class](#)

[Device Overload](#)

[IO.Remote Namespace](#)

# Device Properties

The [Device](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">Capabilities</a>	Capability string from the Remote I/O device.
	<a href="#">Version</a>	Version string from the Remote I/O device.

[Top](#)

## See Also

**Reference**

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceCapabilities Property

Capability string from the Remote I/O device.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public string Capabilities { get; }
```

### Property Value

Type: [String](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceVersion Property

Version string from the Remote I/O device.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public string Version { get; }
```

### Property Value

Type: [String](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Remote Namespace](#)

# Device Methods

The [Device](#) type exposes the following members.

## Methods

	Name	Description
≡	<a href="#">ADC_Available</a>	Query available A/D inputs.
≡	<a href="#">ADC_Create</a>	Create a remote A/D input.
≡	<a href="#">DAC_Available</a>	Query available D/A outputs.
≡	<a href="#">DAC_Create</a>	Create a remote D/A output.
≡	<a href="#">Dispatcher</a>	Command dispatcher.
≡	<a href="#">GPIO_Available</a>	Query available GPIO pins.
≡	<a href="#">GPIO_Create</a>	Create a remote GPIO pin object.
≡	<a href="#">I2C_Available</a>	Query available I <sup>2</sup> C buses.
≡	<a href="#">I2C_Create</a>	Create a remote I <sup>2</sup> C bus controller.
≡	<a href="#">PWM_Available</a>	Query available PWM outputs.
≡	<a href="#">PWM_Create</a>	Create a remote PWM output.
≡	<a href="#">SPI_Available</a>	Query available SPI slave devices.

[SPI\\_Create](#)

Create a remote SPI slave device.

---

[Top](#)

## ◀ See Also

**Reference**

[Device Class](#)

[IO.Remote Namespace](#)

---

# DeviceADC\_Available Method

Query available A/D inputs.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public List<int> ADC_Available()
```

### Return Value

Type: [ListInt32](#)

List of available A/D input numbers.

## ▪ See Also

### Reference

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceADC\_Create Method

Create a remote A/D input.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ► Syntax

C#    VB    F#

[Copy](#)

```
public Sample ADC_Create(  
    int num  
)
```

## Parameters

*num*

Type: [SystemInt32](#)

A/D input number: 0 to 127.

## Return Value

Type: [Sample](#)

A/D input object.

## ► See Also

[Reference](#)

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceDAC\_Available Method

Query available D/A outputs.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public List<int> DAC_Available()
```

### Return Value

Type: [ListInt32](#)

List of available D/A output numbers.

## ▪ See Also

### Reference

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceDAC\_Create Method

Create a remote D/A output.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Sample DAC_Create(  
    int num,  
    int sample = 0  
)
```

## Parameters

*num*

Type: [SystemInt32](#)

D/A output number: 0 to 127.

*sample* (Optional)

Type: [SystemInt32](#)

Initial DAC output sample.

## Return Value

Type: [Sample](#)

D/A output object.

## ◀ See Also

### Reference

[Device Class](#)

## IO.Remote Namespace

---

# DeviceDispatcher Method

Command dispatcher.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Dispatcher(  
    Message cmd,  
    Message resp  
)
```

## Parameters

*cmd*

Type: [IO.Interfaces.Message64Message](#)

Command to be sent.

*resp*

Type: [IO.Interfaces.Message64Message](#)

Response to be received.

## ◀ See Also

[Reference](#)

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceGPIO\_Available Method

Query available GPIO pins.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public List<int> GPIO_Available()
```

### Return Value

Type: [ListInt32](#)

List of available GPIO pin numbers.

## ▪ See Also

### Reference

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceGPIO\_Create Method

Create a remote GPIO pin object.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Pin GPIO_Create(  
    int num,  
    Direction dir,  
    bool state = false  
)
```

## Parameters

*num*

Type: [SystemInt32](#)

GPIO pin number: 0 to 127.

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin data direction: Input or Output.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## Return Value

Type: [Pin](#)

GPIO pin object.

## ↳ See Also

[Reference](#)

[Device Class](#)

[IO.Remote Namespace](#)

---

# DeviceI2C\_Available Method

Query available I<sup>2</sup>C buses.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public List<int> I2C_Available()
```

### Return Value

Type: [ListInt32](#)

List of available I<sup>2</sup>C bus numbers.

## ▪ See Also

### Reference

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceI2C\_Create Method

Create a remote I<sup>2</sup>C bus controller.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Bus I2C_Create(  
    int num,  
    int speed = 100000  
)
```

## Parameters

*num*

Type: [SystemInt32](#)

I<sup>2</sup>C bus number: 0 to 127.

*speed* (Optional)

Type: [SystemInt32](#)

I<sup>2</sup>C bus clock frequency in Hz

## Return Value

Type: [Bus](#)

I<sup>2</sup>C bus controller object.

## ◀ See Also

### Reference

[Device Class](#)

## IO.Remote Namespace

---

# DevicePWM\_Available Method

Query available PWM outputs.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public List<int> PWM_Available()
```

### Return Value

Type: [ListInt32](#)

List of available PWM output numbers.

## ◀ See Also

### Reference

[Device Class](#)

[IO.Remote Namespace](#)

# DevicePWM\_Create Method

Create a remote PWM output.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Output PWM_Create(  
    int num,  
    int freq,  
    double duty = 0  
)
```

## Parameters

*num*

Type: [SystemInt32](#)

PWM output number: 0 to 127.

*freq*

Type: [SystemInt32](#)

PWM pulse frequency in Hz.

*duty* (Optional)

Type: [SystemDouble](#)

Initial PWM output duty cycle.

## Return Value

Type: [Output](#)

PWM output object.

## ↳ See Also

[Reference](#)

[Device Class](#)

[IO.Remote Namespace](#)

---

# DeviceSPI\_Available Method

Query available SPI slave devices.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public List<int> SPI_Available()
```

### Return Value

Type: [ListInt32](#)

List of available SPI slave device numbers.

## ▪ See Also

### Reference

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceSPI\_Create Method

Create a remote SPI slave device.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Device SPI_Create(  
    int num,  
    int mode,  
    int wordsize,  
    int speed  
)
```

## Parameters

*num*

Type: [SystemInt32](#)

SPI slave device number: 0 to 127.

*mode*

Type: [SystemInt32](#)

SPI transfer mode: 0 to 3.

*wordsize*

Type: [SystemInt32](#)

SPI transfer word size: 8, 16, or 32.

*speed*

Type: [SystemInt32](#)

SPI transfer speed in bits per second.

## **Return Value**

Type: [Device](#)

SPI slave device object.

## **Remarks**

The actual SPI transfer rate will be the highest realizable rate that does not exceed the value specified in [speed](#).

## **See Also**

### **Reference**

[Device Class](#)

[IO.Remote Namespace](#)

# Device Fields

The [Device](#) type exposes the following members.

## Fields

	Name	Description
• 	<a href="#">MAX_CHANNELS</a>	Maximum number of channels each subsystem can support.
• 	<a href="#">Unavailable</a>	Designator for an unavailable channel.

[Top](#)

## See Also

**Reference**

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceMAX\_CHANNELS Field

Maximum number of channels each subsystem can support.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int MAX_CHANNELS = 128
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Remote Namespace](#)

# DeviceUnavailable Field

Designator for an unavailable channel.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public const int Unavailable = -1
```

### Field Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Device Class](#)

[IO.Remote Namespace](#)

Munts Technologies Linux Simple I/O Library .Net Standard 2.0 Class  
Library 2.2020.135.1

# GPIO Class

Encapsulates remote GPIO pins.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.RemoteGPIO](#)

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class GPIO : Pin
```

The [GPIO](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">GPIO</a>	Create a remote GPIO pin.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## ◀ See Also

**Reference**

[IO.Remote Namespace](#)

# GPIO Constructor

Create a remote GPIO pin.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public GPIO(  
    Device dev,  
    int num,  
    Direction dir,  
    bool state = false  
)
```

## Parameters

*dev*

Type: [IO.RemoteDevice](#)

Remote I/O device object.

*num*

Type: [SystemInt32](#)

GPIO pin number: 0 to 127.

*dir*

Type: [IO.Interfaces.GPIODirection](#)

GPIO pin data direction: Input or Output.

*state (Optional)*

Type: [SystemBoolean](#)

Initial GPIO output state.

## ▪ Remarks

Use [Device.GPIO\\_Create\(\)](#) instead of this constructor.

## ▪ See Also

### Reference

[GPIO Class](#)

[IO.Remote Namespace](#)

# GPIO Properties

The [GPIO](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">state</a>	Read/Write GPIO state property.

[Top](#)

## See Also

[Reference](#)

[GPIO Class](#)

[IO.Remote Namespace](#)

# GPIOstate Property

Read/Write GPIO state property.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public bool state { get; set; }
```

### Property Value

Type: [Boolean](#)

### Implements

[Pinstate](#)

## ▪ See Also

[Reference](#)

[GPIO Class](#)

[IO.Remote Namespace](#)

# I2C Class

Encapsulates remote I<sup>2</sup>C buses.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.RemoteI2C](#)

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public class I2C : Bus
```

The [I2C](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">I2C</a>	Create a remote I <sup>2</sup> C bus controller.

[Top](#)

## ▪ Methods

	Name	Description
≡	<a href="#">Read</a>	Read bytes from an I <sup>2</sup> C slave device.
≡		

[Transaction](#) Write and read bytes to and from an I<sup>2</sup>C slave device.



[Write](#)

Write bytes to an I<sup>2</sup>C slave device.

[Top](#)

## ◀ See Also

**Reference**

[IO.Remote Namespace](#)

# I2C Constructor

Create a remote I<sup>2</sup>C bus controller.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public I2C(  
    Device dev,  
    int num,  
    int speed = 100000  
)
```

## Parameters

*dev*

Type: [IO.RemoteDevice](#)

Remote I/O device object.

*num*

Type: [SystemInt32](#)

I<sup>2</sup>C bus number: 0 to 127.

*speed (Optional)*

Type: [SystemInt32](#)

I<sup>2</sup>C bus clock frequency in Hz

## ◀ Remarks

Use [Device.I2C\\_Create\(\)](#) instead of this constructor.

## ↳ See Also

[Reference](#)

[I2C Class](#)

[IO.Remote Namespace](#)

---

# I2C Methods

The [I2C](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Read</a>	Read bytes from an I <sup>2</sup> C slave device.
	<a href="#">Transaction</a>	Write and read bytes to and from an I <sup>2</sup> C slave device.
	<a href="#">Write</a>	Write bytes to an I <sup>2</sup> C slave device.

[Top](#)

## See Also

### Reference

[I2C Class](#)

[IO.Remote Namespace](#)

# I2CRead Method

Read bytes from an I<sup>2</sup>C slave device.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public void Read(  
    int slaveaddr,  
    byte[] resp,  
    int resplen  
)
```

## Parameters

*slaveaddr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

## Implements

[BusRead\(Int32, Byte, Int32\)](#)

## ↳ See Also

**Reference**

[I2C Class](#)

[IO.Remote Namespace](#)

---

# I2CTransaction Method

Write and read bytes to and from an I<sup>2</sup>C slave device.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Transaction(  
    int slaveaddr,  
    byte[] cmd,  
    int cmdlen,  
    byte[] resp,  
    int resplen,  
    int delayus  
)
```

## Parameters

*slaveaddr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read.

*delayus*

Type: [SystemInt32](#)

Delay in microseconds between the I<sup>2</sup>C write and read cycles.

Allowed values are 0 to 65535 microseconds.

## Implements

[BusTransaction\(Int32, Byte, Int32, Byte, Int32, Int32\)](#)

## See Also

### Reference

[I2C Class](#)

[IO.Remote Namespace](#)

# I2CWrite Method

Write bytes to an I<sup>2</sup>C slave device.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Write(  
    int slaveaddr,  
    byte[] cmd,  
    int cmdlen  
)
```

## Parameters

*slaveaddr*

Type: [SystemInt32](#)

I<sup>2</sup>C slave address.

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write.

## Implements

[BusWrite\(Int32, Byte, Int32\)](#)

## ↳ See Also

**Reference**

[I2C Class](#)

[IO.Remote Namespace](#)

---

# MessageTypes Enumeration

Remote I/O protocol message types

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum MessageTypes
```

## ▪ Members

Member name	Value	Description
LOOPBACK_REQUEST	0	Loopback request
LOOPBACK_RESPONSE	1	Loopback response
VERSION_REQUEST	2	Version string request
VERSION_RESPONSE	3	Version string response
CAPABILITY_REQUEST	4	Capability

			string request
CAPABILITY_RESPONSE	5	Capability string response	
GPIO_PRESENT_REQUEST	6	GPIO pins available request	
GPIO_PRESENT_RESPONSE	7	GPIO pins available response	
GPIO_CONFIGURE_REQUEST	8	GPIO pins configure request	
GPIO_CONFIGURE_RESPONSE	9	GPIO pins configure response	
GPIO_READ_REQUEST	10	GPIO pins read request	
GPIO_READ_RESPONSE	11	GPIO pins read response	
GPIO_WRITE_REQUEST	12	GPIO pins write request	

GPIO_WRITE_RESPONSE	13	GPIO pins write response
I2C_PRESENT_REQUEST	14	I <sup>2</sup> C buses available request
I2C_PRESENT_RESPONSE	15	I <sup>2</sup> C buses available response
I2C_CONFIGURE_REQUEST	16	I <sup>2</sup> C bus configure request
I2C_CONFIGURE_RESPONSE	17	I <sup>2</sup> C bus configure response
I2C_TRANSACTION_REQUEST	18	I <sup>2</sup> C bus transaction request
I2C_TRANSACTION_RESPONSE	19	I <sup>2</sup> C bus transaction response
SPI_PRESENT_REQUEST	20	SPI slave devices available request
SPI_PRESENT_RESPONSE	21	SPI slave devices

			available response
SPI_CONFIGURE_REQUEST	22	SPI slave device configure request	
SPI_CONFIGURE_RESPONSE	23	SPI slave device configure response	
SPI_TRANSACTION_REQUEST	24	SPI bus transaction request	
SPI_TRANSACTION_RESPONSE	25	SPI bus transaction response	
ADC_PRESENT_REQUEST	26	ADC inputs available request	
ADC_PRESENT_RESPONSE	27	ADC inputs available response	
ADC_CONFIGURE_REQUEST	28	ADC input configure request	
ADC_CONFIGURE_RESPONSE	29	ADC input configure	

			response
ADC_READ_REQUEST	30		ADC input read request
ADC_READ_RESPONSE	31		ADC input read response
DAC_PRESENT_REQUEST	32		DAC outputs available request
DAC_PRESENT_RESPONSE	33		DAC outputs available response
DAC_CONFIGURE_REQUEST	34		DAC input configure request
DAC_CONFIGURE_RESPONSE	35		DAC input configure response
DAC_WRITE_REQUEST	36		DAC output write request
DAC_WRITE_RESPONSE	37		DAC output write response

PWM_PRESENT_REQUEST	38	PWM outputs available request
PWM_PRESENT_RESPONSE	39	PWM outputs available response
PWM_CONFIGURE_REQUEST	40	PWM input configure request
PWM_CONFIGURE_RESPONSE	41	PWM input configure response
PWM_WRITE_REQUEST	42	PWM output write request
PWM_WRITE_RESPONSE	43	PWM output write response

## See Also

**Reference**

[IO.Remote Namespace](#)

# PeripheralTypes Enumeration

Types of remote peripherals

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum PeripheralTypes
```

## ▪ Members

Member name	Value	Description
ADC	0	A/D inputs
DAC	1	D/A outputs
GPIO	2	GPIO pins
I2C	3	I <sup>2</sup> C bus controllers
PWM	4	SPI slave devices
SPI	5	PWM outputs

## ▪ See Also

**Reference**

[IO.Remote Namespace](#)



# PWM Class

Encapsulates remote PWM outputs.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.RemotePWM](#)

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class PWM : Output
```

The [PWM](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">PWM</a>	Create a remote PWM output.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">dutycycle</a>	Write-only property for setting the PWM output duty cycle.

[Top](#)

## ◀ See Also

**Reference**

[IO.Remote Namespace](#)

# PWM Constructor

Create a remote PWM output.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public PWM(  
    Device dev,  
    int num,  
    int freq,  
    double duty = 0  
)
```

## Parameters

*dev*

Type: [IO.RemoteDevice](#)

Remote I/O device object.

*num*

Type: [SystemInt32](#)

PWM output number: 0 to 127.

*freq*

Type: [SystemInt32](#)

PWM pulse frequency in Hz.

*duty* (Optional)

Type: [SystemDouble](#)

Initial PWM output duty cycle.

## ▪ Remarks

Use `Device.PWM_Create()` instead of this constructor.

## ▪ See Also

### Reference

[PWM Class](#)

[IO.Remote Namespace](#)

# PWM Properties

The [PWM](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">dutycycle</a>	Write-only property for setting the PWM output duty cycle.

[Top](#)

## See Also

[Reference](#)

[PWM Class](#)

[IO.Remote Namespace](#)

# PWMdutycycle Property

Write-only property for setting the PWM output duty cycle.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public double dutycycle { set; }
```

### Property Value

Type: [Double](#)

### Implements

[Outputdutycycle](#)

## ▪ See Also

[Reference](#)

[PWM Class](#)

[IO.Remote Namespace](#)

# SPI Class

Encapsulates remote SPI slave devices.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.RemoteSPI](#)

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class SPI : Device
```

The [SPI](#) type exposes the following members.

## ▪ Constructors

	Name	Description
≡	<a href="#">SPI</a>	Create a remote SPI slave device.

[Top](#)

## ▪ Methods

	Name	Description
≡	<a href="#">Read</a>	Read bytes from an SPI slave device.
≡		

**Transaction** Write and read bytes to and from an SPI slave device.



**Write** Write bytes to an SPI slave device.

[Top](#)

## ◀ See Also

**Reference**

[IO.Remote Namespace](#)

# SPI Constructor

Create a remote SPI slave device.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public SPI(  
    Device dev,  
    int num,  
    int mode,  
    int wordsize,  
    int speed  
)
```

## Parameters

*dev*

Type: [IO.RemoteDevice](#)

Remote I/O device object.

*num*

Type: [SystemInt32](#)

SPI slave device number: 0 to 127.

*mode*

Type: [SystemInt32](#)

SPI transfer mode: 0 to 3.

*wordsize*

Type: [SystemInt32](#)

SPI transfer word size: 8, 16, or 32.

*speed*

Type: [SystemInt32](#)

SPI transfer speed in bits per second.

## ▪ Remarks

Use [Device.SPI\\_Create\(\)](#) instead of this constructor.

The actual SPI transfer rate will be the highest realizable rate that does not exceed the value specified in *speed*.

## ▪ See Also

**Reference**

[SPI Class](#)

[IO.Remote Namespace](#)

# SPI Methods

The [SPI](#) type exposes the following members.

## Methods

	Name	Description
	<a href="#">Read</a>	Read bytes from an SPI slave device.
	<a href="#">Transaction</a>	Write and read bytes to and from an SPI slave device.
	<a href="#">Write</a>	Write bytes to an SPI slave device.

[Top](#)

## See Also

### Reference

[SPI Class](#)

[IO.Remote Namespace](#)

# SPIRead Method

Read bytes from an SPI slave device.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Read(  
    byte[] resp,  
    int resplen  
)
```

## Parameters

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read: 1 to 60.

## Implements

[DeviceRead\(Byte, Int32\)](#)

## ◀ See Also

### Reference

[SPI Class](#)

[IO.Remote Namespace](#)

---

# SPITransaction Method

Write and read bytes to and from an SPI slave device.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Transaction(  
    byte[] cmd,  
    int cmdLen,  
    byte[] resp,  
    int resplen,  
    int delayus = 0  
)
```

## Parameters

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write: 0 to 57.

*resp*

Type: [SystemByte](#)

Response buffer.

*resplen*

Type: [SystemInt32](#)

Number of bytes to read: 0 to 60.

*delayus* (Optional)

Type: [SystemInt32](#)

Delay in microseconds between write and read operations: 0 to 65535.

## Implements

[DeviceTransaction\(Byte, Int32, Byte, Int32, Int32\)](#)

## See Also

### Reference

[SPI Class](#)

[IO.Remote Namespace](#)

# SPIWrite Method

Write bytes to an SPI slave device.

**Namespace:** [IO.Remote](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public void Write(  
    byte[] cmd,  
    int cmdLen  
)
```

## Parameters

*cmd*

Type: [SystemByte](#)

Command buffer.

*cmdlen*

Type: [SystemInt32](#)

Number of bytes to write: 1 to 57.

## Implements

[DeviceWrite\(Byte, Int32\)](#)

## ◀ See Also

### Reference

[SPI Class](#)

[IO.Remote Namespace](#)

---

# IO.Remote.mikroBUS Namespace

Mikroelektronika mikroBUS (<https://www.mikroe.com/mikrobus>)  
Remote I/O protocol Server and Socket Services

## ▪ Classes

	Class	Description
	<a href="#">Shield</a>	Encapsulates mikroBUS shields on Remote I/O Protocol servers providing mikroBUS sockets).
	<a href="#">Socket</a>	Encapsulates mikroBUS sockets.

## ▪ Enumerations

	Enumeration	Description
	<a href="#">ShieldKinds</a>	Supported mikroBUS shields.

# Shield Class

Encapsulates mikroBUS shields on Remote I/O Protocol servers providing [mikroBUS](#) sockets).

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Remote.mikroBUSShield](#)

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public static class Shield
```

The [Shield](#) type exposes the following members.

## ▪ Properties

	Name	Description
 <a href="#">S</a>	<a href="#">kind</a>	Returns the kind of mikroBUS shield that is installed on the Remote I/O Protocol server, as obtained from the <a href="#">SHIELDNAME</a> environment variable.

[Top](#)

## ▪ Fields

	<b>Name</b>	<b>Description</b>
 	<a href="#">I2CBus</a>	Shared I <sup>2</sup> C bus that is common to all sockets on this shield.

[Top](#)

## ◀ See Also

### Reference

[IO.Remote.mikroBUS Namespace](#)

# Shield Properties

The [Shield](#) type exposes the following members.

## Properties

	Name	Description
 	<a href="#">kind</a>	Returns the kind of mikroBUS shield that is installed on the Remote I/O Protocol server, as obtained from the <code>SHIELDNAME</code> environment variable.

[Top](#)

## See Also

[Reference](#)

[Shield Class](#)

[IO.Remote.mikroBUS Namespace](#)

# Shieldkind Property

Returns the kind of mikroBUS shield that is installed on the Remote I/O Protocol server, as obtained from the **SHIELDNAME** environment variable.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public static ShieldKinds kind { get; }
```

## Property Value

Type: [ShieldKinds](#)

## ↳ See Also

[Reference](#)

[Shield Class](#)

[IO.Remote.mikroBUS Namespace](#)

Munts Technologies Linux Simple I/O Library .Net Standard 2.0 Class  
Library 2.2020.135.1

# Shield Fields

The [Shield](#) type exposes the following members.

## ↳ Fields

Name	Description
 <a href="#">I2CBus</a>	Shared I <sup>2</sup> C bus that is common to all sockets on this shield.

[Top](#)

## ↳ See Also

[Reference](#)

[Shield Class](#)

[IO.Remote.mikroBUS Namespace](#)

# ShieldI2CBus Field

Shared I<sup>2</sup>C bus that is common to all sockets on this shield.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

[C#](#)    [VB](#)    [F#](#)

[Copy](#)

```
public static Bus I2CBus
```

## Field Value

Type: [Bus](#)

## ↳ See Also

[Reference](#)

[Shield Class](#)

[IO.Remote.mikroBUS Namespace](#)

# ShieldKinds Enumeration

Supported mikroBUS shields.

**Namespace:** `IO.Remote.mikroBUS`

**Assembly:** `libsimpleio` (in `libsimpleio.dll`) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public enum Kinds
```

## ▪ Members

Member name	Value	Description
<code>Clicker</code>	0	<a href="#">Mikroelektronika STM32F4 Clicker</a> with MUNTS-0011 Remote I/O Server firmware, with one mikroBUS socket.
<code>PiClick1</code>	1	Raspberry Pi with Mikroelektronika Pi Click Shield <a href="#">MIKROE-1512/1513</a> for 26-pin expansion header, with one mikroBUS socket (Obsolete.)

---

PiClick2	2	Raspberry Pi with Mikroelektronika Pi 2 Click Shield <a href="#">MIKROE-1879</a> for 40-pin expansion header, with two mikroBUS sockets.
PiClick3	3	Mikroelektronika Pi 3 Click Shield <a href="#">MIKROE-2756</a> for 40-pin expansion header, with selectable on-board A/D converter and two mikroBUS sockets.
PocketBeagle	4	<a href="#">PocketBeagle</a> with female headers on top, with two mikroBUS sockets.
Unknown	2147483647	No known mikroBUS shield installed.

---

## See Also

**Reference**

[IO.Remote.mikroBUS Namespace](#)

# Socket Class

Encapsulates mikroBUS sockets.

## ▪ Inheritance Hierarchy

[SystemObject](#) [IO.Remote.mikroBUSSocket](#)

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ▪ Syntax

C#    VB    F#

[Copy](#)

```
public class Socket
```

The [Socket](#) type exposes the following members.

## ▪ Constructors

	Name	Description
	<a href="#">Socket</a>	Constructor for a single mikroBUS socket.

[Top](#)

## ▪ Properties

	Name	Description
	<a href="#">AIN</a>	Returns the ADC input designator for AN.

	<a href="#">AN</a>	Returns the GPIO pin designator for AN.
	<a href="#">CS</a>	Returns the GPIO pin designator for CS.
	<a href="#">I2CBus</a>	Returns the I <sup>2</sup> C bus designator for this socket.
	<a href="#">INT</a>	Returns the GPIO pin designator for INT.
	<a href="#">MISO</a>	Returns the GPIO pin designator for MISO.
	<a href="#">MOSI</a>	Returns the GPIO pin designator for MOSI.
	<a href="#">PWM</a>	Returns the GPIO pin designator for PWM.
	<a href="#">PWMOut</a>	Returns the PWM output designator for PWM.
	<a href="#">RST</a>	Returns the GPIO pin designator for RST.
	<a href="#">RX</a>	Returns the GPIO pin designator for RX.
	<a href="#">SCK</a>	Returns the GPIO pin designator for SCK.
	<a href="#">SCL</a>	Returns the GPIO pin designator for SCL.
	<a href="#">SDA</a>	Returns the GPIO pin designator for SDA.
	<a href="#">SPIDev</a>	Returns the SPI device designator for

this socket.



[TX](#)

Returns the GPIO pin designator for TX.

[Top](#)

## ◀ See Also

### Reference

[IO.Remote.mikroBUS Namespace](#)

# Socket Constructor

Constructor for a single mikroBUS socket.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public Socket(  
    int num,  
    ShieldKinds shield = ShieldKinds.Unknown  
)
```

## Parameters

*num*

Type: [SystemInt32](#)

Socket number.

*shield* (Optional)

Type: [IO.Remote.mikroBUSShieldKinds](#)

mikroBUS shield kind. Zero indicates automatic detection using the [Shield.kind](#) property.

## ◀ See Also

[Reference](#)

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# Socket Properties

The [Socket](#) type exposes the following members.

## Properties

	Name	Description
	<a href="#">AIN</a>	Returns the ADC input designator for AN.
	<a href="#">AN</a>	Returns the GPIO pin designator for AN.
	<a href="#">CS</a>	Returns the GPIO pin designator for CS.
	<a href="#">I2CBus</a>	Returns the I <sup>2</sup> C bus designator for this socket.
	<a href="#">INT</a>	Returns the GPIO pin designator for INT.
	<a href="#">MISO</a>	Returns the GPIO pin designator for MISO.
	<a href="#">MOSI</a>	Returns the GPIO pin designator for MOSI.
	<a href="#">PWM</a>	Returns the GPIO pin designator for PWM.
	<a href="#">PWMOut</a>	Returns the PWM output designator for PWM.

	RST	Returns the GPIO pin designator for RST.
	RX	Returns the GPIO pin designator for RX.
	SCK	Returns the GPIO pin designator for SCK.
	SCL	Returns the GPIO pin designator for SCL.
	SDA	Returns the GPIO pin designator for SDA.
	SPIDev	Returns the SPI device designator for this socket.
	TX	Returns the GPIO pin designator for TX.

[Top](#)

## ◀ See Also

[Reference](#)

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketAIN Property

Returns the ADC input designator for AN.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int AIN { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketAN Property

Returns the GPIO pin designator for AN.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public int AN { get; }
```

### Property Value

Type: [Int32](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketCS Property

Returns the GPIO pin designator for CS.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public int CS { get; }
```

### Property Value

Type: [Int32](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketI2CBus Property

Returns the I<sup>2</sup>C bus designator for this socket.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ↳ Syntax

C#    VB    F#

[Copy](#)

```
public int I2CBus { get; }
```

## Property Value

Type: [Int32](#)

## ↳ See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketINT Property

Returns the GPIO pin designator for INT.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public int INT { get; }
```

### Property Value

Type: [Int32](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketMISO Property

Returns the GPIO pin designator for MISO.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int MISO { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketMOSI Property

Returns the GPIO pin designator for MOSI.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int MOSI { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketPWM Property

Returns the GPIO pin designator for PWM.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int PWM { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketPWMOut Property

Returns the PWM output designator for PWM.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int PWMOut { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketRST Property

Returns the GPIO pin designator for RST.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int RST { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketRX Property

Returns the GPIO pin designator for RX.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public int RX { get; }
```

### Property Value

Type: [Int32](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketSCK Property

Returns the GPIO pin designator for SCK.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public int SCK { get; }
```

### Property Value

Type: [Int32](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketSCL Property

Returns the GPIO pin designator for SCL.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public int SCL { get; }
```

### Property Value

Type: [Int32](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketSDA Property

Returns the GPIO pin designator for SDA.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int SDA { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketSPIDev Property

Returns the SPI device designator for this socket.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## ◀ Syntax

C#    VB    F#

[Copy](#)

```
public int SPIDev { get; }
```

### Property Value

Type: [Int32](#)

## ◀ See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# SocketTX Property

Returns the GPIO pin designator for TX.

**Namespace:** [IO.Remote.mikroBUS](#)

**Assembly:** libsimpleio (in libsimpleio.dll) Version: 2.2020.135.1

## « Syntax

C#    VB    F#

[Copy](#)

```
public int TX { get; }
```

### Property Value

Type: [Int32](#)

## « See Also

### Reference

[Socket Class](#)

[IO.Remote.mikroBUS Namespace](#)

# Index

IO.Bindings.libsimpleio	1
libADC Class	3
libADC Constructor	5
libADC Methods	6
ADC_close Method	7
ADC_get_name Method	8
ADC_open Method	10
ADC_read Method	12
libDAC Class	14
libDAC Constructor	16
libDAC Methods	17
DAC_close Method	18
DAC_get_name Method	19
DAC_open Method	21
DAC_write Method	23
libEvent Class	25
libEvent Constructor	27
libEvent Methods	28
EVENT_close Method	30
EVENT_modify_fd Method	31
EVENT_open Method	33
EVENT_register_fd Method	34
EVENT_unregister_fd Method	36
EVENT_wait Method	38
libGPIO Class	40
libGPIO Constructor	45
libGPIO Methods	46

GPIO_chip_info Method	48
GPIO_close Method	50
GPIO_configure Method	51
GPIO_line_close Method	53
GPIO_line_event Method	54
GPIO_line_info Method	56
GPIO_line_open Method	58
GPIO_line_read Method	60
GPIO_line_write Method	62
GPIO_open Method	64
GPIO_read Method	66
GPIO_write Method	68
libGPIO Fields	70
DIRECTION_INPUT Field	74
DIRECTION_OUTPUT Field	75
DRIVER_OPENDRAIN Field	76
DRIVER_OPENSOURCE Field	77
DRIVER_PUSH_PULL Field	78
EDGE_BOTH Field	79
EDGE_FALLING Field	80
EDGE_NONE Field	81
EDGE_RISING Field	82
EVENT_REQUEST_BOTH Field	83
EVENT_REQUEST_FALLING Field	84
EVENT_REQUEST_NONE Field	85
EVENT_REQUEST_RISING Field	86
LINE_INFO_ACTIVE_LOW Field	87
LINE_INFO_KERNEL Field	88
LINE_INFO_OPEN_DRAIN Field	89

LINE_INFO_OPEN_SOURCE Field	90
LINE_INFO_OUTPUT Field	91
LINE_REQUEST_ACTIVE_HIGH Field	92
LINE_REQUEST_ACTIVE_LOW Field	93
LINE_REQUEST_INPUT Field	94
LINE_REQUEST_OPEN_DRAIN Field	95
LINE_REQUEST_OPEN_SOURCE Field	96
LINE_REQUEST_OUTPUT Field	97
LINE_REQUEST_PUSH_PULL Field	98
POLARITY_ACTIVEHIGH Field	99
POLARITY_ACTIVELOW Field	100
libHIDRaw Class	101
libHIDRaw Constructor	103
libHIDRaw Methods	104
HIDRAW_close Method	106
HIDRAW_get_info Method	107
HIDRAW_get_name Method	109
HIDRAW_open Method	111
HIDRAW_open_id Method	113
HIDRAW_receive Method	115
HIDRAW_send Method	117
libI2C Class	119
libI2C Constructor	121
libI2C Methods	122
I2C_close Method	123
I2C_open Method	124
I2C_transaction Method	126
libIPV4 Class	128
libIPV4 Constructor	131

libIPV4 Methods	132
IPV4_ntoa Method	134
IPV4_resolve Method	136
TCP4_accept Method	138
TCP4_close Method	140
TCP4_connect Method	141
TCP4_receive Method	143
TCP4_send Method	145
TCP4_server Method	147
UDP4_close Method	149
UDP4_open Method	150
UDP4_receive Method	152
UDP4_send Method	154
libIPV4 Fields	156
INADDR_ANY Field	158
INADDR_BROADCAST Field	159
INADDR_LOOPBACK Field	160
MSG_DONTROUTE Field	161
MSG_DONTWAIT Field	162
MSG_MORE Field	163
libLinux Class	164
libLinux Constructor	169
libLinux Methods	170
LINUX_command Method	172
LINUX_detach Method	173
LINUX_drop_privileges Method	174
LINUX_errno Method	176
LINUX_openlog Method	177
LINUX_poll Method	179

LINUX_strerror Method	181
LINUX_syslog Method	183
LINUX_usleep Method	185
libLinux Fields	186
LOG_ALERT Field	189
LOG_AUTH Field	190
LOG_AUTHPRIV Field	191
LOG_CRIT Field	192
LOG_CRON Field	193
LOG_DAEMON Field	194
LOG_DEBUG Field	195
LOG_EMERG Field	196
LOG_ERR Field	197
LOG_FTP Field	198
LOG_INFO Field	199
LOG_KERN Field	200
LOG_LOCAL0 Field	201
LOG_LOCAL1 Field	202
LOG_LOCAL2 Field	203
LOG_LOCAL3 Field	204
LOG_LOCAL4 Field	205
LOG_LOCAL5 Field	206
LOG_LOCAL6 Field	207
LOG_LOCAL7 Field	208
LOG_LPR Field	209
LOG_MAIL Field	210
LOG_NEWS Field	211
LOG_NOTICE Field	212
LOG_PROGNAME Field	213

LOG_SYSLOG Field	214
LOG_USER Field	215
LOG_UUCP Field	216
LOG_WARNING Field	217
POLLERR Field	218
POLLHUP Field	219
POLLIN Field	220
POLLNVAL Field	221
POLLOUT Field	222
POLLPRI Field	223
libPWM Class	224
libPWM Constructor	226
libPWM Methods	227
PWM_close Method	228
PWM_configure Method	229
PWM_open Method	231
PWM_write Method	233
libPWM Fields	235
ActiveHigh Field	236
ActiveLow Field	237
libSerial Class	238
libSerial Constructor	240
libSerial Methods	241
SERIAL_close Method	242
SERIAL_open Method	243
SERIAL_receive Method	245
SERIAL_send Method	247
libSerial Fields	249
PARITY EVEN Field	250

PARITY_NONE Field	251
PARITY_ODD Field	252
libSPI Class	253
libSPI Constructor	255
libSPI Methods	256
SPI_close Method	257
SPI_open Method	258
SPI_transaction Method	260
libSPI Fields	262
SPI_AUTO_CS Field	263
libStream Class	264
libStream Constructor	266
libStream Methods	267
STREAM_decode_frame Method	268
STREAM_encode_frame Method	270
STREAM_receive_frame Method	272
STREAM_send_frame Method	274
libWatchdog Class	276
libWatchdog Constructor	278
libWatchdog Methods	279
WATCHDOG_close Method	280
WATCHDOG_get_timeout Method	282
WATCHDOG_kick Method	284
WATCHDOG_open Method	286
WATCHDOG_set_timeout Method	288
IO.Devices.AD5593R	290
Device Class	291
Device Constructor	294
Device Properties	295

ADC_Reference Property	297
DAC_Reference Property	298
GPIO_Inputs Property	299
GPIO_Outputs Property	300
Device Methods	301
ADC_Create Method	302
ConfigureChannel Method	303
DAC_Create Method	304
GPIO_Create Method	306
Read_ADC Method	308
Write_DAC Method	309
Device Fields	310
ADC_Resolution Field	311
DAC_Resolution Field	312
MaxChannel Field	313
MinChannel Field	314
PinMode Enumeration	315
ReferenceMode Enumeration	317
IO.Devices.AD5593R.ADC	319
Sample Class	320
Sample Constructor	322
Sample Properties	323
resolution Property	324
sample Property	325
IO.Devices.AD5593R.DAC	326
Sample Class	327
Sample Constructor	329
Sample Properties	331
resolution Property	332

sample Property	333
IO.Devices.AD5593R.GPIO	334
Pin Class	335
Pin Constructor	337
Pin Properties	339
state Property	340
IO.Devices.ADC121C021	341
Sample Class	342
Sample Constructor	344
Sample Properties	345
resolution Property	346
sample Property	347
IO.Devices.ClickBoards.RemoteIO.ADAC	348
Board Class	349
Board Constructor	351
Board Properties	353
device Property	354
Board Methods	355
ADC Method	356
DAC Method	357
GPIO Method	359
Reset Method	361
Board Fields	362
DefaultAddress Field	363
IO.Devices.ClickBoards.RemoteIO.Expand	364
Board Class	365
Board Constructor	367
Board Properties	368
device Property	369

Board Methods	370
GPIO Method	371
Reset Method	373
IO.Devices.ClickBoards.RemoteIO.Expand2	374
Board Class	375
Board Constructor	377
Board Properties	379
device Property	380
Board Methods	381
GPIO Method	382
Reset Method	384
Board Fields	385
DefaultAddress Field	386
IO.Devices.ClickBoards.RemoteIO.PWM	387
Board Class	388
Board Constructor	390
Board Properties	392
dev Property	393
Board Methods	394
GPIO Method	395
PWM Method	397
Servo Method	399
Board Fields	401
DefaultAddress Field	402
IO.Devices.ClickBoards.RemoteIO.SevenSegment	403
Board Class	404
Board Constructor	407
Board Properties	409
blanking Property	411

brightness Property	412
leftdp Property	413
radix Property	414
rightdp Property	415
state Property	416
Board Methods	417
Clear Method	418
Reset Method	419
Board.Base Enumeration	420
Board.ZeroBlanking Enumeration	421
IO.Devices.ClickBoards.SimpleIO.ADAC	422
Board Class	423
Board Constructor	425
Board Properties	426
device Property	427
Board Methods	428
ADC Method	429
DAC Method	430
GPIO Method	432
Reset Method	434
Board Fields	435
DefaultAddress Field	436
IO.Devices.ClickBoards.SimpleIO.Expand	437
Board Class	438
Board Constructor	440
Board Properties	441
device Property	442
Board Methods	443
GPIO Method	444

Reset Method	446
IO.Devices.ClickBoards.SimpleIO.Expand2	447
Board Class	448
Board Constructor	450
Board Properties	451
device Property	452
Board Methods	453
GPIO Method	454
Reset Method	456
Board Fields	457
DefaultAddress Field	458
IO.Devices.ClickBoards.SimpleIO.PWM	459
Board Class	460
Board Constructor	462
Board Properties	464
dev Property	465
Board Methods	466
GPIO Method	467
PWM Method	469
Servo Method	471
Board Fields	473
DefaultAddress Field	474
IO.Devices.ClickBoards.SimpleIO.SevenSegment	475
Board Class	476
Board Constructor	479
Board Properties	481
blanking Property	483
brightness Property	484
leftdp Property	485

radix Property	486
rightdp Property	487
state Property	488
Board Methods	489
Clear Method	490
Board.Base Enumeration	491
Board.ZeroBlanking Enumeration	492
IO.Devices.Grove.ADC	493
Device Class	494
Device Constructor	496
Device Properties	497
voltage Property	498
IO.Devices.Grove.Temperature	499
Device Class	500
Device Constructor	502
Device Properties	503
Celsius Property	504
Fahrenheit Property	505
Kelvins Property	506
IO.Devices.Grove.Temperature_Humidity	507
Device Class	508
Device Constructor	510
Device Properties	511
IO.Devices.HDC1080	513
Device Class	514
Device Constructor	517
Device Properties	518
Celsius Property	520
DeviceID Property	521

Fahrenheit Property	522
Humidity Property	523
Kelvins Property	524
ManufacturerID Property	525
Device Methods	526
Read Method	527
Write Method	528
Device Fields	529
RegConfiguration Field	531
RegDeviceID Field	532
RegHumidity Field	533
RegManufacturerID Field	534
RegSerialNumberFirst Field	535
RegSerialNumberLast Field	536
RegSerialNumberMid Field	537
RegTemperature Field	538
IO.Devices.MCP23017	539
Device Class	540
Device Constructor	543
Device Properties	544
Direction Property	546
DirectionA Property	547
DirectionB Property	548
Polarity Property	549
PolarityA Property	550
PolarityB Property	551
Port Property	552
PortA Property	553
PortB Property	554

Pullups Property	555
PullupsA Property	556
PullupsB Property	557
Device Methods	558
GPIO_Create Method	559
Device Fields	561
MaxChannel Field	562
MinChannel Field	563
IO.Devices.MCP23017.GPIO	564
Pin Class	565
Pin Constructor	567
Pin Properties	569
state Property	570
IO.Devices.MCP23S17	571
Device Class	572
Device Constructor	576
Device Properties	577
Direction Property	579
DirectionA Property	580
DirectionB Property	581
Polarity Property	582
PolarityA Property	583
PolarityB Property	584
Port Property	585
PortA Property	586
PortB Property	587
Pullups Property	588
PullupsA Property	589
PullupsB Property	590

Device Methods	591
GPIO_Create Method	592
Device Fields	594
MaxChannel Field	595
MinChannel Field	596
SPI_Frequency Field	597
SPI_Mode Field	598
SPI_WordSize Field	599
IO.Devices.MCP23S17.GPIO	600
Pin Class	601
Pin Constructor	603
Pin Properties	605
state Property	606
IO.Devices.PCA8574	607
Device Class	608
Device Constructor	610
Device Properties	612
Latch Property	613
Device Methods	614
Read Method	615
Write Method	616
Device Fields	617
MAX_PINS Field	618
IO.Devices.PCA8574.GPIO	619
Pin Class	620
Pin Constructor	622
Pin Properties	624
state Property	625
IO.Devices.PCA9534	626

Device Class	627
Device Constructor	630
Device Properties	632
Config Property	633
Latch Property	634
Device Methods	635
Read Method	636
Read Method	637
Read Method (Byte)	638
Write Method	639
Write Method (Byte)	640
Write Method (Byte, Byte)	641
Device Fields	642
AllInputs Field	644
AllNormal Field	645
AllOff Field	646
AllOutputs Field	647
ConfigurationReg Field	648
InputPolarityReg Field	649
InputPortReg Field	650
MAX_PINS Field	651
OutputPortReg Field	652
IO.Devices.PCA9534.GPIO	653
Pin Class	654
Pin Constructor	656
Pin Properties	658
state Property	659
IO.Devices.PCA9685	660
Device Class	661

Device Constructor	664
Device Properties	666
Frequency Property	667
Device Methods	668
ReadChannel Method	669
WriteChannel Method	670
Device Fields	671
INTERNAL_CLOCK Field	672
MAX_CHANNEL Field	673
MAX_CLOCK Field	674
MIN_CHANNEL Field	675
MIN_CLOCK Field	676
IO.Devices.PCA9685.GPIO	677
Pin Class	678
Pin Constructor	680
Pin Properties	682
state Property	683
IO.Devices.PCA9685.PWM	684
Output Class	685
Output Constructor	687
Output Properties	689
dutycycle Property	690
IO.Devices.PCA9685.Servo	691
Output Class	692
Output Constructor	694
Output Properties	696
position Property	697
IO.Devices.Pmod.HYGRO	698
Device Class	699

Device Constructor	702
Device Properties	703
Device Methods	705
IO.Devices.SN74HC595	706
Device Class	707
Device Constructor	710
Device Properties	712
Length Property	713
state Property	714
Device Methods	715
ClrBit Method	716
ReadBit Method	717
SetBit Method	719
Device Fields	720
SPI_MaxFreq Field	721
SPI_Mode Field	722
IO.Devices.SN74HC595.GPIO	723
Pin Class	724
Pin Constructor	726
Pin Properties	728
state Property	729
IO.Devices.TH02	730
Device Class	731
Device Constructor	733
Device Properties	734
Celsius Property	735
DeviceID Property	736
Fahrenheit Property	737
Humidity Property	738

Kelvins Property	739
IO.Devices.Thermistor	740
NTC_B Class	741
NTC_B Constructor	743
NTC_B Methods	745
Kelvins Method	746
IO.Devices.USB.Munts	747
HID Class	748
HID Fields	750
Product Field	751
Vendor Field	752
Serial Class	753
Serial Fields	755
Product Field	756
Vendor Field	757
IO.Interfaces.ADC	758
Input Class	759
Input Constructor	761
Input Properties	763
voltage Property	764
Sample Interface	765
Sample Properties	766
resolution Property	767
sample Property	768
Voltage Interface	769
Voltage Properties	770
voltage Property	771
IO.Interfaces.DAC	772
Output Class	773

Output Constructor	775
Output Properties	777
voltage Property	778
Sample Interface	779
Sample Properties	781
resolution Property	782
sample Property	783
Voltage Interface	784
Voltage Properties	785
voltage Property	786
IO.Interfaces.GPIO	787
Direction Enumeration	788
Pin Interface	789
Pin Properties	790
state Property	791
IO.Interfaces.Humidity	792
Sensor Interface	793
Sensor Properties	794
Humidity Property	795
IO.Interfaces.I2C	796
Bus Interface	797
Bus Methods	799
Read Method	800
Transaction Method	802
Write Method	804
Device Class	806
Device Constructor	808
Device Methods	809
Read Method	810

Transaction Method	811
Write Method	813
Speeds Class	814
Speeds Fields	816
FastMode Field	817
FastModePlus Field	818
StandardMode Field	819
IO.Interfaces.Message64	820
Message Class	821
Message Constructor	823
Message Constructor	824
Message Constructor (Byte)	825
Message Fields	826
payload Field	827
Size Field	828
Messenger Interface	829
Messenger Methods	831
Receive Method	832
Send Method	833
Transaction Method	834
IO.Interfaces.Motor	835
Output Interface	836
Output Properties	837
velocity Property	838
Velocities Class	839
Velocities Fields	841
Maximum Field	842
Minimum Field	843
Stop Field	844

IO.Interfaces.PWM	845
DutyCycles Class	846
DutyCycles Fields	848
Maximum Field	849
Minimum Field	850
Output Interface	851
Output Properties	852
dutycycle Property	853
IO.Interfaces.Servo	854
Output Interface	855
Output Properties	856
position Property	857
Positions Class	858
Positions Fields	860
Maximum Field	861
Minimum Field	862
Neutral Field	863
IO.Interfaces.SPI	864
Device Interface	865
Device Methods	867
Read Method	868
Transaction Method	869
Write Method	871
IO.Interfaces.Temperature	872
Conversions Class	873
Conversions Constructor	875
Conversions Methods	876
CelsiusToFahrenheit Method	878
CelsiusToKelvins Method	879

FahrenheitToCelsius Method	880
FahrenheitToKelvins Method	881
KelvinsToCelsius Method	882
KelvinsToFahrenheit Method	883
Sensor Interface	884
Sensor Properties	886
Celsius Property	887
Fahrenheit Property	888
Kelvins Property	889
IO.Interfaces.Watchdog	890
Timer Interface	891
Timer Properties	893
timeout Property	894
Timer Methods	895
Kick Method	896
IO.Objects.libsimpleio.ADC	897
Sample Class	898
Sample Constructor	900
Sample Properties	901
fd Property	902
resolution Property	903
sample Property	904
Sample Methods	905
name Method	906
IO.Objects.libsimpleio.DAC	907
Sample Class	908
Sample Constructor	910
Sample Properties	912
fd Property	913

resolution Property	914
sample Property	915
Sample Methods	916
name Method	917
IO.Objects.libsimpleio.Device	918
Designator Structure	919
Designator Constructor	921
Designator Properties	922
available Property	923
Designator Fields	924
chan Field	925
chip Field	926
Unavailable Field	927
IO.Objects.libsimpleio.Exceptions	928
Exception Class	929
Exception Constructor	931
Exception Constructor (String)	932
Exception Constructor (String, Int32)	933
IO.Objects.libsimpleio.GPIO	935
Pin Class	936
Pin Constructor	938
Pin Properties	940
fd Property	941
state Property	942
Pin.Driver Enumeration	943
Pin.Edge Enumeration	944
Pin.Polarity Enumeration	946
IO.Objects.libsimpleio.HID	947
Messenger Class	948

Messenger Constructor	951
Messenger Constructor (String, Int32)	952
Messenger Constructor (Int32, Int32, Int32)	954
Messenger Properties	956
bustype Property	957
fd Property	958
name Property	959
product Property	960
vendor Property	961
Messenger Methods	962
Receive Method	963
Send Method	964
Transaction Method	965
IO.Objects.libsimpleio.I2C	967
Bus Class	968
Bus Constructor	970
Bus Constructor (String)	971
Bus Constructor (Designator)	972
Bus Properties	973
fd Property	974
Bus Methods	975
Read Method	976
Transaction Method	978
Write Method	980
IO.Objects.libsimpleio.mikroBUS	982
Shield Class	983
Shield Properties	985
kind Property	986
Shield Fields	987

I2CBus Field	988
Shield.Kinds Enumeration	989
Socket Class	992
Socket Constructor	995
Socket Properties	996
AIN Property	998
AN Property	999
CS Property	1000
I2CBus Property	1001
INT Property	1002
MISO Property	1003
MOSI Property	1004
PWM Property	1005
PWMOut Property	1006
RST Property	1007
RX Property	1008
SCK Property	1009
SCL Property	1010
SDA Property	1011
SPIDev Property	1012
TX Property	1013
UART Property	1014
IO.Objects.libsimpleio.Platforms	1015
BeagleBone Class	1016
BeagleBone Fields	1022
AIN0 Field	1027
AIN1 Field	1028
AIN2 Field	1029
AIN3 Field	1030

AIN4 Field	1031
AIN5 Field	1032
AIN6 Field	1033
GPIO10 Field	1034
GPIO11 Field	1035
GPIO110 Field	1036
GPIO111 Field	1037
GPIO112 Field	1038
GPIO113 Field	1039
GPIO115 Field	1040
GPIO117 Field	1041
GPIO12 Field	1042
GPIO13 Field	1043
GPIO14 Field	1044
GPIO15 Field	1045
GPIO2 Field	1046
GPIO20 Field	1047
GPIO22 Field	1048
GPIO23 Field	1049
GPIO26 Field	1050
GPIO27 Field	1051
GPIO3 Field	1052
GPIO30 Field	1053
GPIO31 Field	1054
GPIO32 Field	1055
GPIO33 Field	1056
GPIO34 Field	1057
GPIO35 Field	1058
GPIO36 Field	1059

GPIO37 Field	1060
GPIO38 Field	1061
GPIO39 Field	1062
GPIO4 Field	1063
GPIO44 Field	1064
GPIO45 Field	1065
GPIO46 Field	1066
GPIO47 Field	1067
GPIO48 Field	1068
GPIO49 Field	1069
GPIO5 Field	1070
GPIO50 Field	1071
GPIO51 Field	1072
GPIO60 Field	1073
GPIO61 Field	1074
GPIO62 Field	1075
GPIO63 Field	1076
GPIO65 Field	1077
GPIO66 Field	1078
GPIO67 Field	1079
GPIO68 Field	1080
GPIO69 Field	1081
GPIO7 Field	1082
GPIO70 Field	1083
GPIO71 Field	1084
GPIO72 Field	1085
GPIO73 Field	1086
GPIO74 Field	1087
GPIO75 Field	1088

GPIO76 Field	1089
GPIO77 Field	1090
GPIO78 Field	1091
GPIO79 Field	1092
GPIO8 Field	1093
GPIO80 Field	1094
GPIO81 Field	1095
GPIO86 Field	1096
GPIO87 Field	1097
GPIO88 Field	1098
GPIO89 Field	1099
GPIO9 Field	1100
I2C2 Field	1101
SPI2_0 Field	1102
SPI2_1 Field	1103
PocketBeagle Class	1104
PocketBeagle Fields	1109
AIN0 Field	1113
AIN1 Field	1114
AIN2 Field	1115
AIN3 Field	1116
AIN4 Field	1117
AIN5 Field	1118
AIN6 Field	1119
AIN7 Field	1120
GPIO110 Field	1121
GPIO111 Field	1122
GPIO112 Field	1123
GPIO113 Field	1124

GPIO114 Field	1125
GPIO115 Field	1126
GPIO116 Field	1127
GPIO117 Field	1128
GPIO12 Field	1129
GPIO13 Field	1130
GPIO14 Field	1131
GPIO15 Field	1132
GPIO19 Field	1133
GPIO2 Field	1134
GPIO20 Field	1135
GPIO23 Field	1136
GPIO26 Field	1137
GPIO27 Field	1138
GPIO3 Field	1139
GPIO30 Field	1140
GPIO31 Field	1141
GPIO4 Field	1142
GPIO40 Field	1143
GPIO41 Field	1144
GPIO42 Field	1145
GPIO43 Field	1146
GPIO44 Field	1147
GPIO45 Field	1148
GPIO46 Field	1149
GPIO47 Field	1150
GPIO5 Field	1151
GPIO50 Field	1152
GPIO52 Field	1153

GPIO57 Field	1154
GPIO58 Field	1155
GPIO59 Field	1156
GPIO60 Field	1157
GPIO64 Field	1158
GPIO65 Field	1159
GPIO7 Field	1160
GPIO86 Field	1161
GPIO87 Field	1162
GPIO88 Field	1163
GPIO89 Field	1164
I2C1 Field	1165
I2C2 Field	1166
PWM0_0 Field	1167
PWM2_0 Field	1168
SPI0_0 Field	1169
SPI1_1 Field	1170
RaspberryPi Class	1171
RaspberryPi Fields	1174
AIN0 Field	1177
AIN1 Field	1178
GPIO10 Field	1179
GPIO11 Field	1180
GPIO12 Field	1181
GPIO13 Field	1182
GPIO14 Field	1183
GPIO15 Field	1184
GPIO16 Field	1185
GPIO17 Field	1186

GPIO18 Field	1187
GPIO19 Field	1188
GPIO2 Field	1189
GPIO20 Field	1190
GPIO21 Field	1191
GPIO22 Field	1192
GPIO23 Field	1193
GPIO24 Field	1194
GPIO25 Field	1195
GPIO26 Field	1196
GPIO27 Field	1197
GPIO3 Field	1198
GPIO4 Field	1199
GPIO5 Field	1200
GPIO6 Field	1201
GPIO7 Field	1202
GPIO8 Field	1203
GPIO9 Field	1204
I2C1 Field	1205
PWM0_0 Field	1206
PWM0_1 Field	1207
SPI0_0 Field	1208
SPI0_1 Field	1209
IO.Objects.libsimpleio.PWM	1210
Output Class	1211
Output Constructor	1213
Output Properties	1215
dutycycle Property	1216
fd Property	1217

IO.Objects.libsimpleio.Servo	1218
Output Class	1219
Output Constructor	1221
Output Properties	1223
fd Property	1224
position Property	1225
IO.Objects.libsimpleio.SPI	1226
Device Class	1227
Device Constructor	1229
Device Constructor (String, Int32, Int32, Int32, Pin)	1230
Device Constructor (Designator, Int32, Int32, Int32, Pin)	1232
Device Properties	1234
fd Property	1235
Device Methods	1236
Read Method	1237
Transaction Method	1239
Write Method	1241
Device Fields	1243
AUTOCHIPSELECT Field	1244
IO.Objects.libsimpleio.Watchdog	1245
Timer Class	1246
Timer Constructor	1248
Timer Properties	1249
fd Property	1250
timeout Property	1251
Timer Methods	1252
Kick Method	1253
Timer Fields	1254
DefaultDevice Field	1255

DefaultTimeout Field	1256
<b>IO.Objects.Message64.UDP</b>	1257
Messenger Class	1258
Messenger Constructor	1260
Messenger Methods	1262
Receive Method	1263
Send Method	1264
Transaction Method	1265
<b>IO.Objects.Motor.PWM</b>	1267
Output Class	1268
Output Constructor	1270
Output Constructor (Output, Output, Double)	1271
Output Constructor (Pin, Output, Double)	1273
Output Properties	1275
velocity Property	1276
<b>IO.Objects.Motor.Servo</b>	1277
Output Class	1278
Output Constructor	1280
Output Properties	1281
velocity Property	1282
<b>IO.Objects.Servo.PWM</b>	1283
Output Class	1284
Output Constructor	1286
Output Properties	1288
position Property	1289
<b>IO.Remote</b>	1290
ADC Class	1292
ADC Constructor	1294
ADC Properties	1296

resolution Property	1297
sample Property	1298
DAC Class	1299
DAC Constructor	1301
DAC Properties	1303
resolution Property	1304
sample Property	1305
Device Class	1306
Device Constructor	1309
Device Constructor	1310
Device Constructor (Messenger)	1311
Device Properties	1312
Capabilities Property	1313
Version Property	1314
Device Methods	1315
ADC_Available Method	1317
ADC_Create Method	1318
DAC_Available Method	1319
DAC_Create Method	1320
Dispatcher Method	1322
GPIO_Available Method	1323
GPIO_Create Method	1324
I2C_Available Method	1326
I2C_Create Method	1327
PWM_Available Method	1329
PWM_Create Method	1330
SPI_Available Method	1332
SPI_Create Method	1333
Device Fields	1335

MAX_CHANNELS Field	1336
Unavailable Field	1337
GPIO Class	1338
GPIO Constructor	1340
GPIO Properties	1342
state Property	1343
I2C Class	1344
I2C Constructor	1346
I2C Methods	1348
Read Method	1349
Transaction Method	1351
Write Method	1353
MessageTypes Enumeration	1355
PeripheralTypes Enumeration	1361
PWM Class	1363
PWM Constructor	1365
PWM Properties	1367
dutycycle Property	1368
SPI Class	1369
SPI Constructor	1371
SPI Methods	1373
Read Method	1374
Transaction Method	1376
Write Method	1378
IO.Remote.mikroBUS	1380
Shield Class	1381
Shield Properties	1383
kind Property	1384
Shield Fields	1385

I2CBus Field	1386
Shield.Kinds Enumeration	1387
Socket Class	1389
Socket Constructor	1392
Socket Properties	1393
AIN Property	1395
AN Property	1396
CS Property	1397
I2CBus Property	1398
INT Property	1399
MISO Property	1400
MOSI Property	1401
PWM Property	1402
PWMOut Property	1403
RST Property	1404
RX Property	1405
SCK Property	1406
SCL Property	1407
SDA Property	1408
SPIDev Property	1409
TX Property	1410