Arduino Interface Library

Generated by Doxygen 1.8.9.1

Wed Aug 19 2015 01:07:10

Contents

1 Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

ArduinoRemoteIO	**1**

2 File Index

2.1 File List

Here is a list of all files with brief descriptions:

ArduinoInterface.cpp	**4**
ArduinoInterface.h	**4**

3 Class Documentation

3.1 ArduinoRemoteIO Class Reference

```
#include <ArduinoInterface.h>
```

Private Types

- enum Mode { INPUT = 0, OUTPUT = 1 }
- enum AnalogReference {
  DEFAULT, INTERNAL, INTERNAL1V1, INTERNAL2V56,
  EXTERNAL }
- enum BitOrder { MSBFIRST, LSBFIRST }

Private Member Functions

- void pinMode (unsigned char pin, Mode mode)
- void digitalWrite (unsigned char pin, bool value)
- bool digitalRead (unsigned char pin)
- void analogReference (AnalogReference reference)
- int analogRead (unsigned char pin)
- void analogWrite (unsigned char pin, unsigned char value)
- void tone (unsigned char pin, unsigned int frequency)
- void tone (unsigned char pin, unsigned int frequency, unsigned long duration)
- void noTone (unsigned char pin)
- void shiftOut (unsigned char dataPin, unsigned char clockPin, BitOrder order, unsigned char b)
- unsigned char shiftIn (unsigned char dataPin, unsigned char clockPin, BitOrder order)
- unsigned long pulseIn (unsigned char pin, unsigned char value)
- unsigned long pulseIn (unsigned char pin, unsigned char value, unsigned long timeout)

3.1.1 Detailed Description

Definition at line 8 of file ArduinoInterface.h.

3.1.2 Member Enumeration Documentation

3.1.2.1 enum ArduinoRemoteIO::AnalogReference [private]

Enumerator

DEFAULT

INTERNAL

INTERNAL1V1

INTERNAL2V56

EXTERNAL

Definition at line 15 of file ArduinoInterface.h.

3.1.2.2 enum ArduinoRemoteIO::BitOrder [private]

Enumerator

MSBFIRST

LSBFIRST

Definition at line 23 of file ArduinoInterface.h.

3.1.2.3 enum ArduinoRemoteIO::Mode `[private]`

Enumerator

INPUT

OUTPUT

Definition at line 10 of file ArduinoInterface.h.

3.1.3 Member Function Documentation

3.1.3.1 int ArduinoRemoteIO::analogRead (unsigned char *pin*) `[private]`

3.1.3.2 void ArduinoRemoteIO::analogReference (AnalogReference *reference*) `[private]`

3.1.3.3 void ArduinoRemoteIO::analogWrite (unsigned char *pin,* unsigned char *value*) `[private]`

3.1.3.4 bool ArduinoRemoteIO::digitalRead (unsigned char *pin*) `[private]`

3.1.3.5 void ArduinoRemoteIO::digitalWrite (unsigned char *pin,* bool *value*) `[private]`

Write a HIGH or a LOW value to a digital pin.

If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW.

If the pin is configured as an INPUT, writing a HIGH value with digitalWrite() will enable an internal 20K pullup resistor (see the tutorial on digital pins). Writing LOW will disable the pullup. The pullup resistor is enough to light an LED dimly, so if LEDs appear to work, but very dimly, this is a likely cause. The remedy is to set the pin to an output with the pinMode() function.

NOTE: Digital pin 13 is harder to use as a digital input than the other digital pins because it has an LED and resistor attached to it that's soldered to the board on most boards. If you enable its internal 20k pull-up resistor, it will hang at around 1.7 V instead of the expected 5V because the onboard LED and series resistor pull the voltage level down, meaning it always returns LOW. If you must use pin 13 as a digital input, use an external pull down resistor.

Parameters

pin	The pin number.
value	LOW or HIGH.

3.1.3.6 void ArduinoRemoteIO::noTone (unsigned char *pin*) `[private]`

3.1.3.7 void ArduinoRemoteIO::pinMode (unsigned char *pin,* Mode *mode*) `[private]`

Configures the specified pin to behave either as an input or an output.

See the description of digital pins for details.

Parameters

pin	The number of the pin whose mode you wish to set
mode	Either INPUT or OUTPUT

3.1.3.8 unsigned long ArduinoRemoteIO::pulseIn (unsigned char *pin,* unsigned char *value*) `[private]`

3.1.3.9 unsigned long ArduinoRemoteIO::pulseIn (unsigned char *pin,* unsigned char *value,* unsigned long *timeout*) `[private]`

3.1.3.10 unsigned char ArduinoRemoteIO::shiftIn (unsigned char *dataPin,* unsigned char *clockPin,* BitOrder *order*) `[private]`

3.1.3.11 void ArduinoRemoteIO::shiftOut (unsigned char *dataPin,* unsigned char *clockPin,* BitOrder *order,* unsigned char *b*) [private]

3.1.3.12 void ArduinoRemoteIO::tone (unsigned char *pin,* unsigned int *frequency*) [private]

3.1.3.13 void ArduinoRemoteIO::tone (unsigned char *pin,* unsigned int *frequency,* unsigned long *duration*) [private]

The documentation for this class was generated from the following file:

- ArduinoInterface.h

4 File Documentation

4.1 ArduinoInterface.cpp File Reference

4.2 ArduinoInterface.cpp

```
00001
```

4.3 ArduinoInterface.h File Reference

Classes

- class ArduinoRemoteIO

Macros

- #define A0 14
- #define A1 15
- #define A2 16
- #define A3 17
- #define A4 18
- #define A5 19

4.3.1 Macro Definition Documentation

4.3.1.1 #define A0 14

Definition at line 1 of file ArduinoInterface.h.

4.3.1.2 #define A1 15

Definition at line 2 of file ArduinoInterface.h.

4.3.1.3 #define A2 16

Definition at line 3 of file ArduinoInterface.h.

4.3.1.4 #define A3 17

Definition at line 4 of file ArduinoInterface.h.

4.3.1.5 #define A4 18

Definition at line 5 of file ArduinoInterface.h.

4.3.1.6 #define A5 19

Definition at line 6 of file ArduinoInterface.h.

4.4 ArduinoInterface.h

```
00001 #define A0          14
00002 #define A1          15
00003 #define A2          16
00004 #define A3          17
00005 #define A4          18
00006 #define A5          19
00007
00008 class ArduinoRemoteIO {
00009
00010     enum Mode {
00011         INPUT = 0,
00012         OUTPUT = 1
00013     };
00014
00015     enum AnalogReference {
00016         DEFAULT,
00017         INTERNAL,
00018         INTERNAL1V1,
00019         INTERNAL2V56,
00020         EXTERNAL
00021     };
00022
00023     enum BitOrder {
00024         MSBFIRST,
00025         LSBFIRST
00026     };
00027
00035     void pinMode(unsigned char pin, Mode mode);
00036
00063     void digitalWrite(unsigned char pin, bool value);
00064
00068     bool digitalRead(unsigned char pin);
00069
00073     void analogReference(AnalogReference reference);
00074
00078     int analogRead(unsigned char pin);
00079
00083     void analogWrite(unsigned char pin, unsigned char value);
00084
00088     void tone(unsigned char pin, unsigned int frequency);
00089
00093     void tone(unsigned char pin, unsigned int frequency, unsigned long duration);
00094
00098     void noTone(unsigned char pin);
00099
00103     void shiftOut(unsigned char dataPin, unsigned char clockPin,
      BitOrder order, unsigned char b);
00104
00108     unsigned char shiftIn(unsigned char dataPin, unsigned char clockPin,
      BitOrder order);
00109
00113     unsigned long pulseIn(unsigned char pin, unsigned char value);
00114
00118     unsigned long pulseIn(unsigned char pin, unsigned char value, unsigned long timeout);
00119 };
```

Index