Arduino Gyroscope Driver

Generated by Doxygen 1.8.9.1

Tue Aug 18 2015 22:51:49

Contents

1 Class Index 1

1.1 Class List . . . 1

2 File Index 2

2.1 File List . . . 2

3 Class Documentation 2

3.1 IOExpander Class Reference . . . 2

3.1.1 Detailed Description . . . 2

3.2 IOExpanderMCP23X17 Class Reference . . . 2

3.2.1 Detailed Description . . . 3

3.2.2 Member Enumeration Documentation . . . 3

3.2.3 Member Function Documentation . . . 5

3.2.4 Member Data Documentation . . . 7

4 File Documentation 8

4.1 IOExpander.cpp File Reference . . . 8

4.1.1 Macro Definition Documentation . . . 8

4.2 IOExpander.cpp . . . 8

4.3 IOExpander.h File Reference . . . 9

4.4 IOExpander.h . . . 9

4.5 IOExpanderMCP23X17.cpp File Reference . . . 10

4.5.1 Macro Definition Documentation . . . 10

4.6 IOExpanderMCP23X17.cpp . . . 10

4.7 IOExpanderMCP23X17.h File Reference . . . 11

4.7.1 Macro Definition Documentation . . . 12

4.8 IOExpanderMCP23X17.h . . . 13

Index 17

1 Class Index

1.1 Class List

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

IOExpander
Arduino - IO Expander Driver 2

IOExpanderMCP23X17 2

2 File Index

2.1 File List

Here is a list of all files with brief descriptions:

IOExpander.cpp 8

IOExpander.h 9

IOExpanderMCP23X17.cpp 10

IOExpanderMCP23X17.h 11

3 Class Documentation

3.1 IOExpander Class Reference

`#include <IOExpander.h>`

3.1.1 Detailed Description

Arduino - IO Expander Driver.

IoExpander.h

The generic io expander driver functions

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 14 of file IOExpander.h.

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

- IOExpander.h

3.2 IOExpanderMCP23X17 Class Reference

`#include <IOExpanderMCP23X17.h>`

Public Types

- enum SequentialOperationMode { SEQUENTIAL_MODE_ENABLE = 0x00, SEQUENTIAL_MODE_DISABLE = 0xff }
- enum Register {
  IODIRA = 0x00, IODIRB = 0x01, IPOLA = 0x02, IPOLB = 0x03,
  GPINTENA = 0x04, GPINTENB = 0x05, DEFVALA = 0x06, DEFVALB = 0x07,
  INTCONA = 0x08, INTCONB = 0x09, IOCON = 0x0a, GPPUA = 0x0c,
  GPPUB = 0x0d, INTFA = 0x0e, INTFB = 0x0f, INTCAPA = 0x10,
  INTCAPB = 0x11, GPIOA = 0x12, GPIOB = 0x13, OLATA = 0x14,
  OLATB = 0x15 }
- enum Port { PORT_A = GPIOA, PORT_B = GPIOB }
- enum Direction { OUT = 0x00, IN = 0xff }

- enum Mask {
  IOCON_BANK = 0x80, IOCON_MIRROR = 0x40, IOCON_SEQOP = 0x20, IOCON_DISSLW = 0x10,
  IOCON_HAEN = 0x08, IOCON_ODR = 0x04, IOCON_INTPOL = 0x02 }
- enum Pin {
  PIN_A0 = 0, PIN_A1 = 1, PIN_A2 = 2, PIN_A3 = 3,
  PIN_A4 = 4, PIN_A5 = 5, PIN_A6 = 6, PIN_A7 = 7,
  PIN_B0 = 8, PIN_B1 = 9, PIN_B2 = 10, PIN_B3 = 11,
  PIN_B4 = 12, PIN_B5 = 13, PIN_B6 = 14, PIN_B7 = 15 }

Public Member Functions

- void begin (unsigned char device)
- void pinMode (unsigned char pin, bool mode)
- void portMode (Port port, unsigned char mode)
- void digitalWrite (unsigned char pin, bool value)
- bool digitalRead (unsigned char pin)
- void portWrite (Port port, unsigned char value)
- unsigned char portRead (Port port)
- void setPinPullUp (unsigned char pin, bool pullUp)
- void setPinPolarity (unsigned char pin, bool polarity)
- void setPinInterrupt (unsigned char pin, bool interrupt)
- void setSequentialOperationMode (SequentialOperationMode mode)
- void configureRegisterBits (Register reg, unsigned char mask, unsigned char value)
- int writeRegister (Register reg, unsigned char value)
- int readRegister (Register reg)

Private Attributes

- unsigned char device

3.2.1 Detailed Description

Definition at line 29 of file IOExpanderMCP23X17.h.

3.2.2 Member Enumeration Documentation

3.2.2.1 enum IOExpanderMCP23X17::Direction

Enumerator

OUT

IN

Definition at line 71 of file IOExpanderMCP23X17.h.

3.2.2.2 enum IOExpanderMCP23X17::Mask

Enumerator

IOCON_BANK

IOCON_MIRROR

IOCON_SEQOP

IOCON_DISSLW

IOCON_HAEN

IOCON_ODR

IOCON_INTPOL

Definition at line 75 of file IOExpanderMCP23X17.h.

3.2.2.3 enum IOExpanderMCP23X17::Pin

Enumerator

PIN_A0

PIN_A1

PIN_A2

PIN_A3

PIN_A4

PIN_A5

PIN_A6

PIN_A7

PIN_B0

PIN_B1

PIN_B2

PIN_B3

PIN_B4

PIN_B5

PIN_B6

PIN_B7

Definition at line 85 of file IOExpanderMCP23X17.h.

3.2.2.4 enum IOExpanderMCP23X17::Port

Enumerator

PORT_A

PORT_B

Definition at line 67 of file IOExpanderMCP23X17.h.

3.2.2.5 enum IOExpanderMCP23X17::Register

Enumerator

IODIRA

IODIRB

IPOLA

IPOLB

GPINTENA

GPINTENB

DEFVALA

DEFVALB

INTCONA

INTCONB

IOCON

GPPUA

GPPUB

INTFA

INTFB

INTCAPA

INTCAPB

GPIOA

GPIOB

OLATA

OLATB

Definition at line 42 of file IOExpanderMCP23X17.h.

3.2.2.6 enum IOExpanderMCP23X17::SequentialOperationMode

Enumerator

SEQUENTIAL_MODE_ENABLE

SEQUENTIAL_MODE_DISABLE

Definition at line 38 of file IOExpanderMCP23X17.h.

3.2.3 Member Function Documentation

3.2.3.1 void IOExpanderMCP23X17::begin (unsigned char *device*)

Begins the IO expander divice.

Parameters

device	The device address, or just the last 3 pins combination;

Definition at line 16 of file IOExpanderMCP23X17.cpp.

3.2.3.2 void IOExpanderMCP23X17::configureRegisterBits (Register *reg*, unsigned char *mask*, unsigned char *value*)

Configures a registers.

Parameters

reg	The register number.
mask	The mask to be used.
v	The value to be used.

Definition at line 41 of file IOExpanderMCP23X17.cpp.

3.2.3.3 bool IOExpanderMCP23X17::digitalRead (unsigned char *pin*)

Reads the value from a specified pin, either HIGH or LOW.

Parameters

pin	The pin number.

Definition at line 36 of file IOExpanderMCP23X17.cpp.

3.2.3.4 void IOExpanderMCP23X17::digitalWrite (unsigned char *pin*, bool *value*)

Write a HIGH or a LOW value to a pin.

Parameters

pin	The pin number.
value	LOW or WRITE.

Definition at line 31 of file IOExpanderMCP23X17.cpp.

3.2.3.5 void IOExpanderMCP23X17::pinMode (unsigned char *pin*, bool *mode*)

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

Parameters

pin	The pin number.
mode	1 means input, 0 means output.

Definition at line 21 of file IOExpanderMCP23X17.cpp.

3.2.3.6 void IOExpanderMCP23X17::portMode (Port *port*, unsigned char *mode*)

Configures a port to behave either as an input or an output.

Parameters

port	The port.
mode	The mode.

Definition at line 26 of file IOExpanderMCP23X17.cpp.

3.2.3.7 unsigned char IOExpanderMCP23X17::portRead (Port *port*) [inline]

Reads a port.

Parameters

port	The port to write.

Returns

The value associated with the port.

Definition at line 158 of file IOExpanderMCP23X17.h.

3.2.3.8 void IOExpanderMCP23X17::portWrite (Port *port*, unsigned char *value*) [inline]

Writes value to a port.

Parameters

port	The port to write.
value	The value to write to the port.

Definition at line 148 of file IOExpanderMCP23X17.h.

3.2.3.9 int IOExpanderMCP23X17::readRegister (Register *reg*)

Reades a value from a register.

Parameters

reg	The register number.

Returns

The register value.

Definition at line 56 of file IOExpanderMCP23X17.cpp.

3.2.3.10 void IOExpanderMCP23X17::setPinInterrupt (unsigned char *pin*, bool *interrupt*)

Configures a pin to clear or set the interrupt.

Parameters

pin	The pin number.
pullUp	0 means interrupt disable, 1 means interrupt enable.

Definition at line 84 of file IOExpanderMCP23X17.cpp.

3.2.3.11 void IOExpanderMCP23X17::setPinPolarity (unsigned char *pin*, bool *polarity*)

Configures the polarity for a given pin.

Parameters

pin	The pin number.
pullUp	0 means normal, 1 means inverted polarity.

Definition at line 79 of file IOExpanderMCP23X17.cpp.

3.2.3.12 void IOExpanderMCP23X17::setPinPullUp (unsigned char *pin*, bool *pullUp*)

Configures the pullup resistor for a given pin.

Parameters

pin	The pin number.
pullUp	0 means with, 1 means witout pullup.

Definition at line 74 of file IOExpanderMCP23X17.cpp.

3.2.3.13 void IOExpanderMCP23X17::setSequentialOperationMode (SequentialOperationMode *mode*)

Configures the sequential/continuous operation mode.

Parameters

mode	The operation mode.

3.2.3.14 int IOExpanderMCP23X17::writeRegister (Register *reg*, unsigned char *value*)

Writes a value to a register.

Parameters

reg	The register number.
v	The value to be used.

Definition at line 49 of file IOExpanderMCP23X17.cpp.

3.2.4 Member Data Documentation

3.2.4.1 unsigned char IOExpanderMCP23X17::device [private]

I2C device address.

Definition at line 34 of file IOExpanderMCP23X17.h.

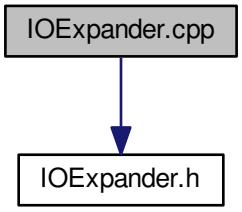
The documentation for this class was generated from the following files:

- IOExpanderMCP23X17.h
- IOExpanderMCP23X17.cpp

4 File Documentation

4.1 IOExpander.cpp File Reference

```
#include "IOExpander.h"
```
Include dependency graph for IOExpander.cpp:



Macros

- #define __ARDUINO_DRIVER_IO_EXPANDER_CPP__ 1

4.1.1 Macro Definition Documentation

4.1.1.1 #define __ARDUINO_DRIVER_IO_EXPANDER_CPP__ 1

Arduino - IO Expander Driver.

IoExpander.cpp

The generic io expander driver functions

Author

Dalmir da Silva dalmirdasilva@gmail.com

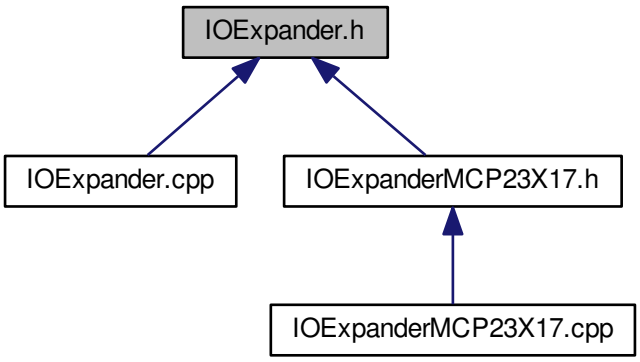
Definition at line 12 of file IOExpander.cpp.

4.2 IOExpander.cpp

```
00001
00011 #ifndef __ARDUINO_DRIVER_IO_EXPANDER_CPP__
00012 #define __ARDUINO_DRIVER_IO_EXPANDER_CPP__ 1
00013
00014 #include "IOExpander.h"
00015
00016 #endif /* __ARDUINO_DRIVER_IO_EXPANDER_CPP__ */
```

4.3 IOExpander.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

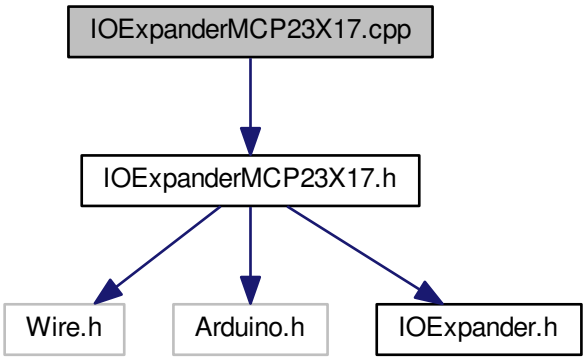
- class IOExpander

4.4 IOExpander.h

```
00001
00011 #ifndef __ARDUINO_DRIVER_IO_EXPANDER_H__
00012 #define __ARDUINO_DRIVER_IO_EXPANDER_H__ 1
00013
00014 class IOExpander {
00015 };
00016
00017 #endif /* __ARDUINO_DRIVER_IO_EXPANDER_H__ */
```

4.5 IOExpanderMCP23X17.cpp File Reference

`#include "IOExpanderMCP23X17.h"`
Include dependency graph for IOExpanderMCP23X17.cpp:



Macros

- #define __ARDUINO_DRIVER_IO_EXPANDER_MCP23X17_CPP__ 1

4.5.1 Macro Definition Documentation

4.5.1.1 #define __ARDUINO_DRIVER_IO_EXPANDER_MCP23X17_CPP__ 1

Arduino - IO Expander Driver.

IoExpanderMcp23x17.cpp

The MCP23X17 driver functions

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file IOExpanderMCP23X17.cpp.

4.6 IOExpanderMCP23X17.cpp

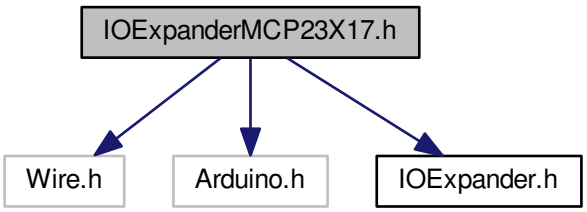
```
00001
00011 #ifndef __ARDUINO_DRIVER_IO_EXPANDER_MCP23X17_CPP__
00012 #define __ARDUINO_DRIVER_IO_EXPANDER_MCP23X17_CPP__ 1
00013
00014 #include "IOExpanderMCP23X17.h"
00015
00016 void IOExpanderMCP23X17::begin(unsigned char device) {
00017     this->device = 0x20 | (device & 0x07);
00018     Wire.begin();
00019 }
00020
00021 void IOExpanderMCP23X17::pinMode(unsigned char pin, bool mode) {
00022     Register reg = IO_EXP_PIN_TO_IODIR_REG(pin);
00023     configureRegisterBits(reg, (1 << (pin % 8)), ((mode) ? 0xff : 0x00));
00024 }
00025
```

```
00026 void IOExpanderMCP23X17::portMode(Port port, unsigned char mode) {
00027     Register reg = IO_EXP_PORT_TO_IODIR_REG(port);
00028     writeRegister(reg, mode);
00029 }
00030
00031 void IOExpanderMCP23X17::digitalWrite(unsigned char pin, bool value) {
00032     Register reg = IO_EXP_PIN_TO_GPIO_REG(pin);
00033     configureRegisterBits(reg, (1 << (pin % 8)), ((value) ? 0xff : 0x00));
00034 }
00035
00036 bool IOExpanderMCP23X17::digitalRead(unsigned char pin) {
00037     Register reg = IO_EXP_PIN_TO_GPIO_REG(pin);
00038     return (bool) (readRegister(reg) & (1 << (pin % 8)));
00039 }
00040
00041 void IOExpanderMCP23X17::configureRegisterBits(
      Register reg, unsigned char mask, unsigned char v) {
00042     unsigned char n;
00043     n = readRegister(reg);
00044     n &= ~(mask);
00045     n |= v & mask;
00046     writeRegister(reg, n);
00047 }
00048
00049 int IOExpanderMCP23X17::writeRegister(Register reg, unsigned char
      v) {
00050     Wire.beginTransmission(device);
00051     Wire.write((unsigned char) reg);
00052     Wire.write(v);
00053     return Wire.endTransmission();
00054 }
00055
00056 int IOExpanderMCP23X17::readRegister(Register reg) {
00057     char tries = 10;
00058     Wire.beginTransmission(device);
00059     Wire.write((unsigned char) reg);
00060     char status = Wire.endTransmission(false);
00061     if (status != 0) {
00062         return -(status);
00063     }
00064     Wire.requestFrom(device, (unsigned char) 1);
00065     while (!Wire.available() && --tries > 0) {
00066         delayMicroseconds(1);
00067     }
00068     if (tries == 0) {
00069         return -5;
00070     }
00071     return Wire.read();
00072 }
00073
00074 void IOExpanderMCP23X17::setPinPullUp(unsigned char pin, bool pullUp) {
00075     Register reg = IO_EXP_PIN_TO_GPPU_REG(pin);
00076     configureRegisterBits(reg, (1 << (pin % 8)), ((pullUp) ? 0xff : 0x00));
00077 }
00078
00079 void IOExpanderMCP23X17::setPinPolarity(unsigned char pin, bool polarity)
       {
00080     Register reg = IO_EXP_PIN_TO_IPOL_REG(pin);
00081     configureRegisterBits(reg, (1 << (pin % 8)), ((polarity) ? 0xff : 0x00));
00082 }
00083
00084 void IOExpanderMCP23X17::setPinInterrupt(unsigned char pin, bool
      interrupt) {
00085     Register reg = IO_EXP_PIN_TO_GPINTEN_REG(pin);
00086     configureRegisterBits(reg, (1 << (pin % 8)), ((interrupt) ? 0xff : 0x00));
00087 }
00088
00089 #endif /* __ARDUINO_DRIVER_IO_EXPANDER_MCP23X17_CPP__ */
```

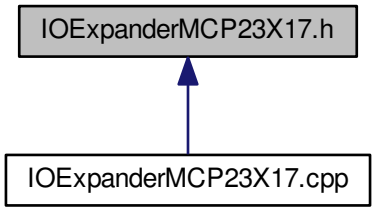
4.7 IOExpanderMCP23X17.h File Reference

```
#include <Wire.h>
#include <Arduino.h>
#include <IOExpander.h>
```

Include dependency graph for IOExpanderMCP23X17.h:



This graph shows which files directly or indirectly include this file:



Classes

- class IOExpanderMCP23X17

Macros

- #define IO_EX_MAX_PINS 16
- #define IO_EXP_NORMALIZE_PIN(p) (((p) >= IO_EX_MAX_PINS ? IO_EX_MAX_PINS - 1 : (((p) < 0) ? 0 : (p)))
- #define IO_EXP_IS_PIN_VALID(p) ((p) < IO_EX_MAX_IO && (p) >= 0)
- #define IO_EXP_PIN_TO_GPIO_REG(p) ((p) < 8 ? GPIOA : GPIOB)
- #define IO_EXP_PIN_TO_IODIR_REG(p) ((p) < 8 ? IODIRA : IODIRB)
- #define IO_EXP_PORT_TO_IODIR_REG(p) ((p) == PORT_A ? IODIRA : IODIRB)
- #define IO_EXP_PIN_TO_GPPU_REG(p) ((p) < 8 ? GPPUA : GPPUB)
- #define IO_EXP_PIN_TO_IPOL_REG(p) ((p) < 8 ? IPOLA : IPOLB)
- #define IO_EXP_PIN_TO_GPINTEN_REG(p) ((p) < 8 ? GPINTENA : GPINTENB)

4.7.1 Macro Definition Documentation

4.7.1.1 #define IO_EX_MAX_PINS 16

Arduino - IO Expander Driver.

IoExpanderMcp23x17.h

The MCP23X17 driver functions

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 18 of file IOExpanderMCP23X17.h.

4.7.1.2 #define IO_EXP_IS_PIN_VALID(p) ((p) < IO_EX_MAX_IO && (p) >= 0)

Definition at line 21 of file IOExpanderMCP23X17.h.

4.7.1.3 #define IO_EXP_NORMALIZE_PIN(p) (((p) >= IO_EX_MAX_PINS ? IO_EX_MAX_PINS - 1 : (((p) < 0) ? 0 : (p)))

Definition at line 20 of file IOExpanderMCP23X17.h.

4.7.1.4 #define IO_EXP_PIN_TO_GPINTEN_REG(p) ((p) < 8 ? GPINTENA : GPINTENB)

Definition at line 27 of file IOExpanderMCP23X17.h.

4.7.1.5 #define IO_EXP_PIN_TO_GPIO_REG(p) ((p) < 8 ? GPIOA : GPIOB)

Definition at line 22 of file IOExpanderMCP23X17.h.

4.7.1.6 #define IO_EXP_PIN_TO_GPPU_REG(p) ((p) < 8 ? GPPUA : GPPUB)

Definition at line 25 of file IOExpanderMCP23X17.h.

4.7.1.7 #define IO_EXP_PIN_TO_IODIR_REG(p) ((p) < 8 ? IODIRA : IODIRB)

Definition at line 23 of file IOExpanderMCP23X17.h.

4.7.1.8 #define IO_EXP_PIN_TO_IPOL_REG(p) ((p) < 8 ? IPOLA : IPOLB)

Definition at line 26 of file IOExpanderMCP23X17.h.

4.7.1.9 #define IO_EXP_PORT_TO_IODIR_REG(p) ((p) == PORT_A ? IODIRA : IODIRB)

Definition at line 24 of file IOExpanderMCP23X17.h.

4.8 IOExpanderMCP23X17.h

```
00001
00011 #ifndef __ARDUINO_DRIVER_IO_EXPANDER_MCP23X17_H__
00012 #define __ARDUINO_DRIVER_IO_EXPANDER_MCP23X17_H__ 1
00013
00014 #include <Wire.h>
00015 #include <Arduino.h>
00016 #include <IOExpander.h>
00017
00018 #define IO_EX_MAX_PINS                          16
00019
00020 #define IO_EXP_NORMALIZE_PIN(p)                 (((p) >= IO_EX_MAX_PINS ? IO_EX_MAX_PINS - 1 : (((p) < 0) ?
       0 : (p)))
00021 #define IO_EXP_IS_PIN_VALID(p)                  ((p) < IO_EX_MAX_IO && (p) >= 0)
00022 #define IO_EXP_PIN_TO_GPIO_REG(p)               ((p) < 8 ? GPIOA : GPIOB)
00023 #define IO_EXP_PIN_TO_IODIR_REG(p)              ((p) < 8 ? IODIRA : IODIRB)
00024 #define IO_EXP_PORT_TO_IODIR_REG(p)             ((p) == PORT_A ? IODIRA : IODIRB)
00025 #define IO_EXP_PIN_TO_GPPU_REG(p)               ((p) < 8 ? GPPUA : GPPUB)
00026 #define IO_EXP_PIN_TO_IPOL_REG(p)               ((p) < 8 ? IPOLA : IPOLB)
00027 #define IO_EXP_PIN_TO_GPINTEN_REG(p)            ((p) < 8 ? GPINTENA : GPINTENB)
00028
00029 class IOExpanderMCP23X17 {
00030
00034     unsigned char device;
00035
00036 public:
```

```
00037
00038     enum SequentialOperationMode {
00039         SEQUENTIAL_MODE_ENABLE = 0x00,
      SEQUENTIAL_MODE_DISABLE = 0xff,
00040     };
00041
00042     enum Register {
00043         IODIRA = 0x00,
00044         IODIRB = 0x01,
00045         IPOLA = 0x02,
00046         IPOLB = 0x03,
00047         GPINTENA = 0x04,
00048         GPINTENB = 0x05,
00049         DEFVALA = 0x06,
00050         DEFVALB = 0x07,
00051         INTCONA = 0x08,
00052         INTCONB = 0x09,
00053         IOCON = 0x0a,
00054 //         IOCON = 0x0b,
00055         GPPUA = 0x0c,
00056         GPPUB = 0x0d,
00057         INTFA = 0x0e,
00058         INTFB = 0x0f,
00059         INTCAPA = 0x10,
00060         INTCAPB = 0x11,
00061         GPIOA = 0x12,
00062         GPIOB = 0x13,
00063         OLATA = 0x14,
00064         OLATB = 0x15
00065     };
00066
00067     enum Port {
00068         PORT_A = GPIOA, PORT_B = GPIOB
00069     };
00070
00071     enum Direction {
00072         OUT = 0x00, IN = 0xff
00073     };
00074
00075     enum Mask {
00076         IOCON_BANK = 0x80,
00077         IOCON_MIRROR = 0x40,
00078         IOCON_SEQOP = 0x20,
00079         IOCON_DISSLW = 0x10,
00080         IOCON_HAEN = 0x08,
00081         IOCON_ODR = 0x04,
00082         IOCON_INTPOL = 0x02
00083     };
00084
00085     enum Pin {
00086         PIN_A0 = 0,
00087         PIN_A1 = 1,
00088         PIN_A2 = 2,
00089         PIN_A3 = 3,
00090         PIN_A4 = 4,
00091         PIN_A5 = 5,
00092         PIN_A6 = 6,
00093         PIN_A7 = 7,
00094         PIN_B0 = 8,
00095         PIN_B1 = 9,
00096         PIN_B2 = 10,
00097         PIN_B3 = 11,
00098         PIN_B4 = 12,
00099         PIN_B5 = 13,
00100         PIN_B6 = 14,
00101         PIN_B7 = 15
00102     };
00103
00109     void begin(unsigned char device);
00110
00117     void pinMode(unsigned char pin, bool mode);
00118
00125     void portMode(Port port, unsigned char mode);
00126
00133     void digitalWrite(unsigned char pin, bool value);
00134
00140     bool digitalRead(unsigned char pin);
00141
00148     void inline portWrite(Port port, unsigned char value) {
00149         writeRegister((Register) port, value);
00150     }
00151
00158     unsigned char inline portRead(Port port) {
00159         return readRegister((Register) port);
00160     }
00161
00168     void setPinPullUp(unsigned char pin, bool pullUp);
```

```
00169
00176     void setPinPolarity(unsigned char pin, bool polarity);
00177
00184     void setPinInterrupt(unsigned char pin, bool interrupt);
00185
00191     void setSequentialOperationMode(
      SequentialOperationMode mode);
00192
00200     void configureRegisterBits(Register reg, unsigned char mask,
00201             unsigned char value);
00202
00209     int writeRegister(Register reg, unsigned char value);
00210
00217     int readRegister(Register reg);
00218 };
00219
00220 #endif /* __ARDUINO_DRIVER_IO_EXPANDER_MCP23X17_H__ */
```

Index

__ARDUINO_DRIVER_IO_EXPANDER_CPP__
    IOExpander.cpp, 8
__ARDUINO_DRIVER_IO_EXPANDER_MCP23X17_CPP__
    IOExpanderMCP23X17.cpp, 10

begin
    IOExpanderMCP23X17, 5

configureRegisterBits
    IOExpanderMCP23X17, 5

DEFVALA
    IOExpanderMCP23X17, 4
DEFVALB
    IOExpanderMCP23X17, 4
device
    IOExpanderMCP23X17, 7
digitalRead
    IOExpanderMCP23X17, 5
digitalWrite
    IOExpanderMCP23X17, 5
Direction
    IOExpanderMCP23X17, 3

GPINTENA
    IOExpanderMCP23X17, 4
GPINTENB
    IOExpanderMCP23X17, 4
GPIOA
    IOExpanderMCP23X17, 5
GPIOB
    IOExpanderMCP23X17, 5
GPPUA
    IOExpanderMCP23X17, 4
GPPUB
    IOExpanderMCP23X17, 5

IN
    IOExpanderMCP23X17, 3
INTCAPA
    IOExpanderMCP23X17, 5
INTCAPB
    IOExpanderMCP23X17, 5
INTCONA
    IOExpanderMCP23X17, 4
INTCONB
    IOExpanderMCP23X17, 4
INTFA
    IOExpanderMCP23X17, 5
INTFB
    IOExpanderMCP23X17, 5
IO_EX_MAX_PINS
    IOExpanderMCP23X17.h, 12
IO_EXP_IS_PIN_VALID
    IOExpanderMCP23X17.h, 13
IO_EXP_NORMALIZE_PIN
    IOExpanderMCP23X17.h, 13
IO_EXP_PIN_TO_GPINTEN_REG
    IOExpanderMCP23X17.h, 13
IO_EXP_PIN_TO_GPIO_REG
    IOExpanderMCP23X17.h, 13
IO_EXP_PIN_TO_GPPU_REG
    IOExpanderMCP23X17.h, 13
IO_EXP_PIN_TO_IODIR_REG
    IOExpanderMCP23X17.h, 13
IO_EXP_PIN_TO_IPOL_REG
    IOExpanderMCP23X17.h, 13
IO_EXP_PORT_TO_IODIR_REG
    IOExpanderMCP23X17.h, 13
IOCON
    IOExpanderMCP23X17, 4
IOCON_BANK
    IOExpanderMCP23X17, 3
IOCON_DISSLW
    IOExpanderMCP23X17, 3
IOCON_HAEN
    IOExpanderMCP23X17, 3
IOCON_INTPOL
    IOExpanderMCP23X17, 4
IOCON_MIRROR
    IOExpanderMCP23X17, 3
IOCON_ODR
    IOExpanderMCP23X17, 3
IOCON_SEQOP
    IOExpanderMCP23X17, 3
IODIRA
    IOExpanderMCP23X17, 4
IODIRB
    IOExpanderMCP23X17, 4
IOExpander, 2
IOExpander.cpp, 8
    __ARDUINO_DRIVER_IO_EXPANDER_CPP__, 8
IOExpander.h, 9
IOExpanderMCP23X17, 2
    begin, 5
    configureRegisterBits, 5
    DEFVALA, 4
    DEFVALB, 4
    device, 7
    digitalRead, 5
    digitalWrite, 5
    Direction, 3
    GPINTENA, 4
    GPINTENB, 4
    GPIOA, 5
    GPIOB, 5
    GPPUA, 4
    GPPUB, 5
    IN, 3
    INTCAPA, 5

INTCAPB, 5
INTCONA, 4
INTCONB, 4
INTFA, 5
INTFB, 5
IOCON, 4
IOCON_BANK, 3
IOCON_DISSLW, 3
IOCON_HAEN, 3
IOCON_INTPOL, 4
IOCON_MIRROR, 3
IOCON_ODR, 3
IOCON_SEQOP, 3
IODIRA, 4
IODIRB, 4
IPOLA, 4
IPOLB, 4
Mask, 3
OLATA, 5
OLATB, 5
OUT, 3
PIN_A0, 4
PIN_A1, 4
PIN_A2, 4
PIN_A3, 4
PIN_A4, 4
PIN_A5, 4
PIN_A6, 4
PIN_A7, 4
PIN_B0, 4
PIN_B1, 4
PIN_B2, 4
PIN_B3, 4
PIN_B4, 4
PIN_B5, 4
PIN_B6, 4
PIN_B7, 4
PORT_A, 4
PORT_B, 4
Pin, 4
pinMode, 6
Port, 4
portMode, 6
portRead, 6
portWrite, 6
readRegister, 6
Register, 4
SEQUENTIAL_MODE_DISABLE, 5
SEQUENTIAL_MODE_ENABLE, 5
SequentialOperationMode, 5
setPinInterrupt, 7
setPinPolarity, 7
setPinPullUp, 7
setSequentialOperationMode, 7
writeRegister, 7
IOExpanderMCP23X17.cpp, 10
__ARDUINO_DRIVER_IO_EXPANDER_MC↩
P23X17_CPP__, 10
IOExpanderMCP23X17.h, 11, 13
IO_EX_MAX_PINS, 12
IO_EXP_IS_PIN_VALID, 13
IO_EXP_NORMALIZE_PIN, 13
IO_EXP_PIN_TO_GPINTEN_REG, 13
IO_EXP_PIN_TO_GPIO_REG, 13
IO_EXP_PIN_TO_GPPU_REG, 13
IO_EXP_PIN_TO_IODIR_REG, 13
IO_EXP_PIN_TO_IPOL_REG, 13
IO_EXP_PORT_TO_IODIR_REG, 13
IPOLA
IOExpanderMCP23X17, 4
IPOLB
IOExpanderMCP23X17, 4

Mask
IOExpanderMCP23X17, 3

OLATA
IOExpanderMCP23X17, 5
OLATB
IOExpanderMCP23X17, 5
OUT
IOExpanderMCP23X17, 3

PIN_A0
IOExpanderMCP23X17, 4
PIN_A1
IOExpanderMCP23X17, 4
PIN_A2
IOExpanderMCP23X17, 4
PIN_A3
IOExpanderMCP23X17, 4
PIN_A4
IOExpanderMCP23X17, 4
PIN_A5
IOExpanderMCP23X17, 4
PIN_A6
IOExpanderMCP23X17, 4
PIN_A7
IOExpanderMCP23X17, 4
PIN_B0
IOExpanderMCP23X17, 4
PIN_B1
IOExpanderMCP23X17, 4
PIN_B2
IOExpanderMCP23X17, 4
PIN_B3
IOExpanderMCP23X17, 4
PIN_B4
IOExpanderMCP23X17, 4
PIN_B5
IOExpanderMCP23X17, 4
PIN_B6
IOExpanderMCP23X17, 4
PIN_B7
IOExpanderMCP23X17, 4
PORT_A
IOExpanderMCP23X17, 4

PORT_B
    IOExpanderMCP23X17, 4
Pin
    IOExpanderMCP23X17, 4
pinMode
    IOExpanderMCP23X17, 6
Port
    IOExpanderMCP23X17, 4
portMode
    IOExpanderMCP23X17, 6
portRead
    IOExpanderMCP23X17, 6
portWrite
    IOExpanderMCP23X17, 6

readRegister
    IOExpanderMCP23X17, 6
Register
    IOExpanderMCP23X17, 4

SEQUENTIAL_MODE_DISABLE
    IOExpanderMCP23X17, 5
SEQUENTIAL_MODE_ENABLE
    IOExpanderMCP23X17, 5
SequentialOperationMode
    IOExpanderMCP23X17, 5
setPinInterrupt
    IOExpanderMCP23X17, 7
setPinPolarity
    IOExpanderMCP23X17, 7
setPinPullUp
    IOExpanderMCP23X17, 7
setSequentialOperationMode
    IOExpanderMCP23X17, 7

writeRegister
    IOExpanderMCP23X17, 7