StepMotor library for Arduino & ARM Cortex-M3 (STM32)

Generated by Doxygen 1.8.17

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 StepMotor Class Reference

StepMotor Class.

`#include <StepMotor.h>`

Public Member Functions

- StepMotor (SM_motortype_t motorType, SM_torqueforce_t torqueForce, uint8_t in1Pin, uint8_t in2Pin, uint8_t in3Pin, uint8_t in4Pin)

  StepMotor Class constructor.
- virtual void begin ()

  Initialize StepMotor controller pins.
- virtual void end ()

  Release StepMotor controller pins.
- virtual void setMov (uint16_t nSteps, SM_stepdelay_t delay_ms, SM_direction_t direction)

  Rotate StepMotor by nSteps at the expected direction.

Private Member Functions

- void _setMotorType (SM_motortype_t motorType)

  Set StepMotor type to be used controlled.
- void _setTorqueForce (SM_torqueforce_t torqueForce)

  Set torque force type to be used by StepMotor.
- void _control_stepCmd (uint8_t *stepSequenceMatrix, bool is4stepMatrix, uint16_t nSteps, SM_stepdelay_t delay_ms)

  Control sequenced steps & speed applied to the StepMotor.
- void _set_stepCmd (uint8_t nibble_cmd)

  Set a step command to the StepMotor pins.

Private Attributes

- SM_motortype_t _motorType
  Defines StepMotor motor type.
- SM_torqueforce_t _torqueForce
  Defines StepMotor torque force.
- uint8_t _pin1Port
  PORT register for pin1.
- uint8_t _pin2Port
  PORT register for pin2.
- uint8_t _pin3Port
  PORT register for pin3.
- uint8_t _pin4Port
  PORT register for pin4.
- uint8_t _pin1PortBit
  Bit number in IO register for pin1.
- uint8_t _pin2PortBit
  Bit number in IO register for pin2.
- uint8_t _pin3PortBit
  Bit number in IO register for pin3.
- uint8_t _pin4PortBit
  Bit number in IO register for pin4.

3.1.1 Detailed Description

StepMotor Class.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 StepMotor()

```
StepMotor::StepMotor (
            SM_motortype_t motorType,
            SM_torqueforce_t torqueForce,
            uint8_t in1Pin,
            uint8_t in2Pin,
            uint8_t in3Pin,
            uint8_t in4Pin )
```

StepMotor Class constructor.

3.1.3 Member Function Documentation

3.1.3.1 _control_stepCmd()

```
void StepMotor::_control_stepCmd (
            uint8_t * stepSequenceMatrix,
            bool is4stepMatrix,
            uint16_t nSteps,
            SM_stepdelay_t delay_ms )  [private]
```

Control sequenced steps & speed applied to the StepMotor.

3.1.3.2 _set_stepCmd()

```
void StepMotor::_set_stepCmd (
            uint8_t nibble_cmd )  [private]
```

Set a step command to the StepMotor pins.

3.1.3.3 _setMotorType()

```
void StepMotor::_setMotorType (
            SM_motortype_t motorType )  [private]
```

Set StepMotor type to be used controlled.

3.1.3.4 _setTorqueForce()

```
void StepMotor::_setTorqueForce (
            SM_torqueforce_t torqueForce )  [private]
```

Set torque force type to be used by StepMotor.

3.1.3.5 begin()

```
void StepMotor::begin ( )  [virtual]
```

Initialize StepMotor controller pins.

3.1.3.6 end()

```
void StepMotor::end ( )  [virtual]
```

Release StepMotor controller pins.

3.1.3.7 setMov()

```
void StepMotor::setMov (
            uint16_t nSteps,
            SM_stepdelay_t delay_ms,
            SM_direction_t direction )  [virtual]
```

Rotate StepMotor by nSteps at the expected direction.

- Following table documents how internal and external variables are mapped to set an adequate movement command.

_motorType	_torqueForce	direction	cmd (BIN)
UNI_4PHASE	MIN_TORQUE	CLK	0b0000
UNI_4PHASE	MIN_TORQUE	CTR_CLK	0b0001
UNI_4PHASE	MAX_TORQUE	CLK	0b0010
UNI_4PHASE	MAX_TORQUE	CTR_CLK	0b0011
BI_2PHASE	MIN_TORQUE	CLK	0b0100
BI_2PHASE	MIN_TORQUE	CTR_CLK	0b0101
BI_2PHASE	MAX_TORQUE	CLK	0b0110
BI_2PHASE	MAX_TORQUE	CTR_CLK	0b0111

3.1.4 Member Data Documentation

3.1.4.1 _motorType

```
SM_motortype_t StepMotor::_motorType  [private]
```

Defines StepMotor motor type.

3.1.4.2 _pin1Port

```
uint8_t StepMotor::_pin1Port  [private]
```

PORT register for pin1.

3.1.4.3 _pin1PortBit

```
uint8_t StepMotor::_pin1PortBit  [private]
```

Bit number in IO register for pin1.

3.1.4.4 _pin2Port

```
uint8_t StepMotor::_pin2Port  [private]
```

PORT register for pin2.

3.1.4.5 _pin2PortBit

```
uint8_t StepMotor::_pin2PortBit  [private]
```

Bit number in IO register for pin2.

3.1.4.6 _pin3Port

```
uint8_t StepMotor::_pin3Port  [private]
```

PORT register for pin3.

3.1.4.7 _pin3PortBit

```
uint8_t StepMotor::_pin3PortBit  [private]
```

Bit number in IO register for pin3.

3.1.4.8 _pin4Port

```
uint8_t StepMotor::_pin4Port  [private]
```

PORT register for pin4.

3.1.4.9 _pin4PortBit

```
uint8_t StepMotor::_pin4PortBit  [private]
```

Bit number in IO register for pin4.

3.1.4.10 _torqueForce

```
SM_torqueforce_t StepMotor::_torqueForce  [private]
```

Defines StepMotor torque force.

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

- StepMotor.h
- StepMotor.cpp

Chapter 4

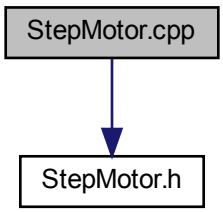
File Documentation

4.1 StepMotor.cpp File Reference

StepMotor library source code for Arduino & ARM Cortex-M3 (STM32)

```
#include "StepMotor.h"
```

Include dependency graph for StepMotor.cpp:



4.1.1 Detailed Description

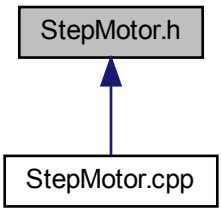
StepMotor library source code for Arduino & ARM Cortex-M3 (STM32)

Source: https://github.com/1lucas1gabriel/StepMotor

4.2 StepMotor.h File Reference

StepMotor library header file for Arduino & ARM Cortex-M3 (STM32)

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



Classes

- class StepMotor

  StepMotor Class.

Enumerations

- enum SM_motortype_t { UNIPOLAR_4PHASE = 0b00, BIPOLAR_2PHASE = 0b01 }
- enum SM_torqueforce_t { MIN_TORQUE = 0b00, MAX_TORQUE = 0b01 }
- enum SM_direction_t { CLOCKWISE = 0b00, COUNTER_CLOCKWISE = 0b01 }
- enum SM_stepdelay_t { SLOW_MS = 30, MEDIUM_MS = 15, FAST_MS = 5 }

Variables

- static const uint8_t uni_4phase_fullstep_maxtorque_clk [4] = {0x9, 0x3, 0x6, 0xC}
- static const uint8_t uni_4phase_fullstep_maxtorque_ctr_clk [4] = {0xC, 0x6, 0x3, 0x9}
- static const uint8_t uni_4phase_fullstep_mintorque_clk [4] = {0x8, 0x1, 0x2, 0x4}
- static const uint8_t uni_4phase_fullstep_mintorque_ctr_clk [4] = {0x4, 0x2, 0x1, 0x8}
- static const uint8_t bi_2phase_fullstep_maxtorque_clk [4] = {0xA, 0x9, 0x5, 0x6}
- static const uint8_t bi_2phase_fullstep_maxtorque_ctr_clk [4] = {0x6, 0x5, 0x9, 0xA}
- static const uint8_t bi_2phase_fullstep_mintorque_clk [4] = {0x8, 0x1, 0x4, 0x2}
- static const uint8_t bi_2phase_fullstep_mintorque_ctr_clk [4] = {0x2, 0x4, 0x1, 0x8}

4.2.1 Detailed Description

StepMotor library header file for Arduino & ARM Cortex-M3 (STM32)

Source: https://github.com/1lucas1gabriel/StepMotor

```
*   Required mapping between uC Port and Step Motor connection:
*
*   0. One Single PORT must be used to connect all 4 pins.
*   1. Pin connection must be sequenced over a same PORT.
*  2. Written nibble to a Port refers to a step command.
*
*
*  Pin connection example
*
*                       ----------------------------------------
*  uC: 8 LSB Port       | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
*                       ----------------------------------------
*   StepMotor:                          ---------------------
*  pin connection (nibble_cmd)          | In1| In2| In3| In4|
*                                       ---------------------
*
```

4.2.2 Enumeration Type Documentation

4.2.2.1 SM_direction_t

`enum SM_direction_t`

Defines rotational direction for the StepMotor.

Enumerator

CLOCKWISE	
COUNTER_CLOCKWISE	

4.2.2.2 SM_motortype_t

`enum SM_motortype_t`

Defines the StepMotor type used.

Enumerator

UNIPOLAR_4PHASE	
BIPOLAR_2PHASE	

4.2.2.3 SM_stepdelay_t

`enum SM_stepdelay_t`

Defines delay time applied to the StepMotor between each step command.

Enumerator

SLOW_MS	
MEDIUM_MS	
FAST_MS	

4.2.2.4 SM_torqueforce_t

```
enum SM_torqueforce_t
```

Defines torque force type applied by the StepMotor.

Enumerator

MIN_TORQUE	
MAX_TORQUE	

4.2.3 Variable Documentation

4.2.3.1 bi_2phase_fullstep_maxtorque_clk

```
const uint8_t bi_2phase_fullstep_maxtorque_clk[4] = {0xA, 0x9, 0x5, 0x6}  [static]
```

4.2.3.2 bi_2phase_fullstep_maxtorque_ctr_clk

```
const uint8_t bi_2phase_fullstep_maxtorque_ctr_clk[4] = {0x6, 0x5, 0x9, 0xA}  [static]
```

4.2.3.3 bi_2phase_fullstep_mintorque_clk

```
const uint8_t bi_2phase_fullstep_mintorque_clk[4] = {0x8, 0x1, 0x4, 0x2}  [static]
```

4.2.3.4 bi_2phase_fullstep_mintorque_ctr_clk

```
const uint8_t bi_2phase_fullstep_mintorque_ctr_clk[4] = {0x2, 0x4, 0x1, 0x8}  [static]
```

4.2.3.5 uni_4phase_fullstep_maxtorque_clk

```
const uint8_t uni_4phase_fullstep_maxtorque_clk[4] = {0x9, 0x3, 0x6, 0xC}  [static]
```

4.2.3.6 uni_4phase_fullstep_maxtorque_ctr_clk

```
const uint8_t uni_4phase_fullstep_maxtorque_ctr_clk[4] = {0xC, 0x6, 0x3, 0x9}  [static]
```

4.2.3.7 uni_4phase_fullstep_mintorque_clk

```
const uint8_t uni_4phase_fullstep_mintorque_clk[4] = {0x8, 0x1, 0x2, 0x4}  [static]
```

4.2.3.8 uni_4phase_fullstep_mintorque_ctr_clk

```
const uint8_t uni_4phase_fullstep_mintorque_ctr_clk[4] = {0x4, 0x2, 0x1, 0x8}  [static]
```

Index