3 speed Rotary Encoder Half Step library for Arduino

This is an optimized three speed Rotary Encoder library for Arduino which supports:

- Half step Rotary Encoder types.
- Detect three rotation speeds.
- Configurable rotation speed sensitivity.
- Polling and interrupt based.
- Single or multiple Rotary Encoders.
- Optional Rotary button.
- Pin state table in flash.

Half step / half step Rotary Encoders

The difference between a half step or half step Rotary Encoder type is how the data signals of the two pins are generated. It depends on the mechanical construction of the notches and contacts inside the Rotary Encoder.

Please refer to the ErriezRotaryEncoderFullStep library for full step Rotary Encoders. Experiment with the full step and half step libraries which works optimal for your Rotary Encoder.

Hardware

Connect the two rotary pins to the DIGITAL pins of an Arduino board.

A third rotary button pin is not used in the Rotary library, but can be used in the sketch.

Tested with Arduino IDE v1.8.5 on hardware:

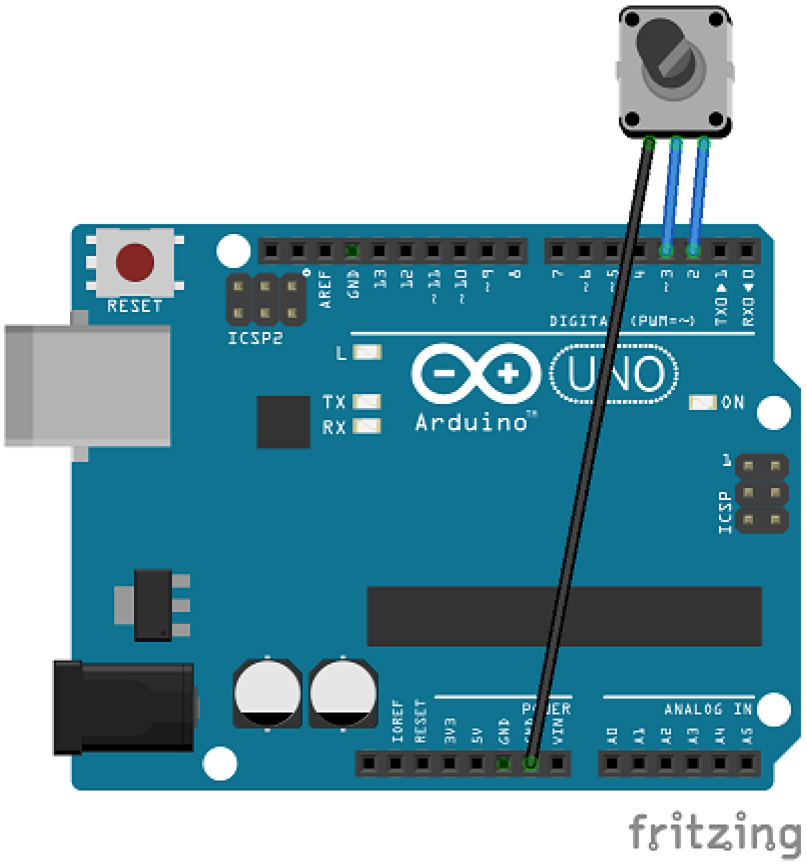
- Arduino UNO
- Arduino Nano
- Arduino Micro
- Arduino Pro or Pro Mini
- Arduino Mega or Mega2560
- Arduino Leonardo
- WeMos D1 R2 & mini (ESP8266)

Interrupts

Both rotary pins must be connected to a DIGITAL pin with interrupt support, such as `INT0` or `INT1`. This is chip specific. Please refer to the documentation of your board or attachInterrupt().

Arduino UNO hardware

The connection below can be used for polled and interrupts. An optional button pin can be connected to DIGITAL pin 4.



Rotary pin	Arduino UNO/NANO/Mega2560/Leonardo board
1	D2 (INT0)
2	D3 (INT1)
Button (optional)	D4
GND	GND

Arduino WeMos D1 R2 & mini (ESP8266) hardware

Note that some ESP8266 pins mixes ESP8622 GPIO pins with Arduino digital pins. Connect a Rotary Encoder to the following pins which can be used with polled and interrupt examples:

Rotary pin	ESP8622 pin	Text on board WeMos D1 R2
1	GPIO13	D7 MOSI
2	GPIO12	D6 MISO
Button (optional)	GPIO14	D5 SCK
LED (Not used)	GPIO2	D4
GND	GND	GND

Note: An external pull-up resistor is required when a pin does not have an internal pull-up.

```
// Connect the rotary pins to the WeMos D1 R2 board:
#define ROTARY_PIN1         12
#define ROTARY_PIN2         13
#define ROTARY_BUTTON_PIN   14
```

Examples

The following examples are available:

- Rotary | Interrupt | InterruptHalfStepBasic
- Rotary | Interrupt | InterruptHalfStepButton
- Rotary | Interrupt | InterruptHalfStepCounter
- Rotary | Polled | PolledHalfStepBasic
- Rotary | Polled | PolledHalfStepButton
- Rotary | Polled | PolledHalfStepCounter
- Rotary | Polled | PolledHalfStepMultiple

Usage

Read rotary with polling

```
#include <RotaryHalfStep.h>

// Connect rotary pins to the DIGITAL pins of the Arduino board
#define ROTARY_PIN1   2
#define ROTARY_PIN2   3

// Enable ONE of the three constructors below with different number of arguments:

// Initialize half step rotary encoder, default pull-up enabled, default
// sensitive=100
```

```
RotaryHalfStep rotary(ROTARY_PIN1, ROTARY_PIN2);

// Or initialize half step rotary encoder, pull-up disabled, default sensitive=100
// RotaryHalfStep rotary(ROTARY_PIN1, ROTARY_PIN2, false);

// Or initialize half step rotary encoder, pull-up enabled, sensitive 1..255
// A higher value is more sensitive
// RotaryHalfStep rotary(ROTARY_PIN1, ROTARY_PIN2, true, 150);

void loop()
{
  int rotaryState = rotary.read();

  // rotaryState = -3: Counter clockwise turn, multiple notches fast
  // rotaryState = -2: Counter clockwise turn, multiple notches
  // rotaryState = -1: Counter clockwise turn, single notch
  // rotaryState = 0:  No change
  // rotaryState = 1:  Clockwise turn, single notch
  // rotaryState = 2:  Clockwise turn, multiple notches
  // rotaryState = 3:  Clockwise turn, multiple notches fast
}
```

Read rotary with interrupts

```
#include <RotaryHalfStep.h>

// Connect rotary pins to Arduino DIGITAL pins with interrupt support:
//
// +-----------------------------------+--------------------------+
// |              Board                |  DIGITAL interrupt pins  |
// +-----------------------------------+--------------------------+
// | Uno, Nano, Mini, other 328-based  |  2, 3                    |
// | Mega, Mega2560, MegaADK           |  2, 3, 18, 19, 20, 21    |
// | Micro, Leonardo, other 32u4-based |  0, 1, 2, 3, 7           |
// +-----------------------------------+--------------------------+
//
#define ROTARY_PIN1   2
#define ROTARY_PIN2   3

// Enable ONE of the three constructors below with different number of arguments:

// Initialize half step rotary encoder, default pull-up enabled, default
// sensitive=100
RotaryHalfStep rotary(ROTARY_PIN1, ROTARY_PIN2);

// Or initialize half step rotary encoder, pull-up disabled, default sensitive=100
// RotaryHalfStep rotary(ROTARY_PIN1, ROTARY_PIN2, false);

// Or initialize half step rotary encoder, pull-up enabled, sensitive 1..255
// A higher value is more sensitive
// RotaryHalfStep rotary(ROTARY_PIN1, ROTARY_PIN2, true, 150);

void setup()
```

```
{
  // Initialize pin change interrupt on both rotary encoder pins
  attachInterrupt(digitalPinToInterrupt(ROTARY_PIN1), rotaryInterrupt, CHANGE);
  attachInterrupt(digitalPinToInterrupt(ROTARY_PIN2), rotaryInterrupt, CHANGE);
}

void rotaryInterrupt()
{
  int rotaryState = rotary.read();

  // rotaryState = -3: Counter clockwise turn, multiple notches fast
  // rotaryState = -2: Counter clockwise turn, multiple notches
  // rotaryState = -1: Counter clockwise turn, single notch
  // rotaryState = 0:  No change
  // rotaryState = 1:  Clockwise turn, single notch
  // rotaryState = 2:  Clockwise turn, multiple notches
  // rotaryState = 3:  Clockwise turn, multiple notches fast
}
```

Libary installation using Git

`Git` is the preferred way to keep this library up to date, because the Arduino Library manager does not update as long as this library is not added to the official Arduino Library database.

Install Git client for Windows

Install a Git client for Windows.

Install Git client for Linux

Open a command prompt and install a Git client for Linux, such as Debian Ubuntu:

```
sudo apt-get install git
```

Get Arduino libraries directory

This library must be installed in the Arduino Sketchbook library subdirectory.

To retrieve the Arduino Sketchbook directory, open the Arduino IDE Preferences dialog box via: `File` | `Preferences` | `Settings tab` and copy the Sketchbook location.

For example on:

- Windows : `C:\Users\User\Documents\Arduino`
- Linux: `/home/user/Arduino`

Clone this library

Clone this library by opening a command prompt:

- Windows: (`Windows key + R`, Type `cmd` + `[ENTER]`)

- Linux: Depends on your version.

Then type:

```
# Change directory to the sketchbook directory as configured in the Arduino IDE:
# Windows:
cd C:\Users\User\Documents\Arduino
# Linux:
cd ~/Arduino

# Go to the libraries subdirectory
cd libraries

# Run the git clone library once:
git clone https://github.com/Erriez/ErriezRotaryEncoderHalfStep.git
```

IMPORTANT: Restart the Arduino IDE.

Update this library

Open a command prompt and type:

```
# Change directory to the sketchbook directory as configured in the Arduino IDE:
# Windows:
cd C:\Users\User\Documents\Arduino
# Linux:
cd ~/Arduino

# Go to the libraries subdirectory
cd libraries

# Update the library:
git pull
```

IMPORTANT: Restart the Arduino IDE.

Libary installation with a ZIP

This method is not preferred, because updates should be manually installed.

1. Download the latest version here.
2. Open the Arduino IDE.
3. Sketch | Include Library | Add .ZIP library...
4. Browse to the downloaded ZIP.
5. Restart the IDE.