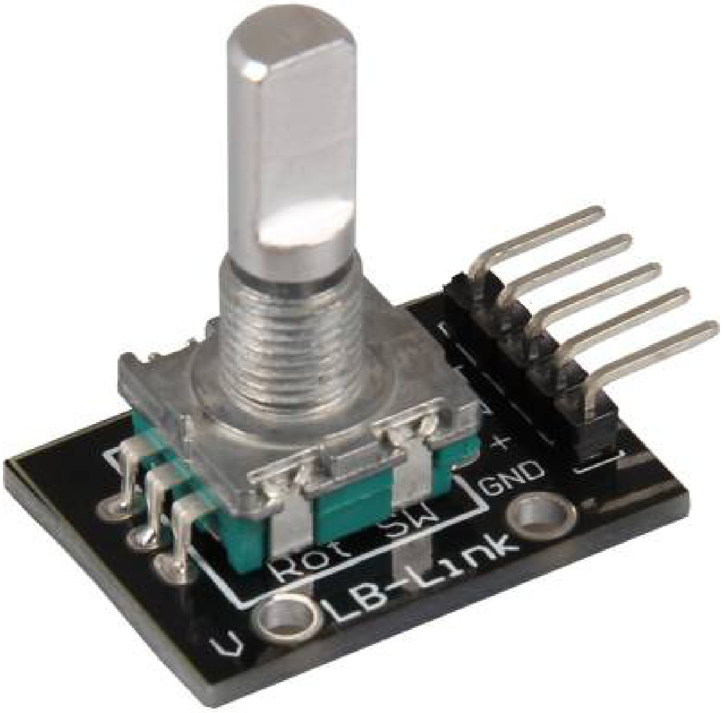
KY-040 Rotary encoder

Contents

Picture



Technical data / Short description

The current position of the rotary switch will be send encoded to the output.

Encoding

The idea of the rotary switch is that with every "step" only one of the conditions will change. In order of which status have changed first, you can see the rotational direction if you look at the following encoding.

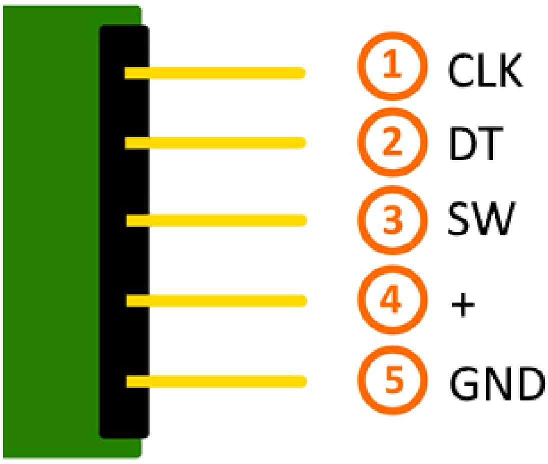
Clockwise [A will change first] -> Pin_CLK

A	B
0	0
1	0
1	1
0	1
0	0

<u>Counterclockwise</u> [B will change first] -> Pin_DT

A	B
0	0
0	1
1	1
1	0
0	0

Pinout



Code example Arduino

The program checks which pin has changed first, to get the rotational direction, after detecting a change at the pin status. You will get the rotational direction after you compare the current status of the pins with the last status. After detecting the rotational direction, the steps from the start will be counted and outputted. Pushing the button of the rotary encoder will reset the current position.

For serial output: Baudrate = 115200

```
// Initialization of the needed variables<br />
int Counter = 0;
boolean Richtung;
int Pin_clk_Letzter;
int Pin_clk_Aktuell;

// Definition of the input-pins
int pin_clk = 3;
int pin_dt = 4;
int button_pin = 5;


void setup()
{
   // Initialization of the input-pins...
   pinMode (pin_clk,INPUT);
   pinMode (pin_dt,INPUT);
   pinMode (button_pin,INPUT);

   // ...and activating of their pull up resistors
   digitalWrite(pin_clk, true);
   digitalWrite(pin_dt, true);
   digitalWrite(button_pin, true);

   // Initial reading of the Pin_CLK
   Pin_clk_Letzter = digitalRead(pin_clk);
   Serial.begin (115200);
 }

// The program checks, which of the status pins have changed first

void loop()
{
   // Reading of the current status
   Pin_clk_Aktuell = digitalRead(pin_clk);

   // Check for a Change
   if (Pin_clk_Aktuell != Pin_clk_Letzter)
   {

                if (digitalRead(pin_dt) != Pin_clk_Aktuell)
                {
                        // Pin_CLK has changed first
                        Counter ++;
                        Richtung = true;
                }

                else
                {       // Else Pin_DT changed first
                        Richtung = false;
                        Counter--;
                }

                Serial.println ("Rotation detected: ");
```

```
                  Serial.println ("Rotation detected: ");
                  Serial.print ("Rotational direction: ");

                  if (Richtung)
                  {
                     Serial.println ("Clockwise");
                  }
                  else
                  {
                     Serial.println("Counterclockwise");
                  }

                  Serial.print("Current position: ");
                  Serial.println(Counter);
                  Serial.println("------------------------------");

     }

     // Preparation for the next run:
     // The current value will be the last value for the next run.
     Pin_clk_Letzter = Pin_clk_Aktuell;

     // Reset funciton to save the current position
     if (!digitalRead(button_pin) && Counter!=0)
       {
         Counter = 0;
         Serial.println("Position resetted");
       }

   }
```

Connections Arduino:

CLK	=	[Pin 3]
DT	=	[Pin 4]
Button	=	[Pin 5]
+	=	[Pin 5V]
GND	=	[Pin GND]

Code example download

KY-40_rotary-encoder_ARD

Code example Raspberry Pi

The program checks which pin has changed first, to get the rotational direction, after detecting a change at the pin status. You will get the rotational direction after you compare the current status of the pins with the last status. After detecting the rotational direction, the steps from the start will be counted and outputted. Pushing the button of the rotary encoder will reset the current position.

```
# coding=utf-8
# Needed modules will be imported and configured
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

# Declaration and initialisation of the input pins which are connected with the sensor.
PIN_CLK = 16
```

```
PIN_DT = 15
BUTTON_PIN = 14

GPIO.setup(PIN_CLK, GPIO.IN, pull_up_down = GPIO.PUD_UP)
GPIO.setup(PIN_DT, GPIO.IN, pull_up_down = GPIO.PUD_UP)
GPIO.setup(BUTTON_PIN, GPIO.IN, pull_up_down = GPIO.PUD_UP)

# Needed variables will be initialised
Counter = 0
Richtung = True
PIN_CLK_LETZTER = 0
PIN_CLK_AKTUELL = 0
delayTime = 0.01

# Initial reading of Pin_CLK
PIN_CLK_LETZTER = GPIO.input(PIN_CLK)

# This output function will start at signal detection
def ausgabeFunktion(null):
    global Counter

    PIN_CLK_AKTUELL = GPIO.input(PIN_CLK)

    if PIN_CLK_AKTUELL != PIN_CLK_LETZTER:

        if GPIO.input(PIN_DT) != PIN_CLK_AKTUELL:
            Counter += 1
            Richtung = True;
        else:
            Richtung = False
            Counter = Counter - 1

        print "Rotation detected: "

        if Richtung:
            print "Rotational direction: Clockwise"
        else:
            print "Rotational direction: Counterclockwise"

        print "Current position: ", Counter
        print "------------------------------"

def CounterReset(null):
    global Counter

    print "Position reset!"
    print "------------------------------"
    Counter = 0

# To include a debounce, the output function will be initialised from the GPIO Python Module
GPIO.add_event_detect(PIN_CLK, GPIO.BOTH, callback=ausgabeFunktion, bouncetime=50)
GPIO.add_event_detect(BUTTON_PIN, GPIO.FALLING, callback=CounterReset, bouncetime=50)


print "Sensor-Test [press ctrl-c to end]"

# Main program loop
try:
        while True:
            time.sleep(delayTime)

# Scavenging work after the end of the program
except KeyboardInterrupt:
        GPIO.cleanup()
```

Connections Raspberry Pi:

CLK	=	GPIO16	[Pin 36]
DT	=	GPIO15	[Pin 10]
SW	=	GPIO14	[Pin 8]
+	=	3,3V	[Pin 1]
GND	=	GND	[Pin 6]

Example program download

KY-040_rotary-encoder_RPi

To start, enter the command:

```
sudo python KY-040_rotary-encoder_RPi.py
```