

Mehr[Blog erstellen](#) [Anmelden](#)

Cool-Emerald

Our thoughts and memories...

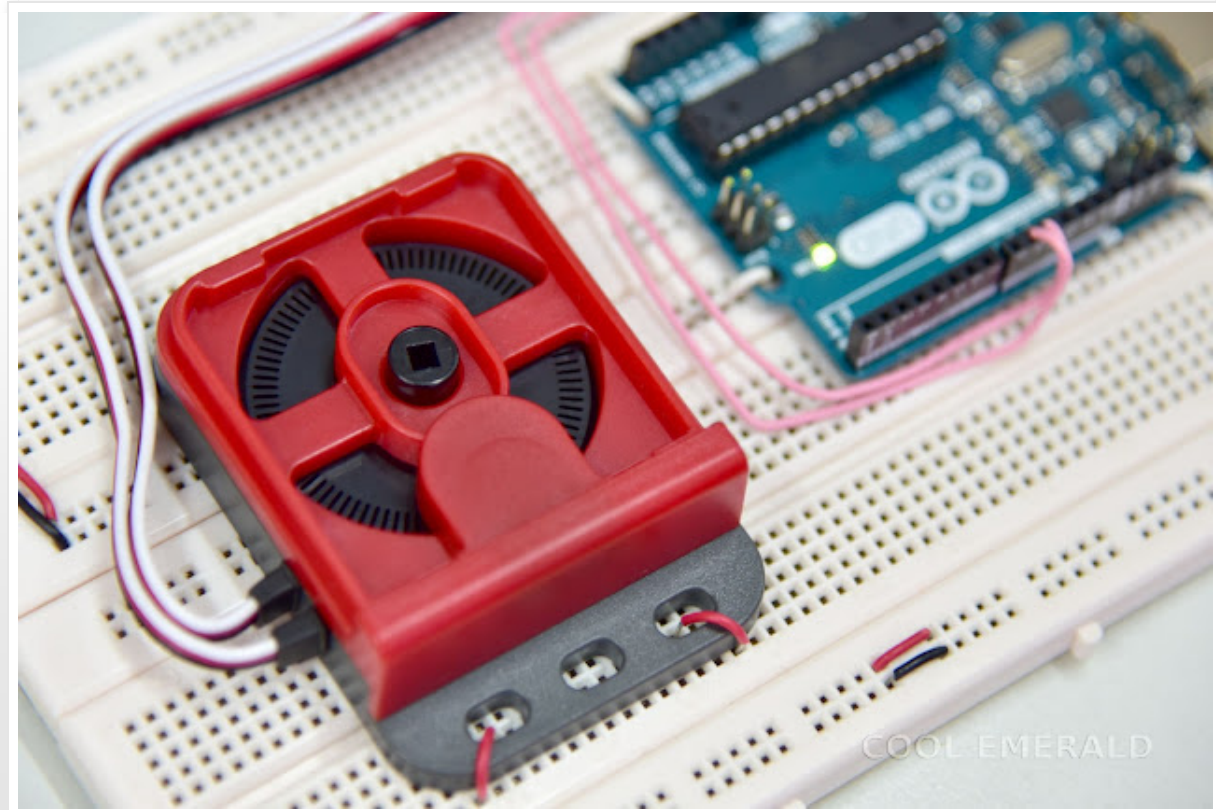
Sasana Year 2566 , Myanmar Year 1384 , Tagu new moon, Friday , Sabbath.

[Home](#)[Contact Us](#)[Profile](#)[About](#)

Thursday, March 6, 2014

Reading Rotary Encoder Using Microcontroller

Rotary encoders are commonly used for measuring angular position or motion sensing. An optical encoder has a disc with a pattern of cutouts. As the disc rotated, an LED light that shines on photo detector is turned on and off accordingly to produce a digital waveform.



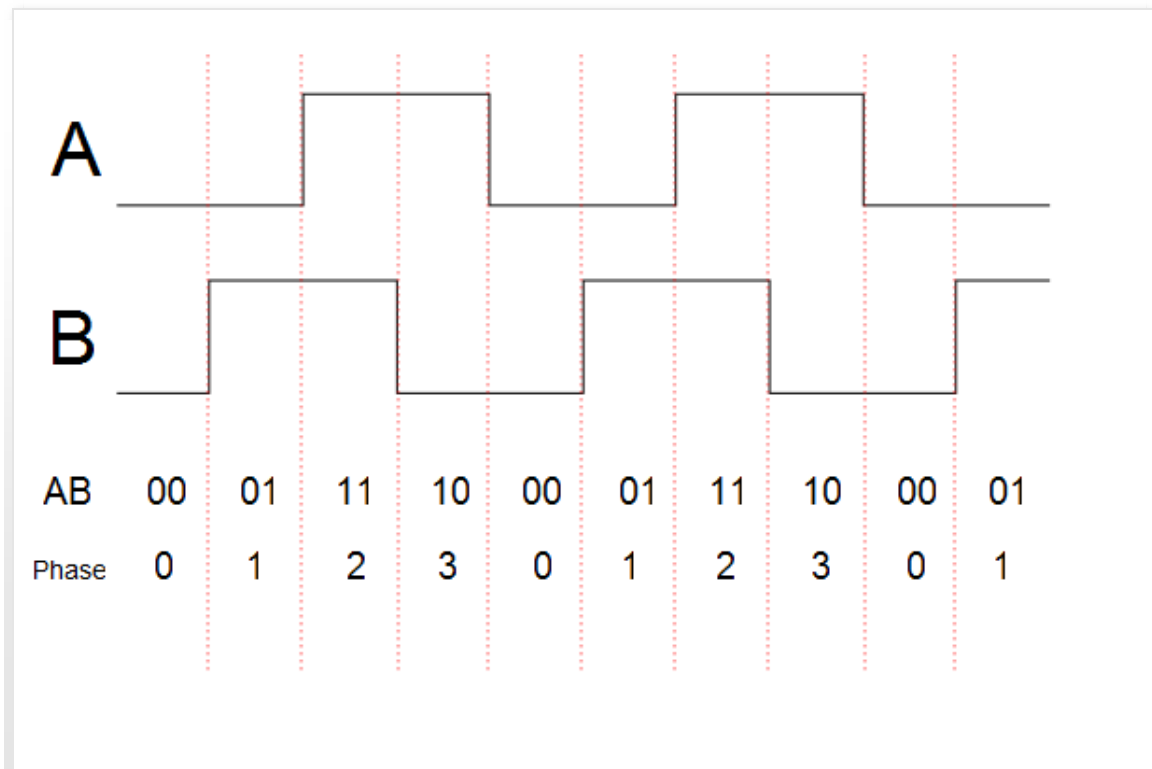
An optical encoder produced by VEX Robotics Design System

Gray code is normally used in encoders instead of ordinary binary code to prevent glitches. In Gray code, the number of changing bits between successive numbers is only 1. The following table shows 2 bit Gray code from 0 to 3.

Gray code

Number	2-bit Gray Code
0	00
1	01
2	11
3	10

Encoders typically have two outputs called A and B. When it is turned clockwise, the waveform as shown in the following figure is produced. Its phase increases from 0 to 3. When it is turned counter-clockwise, the output phases are produced in reverse order.



I have found several example programs in the Internet to read an encoder from a microcontroller. But I think, those programs are long and inefficient. The program presented here is simple, short and efficient. The resulting resolution of the program is 4 times the pulses per revolution of the encoder.

I use the state machine design. I define the output phases of the encoder 0, 1, 2, and 3 as states - s0, s1, s2, and s3 respectively. Then, the counting of the encoder states is shown in the following table.

Counting encoder states

Next state	Present state	Count
s0	s0	No change
s0	s1	Down
s0	s3	Up
s0	s2	Don't care
s1	s0	Up
s1	s1	No change
s1	s3	Don't care
s1	s2	Down
s3	s0	Down
s3	s1	Don't care
s3	s3	No change
s3	s2	Up
s2	s0	Don't care
s2	s1	Up
s2	s3	Down
s2	s2	No change

When the states are replaced by the corresponding binary bits, the following truth table is obtained.

Truth table

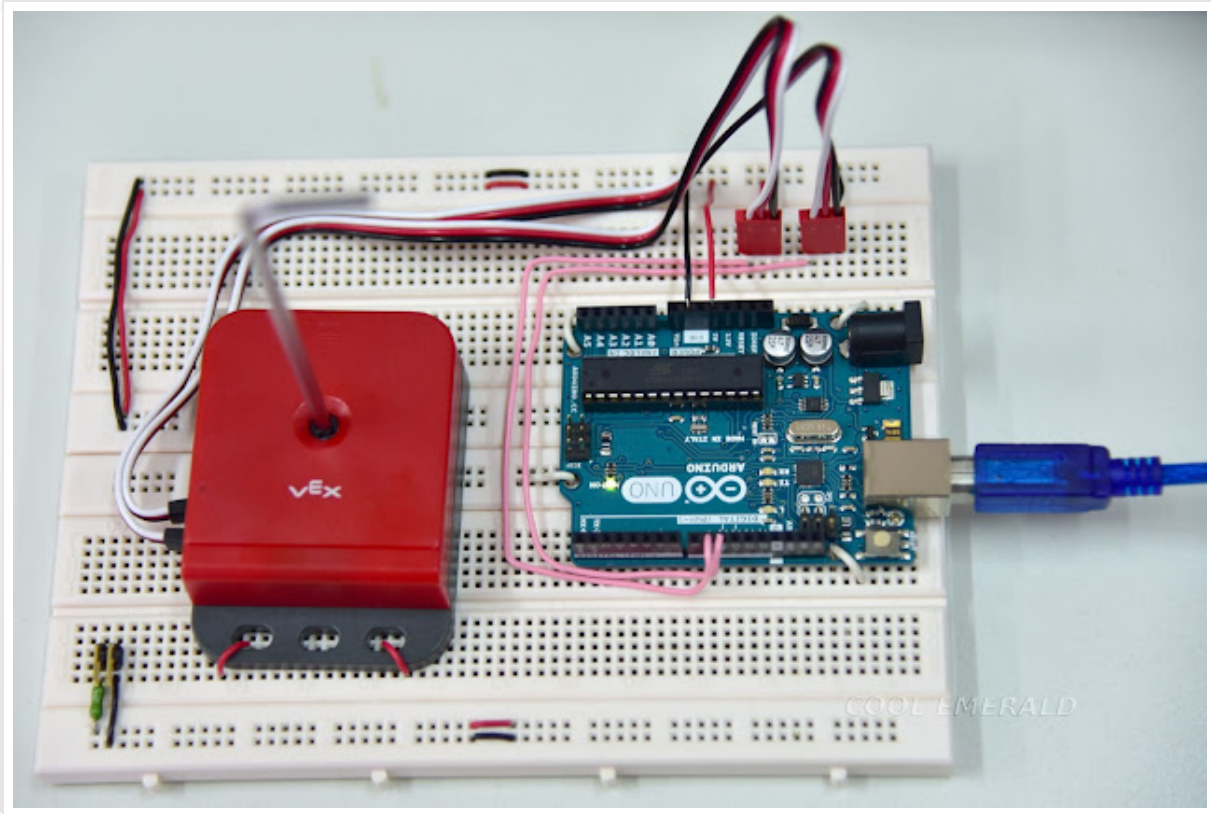
Decimal number	Next state - Present state	Up	Down
0	00 00	0	0
1	00 01	0	1
2	00 10	1	0
3	00 11	0	0
4	01 00	1	0
5	01 01	0	0
6	01 10	0	0
7	01 11	0	1
8	10 00	0	1
9	10 01	0	0
10	10 10	0	0
11	10 11	1	0

12	11 00	0	0
13	11 01	1	0
14	11 10	0	1
15	11 11	0	0

By considering next state and present state in the truth table as binary code, it is counting up at 2, 4, 11, and 13. Similarly, it is counting down at 1, 7, 8, and 14. An array can be declared to represent the truth table as follows.

```
int En_TruthTable[] = {0,-1,1,0,1,0,0,-1,-1,0,0,1,0,1,-1,0};
```

Arduino UNO single board microcontroller and an optical shaft encoder from VEX Robotics Design System are used in this example.



In our circuit, pin 9 of the microcontroller is connected to channel A of the encoder and pin 10 is connected to channel B. The code to get the next state (NS) from the bitwise reading of channel A and channel B is shown below.

```
NS = (digitalRead(pinA)<<1) | digitalRead(pinB);
```

As we turn the shaft, the encoder produces a series of digital pulses and the microcontroller has to

constantly check and update the position everytime the next state (NS) is different from the present state (PS). If c is a variable to keep track of the encoder position, its values should reach back to zero when the encoder has been turned a complete revolution. The number of state changes for one revolution (SPR) is four times the pulses per revolution (PPR) of the encoder ($SPR = 4 \cdot PPR$).

$$c = (c + \text{En_TruthTable}[(NS \ll 2) | PS] + SPR) \% SPR;$$

The angle (a) that corresponds to c in degree ($0 \leq a < 360$) is calculated as follows.

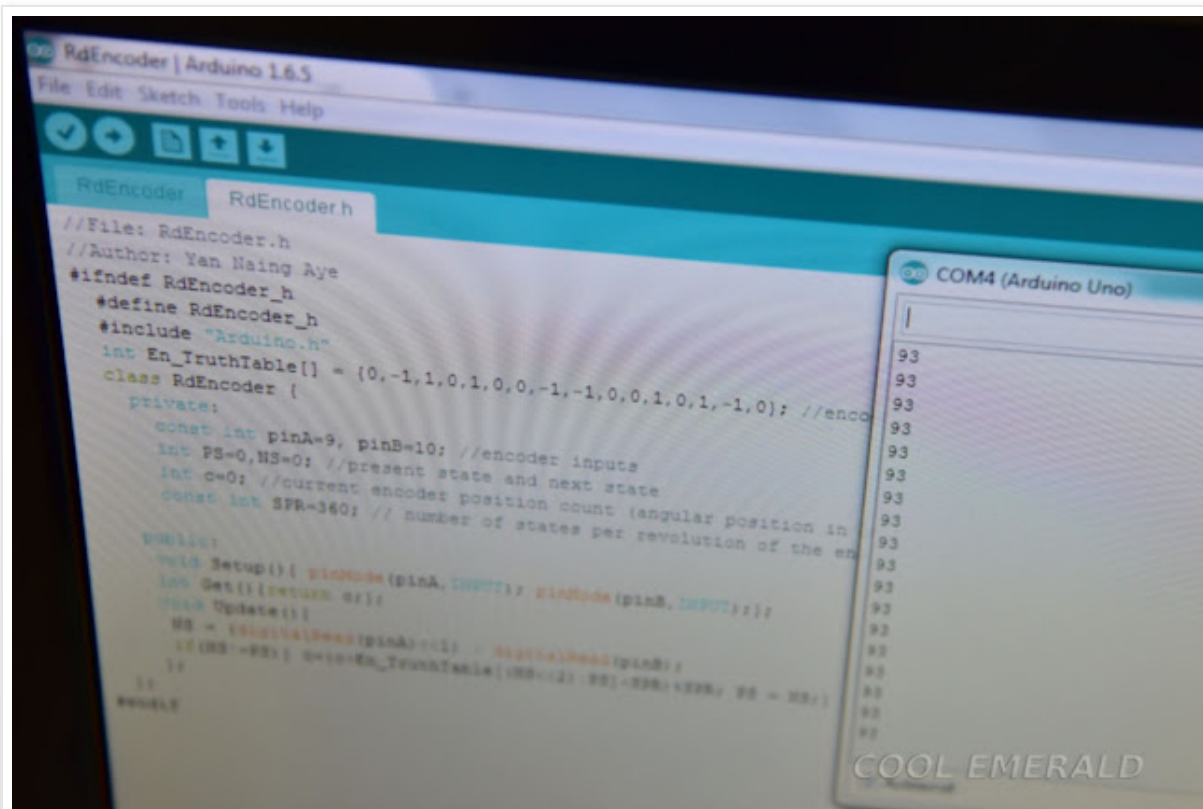
$$a = c * 360.0 / SPR$$

Similarly, the angle (b) in degree ($-180 \leq a < 180$) can be calculated from the angle (a).

$$b = a - \text{floor}(a/180) * 360.$$

An example program is available at [Rotary Encoder using Arduino to get absolute value \(on GitHub\)](#).

Arduino software (IDE) is available for free and I found it very easy to use. In my case, after I have chosen the correct board and correct COM port in the "Tools" menu, it worked without any problem.



Posted by Yan Naing Aye at Thursday, March 06, 2014



Labels: C, Circuit, Electronics, Embedded System, Encoder, Hardware, Microcontroller, Robotics

No comments:

Post a Comment

Comments are moderated and don't be surprised if your comment does not appear promptly.



Enter Comment

[Newer Post](#)

[Home](#)

[Older Post](#)

Subscribe to: [Post Comments \(Atom\)](#)

Blog Archive

- ▶ [2022](#) (2)
- ▶ [2021](#) (5)
- ▶ [2020](#) (12)
- ▶ [2018](#) (12)
- ▶ [2017](#) (13)
- ▶ [2016](#) (10)
- ▶ [2015](#) (7)
- ▼ [2014](#) (6)
 - ▶ [September](#) (2)
 - ▶ [August](#) (2)
 - ▼ [March](#) (2)
 - [Travel Checklist](#)

Reading Rotary Encoder Using Microcontroller

- [2013](#) (14)
- [2012](#) (4)
- [2011](#) (11)
- [2010](#) (10)
- [2009](#) (10)
- [2008](#) (11)

Labels

.NET 2.4 GHz 2005 6 DOF 8051 A4988 Accelerometer Actuator Adaptive Filter **ADC** ADS1115 Aircon AJAX **Algorithm** Amplifier **Arduino** ARM Armbian Astrological AT89C51CC03 BeagleBoard BeagleBone Bibliography Bluetooth BMFLC Book Burmese byte stuffing **C** **C++** Calendar CAN Capacitor CC2530 CCTV Centroid of Myanmar **Circuit** Clock CMake CMUcam5 Code Code::Blocks CodeLite Color Com **Comm** **Communication** Compensation Computer Vision Control CoreXY cpprestsdk CRC Cross-compiling Cross-platform CRP Crystal Oscillator Current Transformer Curve Fitting D-H model DAC Database Daughters Device Driver Diode Discrete Backlash Operator DIY DLL DOF DrawBot Driver DRV2700 DRV8825 DRV8834 DS1307 Dynamic Link Library **Electronics** email Embedded System Encoder Energy Ethernet face-detection Family FAT32 Feed-forward **firmware** Flash Fourier FRDM-K64F FRDM-K82F **Free** **Software** FreeTDS Freeware Fujitsu Function generator Fusion g++ galaxy note 2 gcc gcode **GIMP** Gnuplot grbl Gregorian Gyroscope Haar Cascades **Hardware** HDR Health HFMD High pass filter Human Arm Hysteresis I2C IAR IEEE 802.15.4 ILI9341 image **Image processing** IMU Inertial Measurement Unit Inkscape Integration **Interface** internet interrupt **JavaScript** JN5168 JSON JsonCpp Julian k-means K64 K82 KiCad Kids **Kinematics** L3G4200D LabVIEW Laser LaTeX LCD **Linux** LM337 LM358N Low pass filter LPC54102 LPC824 LPCXpresso824-MAX LuaTeX Mathematics **MatLab** Matplotlib MCP3008 **MCU** **MCUXpresso** Mechanical Mechatronics Medical Mesh Bee Meter **Microcontroller** ModBus Model Monitor Monte Carlo Integration Motor Driver **MySQL** multidrop multithreading **Myanmar** **MySQL** Network Notiflycon NXP **Octave** ODBC Odroid Op-amp op-amp oscillator Open source **opencv** opencv-3.2.0 opencv3 OPi Orange Pi **Parallel** Port password pca9535 PCB PCF8591 personal phone **Photography** Piezoelectric PIR Sensor Pixy pkg-config Prandtl-Ishlinskii **Programming** Qt **Raspberry Pi** RaspiCam RC real-time Reference Regulator Relay repair REST robot **Robotics** Rotation RS232 RS422 RS485 SB-900 SBC schroot SD Card SDCC **Sensor** Serial Serial Port **Signal** **Processing** Simple Smart Home Socket soft-timer **SPI** **SQL** Stages Stepper Motor Surveillance SVG System TCP Template tesseract tesseract-ocr **TeXstudio** Thesis TI Timer Tool Touch Transceiver Translation Travel **UART** Ubuntu UDP USB v4l2 **VB** vector graphic Velvia Virtual Com **Visual Basic** Visual Studio Visual Studio 2015 VNC VS14 Web Windows **Wireless** Writing wxDev-C++ wxJSON **wxWidgets** X DevAPI XeLaTeX XeTeX XY Plotter **Zigbee**

00042647

Powered by **Blogger**.