Arudino notes

***www.arexx.com --> Forum***

***www.roboternetz.de --> Forum***





/\*

ArdBot bumper switch demo

Requires Arduino IDE version 0017

or later (0019 or later preferred)

\*/

#include <Servo.h>

const int ledPin = 13; // Built-in LED

const int bumpLeft = 2; // Left bumper pin 2

const int bumpRight = 3; // Left bumper pin 3

int pbLeft = 0; // Var for left bump

int pbRight = 0; // Var for left bump

Servo servoLeft; // Define left servo

Servo servoRight; // Define right servo

void setup() {

servoLeft.attach(10); // Left servo pin D10

servoRight.attach(9); // Right servo pin D9

// Set pin modes

pinMode(bumpLeft, INPUT);

pinMode(bumpRight, INPUT);

pinMode(ledPin, OUTPUT);

}

void loop() {

forward(); // Start forward

// Test bumper switches

pbLeft = digitalRead(bumpLeft);

pbRight = digitalRead(bumpRight);

// Show LED indicator

showLED();

// If left bumper hit

if (pbLeft == HIGH) {

reverse();

delay(500);

turnRight();

delay(1500);

}

// If right bumper hit

if (pbRight == HIGH) {

reverse();

delay(500);

turnLeft();

delay(1500);

}

}

// Motion routines

void forward() {

servoLeft.write(180);

servoRight.write(0);

}

void reverse() {

servoLeft.write(0);

servoRight.write(180);

}

void turnRight() {

servoLeft.write(180);

servoRight.write(180);

}

void turnLeft() {

servoLeft.write(0);

servoRight.write(0);

}

void stopRobot() {

servoLeft.write(90);

servoRight.write(90);

}

void showLED() {

// Show LED if a bumper is hit

if (pbRight == HIGH || pbLeft == HIGH) {

// Turn LED on

digitalWrite(ledPin, HIGH);

}

else {

// Turn LED off

digitalWrite(ledPin, LOW);

}

}

**3 Reading a switch**

The LED exercise shows how the Arduino can control

the outside world. Many applications require reading the

state of sensors, including switches. The figure to the

right shows a picture of a pushbutton switch and its

schematic symbol. Note that the symbol represents a

switch whose contacts are normally open, but then are

shorted when the button is pushed. If you have a switch,

use the continuity (beeper) function of a digital multimeter

(DMM) to understand when the leads are open and

when they are connected as the button is pushed.

For this exercise, the Arduino will read the state of a normally-open push button switch and

display the results on the PC using the serial.println() command. You will need a switch, a 10

kohm resistor and some pieces of 22 g hookup wire. If you don't have a switch, substitute two

wires and manually connect their free ends to simulate a switch closure. The figure below shows

the schematic for the circuit on the left and a realization on the right.

Create and run this Arduino program

void setup()

{

Serial.begin(9600);

}

void loop()

{

Serial.println(digitalRead(3));

delay(250);

}

Open the Serial Monitor window. When the switch is open, you should see a train of 1's on the

screen. When closed, the 1's change to 0's. On the hardware side, when the switch is open, no

current flows through the resistor. When no current flows through a resistor, there is no voltage

drop across the resistor, which means the voltage on each side is the same. In your circuit, when

the switch is open, pin 3 is at 5 volts which the computer reads as a 1 state. When the switch is

closed, pin 3 is directly connected to ground, which is at 0 volts. The computer reads this as a 0

state.

Now try this program which is an example of how you can have the computer sit and wait for a

sensor to change state.

void setup()

{

Serial.begin(9600);

}

void loop()

{

while (digitalRead(3) == HIGH)

;

Serial.println("Somebody closed the switch!");

while (digitalRead(3) == LOW)

;

Serial.println("The switch is now open!");

}

Watch the activity in the Serial Monitor window as you press and release the switch.